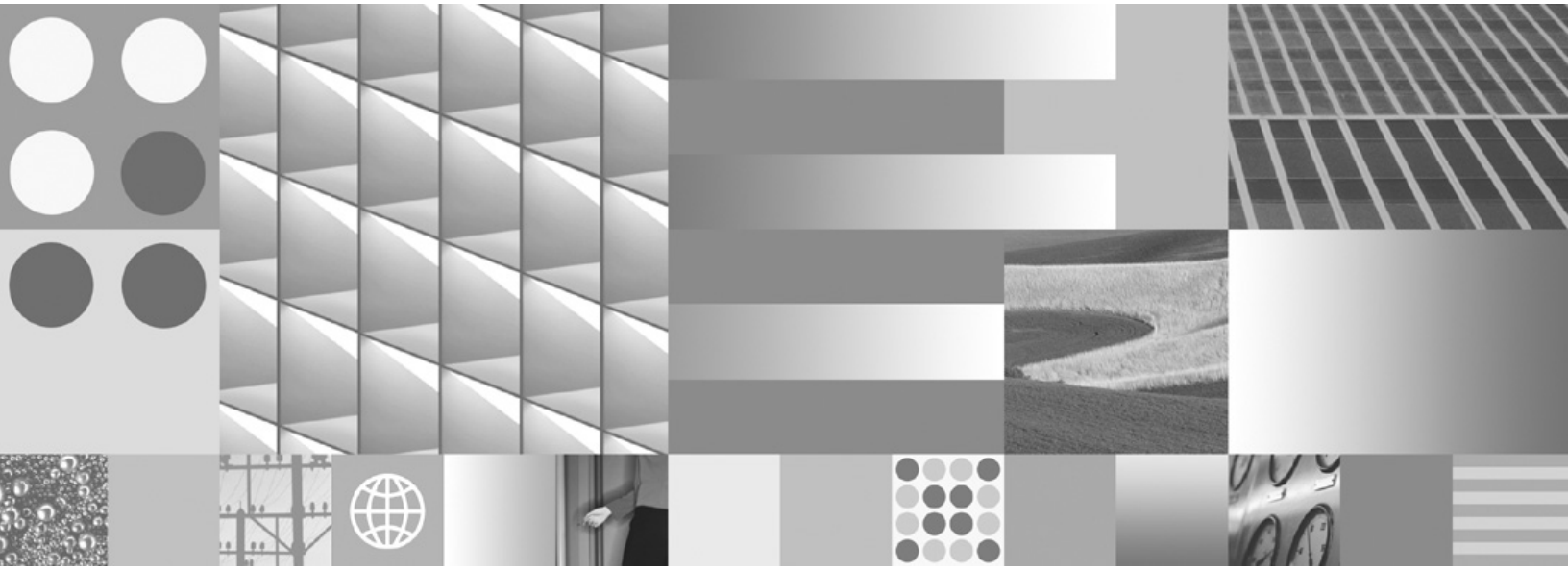
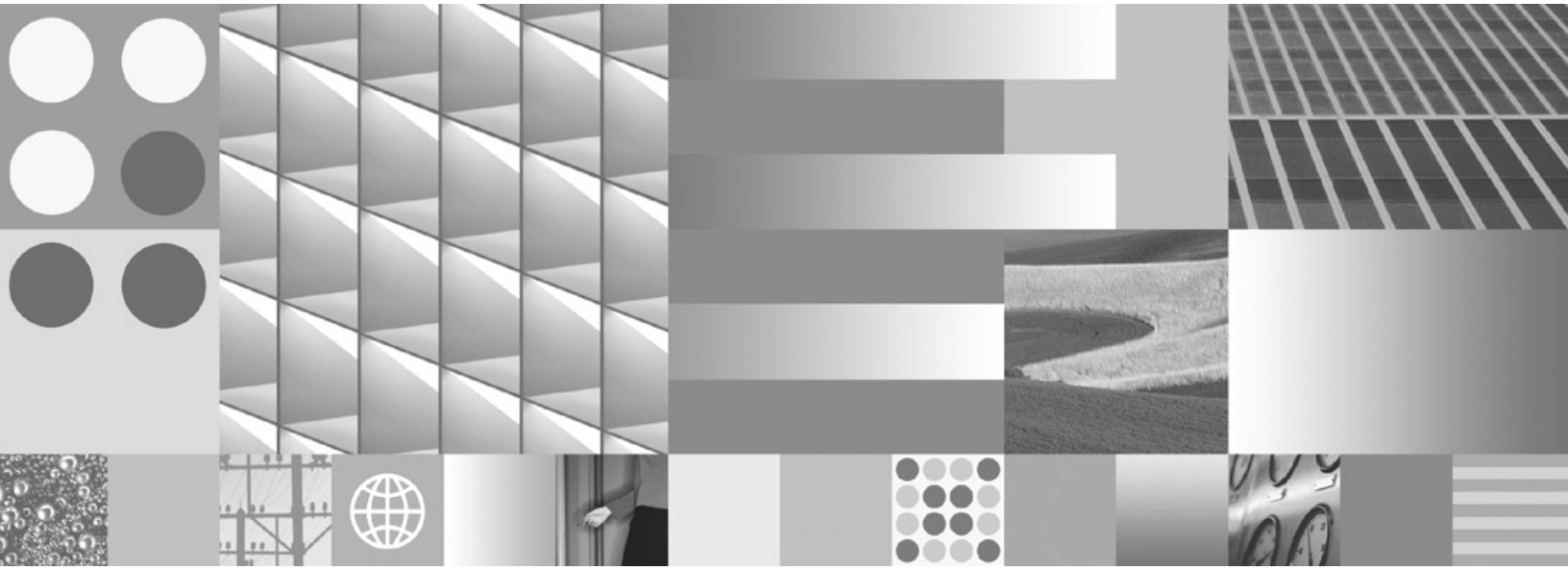
IBM DB2 9.7
Linux 版、UNIX 版和 Windows 版



数据库监视指南和参考

S151-1165-00

IBM DB2 9.7
Linux 版、UNIX 版和 Windows 版



数据库监视指南和参考

S151-1165-00

注意

使用此信息及其支持的产品前，请先阅读第 719 页的附录 B，『声明』下的常规信息。

修订版声明

此文档包含 IBM 的所有权信息。它在许可协议中提供，且受版权法的保护。本出版物中包含的信息不包括对任何产品的保证，且提供的任何语句都不需要如此解释。

您可在线或通过当地的 IBM 代表处订购 IBM 出版物。

- 要在线订购出版物，请转至 IBM 出版物中心，网址为: www.ibm.com/shop/publications/order
- 要查找当地的 IBM 代表处，请转至 IBM 全球联系人目录，网址为: www.ibm.com/planetwide

要从美国或加拿大的 DB2 市场和销售部订购 DB2 出版物，请致电 1-800-IBM-4YOU（426-4968）。

当您向 IBM 发送信息时，即同意授予 IBM 独一无二的权力以它认为适当且不会对您造成任何影响的方式使用或分发该信息。

目录

关于本书

《*系统监视器指南和参考*》描述了如何收集关于数据库和数据库管理器的不同类型的信息。

它还解释了如何使用收集的信息来了解数据库活动、改善性能和确定问题的原因。

第 1 部分 监视接口

第 1 章 数据库监视

对于数据库管理系统的性能和运行状况的维护而言，数据库监视是一个非常重要的活动。为便于进行监视，DB2® 从数据库管理器、数据库及所有已连接的应用程序收集信息。可借助此信息执行下列及其他任务:

- 根据数据库使用模式预计硬件要求。
- 分析各个应用程序或 SQL 查询的性能。
- 跟踪索引和表的使用情况。
- 查明系统性能下降的原因。
- 评估优化活动（如更改数据库管理器配置参数、添加索引或修改 SQL 查询）的效果。

第 2 章 监视器表函数概述

从 DB2 版本 9.7 开始，可以通过传统系统监视器的轻量级替代项来访问监视器数据。请使用监视器表函数来收集和查看系统、活动或数据对象的数据。

受监视元素的数据将在内存中持续累积并可供查询。您可以选择接收单一对象（例如服务类 A 或表 TABLE1）的数据或者所有对象的数据。

在数据库分区环境中使用这些表函数时，您可以选择接收单一分区的数据或所有分区的数据。如果您选择接收所有分区的数据，那么这些表函数将对每个分区返回一行。通过使用 SQL，可以对各个分区的值进行求和，以获取跨分区的监视元素值。

使用表函数来监视系统信息

系统监视透视图涵盖数据服务器为了处理应用程序请求而执行的全部工作。在此透视图中，您可以确定数据服务器的整体工作以及为特定应用程序请求子集完成的工作。

此透视图的监视元素被称为“请求监视元素”，它们涵盖所有与处理请求相关联的数据服务器操作。

请求监视元素将在内存中持续地进行累积和聚集，因此立即可供查询。请求监视元素将对各个级别的工作负载管理（WLM）对象层次结构的请求进行聚集：按工作单元、按工作负载或者按服务类。它们还按连接进行聚集。

请使用下列表函数来访问当前系统监视信息:

- MON_GET_SERVICE_SUBCLASS 和 MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_WORKLOAD 和 MON_GET_WORKLOAD_DETAILS
- MON_GET_CONNECTION 和 MON_GET_CONNECTION_DETAILS
- MON_GET_UNIT_OF_WORK 和 MON_GET_UNIT_OF_WORK_DETAILS

这组表函数使您能够下寻到或侧重于特定聚集级别的请求监视元素。表函数成对提供：一个表函数用于对常用数据进行关系访问，另一个表函数用于对全部可用的监视元素进行 XML 访问。

缺省情况下，对于新数据库，这些表函数将收集系统监视信息。您可以使用下列设置中的一个或全部来更改缺省设置:

- 数据库配置参数 **mon_req_metrics** 指定所有服务类中的最低收集级别。
- CREATE/ALTER SERVICE CLASS 语句的 COLLECT REQUEST METRICS 子句指定服务超类的收集级别。使用此设置来将给定服务类的收集级别增大为高于对所有服务类设置的最低收集级别。

每个设置的可能值如下所示:

NONE 不收集请求监视元素

BASE 收集所有请求监视元素

例如，要只收集部分服务类的系统监视信息，请执行下列操作:

1. 将数据库配置参数 **mon_req_metrics** 设置为 NONE。
2. 对于每个期望的服务类，将 CREATE/ALTER SERVICE CLASS 语句的 COLLECT REQUEST METRICS 子句设置为 BASE。

使用表函数来监视活动

活动监视透视图侧重于与执行活动相关的数据服务器处理子集。在 SQL 语句的上下文中，术语“活动”是指 SQL 语句节的执行。

此透视图的监视元素被称为“活动监视元素”，它们是请求监视元素的子集。活动监视元素用于度量为了执行语句节而完成的工作的各个方面。活动监视包括其他信息，例如活动的 SQL 语句文本。

对于进行中的活动而言，活动度量值将在内存中累积。对于作为 SQL 语句的活动而言，活动度量值还将在程序包高速缓存中累积。在程序包高速缓存中，活动度量值将针对每个 SQL 语句节的所有执行进行累积。

请使用下列表函数来访问活动的当前数据:

MON_GET_ACTIVITY_DETAILS
返回关于调用此表函数时进行中的各个活动的数据。数据以 XML 格式返回。

MON_GET_PKG_CACHE_STMT
返回对单个 SQL 语句节的所有执行聚集的数据。数据以关系格式返回。

缺省情况下，对于新数据库，将收集活动监视信息。您可以使用下列设置中的一个或全部来更改缺省设置:

- **mon_act_metrics** 数据库配置参数指定所有工作负载中的最低收集级别。
- CREATE/ALTER WORKLOAD 语句的 COLLECT ACTIVITY METRICS 子句用于指定给定工作负载的收集级别（高于对所有工作负载设置的最低收集级别）。

每个设置的可能值如下所示:

NONE 不收集活动监视元素

BASE 收集所有活动监视元素

例如，要仅收集所选工作负载的活动监视元素，请执行下列操作:

1. 将 **mon_act_metrics** 数据库配置参数设置为 NONE。
2. 将 CREATE/ALTER WORKLOAD 语句的 COLLECT ACTIVITY METRICS 子句设置为 BASE。缺省情况下，其他工作负载的值为 NONE。

使用表函数来监视数据对象

数据对象监视透视图提供关于对数据对象（即，表、索引、缓冲池、表空间和容器）执行的操作的信息。

对于每种对象类型，都有一组不同的监视元素。每当一个请求涉及处理数据对象时，该对象的监视元素都将递增。例如，在处理涉及从特定表中读取行的请求时，该表的“读取行数”度量值将递增。

请使用下列表函数来访问数据对象的当前详细信息:

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER
- MON_GET_TABLE
- MON_GET_INDEX

这些表函数以关系格式返回数据。

您无法访问数据对象的历史数据。

缺省情况下，将对新数据库收集数据对象监视元素。您可以使用 **mon_obj_metrics** 数据库配置参数来减少表函数所收集的数据量。

此配置参数的可能值如下所示:

NONE 不收集数据对象监视元素

BASE 收集所有数据对象监视元素

无论对 **mon_obj_metrics** 参数设置哪个值，都将始终为下列表函数所报告的监视元素收集数据:

- MON_GET_TABLE
- MON_GET_INDEX

要停止收集下列表函数所报告的数据对象监视元素，请将 **mon_obj_metrics** 配置参数设置为 NONE。

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER

第 3 章 事件监视器

事件监视器返回有关 CREATE EVENT MONITOR 语句中指定的事件类型的信息。对于每个事件类型，将在特定时间点收集监视信息。

下表列示收集监视数据时的可用事件类型和每个事件类型的可用信息。第一列中的可用事件类型对应定义事件类型的 CREATE EVENT MONITOR 语句中使用的关键字。

除了出现数据的已定义事件外，还可使用 FLUSH EVENT MONITOR SQL 语句来生成事件。此方法生成的事件将使用所有监视器类型（DEADLOCKS 和 DEADLOCKS WITH DETAILS 除外）的当前数据库监视器值写入，这些监视器类型与清空的事件监视器相关联。

使用语句事件监视器监视 SQL 过程的执行时:

- 数据操作语言（DML）语句，例如，INSERT、SELECT、DELETE 和 UPDATE，会生成事件。
- 过程语句，例如，变量赋值和控制结构（例如，WHILE 或 IF），不会以确定性方式生成事件。

表 1. 事件类型

事件类型	收集数据的时间	可用信息
DEADLOCKS[1]	死锁检测	涉及的应用程序及处于争用状态的锁定。
DEADLOCKS WITH DETAILS[1]	死锁检测	有关涉及的应用程序的综合信息，包括参与语句（和语句文本）的标识和要挂起的锁定的列表。如果使用 DEADLOCKS WITH DETAILS 事件监视器而不是 DEADLOCKS 事件监视器，那么会导致发生死锁时性能下降，原因是收集了其他的信息。
DEADLOCKS WITH DETAILS HISTORY[1]	死锁检测	DEADLOCKS WITH DETAILS 事件监视器中报告的所有信息以及每个应用程序的当前工作单元的语句历史记录，这些应用程序拥有的锁定参与了挂起该锁定的数据库分区的死锁方案。如果使用 DEADLOCKS WITH DETAILS HISTORY 事件监视器，那么会导致激活时性能轻微下降，原因是进行了语句历史记录跟踪。
DEADLOCKS WITH DETAILS HISTORY VALUES[1]	死锁检测	带有详细信息的死锁历史记录中报告的所有信息，以及在执行语句时对所有参数标记提供的值。如果使用 DEADLOCKS WITH DETAILS HISTORY VALUES 事件监视器，那么会导致激活时性能较为严重的下降，原因是额外复制数据值。
STATEMENTS	SQL 语句结束	语句启动或停止时间、使用的 CPU、动态 SQL 的文本、SQLCA（SQL 语句的返回码）及其他度量值，如访存计数。 **注:** 当时间戳记开关设置为 OFF 时，语句启动或停止时间不可用。
	子节结束	对于分区数据库: 耗用的 CPU、执行时间以及表和表队列信息。

表 1. 事件类型 (续)

事件类型	收集数据的时间	可用信息
TRANSACTIONS[2]	工作单元结束	UOW 工作启动或停止时间、先前的 UOW 时间、耗用的 CPU 以及锁定和记录度量值。如果使用 XA 运行，那么不会生成事务记录。
CONNECTIONS	连接结束	所有应用程序级别计数器。
DATABASE	数据库释放	所有数据库级别计数器。
BUFFERPOOLS	数据库释放	缓冲池、预取程序、页清除程序和每个缓冲池的直接 I/O 的计数器。
TABLESPACES	数据库释放	缓冲池、预取程序、页清除程序和每个表空间的直接 I/O 的计数器。
TABLES	数据库释放	对每个表读取或写入的行数。
活动	在启用了 COLLECT ACTIVITY DATA 选项的服务类、工作负载或工作类中执行的活动完成时。在执行 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 存储过程时也会对目标活动收集数据。 如果活动超过启用了 COLLECT ACTIVITY DATA选项的阈值，也会收集数据。	活动级别数据。如果指定了 WITH DETAILS 作为 COLLECT ACTIVITY DATA 的一部分，那么将对具有语句和编译环境信息的那些活动包含这些信息。如果还指定了 AND VALUES，那么还将对具有输入数据值的那些活动包含这些值。
统计信息	每隔 *period* 分钟，其中 *period* 为收集统计信息的时间长度。此期间在 WLM_COLLECT_INT 数据库配置参数中定义。 调用 WLM_COLLECT_STATS 存储过程时，也会收集数据。	从在系统中每个服务类、工作负载或工作类内执行的活动计算而来的统计信息。
阈值违例	检测到阈值违例时。	阈值违例信息。
锁定	在检测到下列任何类型的事件之后（取决于配置设置）：锁定超时、死锁以及锁定等待超出指定的持续时间。	锁定事件记录。
工作单元	在工作单元完成之后	工作单元事件记录。可以选择将请求度量值包括在记录中。

[1] 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

[2] 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

注： 将为每个新创建的数据库创建详细的死锁事件监视器。此事件监视器称为 DB2DETAILDEADLOCK，将在激活数据库时启动，并且写至数据库目录中的文件。可通过删除它来避免此事件监视器导致的开销。建议不要使用 DB2DETAILDEADLOCK 事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

写入非格式化事件表的事件监视器

DB2 9.7 引入了一些以非格式化事件表作为目标类型的事件监视器。此类事件监视器提供了更好的性能、新的 CREATE EVENT MONITOR 语句选项以及用于访问数据以进行分析的新接口。

非格式化事件表事件监视器的特征将影响您执行下列任务的方式:

- 创建事件监视器和配置数据收集功能
- 管理事件监视器操作
- 访问事件监视器所捕获的事件数据

通常，您可以通过为每个数据库创建用于给定事件类型（例如锁定事件）的单一事件监视器来满足所有监视需求。您可以更改设置以增加或减少可以通过监视器收集的数据量，以便满足不断变化的监视需求。这与某些旧的事件监视器不同，对于那些事件监视器，更常见的做法是创建多个事件监视器并让每一个用于满足特定的监视需求。

创建与事件监视器相关联的非格式化事件表

创建事件监视器的一个方面是，指定将该监视器所收集的数据写至何位置。此类事件监视器始终以二进制格式将数据写入非格式化事件表。非格式化事件表是 DB2 9.7 中引入的目标类型。每当您创建事件监视器时，都将以隐式方式创建非格式化事件表。用于此类事件监视器的 CREATE EVENT 语句包含 WRITE TO UNFORMATTED EVENT TABLE 子句。

CREATE EVENT MONITOR 语句包含下列用于配置非格式化事件表的选项:

- 表名 – 缺省情况下，非格式化事件表将根据事件监视器名称进行命名。
- 表空间名 – 缺省情况下，如果存在用户对其具有 USE 特权的 IBMDEFAULTGROUP 表空间，那么将在该表空间中创建非格式化表。但是，建议的做法是定义针对事件监视器进行优化的表空间，如下所述。
- PCTDEACTIVATE – 缺省值是 100，这意味着事件监视器将在表空间变满时停用。

您必须考虑下列有关非格式化事件表的表空间的事项:

- 为事件监视器非格式化事件表创建表空间并对其进行配置以提高性能。请使用 CREATE TABLESPACE 的下列子句:
  - 指定尽可能大的页大小（PAGESIZE），其大小可达 32KB。
  - 指定 NO FILE CACHING SYSTEM 选项。
- 在分区数据库环境中，考虑用于存放表空间的分区。如果目标非格式化事件表的表空间在某个数据库分区中不存在，那么该目标非格式化事件表的数据将被忽略。此行为允许用户创建只在特定数据库分区中存在的表空间，从而选择要监视的部分数据库分区。

有关非格式化事件表的其他有用信息包括:

- SYSCAT.EVENTTABLES 目录视图列示了事件监视器、与其相关联的非格式化表以及其他详细信息。
- 相关链接中列示的主题对非格式化事件表的各个列作了描述。

配置事件监视器的数据收集

设置事件监视器的工作涉及指定所要收集的数据。涉及的各个方面包括所要监视的系统工作负载子集、所要收集的事件类型、要为每个事件收集的详细信息量以及启用/禁用数据捕获功能（打开和关闭数据捕获功能）。与配置数据收集相关的注意事项如下所示:

- 对于此类事件监视器，您主要通过使用 CREATE/ALTER WORKLOAD 语句设置各个工作负载定义的属性来配置数据收集。即，可以为不同的工作负载指定不同的数据收集设置。对于特定类型的事件监视器，CREATE/ALTER WORKLOAD 语句包含特定的子句。
- 缺省情况下，此类事件监视器将自动激活。通过在 CREATE EVENT 语句中指定 MANUALSTART 关键字，您可以指定以手动方式激活事件监视器。然后，可以通过 SET EVENT MONITOR STATE 语句来控制事件监视器。
- 如另一个上下文所述，在分区数据库环境中，可以选择要使用事件监视器来监视的部分数据库分区。创建事件监视器时，请为非格式化事件表指定一个只驻留在那些所要监视的分区中的表空间。如果非格式化事件表在给定数据库分区中不存在，那么该事件监视器将不会收集该分区的数据。
- 此类事件监视器的数据收集不受使用 UPDATE MONITOR SWITCHES 语句设置的系统监视器开关设置影响，事件捕获功能也无法通过 SET EVENT MONITOR 语句来打开和关闭。

管理事件监视器操作

下列各点提供与管理事件监视器的进行中的操作相关的准则:

- 在任何时候，都可以使用 ALTER WORKLOAD 语句来更改对所要收集的数据的指定。
- 如果在 CREATE EVENT 语句中指定了 MANUALSTART 选项，那么可以使用 SET EVENT MONITOR STATE 语句来启动和停止数据收集操作。
- 必须以手动方式修剪非格式化事件表。
- 如果非格式化事件表已耗用所分配的最大空间量，那么事件监视器将停用。
- 如果不再需要某个事件监视器，请使用 DROP 语句将其删除。发出 DROP 语句并不会删除与事件监视器相关联的非格式化事件表。在删除事件监视器之后，必须以手动方式删除相关联的非格式化事件表。如果未删除非格式化事件表，并且以后尝试创建与现有表同名的非格式化事件表，那么将发生问题。

访问事件监视器所捕获的事件数据

此类事件监视器以二进制格式将数据写入非格式化事件表。您可以使用 db2evmonfmt 命令或例程来访问此数据。

借助 db2evmonfmt 命令，可以执行下列操作:

- 根据下列属性来选择感兴趣的事件: 事件标识、事件类型、时间段、应用程序、工作负载或服务类。
- 选择是以文本报告形式还是格式化 XML 文档形式来接收输出。
- 通过创建您自己的 XSLT 样式表代替使用 db2evmonfmt 附带提供的 XSLT 样式表，对输出格式进行全面控制。

您还可以使用下列例程从非格式化事件表中抽取数据:

- EVMON_FORMAT_UE_TO_XML – 将非格式化事件表中的数据抽取到 XML 文档。
- EVMON_FORMAT_UE_TO_TABLES – 将非格式化事件表中的数据抽取到一组关系表。

借助这些例程，可以使用 SELECT 语句来准确地指定要从非格式化事件表中抽取的行。

非格式化事件表的列定义

当您发出包含 WRITE TO UNFORMATTED EVENT TABLE 子句的 CREATE EVENT 语句时，将创建非格式化事件表。当您想要抽取数据以进行分析或者要从表中修剪掉不需要的数据时，列定义非常有用。

当您想要使用下列其中一个例程从非格式化事件表中抽取数据时，非格式化事件表的列定义非常有用:

- EVMON_FORMAT_UE_TO_XML – 将非格式化事件表中的数据抽取到 XML 文档。
- EVMON_FORMAT_UE_TO_TABLES – 将非格式化事件表中的数据抽取到一组关系表。

对这些例程的调用接受一个 SELECT 语句，该语句指定要抽取的行。您可以使用非格式化事件表的列定义来帮助构造 SELECT 语句。

写入非格式化事件表的事件数据不会被自动清除。您必须以手动方式从该表中清除数据。当您想要清除特定的一组记录时，非格式化事件表的列定义非常有用。另一个选项是，使用 TRUNCATE TABLE 命令来除去所有表行。

在 CREATE EVENT MONITOR 语句中，可以指定相关联非格式化事件表的名称。如果未指定此名称，那么缺省名称将是事件监视器的名称。SYSCAT.EVENTTABLES 目录视图列示了事件监视器、与其相关联的非格式化表以及其他详细信息。

下表描述非格式化事件表中的列。键列是 event_data 列。其他列表示可用于查找您感兴趣的事件的标识。要了解表列的其他属性，请发出 DESCRIBE 语句。

表 2. 非格式化事件表的列定义

列名	列数据类型	列描述
appl_id	VARCHAR	发生此事件的应用程序的标识。空值表示无法获得应用程序标识。
appl_name	VARCHAR	发生此事件的应用程序的名称。空值表示无法获得应用程序名称。
client_acctng	VARCHAR	此事件的 CLIENT_ACCTNG 专用寄存器的当前值。空值表示无法获得客户机记帐信息。
client_applname	VARCHAR	此事件的 CLIENT_APPLNAME 专用寄存器的当前值。空值表示无法获得客户机应用程序名称。

表 2. 非格式化事件表的列定义 (续)

列名	列数据类型	列描述
client_userid	VARCHAR	此事件的 CLIENT_USERID 专用寄存器的当前值。空值表示无法获得客户机用户标识。
client_wrkstnname	VARCHAR	此事件的 CLIENT_WRKSTNNAME 专用寄存器的当前值。空值表示无法获得客户机工作站名称。
event_correlation_id	BIT DATA	可选的事件相关标识。空值表示无法获得事件相关标识。 此值基于事件监视器类型: • LOCKING - 保留供将来使用 • UOW - 保留供将来使用
event_data	BLOB	事件监视器所捕获的事件的完整事件记录数据，以原始二进制格式存储。
event_id	INTEGER	（对于锁定事件监视器记录）在数据库中唯一的事件标识。此标识将在数据库激活时复位。唯一性由 **event_timestamp**、**event_id**、**member** 与 **event_type** 的组合保证。 （对于 UOW 事件监视器记录）在每个连接中唯一的 UOW 标识的别名。唯一性由 **event_timestamp**、**event_id**、**member**、**event_type** 与 **appl_id** 的组合保证。
event_timestamp	TIMESTAMP	事件监视器生成此事件时的时间戳记。所有子记录都将共享父记录的时间戳记。
event_type	VARCHAR	在所检测的成员上发生的事件类型。
member	SMALLINT	发生此事件的成员。
partitioning_key	INTEGER	表的分区键，用于使插入操作在运行事件监视器的数据库分区中以局部方式执行。
record_seq_num	INTEGER	在 event_data 列中存储的记录的序号。
record_type	INTEGER	在 event_data 列中存储的记录的类型。

表 2. 非格式化事件表的列定义 (续)

列名	列数据类型	列描述
service_subclass_name	VARCHAR	发生此事件的服务子类的名称。空值表示无法获得服务子类名。
service_superclass_name	VARCHAR	发生此事件的服务超类的名称。空值表示无法获得服务超类名。
workload_name	VARCHAR	发生此事件的工作负载的名称。空值表示无法获得工作负载名称。

用于读取事件监视器数据的 db2evmonfmt 工具

根据使用非格式化事件表的事件监视器所生成的数据，基于 Java™ 的通用 XML 解析器工具 db2evmonfmt 将生成可读的纯文本输出（文本版本）或格式化 XML 输出。根据您指定的参数，db2evmonfmt 工具将确定事件监视器数据的解析方式以及所要创建的输出类型。

db2evmonfmt 工具以 Java 源代码形式提供。在使用此工具之前，您必须通过执行下列步骤来设置和编译此工具:

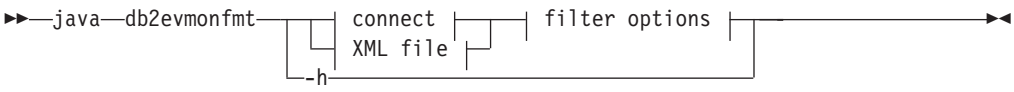
1. 在 sqllib/samples/java/jdbc 目录中找到源代码
2. 按照 Java 源文件中嵌入的指示信息来设置和编译此工具

您可以随意修改源代码以更改输出。

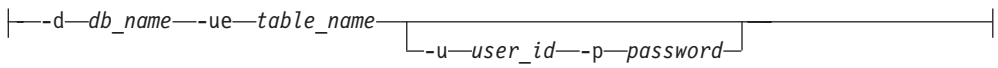
此工具使用 XSLT 样式表将事件数据变换为格式化文本。您不需要理解这些样式表。此工具将根据事件监视器类型自动装入正确的样式表并变换事件数据。每个事件监视器都将在 sqllib/samples/xml/data 目录中提供缺省样式表。此工具还提供了下列过滤选项:

- 事件标识
- 事件时间戳记
- 事件类型
- 工作负载名
- 服务类名
- 应用程序名

工具语法

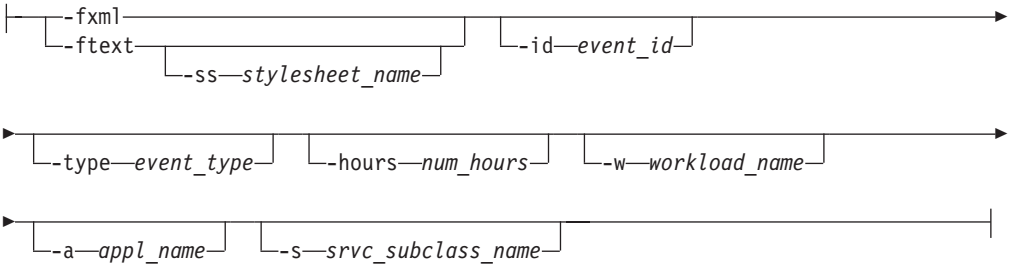


connect:



XML file:

filter options:



工具参数

java
要成功地运行基于 Java 的 db2evmonfmt 工具，必须在工具名前面指定 java 关键字。在 DB2 产品的安装期间，将从 sqllib/java/jdk64 目录中安装成功地运行此工具所需的正确 Java 版本。

-d *db_name*
指定所要连接的数据库的名称。

-ue *table_name*
指定非格式化事件表的名称。

-u *user_id*
指定用户标识。

-p *password*
指定密码。

-f *xml_filename*
指定要格式化的输入 XML 文件的名称。

-fxml
生成格式化 XML 文档（输出到标准输出）。

-ftext
将 XML 文档格式化为文本文档（输出到标准输出）。

-ss *stylesheet_name*
指定用于变换 XML 文档的 XSLT 样式表。

-id *event_id*
显示所有与指定事件标识匹配的事件。

-type *event_type*
显示所有与指定事件类型匹配的事件。

-hours *num_hours*
显示所有在过去指定数目的小时内发生的事件。

-w *workload_name*
显示所有属于所指定工作负载的事件。

-a *appl_name*
显示所有属于所指定应用程序的事件。

-s *srvc_subclass_name*
显示所有属于所指定服务子类的事件。

XSLT 样式表

DB2 数据库管理器提供了缺省的 XSLT 样式表（参见表 1），这些样式表在 sqllib/samples/java/jdbc 目录中。您可以对这些样式表进行更改，以便生成期望的输出。

表 3. 事件监视器的缺省 XSLT 样式表

事件监视器	缺省 XSLT 样式表
锁定	DB2EvmonLocking.xsl
工作单元	DB2EvmonUOW.xsl

您可以创建自己的 XSLT 样式表以变换 XML 文档。可以使用 -ss stylesheet_name 选项将这些样式表传递给基于 Java 的工具。

示例

示例 1

要从数据库 SAMPLE 中的程序包高速缓存非格式化事件表 PKG 中获取过去 32 小时内发生的所有事件的格式化文本输出，请发出以下命令:

```
java db2evmonfmt -d sample -ue pkg -ftext -hours 32
```

示例 2

要从数据库 SAMPLE 中的非格式化事件表 LOCK 中获取过去 24 小时内发生的所有 LOCKTIMEOUT 类型事件的格式化文本输出，请发出以下命令:

```
java db2evmonfmt -d sample -ue LOCK -ftext -hours 24 –type locktimeout
```

示例 3

要从 XML 源文件 LOCK.XML 中获取格式化文本输出，以便抽取过去 5 小时内发生的所有与事件类型 LOCKWAIT 匹配的事件，请发出以下命令:

```
java db2evmonfmt -f lock.xml -ftext -type lockwait -hours 5
```

示例 4

要使用已创建的 XSLT 样式表 SUMMARY.XSL 从数据库 SAMPLE 中的非格式化事件表 UOW 中获取所有事件的格式化文本输出，请发出以下命令:

```
java db2evmonfmt -d sample -ue uow -ftext -ss summary.xsl
```

格式化纯文本输出样本

以下格式化纯文本输出样本是使用锁定事件监视器 XSLT 样式表生成的:

```
------------------------------------------------
Event Entry     : 0
Event ID        : 1
Event Type      : Locktimeout
Event Timestamp : 2008-05-23-12.00.14.132329000
------------------------------------------------
```

```
Lock Details
------------
Lock Name       : 02000401000000000000000054
Lock Type       : Table
Lock Attributes : 00000000
Lock Count      : 1
Lock Hold Count : 0
Lock rrIID      : 0
Lock Status     : Waiting
Cursor Bitmap   : 00000000
Tablespace Name : USERSPACE1
Table Name      : NEWTON  .SARAH

Attributes           Requestor                      Holder
----------           ---------                      ------
Application Handle   [0-35]                         [0-16]
Application ID       *LOCAL.horton.080523160016    *LOCAL.horton.080523155938
Application Name     xaplus0001                     db2bp
Authentication ID    NEWTON                         HORTON
Requesting Agent     65                             21
Coordinating Agent   65                             21
Application Status   SQLM_CONNECTPEND               SQLM_CONNECTPEND
Lock Timeout         5000                           0
Workload Name        XAPLUS0010_WL02                SYSDEFAULTUSERWORKLOAD
Service Subclass     XAPLUS0010_SC02                SYSDEFAULTSUBCLASS
Current Request      Execute                        Execute Immediate
Lock Mode            Intent Exclusive               Exclusive
tpmon Userid
tpmon Wkstn
tpmon App
tpmon Accstring

Lock Requestor Current Activities
---------------------------------
Activity ID     : 2
Uow ID          : 1
Package ID      : 65426E4D4B584659
Package SectNo  : 3
Package Name    : NEWTON
Package Schema  : AKINTERF
Package Version :
Reopt           : always
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Insert/Update/Delete
Stmt Text       : INSERT INTO SARAH VALUES(:H00008, :H00013, :H00014)


Lock Requestor Past Activities
------------------------------
Activity ID     : 1
Uow ID          : 1
Package ID      : 65426E4D4B584659
Package SectNo  : 2
Package Name    : NEWTON
Package Schema  : AKINTERF
Package Version :
Reopt           : always
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
```

```
Stmt Flag       : Dynamic
Stmt Type       : DML, Insert/Update/Delete
Stmt Text       : INSERT INTO NADIA VALUES(:H00007)


Lock Holder Current Activities
------------------------------

Lock Holder Past Activities
------------------------------
Activity ID     : 1
Uow ID          : 2
Package ID      : 41414141414E4758
Package SectNo  : 201
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt           : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Select (blockable)
Stmt Text       : select * from newton.sarah

Activity ID     : 2
Uow ID          : 2
Package ID      : 41414141414E4758
Package SectNo  : 203
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt           : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Lock Table
Stmt Text       : lock table newton.sarah in exclusive mode


-------------------------------------------------
Event Entry     : 1
Event ID        : 2
Event Type      : Locktimeout
Event Timestamp : 2008-05-23-12.04.42.144896000
-------------------------------------------------

...
...
...
```

用法说明

db2evmonfmt 实用程序是基于 Java 的工具，要成功地运行此工具，必须在它前面指定 java 关键字。所需的 Java 版本是从 sqllib/java/jdk64 目录中随 DB2 产品一起安装的版本。

注: 您还可以使用 EVMON_FORMAT_UE_TO_XML 表函数将非格式化事件表 BLOB 列中包含的二进制事件格式化为 XML 文档。

监视数据库锁定

在大型 DB2 环境中，诊断和排除锁定争用情况的工作可能相当复杂且耗时。锁定事件监视器和其他工具通过收集锁定数据来帮助您简化此任务。

简介

锁定事件监视器用于在发生锁定事件时捕获关于那些事件的描述性信息。捕获的信息将标识锁定事件引起的锁定争用情况所涉及的关键应用程序。将同时捕获关于锁定请求者（接收到死锁或锁定超时错误或者等待锁定时的耗用时间超出指定时间长度的应用程序）和当前锁定所有者的信息。

锁定事件监视器所收集的信息将以二进制格式写入数据库中的非格式化事件表。在捕获后执行的一个步骤将对捕获到的数据进行处理，以提高捕获过程的效率。

您还可以使用动态或静态 SQL 来直接访问 DB2 关系监视接口（表函数），以便收集锁定事件信息。

确定是否已发生死锁或锁定超时的过程也有所简化。消息将在其中任何一个事件发生时被写入管理通知日志；这将对返回给应用程序的 SQL0911N（sqlcode 为 -911）错误进行补充。此外，还会将锁定升级通知写入管理通知日志；此信息对于您调整锁定表大小以及应用程序能够使用的锁定表空间量而言十分有用。另外，还可以检查有关锁定超时（**lock_timeouts**）、锁定等待（**lock_waits**）和死锁（**deadlocks**）的计数器。

可以捕获其锁定数据的活动的类型包括:

- SQL 语句，例如:
  - DML
  - DDL
  - CALL
- LOAD 命令
- REORG 命令
- BACKUP DATABASE 命令
- 实用程序请求

锁定事件监视器将不推荐使用的死锁事件监视器（CREATE EVENT MONITOR FOR DEADLOCKS 语句和 DB2DETAILDEADLOCK）以及不推荐使用的锁定超时报告功能（DB2_CAPTURE_LOCKTIMEOUT 注册表变量）替换为简化而一致的锁定事件数据收集接口，并添加了捕获锁定等待数据的功能。

功能概述

要使用锁定事件监视器来捕获锁定事件数据，需要执行两个步骤:

1. 必须使用 CREATE EVENT MONITOR FOR LOCKING 语句来创建 LOCK EVENT 监视器。您需要提供监视器名称以及一个非格式化事件表的名称，锁定事件数据将写入该表。
2. 必须使用下列其中一种方法来指定锁定事件数据的捕获级别:
   - 您可以通过更改现有工作负载或者使用 CREATE 或 ALTER WORKLOAD 语句创建新工作负载来指定特定的工作负载。在工作负载级别，您必须指定要捕获的

锁定事件数据的类型（死锁、锁定超时或锁定等待），并指定锁定是否涉及应用程序的 SQL 语句历史记录和输入值。对于锁定等待，还必须指定应用程序将等待锁定的时间长度（在这段时间过后，将捕获锁定等待数据）。

- 通过设置适当的数据库配置参数，可以在数据库级别收集数据并影响所有 DB2 工作负载:

  mon_lockwait

  此参数控制锁定等待事件的生成方式。

  最佳实践是，在工作负载级别启用锁定等待数据收集功能。

  mon_timeout

  此参数控制锁定超时事件的生成方式。

  最佳实践是，如果这些事件并不是应用程序所预期的事件，请在数据库级别启用锁定超时数据收集功能。否则，请在工作负载级别启用此功能。

  mon_deadlock

  此参数控制死锁事件的生成方式。

  最佳实践是，在数据库级别启用死锁数据收集功能。

  mon_lw_thresh

  此参数控制在生成 **mon_lockwait** 的事件之前等待锁定时耗用的时间。

捕获 SQL 语句历史记录和输入值将产生附加的开销，但要成功地调试锁定问题，通常需要此级别的详细信息。

发生锁定事件之后，通过使用所提供的基于 Java 的应用程序 db2evmonfmt，可以将非格式化事件表中的二进制数据变换为 XML 或文本文档。另外，可以使用 EVMON_FORMAT_UE_TO_XML 表函数将非格式化事件表 BLOB 列中的二进制事件数据格式化为 XML 报告文档，也可以使用 EVMON_FORMAT_UE_TO_TABLES 过程将其格式化为关系表。

为了帮助确定应该对哪些工作负载监视锁定事件，您可以查看管理通知日志。每当遇到死锁或锁定超时情况时，系统都会将一条消息写入日志。这些消息将标识正在其中运行锁定请求者和锁定所有者的工作负载以及锁定事件的类型。另外，还可以在工作负载级别检查有关锁定超时（**lock_timeouts**）、锁定等待（**lock_waits**）和死锁（**deadlocks**）的计数器。

为锁定事件收集的信息

锁定事件监视器所收集的一些锁定事件信息包括:

- 导致发生事件的锁定
- 正在挂起导致发生锁定事件的锁定的应用程序
- 正在等待或请求导致发生锁定事件的锁定的应用程序
- 应用程序在锁定事件期间执行的操作

局限性

- 写入非格式化事件表的锁定事件数据不会被自动清除。您必须定期从该表中清除数据。

- 只能将收集到的事件监视器数据输出到非格式化事件表。不支持输出到文件、管道和表。
- 对于每个数据库，建议您只创建一个锁定事件监视器。每个附加的事件监视器都只创建同一数据的副本。

不推荐使用的锁定监视功能

缺省情况下，将为每个数据库创建不推荐使用的详细死锁事件监视器 DB2DETAILDEADLOCK，并在激活数据库时启动该监视器。您必须禁用 DB2DETAILDEADLOCK 事件监视器并将其除去，否则不推荐使用的事件监视器和新事件监视器都将收集数据，这将显著影响性能。

要除去 DB2DETAILDEADLOCK 事件监视器，请发出下列 SQL 语句:

```
SET EVENT MONITOR DB2DETAILDEADLOCK state 0
DROP EVENT MONITOR DB2DETAILDEADLOCK
```

收集锁定事件数据并生成报告

您可以使用锁定事件监视器来收集锁定超时、锁定等待和死锁信息，以帮助确定和解决锁定问题。本任务描述以不可读格式在非格式化事件表中收集锁定事件数据之后，如何获取可读的文本报告。

开始前

要创建锁定事件监视器并收集锁定事件监视器数据，您必须具有 DBADM 或 SQLADM 权限。

关于此任务

锁定事件监视器用于收集有助于确定和解决锁定问题的相关信息。例如，锁定事件监视器收集的一些锁定事件信息如下:

- 导致锁定事件的锁定
- 正在请求或挂起导致锁定事件的锁定的应用程序
- 应用程序在锁定事件期间执行的操作

本任务提供有关收集指定工作负载的锁定事件数据的指示信息。在下列情况下，您可能想收集锁定事件数据:

- 使用 MON_GET_WORKLOAD 表函数时，您注意到锁定等待值超出正常情况。
- 应用程序返回 SQL 返回码 -911，原因码为 68，这表明“事务由于锁定超时而回滚”。
- 您在管理通知日志中找到死锁事件消息。此日志消息指出在两个应用程序之间发生锁定事件，例如在应用程序 A 与 B 之间发生此事件，其中 A 是工作负载 FINANCE 的组成部分，B 是工作负载 PAYROLL 的组成部分。

限制

要查看数据值，您必须对 EVMON_FORMAT_UE_* 例程具有 EXECUTE 特权（SQLADM 和 DBADM 权限隐式地包含此特权）。您还必须对非格式化事件表具有 SELECT 特权，缺省情况下，具有 DATAACCESS 权限的用户以及事件监视器及相关非格式化事件表的创建者具有此特权。

过程

要收集关于将来有可能发生的锁定事件的详细信息，请执行以下步骤:

1. 使用 CREATE EVENT MONITOR FOR LOCKING 语句来创建名为 lockevmon 的锁定事件监视器，如以下示例所示:

```
CREATE EVENT MONITOR lockevmon FOR LOCKING
   WRITE TO UNFORMATTED EVENT TABLE
```

   注: 以下列示创建事件监视器时要记住的要点:

   - 您可以提前创建事件监视器而不必担心耗用磁盘空间，这是因为，在数据库或工作负载级别激活数据收集操作之前不会写入任何内容
   - 在分区数据库环境中，请确保在分区表空间的所有节点上设置事件监视器。否则，在没有分区表空间的分区中，将丢失锁定事件。
   - 请确保设置表空间和缓冲池，以便最大程度地降低对表进行访问以获取数据期间执行的工作对高性能工作的影响。

2. 要在工作负载级别启用锁定事件数据收集功能，请发出 ALTER WORKLOAD 语句并指定下列其中一个 COLLECT 子句: COLLECT LOCK TIMEOUT DATA、COLLECT DEADLOCK DATA 或 COLLECT LOCK WAIT DATA。请在 COLLECT 子句中指定 WITH HISTORY 选项。设置数据库配置参数将在数据库级别影响锁定事件数据收集，并且所有工作负载都将受影响。

   对于锁定等待事件

   要对 FINANCE 应用程序收集任何超过 5 秒钟才能获取的锁定的锁定等待数据，并且要对 PAYROLL 应用程序收集任何超过 10 秒才能获取的锁定的锁定等待数据，请发出下列语句:

```
ALTER WORKLOAD finance COLLECT LOCK WAIT DATA WITH HISTORY AND VALUES
   FOR LOCKS WAITING MORE THAN 5 SECONDS
ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA
   FOR LOCKS WAITING MORE THAN 10 SECONDS WITH HISTORY
```

   要使用 HIST_AND_VALUES 输入数据值对 SAMPLE 数据库设置 **mon_lockwait** 数据库配置参数，并且要将 **mon_lw_thresh** 数据库配置参数设置为 10 秒，请发出下列命令:

```
db2 update db cfg for sample using mon_lockwait hist_and_values
db2 update db cfg for sample using mon_lw_thresh 10000000
```

   对于锁定超时事件

   要对 FINANCE 和 PAYROLL 应用程序收集锁定超时数据，请发出下列语句:

```
ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA WITH HISTORY
```

   要使用 HIST_AND_VALUES 输入数据值对 SAMPLE 数据库设置 **mon_locktimeout** 数据库配置参数，请发出以下命令:

```
db2 update db cfg for sample using mon_locktimeout hist_and_values
```

   对于死锁事件

   要对 FINANCE 和 PAYROLL 应用程序收集数据，请发出下列语句:

```
ALTER WORKLOAD finance COLLECT DEADLOCK DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT DEADLOCK DATA WITH HISTORY
```

   要使用 HIST_AND_VALUES 输入数据值对 SAMPLE 数据库设置 **mon_deadlock** 数据库配置参数，请发出以下命令:

```
db2 update db cfg for sample using mon_deadlock hist_and_values
```

3. 重新运行工作负载，以便接收另一个锁定事件通知。
4. 连接到数据库。
5. 使用下列其中一个过程来获取锁定事件报告:
   a. 使用 XML 解析器工具 db2evmonfmt 来生成基于非格式化事件表中收集的事件数据并使用缺省样式表的纯文本报告，例如:

      ```
      java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
      ```

   b. 使用 EVMON_FORMAT_UE_TO_XML 表函数来获取 XML 文档。
   c. 使用 EVMON_FORMAT_UE_TO_TABLES 过程将该数据输出到关系表。
6. 分析报告，确定锁定事件问题的原因并解决该问题。
7. 通过运行下列语句或复位数据库配置参数，对 FINANCE 和 PAYROLL 应用程序关闭锁定数据收集功能:

   对于锁定等待事件

   ```
   ALTER WORKLOAD finance COLLECT LOCK WAIT DATA NONE
   ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA NONE
   ```

   要使用缺省的 NONE 输入数据值对 SAMPLE 数据库复位 **mon_lockwait** 数据库配置参数，并且要将 **mon_lw_thresh** 数据库配置参数复位为缺省值（5 秒），请发出下列命令:

   ```
   db2 update db cfg for sample using mon_lockwait none
   db2 update db cfg for sample using mon_lw_thresh 5000000
   ```

   对于锁定超时事件

   ```
   ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA NONE
   ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA NONE
   ```

   要使用缺省的 NONE 输入数据值对 SAMPLE 数据库复位 **mon_locktimeout** 数据库配置参数，请发出以下命令:

   ```
   db2 update db cfg for sample using mon_locktimeout none
   ```

   对于死锁事件

   ```
   ALTER WORKLOAD finance COLLECT DEADLOCK DATA NONE
   ALTER WORKLOAD payroll COLLECT DEADLOCK DATA NONE
   ```

   要使用缺省的 WITHOUT_HIST 输入数据值对 SAMPLE 数据库复位 **mon_deadlock** 数据库配置参数，请发出以下命令:

   ```
   db2 update db cfg for sample using mon_deadlock without_hist
   ```

下一步任务

请重新运行应用程序，以确保锁定问题已不再存在。

为锁定事件监视器写入 XML 的信息

EVMON_FORMAT_UE_TO_XML 表函数为锁定事件监视器写入的信息。sqllib/misc/DB2EvmonLocking.xsd 文件也对此信息作了记录。

db2_lock_event

用于详细描述锁定超时、锁定等待或死锁事件的主要模式。

表 4. db2_lock_event

名称	数据类型	最小出现次数	最大出现次数	描述
db2_lock_event	db2lockevent	1	1	用于详细描述锁定超时、锁定等待或死锁事件的主要模式。

db2lockevent

此模式描述事件监视器所捕获的每个锁定事件的结构。

表 5. db2lockevent 属性

名称	数据类型	用法	描述
id	xs:positiveInteger	必需	用于表示事件标识的整数。
type	xs:string	必需	所发生的锁定事件的类型。锁定事件可以具有下列类型：Locktimeout、Deadlock 或 Lockwait。
timestamp	db2_timestamp_type	必需	用于表示锁定事件发生时间的时间戳记。
member	member_type	必需	锁定事件发生所在的成员。
release	xs:nonNegativeInteger	必需	表示捕获此事件的 DB2 产品的级别。

对于正常事件，将返回下列元素：

表 6. 针对正常事件返回的 db2lockevent 元素

名称	数据类型	最小出现次数	最大出现次数	描述
db2_deadlock_graph	db2dgraph	0	1	此模式元素表示 DB2 死锁图。此图概述死锁涉及的所有参与者。
db2_participant	db2appinfo	1	不受限	此模式元素表示锁定事件所涉及的所有参与者的应用程序信息。

如果发生错误情况，那么将返回下列元素，这些元素将提供消息详细信息：

表 7. 针对错误条件返回的 db2lockevent 元素

名称	数据类型	最小出现次数	最大出现次数	描述
db2_message	xs:string	1	1	错误消息
db2_event_file	xs:string	1	1	事件已被写入的文件的标准路径。

db2appdetails

此模式表示关于参与者的详细信息。

表 8. db2appdetails 元素

名称	数据类型	最小出现次数	最大出现次数	描述
application_handle	application_handle_type	1	1	应用程序在系统范围内的唯一标识。有关更多详细信息，请参阅监视元素 agent_id。
appl_id	appl_id_type	1	1	当应用程序连接到数据库管理器上的数据库时，将生成此标识。有关更多详细信息，请参阅监视元素 appl_id。
appl_name	xs:string	1	1	客户机上运行的应用程序在数据库中的名称。有关更多详细信息，请参阅监视元素 appl_name。
auth_id	authid_type	1	1	调用受监视应用程序的用户的授权标识。有关更多详细信息，请参阅监视元素 auth_id。
agent_tid	xs:nonNegativeInteger	1	1	代理程序的引擎可分派单元（EDU）的唯一标识。有关更多详细信息，请参阅监视元素 agent_pid。
coord_agent_tid	xs:nonNegativeInteger	1	1	应用程序的协调代理程序的引擎可分派单元（EDU）标识。有关更多详细信息，请参阅监视元素 coord_agent_pid。
appl_status	.	1	1	应用程序的当前状态。有关更多详细信息，请参阅监视元素 appl_status。
appl_action	.	1	1	客户机应用程序正在执行的操作/请求。
lock_timeout_val	xs:integer	1	1	数据库配置参数“锁定超时”。值以秒计。有关更多详细信息，请参阅监视元素 lock_timeout_val。
lock_wait_val	xs:integer	1	1	在锁定事件发生期间起作用的锁定等待参数。这是数据库配置参数 MON_LW_THRESH 或者在工作负载级别指定的 COLLECT LOCK WAIT DATA 设置。值以毫秒计。
tentry_state	.	1	1	TEntry 状态。仅供内部使用。
tentry_flag1	xs:hexBinary	1	1	TEntry 标志 1。仅供内部使用。

表 8. db2appdetails 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
tentry_flag2	xs:hexBinary	1	1	TEntry 标志 2。仅供内部使用。
xid	xs:hexBinary	1	1	XID - 全局事务标识。
workload_id	db_object_id_type	1	1	此应用程序所属的工作负载的标识。有关更多详细信息，请参阅监视元素 workload_id。
workload_name	db_object_name_type	1	1	此应用程序所属的工作负载的名称。有关更多详细信息，请参阅监视元素 workload_name。
service_class_id	db_object_id_type	1	1	此应用程序所属的服务子类的标识。有关更多详细信息，请参阅监视元素 service_class_id。
service_subclass_name	db_object_name_type	1	1	此应用程序所属的服务子类的名称。有关更多详细信息，请参阅监视元素 service_subclass_name。
current_request	xs:string	1	1	当前正在处理或最近处理的操作。
lock_escalation	xs:string	1	1	指示锁定请求是否作为锁定升级的组成部分。有关更多详细信息，请参阅监视元素 lock_escalation。可能的值包括: Yes 或 No。
past_activities_wrapped	xs:string	1	1	指示活动列表是否已回绕。对任何一个应用程序所保留的先前活动数的缺省限制是 250。您可以使用注册表变量 DB2_MAX_INACT_STMTS 来更改此缺省值。用户可能希望选择另一个限制值，以便增加或减少用于不活动语句信息的系统监视器堆空间量。
client_userid	tpmon_type	1	1	由事务管理器生成并提供给服务器的客户机用户标识。有关更多详细信息，请参阅监视元素 client_userid。
client_wrkstnname	tpmon_type	1	1	如果在此连接中发出了 sqleseti API，那么此元素标识客户机系统或工作站。有关更多详细信息，请参阅监视元素 client_wrkstnname。

表 8. db2appdetails 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
client_applname	tpmon_type	1	1	如果在此连接中发出了 sqleseti API，那么此元素标识执行事务的服务器事务程序。有关更多详细信息，请参阅监视元素 client_applname。
client_acctng	tpmon_type	1	1	如果在此连接中发出了 sqleseti API，那么此元素是为了进行日志记录和诊断而传递到目标数据库的数据。有关更多详细信息，请参阅监视元素 client_acctng。

db2appinfo

此模式描述正在请求锁定或正在挂起锁定的应用程序的结构。

表 9. db2appinfo 属性

名称	数据类型	用法	描述
no	xs:positiveInteger	必需	用于在死锁事件中唯一地标识此参与者的序号。
type	xs:string	必需	参与者类型。参与者可以是请求者或所有者。
participant_no_holding_lk	xs:positiveInteger	可选	用于标识正在挂起锁定的应用程序的参与者编号。

表 10. db2appinfo 元素

名称	数据类型	最小出现次数	最大出现次数	描述
db2_object_requested	db2objectrequested	0	1	此模式元素表示请求者正在尝试获取但被所有者挂起的 DB2 锁定。
db2_app_details	db2appdetails	1	1	此模式元素表示关于此参与者的详细信息。
db2_activity	db2activity	0	不受限	应用程序当前正在执行或已执行的所有 DB2 活动的列表。

db2objectrequested

表 11. db2objectrequested 属性

名称	数据类型	用法	描述
type	xs:string	必需	正在请求的对象的类型。可能的值包括: lock 或 ticket。

如果属性 db2objectrequested/type 的值为 lock，那么将返回下列元素:

表 12. 针对锁定返回的 db2objectrequested 元素

名称	数据类型	最小出现次数	最大出现次数	描述
lock_name	xs:hexBinary	1	1	内部二进制锁定名称。此元素将充当锁定的唯一标识。有关更多详细信息，请参阅监视元素 lock_name。
lock_object_type	.	1	1	应用程序正在等待锁定的对象的类型。有关更多详细信息，请参阅监视元素 lock_object_type。
lock_specifics	xs:string	1	1	关于锁定的内部具体信息。仅供参考。
lock_attributes	xs:hexBinary	1	1	锁定属性。有关更多详细信息，请参阅监视元素 lock_attributes。
lock_current_mode	.	1	1	在转换前的原始锁定。有关更多详细信息，请参阅监视元素 lock_current_mode。
lock_mode_requested	.	1	1	此参与者所请求的锁定方式。有关更多详细信息，请参阅监视元素 lock_mode_requested。
lock_mode	.	1	1	正在挂起的锁定的类型。有关更多详细信息，请参阅监视元素 lock_mode。
lock_count	xs:integer	1	1	正在挂起的锁定上的锁定数目。有关更多详细信息，请参阅监视元素 lock_count。
lock_hold_count	xs:integer	1	1	挂起锁定的次数。有关更多详细信息，请参阅监视元素 lock_hold_count。
lock_rriid	xs:integer	1	1	行锁定的 IID。仅供内部使用。
lock_status	.	1	1	指示锁定的内部状态。有关更多详细信息，请参阅监视元素 lock_status。
lock_release_flags	xs:hexBinary	1	1	锁定释放标志。有关更多详细信息，请参阅监视元素 lock_release_flags。
tablespace_name	.	1	1	在其中挂起锁定的表空间的名称。有关更多详细信息，请参阅监视元素 tablespace_name。
table_name	.	1	1	在其中挂起锁定的表的名称。有关更多详细信息，请参阅监视元素 table_name。

表 12. 针对锁定返回的 db2objectrequested 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
table_schema	db_object_name_type	1	1	表的模式。有关更多详细信息，请参阅监视元素 table_schema。

如果属性 db2objectrequested/type 的值为 ticket，那么将返回下列元素:

表 13. 针对凭单返回的 db2objectrequested 元素

名称	数据类型	最小出现次数	最大出现次数	描述
threshold_name	db_object_name_type	1	1	阈值队列的名称。
threshold_id	db_object_id_type	1	1	阈值队列的标识。
queued_agents	xs:nonNegativeInteger	1	1	当前正在阈值队列中进行排队的代理程序的总数。

db2dgraph

此模式描述死锁图的结构以及死锁循环所涉及的参与者。

表 14. db2dgraph 属性

名称	数据类型	用法	描述
dl_conns	xs:nonNegativeInteger	必需	死锁所涉及的参与者的总数。有关更多详细信息，请参阅监视元素 dl_conns。
rolled_back_participant_no	xs:nonNegativeInteger	必需	用于标识已回滚的应用程序的参与者编号。有关更多详细信息，请参阅监视元素 rolled_back_participant_no。
type	xs:string	必需	已发生的死锁的类型：局部或全局。对于全局死锁而言，至少一个或多个参与者驻留在另一个成员中。对于局部死锁而言，所有参与者都驻留在同一个成员中。

表 15. db2dgraph 元素

名称	数据类型	最小出现次数	最大出现次数	描述
db2_participant	db2dstack	1	不受限	此模式元素表示死锁图中的单一堆栈条目。

db2dstack

此模式描述死锁图中条目的结构。该条目包含关于死锁循环中正在请求和挂起锁定的参与者的信息。

表 16. db2dstack 属性

名称	数据类型	用法	描述
no	xs:positiveInteger	必需	用于标识正在请求锁定的应用程序的参与者编号。
deadlock_member	member_type	必需	参与者正在其中请求锁定的成员。
participant_no_holding_lk	xs:positiveInteger	必需	用于标识正在挂起锁定的应用程序的参与者编号。
application_handle	application_handle_type	必需	应用程序在系统范围内的唯一标识。有关更多详细信息，请参阅监视元素 agent_id。

db2actdetails

此模式描述关于活动的详细信息。

表 17. db2actdetails 元素

名称	数据类型	最小出现次数	最大出现次数	描述
activity_id	xs:positiveInteger	1	1	用于唯一地标识给定工作单元中某个应用程序的活动的计数器。有关更多详细信息，请参阅监视元素 activity_id。
uow_id	xs:positiveInteger	1	1	此活动记录所应用于的工作单元标识。有关更多详细信息，请参阅监视元素 uow_id。
package_name	xs:string	1	1	当前正在执行的 SQL 语句所在程序包的名称。有关更多详细信息，请参阅监视元素 package_name。
package_schema	xs:string	1	1	预编译应用程序的用户的授权标识。有关更多详细信息，请参阅监视元素 package_schema。
package_version_id	xs:string	1	1	程序包版本指定当前正在执行的 SQL 语句所在程序包的版本标识。有关更多详细信息，请参阅监视元素 package_version_id。
consistency_token	xs:string	1	1	程序包一致性标记帮助标识当前正在执行的 SQL 语句所在程序包的版本。有关更多详细信息，请参阅监视元素 consistency_token。

表 17. db2actdetails 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
section_number	xs:nonNegativeInteger	1	1	程序包中用于当前正在处理或最新处理的 SQL 语句的内部节号。有关更多详细信息，请参阅监视元素 section_number。
reopt	xs:string	1	1	用于预编译此程序包的 REOPT 绑定选项。可能的值包括：NONE、ONCE 和 ALWAYS。有关更多详细信息，请参阅 REOPT 绑定选项。
incremental_bind	xs:string	1	1	此程序包在执行时以增量方式进行绑定。可能的值包括：Yes 或 No。
effective_isolation	.	1	1	运行 SQL 语句时作用于该语句的隔离值。有关更多详细信息，请参阅监视元素 effective_isolation。
effective_query_degree	xs:nonNegativeInteger	1	1	运行 SQL 语句时作用于该语句的程度值。有关更多详细信息，请参阅监视元素 effective_query_degree。
stmt_unicode	xs:string	1	1	SQL 语句 Unicode 标志。可能的值包括：Yes 或 No。
stmt_lock_timeout	xs:integer	1	1	运行 SQL 语句时作用于该语句的锁定超时值。有关更多详细信息，请参阅监视元素 stmt_lock_timeout。
stmt_type	xs:string	1	1	所处理的 SQL 语句的类型。可能的值包括：Dynamic 或 Static。有关更多详细信息，请参阅监视元素 stmt_type。
stmt_operation	xs:string	1	1	SQL 语句操作类型。有关更多详细信息，请参阅监视元素 stmt_operation。
stmt_query_id	xs:nonNegativeInteger	1	1	对任何 SQL 语句指定的内部查询标识。有关更多详细信息，请参阅监视元素 stmt_query_id。
stmt_nest_level	xs:nonNegativeInteger	1	1	此元素包含运行语句时起作用的嵌套级别或递归级别。有关更多详细信息，请参阅监视元素 stmt_nest_level。

表 17. db2actdetails 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
stmt_invocation_id	xs:nonNegativeInteger	1	1	此元素包含运行 SQL 语句的例程调用的标识。有关更多详细信息，请参阅监视元素 stmt_invocation_id。
stmt_source_id	xs:nonNegativeInteger	1	1	此元素包含对已运行的 SQL 语句的源代码指定的内部标识。有关更多详细信息，请参阅监视元素 stmt_source_id。
stmt_pkgcache_id	xs:nonNegativeInteger	1	1	此元素包含动态 SQL 语句的内部程序包高速缓存标识。有关更多详细信息，请参阅监视元素 stmt_pkgcache_id。
stmt_text	xs:string	1	1	QL 语句的文本。有关更多详细信息，请参阅监视元素 stmt_text。

db2activity

此模式描述给定工作单元中某个应用程序的单一 DB2 活动的结构。

表 18. db2activity 属性

名称	数据类型	用法	描述
type	xs:string	必需	此属性表示类型或活动。可能的值包括 current 或 past。

表 19. db2activity 元素

名称	数据类型	最小出现次数	最大出现次数	描述
db2_activity_details	db2actdetails	1	1	此模式元素表示关于此活动的详细信息。
db2_input_variable	db2inputvar	0	不受限	此模式元素表示与 SQL 语句相关联的输入变量的列表。

db2inputvar

此模式描述单一输入变量的结构。

表 20. db2inputvar 元素

名称	数据类型	最小出现次数	最大出现次数	描述
stmt_value_index	xs:positiveInteger	1	1	此元素表示 SQL 语句中使用的输入参数标记或主变量的位置。有关更多详细信息，请参阅监视元素 stmt_value_index。

表 20. db2inputvar 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
stmt_value_isreopt	xs:string	1	1	此元素指示该变量在语句重新优化期间是否已被使用。有关更多详细信息，请参阅监视元素 stmt_value_isreopt。
stmt_value_isnull	xs:string	1	1	此元素指示与 SQL 语句相关联的数据值是否为空值。有关更多详细信息，请参阅监视元素 stmt_value_isnull。
stmt_value_type	xs:string	1	1	此元素包含与 SQL 语句相关联的数据值类型的字符串表示。有关更多详细信息，请参阅监视元素 stmt_value_type。
stmt_value_data	xs:string	1	1	此元素包含与 SQL 语句相关联的数据值的字符串表示。有关更多详细信息，请参阅监视元素 stmt_value_data。

为锁定事件监视器写入关系表的信息

本节概述使用 EVMON_FORMAT_UE_TO_TABLES 过程为锁定事件监视器从 XML 文档写入关系表的信息。sqllib/misc/DB2EvmonLocking.xsd 文件也对此信息作了记录。

为锁定事件监视器写入的信息

表 21. 为锁定事件监视器返回的信息: 表名: LOCK_EVENT

列名	数据类型	描述
XMLID	VARCHAR(1024) NOT NULL	
EVENT_ID	BIGINT NOT NULL	
EVENT_TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	
MEMBER	SMALLINT NOT NULL	member - 数据库成员
DL_CONNS	BIGINT	dl_conns - 死锁中涉及的连接数
ROLLED_BACK_PARTICIPANT_NO	BIGINT	rolled_back_participant_no - 回滚的应用程序参与者

表 22. 为锁定事件监视器返回的信息: 表名: LOCK_PARTICIPANTS

列名	数据类型	描述
XMLID	VARCHAR(1024) NOT NULL	
PARTICIPANT_NO	BIGINT	participant_no - 死锁参与者
PARTICIPANT_TYPE	VARCHAR(10)	
PARTICIPANT_NO_HOLDING_LK	BIGINT	participant_no_holding_lk - 对应用程序所需对象挂起锁定的参与者
APPLICATION_HANDLE	BIGINT	application_handle - 应用程序句柄
APPL_ID	VARCHAR(128)	appl_id - 应用程序标识

表 22. 为锁定事件监视器返回的信息：表名：LOCK_PARTICIPANTS（续）

列名	数据类型	描述
APPL_NAME	VARCHAR(128)	appl_name - 应用程序名称
AUTH_ID	VARCHAR(128)	auth_id - 授权标识
AGENT_TID	BIGINT	agent_pid - 引擎可分派单元（EDU）标识
COORD_AGENT_TID	BIGINT	coord_agent_pid - 协调代理程序标识
APPL_STATUS	BIGINT	appl_status - 应用程序状态
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - 锁定超时值
LOCK_WAIT_VAL	BIGINT	
WORKLOAD_ID	BIGINT	workload_id - 工作负载标识
WORKLOAD_NAME	VARCHAR(128)	workload_name - 工作负载名称
SERVICE_CLASS_ID	BIGINT	service_class_id - 服务类标识
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
CURRENT_REQUEST	VARCHAR(32)	
LOCK_ESCALATION	CHAR(3)	lock_escalation - 锁定升级
PAST_ACTIVITIES_WRAPPED	CHAR(3)	
CLIENT_USERID	VARCHAR(64)	
CLIENT_WRKSTNNAME	VARCHAR(128)	
CLIENT_APPLNAME	VARCHAR(128)	
CLIENT_ACCTNG	VARCHAR(200)	
OBJECT_REQUESTED	VARCHAR(10)	
LOCK_NAME	CHAR(26)	lock_name - 锁定名称
LOCK_OBJECT_TYPE	BIGINT	lock_object_type - 等待的锁定对象类型
LOCK_ATTRIBUTES	CHAR(8)	lock_attributes - 锁定属性
LOCK_CURRENT_MODE	BIGINT	lock_current_mode - 转换前的原始锁定方式
LOCK_MODE_REQUESTED	BIGINT	lock_mode_requested - 请求的锁定方式
LOCK_MODE	BIGINT	lock_mode - 锁定方式
LOCK_COUNT	BIGINT	lock_count - 锁定计数
LOCK_HOLD_COUNT	BIGINT	lock_hold_count - 锁定挂起计数
LOCK_RRIID	BIGINT	
LOCK_STATUS	VARCHAR(10)	lock_status - 锁定状态
LOCK_RELEASE_FLAGS	CHAR(8)	lock_release_flags - 锁定释放标志
TABLE_FILE_ID	BIGIN	table_file_id - 表文件标识
TABLE_NAME	VARCHAR(128)	table_name - 表名
TABLE_SCHEMA	VARCHAR(128)	table_schema - 表模式名
TABLESPACE_NAME	VARCHAR(128)	tablespace_name - 表空间名称
THRESHOLD_ID	BIGINT	
THRESHOLD_NAME	VARCHAR(128)	threshold_name - 阈值名称

表 23. 为锁定事件监视器返回的信息: 表名: LOCK_PARTICIPANT_ACTIVITIES

列名	数据类型	描述
XMLID	VARCHAR(1024) NOT NULL	
PARTICIPANT_NO	BIGINT	participant_no - 死锁参与者
ACTIVITY_ID	BIGINT	activity_id - 活动标识
ACTIVITY_TYPE	VARCHAR(10)	activity_type - 活动类型
UOW_ID	BIGINT	uow_id - 工作单元标识
PACKAGE_NAME	VARCHAR(128)	package_name - 程序包名
PACKAGE_SCHEMA	VARCHAR(128)	package_schema - 程序包模式
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - 程序包版本
CONSISTENCY_TOKEN	VARCHAR(8)	consistency_token - 程序包一致性标记
SECTION_NUMBER	BIGINT	section_number - 节号
REOPT	VARCHAR(10)	
INCREMENTAL_BIND	CHAR(3)	
EFFECTIVE_ISOLATION	BIGINT	effective_isolation - 有效隔离级别
EFFECTIVE_QUERY_DEGREE	BIGINT	effective_query_degree - 有效查询并行度
STMT_LOCK_TIMEOUT	BIGINT	stmt_lock_timeout - 语句锁定超时
STMT_TYPE	VARCHAR(10)	stmt_type - 语句类型
STMT_QUERY_ID	BIGINT	stmt_query_id - 语句查询标识
STMT_NEST_LEVEL	SMALLINT	stmt_nest_level - 语句嵌套级别
STMT_INVOCATION_ID	BIGINT	stmt_invocation_id - 语句调用标识
STMT_SOURCE_ID	BIGINT	stmt_source_id - 语句源标识
STMT_PKGCACHE_ID	BIGINT	stmt_pkgcache_id - 语句程序包高速缓存标识
STMT_TEXT	CLOB(2097152)	stmt_text - SQL 语句文本

表 24. 为锁定事件监视器返回的信息: 表名: LOCK_ACTIVITY_VALUES

列名	数据类型	描述
XMLID	VARCHAR(1024) NOT NULL	
PARTICIPANT_NO	BIGINT	participant_no - 死锁参与者
ACTIVITY_ID	BIGINT	activity_id - 活动标识
UOW_ID	BIGINT	uow_id - 工作单元标识
STMT_VALUE_INDEX	BIGINT	stmt_value_index - 值索引
STMT_VALUE_ISREOPT	CHAR(3)	stmt_value_isreopt - 用于语句重新优化的变量
STMT_VALUE_ISNULL	CHAR(3)	stmt_value_isnull - 包含空值
STMT_VALUE_TYPE	CHAR(16)	stmt_value_type - 值类型
STMT_VALUE_DATA	CLOB (32K)	stmt_value_data - 值数据

监视工作单元事件

每当工作单元完成时，即，每当执行落实或回滚时，工作单元（UoW）事件监视器都将记录一个事件。这些关于各个 UoW 的历史信息对于扣费（按 CPU 使用率收费）以及监视是否符合响应时间服务级别目标而言非常有用。

UoW 事件监视器是一种通过请求度量值来执行系统透视图监视的方法。与工作单元事件监视器最紧密相关的替代项或补充项是统计信息事件监视器或者 MON_GET_UNIT_OF_WORK 和 MON_GET_UNIT_OF_WORK_DETAILS 表函数。

要创建 UoW 事件监视器并收集锁定事件监视器数据，必须具有 DBADM 或 SQLADM 权限。

创建 UoW 事件监视器和配置数据收集功能

在创建 UoW 事件监视器之前，请确定要在其中存储事件监视器的非格式化事件表的表空间。建议的做法是，配置一个专用表空间来存储与任何事件监视器相关联的非格式化事件表。请在页大小至少为 8K 的表空间中创建工作单元事件监视器，以确保事件数据包含在非格式化事件表的直接插入 BLOB 列中。如果不直接插入 BLOB 列，那么对非格式化事件表读写事件的效率可能不高。

数据库管理器会尝试在非格式化事件表中直接插入 BLOB 列 event_data，但这并非始终有可能实现。要检查非格式化事件表中的行是否直接插入型的行，请使用 ADMIN_IS_INLINED 函数。如果这些行不是直接插入型的行，请使用 ADMIN_EST_INLINE_LENGTH 函数来确定这些行所需的空间量。

创建事件监视器时的其他选项包括，指定任何现有表空间或者不指定任何表空间并在缺省情况下选择表空间。

要使用缺省值和最佳实践来设置 UoW 事件监视器，请完成下列步骤:

1. 通过发出 CREATE EVENT MONITOR 语句来创建事件监视器。以下示例尽可能使用缺省值，并指定将非格式化事件表存储在现有的表空间中:

   ```
   CREATE EVENT MONITOR MY_UOW_EVMON
     FOR UNIT OF WORK
      WRITE TO UNFORMATTED EVENT TABLE (IN MY_EVMON_TABLESPACE)
   ```

2. 配置所要收集的数据。以下语句演示了一种简单的方法:

   ```
   db2 update db cfg for dbname using mon_uow_data base
   ```

配置数据收集功能

要配置数据收集功能，您还必须指定要为其捕获事件的系统工作负载子集以及要为每个事件收集的详细信息量。缺省情况下，不收集 UoW 数据。您可以使用下列其中一个设置来更改缺省设置:

- **mon_uow_data** 数据库配置参数
- CREATE/ALTER WORKLOAD 语句的 COLLECT UNIT OF WORK DATA 子句

可用的数据收集级别如下所示:

NONE 不收集 UoW 数据

BASE 收集 UoW 数据

如果将 **mon_uow_data** 数据库配置参数或 CREATE/ALTER WORKLOAD 语句的 COLLECT UNIT OF WORK DATA 子句设置为 BASE，那么这将是对工作负载生效的设置。

如果您只希望对所选工作负载启用数据收集功能，请对期望的工作负载将 **mon_uow_data** 数据库配置参数设置为 NONE 并将级别设置为 BASE。

请求度量值是其中一种可以通过 UOW 事件监视器收集的信息。UoW 事件监视器是其中一个受请求度量值收集设置影响的接口。缺省情况下，系统在适用的表函数和事件监视器（其中包括 UoW 事件监视器）中收集和报告请求度量值。您可以使用下列其中一个设置来更改缺省设置:

- **mon_req_metrics** 数据库配置参数
- 对服务超类执行的 CREATE/ALTER SERVICE CLASS 语句的 COLLECT REQUEST METRICS 子句。

更改这些设置将影响任何能够报告请求度量值的表函数或事件监视器。

访问 UoW 事件监视器所捕获的事件数据

此类事件监视器以二进制格式将数据写入非格式化事件表。您可以使用下列表函数来访问此数据:

- EVMON_FORMAT_UE_TO_XML - 将非格式化事件表中的数据抽取到 XML 文档。
- EVMON_FORMAT_UE_TO_TABLES - 将非格式化事件表中的数据抽取到一组关系表。

使用这些表函数来指定要使用 SELECT 语句抽取的数据。您可以对 SELECT 语句所提供的选择、排序和其他方面功能进行全面控制。

另外，还可以使用 db2evmonfmt 命令来执行下列任务:

- 根据下列属性来选择感兴趣的事件: 事件标识、事件类型、时间段、应用程序、工作负载或服务类。
- 选择是以文本报告形式还是格式化 XML 文档形式来接收输出。
- 通过创建您自己的 XSLT 样式表代替使用 db2evmonfmt 命令所提供的 XSLT 样式表，对输出格式进行控制。

例如，以下命令提供了一个具有下列特性的 UoW 报告:

1. 选择过去 24 小时内在 SAMPLE 数据库中发生的 UoW 事件。将从名为 SAMPLE_UoW_events 的非格式化事件表获取这些事件记录。
2. 使用 DB2EvmonUOW.xsl 样式表来提供格式化文本输出。

```
java db2evmonfmt -d SAMPLE -ue SAMPLE_UOW_EVENTS -ftext -ss DB2EvmonUOW.xsl -hours 24
```

收集工作单元事件数据并生成报告

您可以使用工作单元事件监视器来收集关于事务的数据，这些数据可用于进行扣费。本任务描述以不可读格式在非格式化事件表中收集事务事件数据之后，如何获取可读的文本报告。

开始前

要收集工作单元事件监视器数据，您必须具有 SYSADM 或 SYSCTRL 权限。

关于此任务

工作单元事件监视器收集用于标识应用程序事务和相应 CPU 使用情况的相关信息，此信息可用于进行扣费。例如，工作单元事件监视器收集的一些事务事件信息如下:

- CPU 使用时间总计（TOTAL_CPU_TIME）
- 应用程序句柄（APPLICATION_HANDLE）

本任务提供有关收集指定工作负载的工作单元事件数据的指示信息。

限制

如果您不具有 SYSADM 或 SYSCTRL 权限，那么输入数据值不可查看。

过程

要收集关于工作单元事件的详细信息，请执行以下步骤:

1. 使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来创建名为 uowevmon 的工作单元事件监视器，如以下示例所示:

   ```
   CREATE EVENT MONITOR uowevmon FOR UNIT OF WORK
      WRITE TO UNFORMATTED EVENT TABLE
   ```

2. 借助语句历史记录，使用 ALTER WORKLOAD 语句在工作负载级别启用工作单元事件数据收集功能。 要对 FINANCE 和 PAYROLL 应用程序收集工作单元数据，请发出下列语句:

   ```
   ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA WITH HISTORY
   ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA WITH HISTORY
   ```

3. 重新运行工作负载，以便收集工作单元事务事件。
4. 连接到数据库。
5. 使用以下方法来获取工作单元事件报告:
   a. 使用 XML 解析器工具 db2evmonfmt 来生成基于非格式化事件表中收集的事件数据的纯文本报告，例如:

      ```
      java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
      ```

6. 分析该报告，确定应用程序使用的 CPU 时间量，以便相应地进行扣费。
7. 如果要对 FINANCE 和 PAYROLL 应用程序关闭工作单元数据收集功能，请运行下列语句:

   ```
   ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA NONE
   ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA NONE
   ```

为工作单元事件监视器写入 XML 的信息

EVMON_FORMAT_UE_TO_XML 表函数为工作单元事件监视器写入的信息。sqllib/misc/DB2EvmonUOW.xsd 文件也对此信息作了记录。

为工作单元事件监视器写入的信息

db2_uow_event: 用于描述工作单元事件的主模式

表 25. 元素

名称	数据类型	描述
db2_uow_event	db2uowevent	用于描述工作单元事件的主模式

db2uowevent: 用于描述工作单元事件的主模式

表 26. 属性

名称	数据类型	描述
id	xs:positiveInteger	用于表示事件标识的整数。
type	xs:string	已发生的工作单元事件的类型。事件可以具有以下类型：UOW
timestamp	db2_timestamp_type	用于表示 UOW 事件发生时间的时间戳记。
member	member_type	发生 UOW 事件的成员。
release	xs:nonNegativeInteger	表示捕获此事件的 DB2 产品的级别。

表 27. 元素

名称	数据类型	最小出现次数	最大出现次数	描述
completion_status	xs:string	1	1	工作单元的完成状态。可能的值包括：UNKNOWN、COMMIT、ROLLBACK、GLOBAL_COMMIT、GLOBAL ROLLBACK、XA_END 或 XA_PREPARE
start_time	xs:dateTime	1	1	工作单元的开始时间。有关更多详细信息，请参阅监视元素 uow_start_time。
stop_time	xs:dateTime	1	1	工作单元的停止时间。有关更多详细信息，请参阅监视元素 uow_stop_time。
connection_time	xs:dateTime	1	1	应用程序连接到数据库成员的时间。有关更多详细信息，请参阅监视元素 conn_time。
application_name	xs:string	1	1	客户机上运行的应用程序在数据库中的名称。有关更多详细信息，请参阅监视元素 appl_name。
application_handle	application_handle_type	1	1	应用程序在系统范围内的唯一标识。有关更多详细信息，请参阅监视元素 agent_id。
application_id	appl_id_type	1	1	当应用程序连接到数据库管理器上的数据库时，将生成此标识。有关更多详细信息，请参阅监视元素 appl_id。
uow_id	incrementing_id_type	1	1	此活动记录所应用于的工作单元标识。有关更多详细信息，请参阅监视元素 uow_id。

表 27. 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
workload_occurrence_id	incrementing_id_type	1	1	此活动记录所应用于的工作负载实例标识。有关更多详细信息，请参阅监视元素 workload_occurrence_id。
coord_member	member_type	1	1	此工作单元的协调成员。有关更多详细信息，请参阅监视元素 coord_partitition_num。
member_activation_time	xs:dateTime	1	1	此数据库成员的激活时间。有关更多详细信息，请参阅监视元素 db_conn_time。
workload_name	db_object_name_type	1	1	在其中完成此工作单元的工作负载的名称。有关更多详细信息，请参阅监视元素 workload_name。
workload_id	db_object_id_type	1	1	在其中完成此工作单元的工作负载的工作负载标识。有关更多详细信息，请参阅监视元素 workload_id。
service_superclass_name	db_object_name_type	0	1	在其中完成此工作单元的服务超类的名称。有关更多详细信息，请参阅监视元素 service_superclass_name。
service_subclass_name	db_object_name_type	0	1	在其中完成此工作单元的服务子类的名称。有关更多详细信息，请参阅监视元素 service_subclass_name。
service_class_id	db_object_id_type	0	1	在其中完成此工作单元的服务类的服务类标识。有关更多详细信息，请参阅监视元素 service_class_id。
session_authid	authid_type	0	1	调用所监视应用程序的用户的会话授权标识。有关更多详细信息，请参阅监视元素 auth_id。
system_authid	authid_type	1	1	调用所监视应用程序的用户的系统授权标识。有关更多详细信息，请参阅监视元素 auth_id。
client_pid	xs:nonNegativeInteger	1	1	客户机报告的进程标识。有关更多详细信息，请参阅监视元素 client_pid。
client_product_id	xs:string	1	1	客户机的产品标识。有关更多详细信息，请参阅监视元素 client_prdid。

表 27. 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
client_platform	xs:nonNegativeInteger	1	1	客户机的平台。有关更多详细信息，请参阅监视元素 client_platform。
client_protocol	xs:string	0	1	客户机的产品标识。有关更多详细信息，请参阅监视元素 client_protocol。
client_userid	tpmon_type	0	1	由事务管理器生成并提供给服务器的客户机用户标识。有关更多详细信息，请参阅监视元素 client_userid。
client_wrkstnname	tpmon_type	0	1	如果在此连接中发出了 sqleseti API，那么此元素标识客户机系统或工作站。有关更多详细信息，请参阅监视元素 client_wrkstnname。
client_applname	tpmon_type	0	1	如果在此连接中发出了 sqleseti API，那么此元素标识执行事务的服务器事务程序。有关更多详细信息，请参阅监视元素 client_applname。
client_acctng	tpmon_type	0	1	如果在此连接中发出了 sqleseti API，那么此元素是为了进行日志记录和诊断而传递到目标数据库的数据。有关更多详细信息，请参阅监视元素 client_acctng。
local_transaction_id	xs:hexBinary	1	1	工作单元的局部事务标识。
global_transaction_id	xs:hexBinary	1	1	工作单元的全局事务标识。
system_metrics	system_level_metrics	1	1	工作单元的度量值。这是包含所有系统级度量值的 XML 元素。

为工作单元事件监视器写入关系表的信息

本节概述使用 EVMON_FORMAT_UE_TO_TABLES 过程为工作单元事件监视器从 XML 文档写入关系表的信息。`sqllib/misc/DB2EvmonUOW.xsd` 文件也对此信息作了记录。

为 UOW 事件监视器写入的信息

表 28. 为 UOW 事件监视器返回的信息: 表名: UOW_EVENT

列名	数据类型	描述
EVENT_ID	INTEGER NOT NULL	
TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	
MEMBER	SMALLINT	member - 数据库成员

表 28. 为 UOW 事件监视器返回的信息: 表名: UOW_EVENT (续)

列名	数据类型	描述
COORD_MEMBER	SMALLINT	coord_member - 协调程序成员
COMPLETION_STATUS	VARCHAR(128)	
START_TIME	TIMESTAMP	start_time - 事件开始时间
STOP_TIME	TIMESTAMP	stop_time - 事件停止时间
WORKLOAD_NAME	VARCHAR(128)	workload_name - 工作负载名称
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
SERVICE_CLASS_ID	INTEGER	service_class_id - 服务类标识
UOW_ID	INTEGER	uow_id - 工作单元标识
WORKLOAD_OCCURRENCE_ID	INTEGER	workload_occurrence_id - 工作负载项标识
CONNECTION_TIME	TIMESTAMP	
MEMBER_ACTIVATION_TIME	TIMESTAMP	
APPLICATION_ID	VARCHAR(128)	
APPLICATION_HANDLE	BIGINT	application_handle - 应用程序句柄
APPLICATION_NAME	VARCHAR(128)	
SYSTEM_AUTHID	VARCHAR(128)	
SESSION_AUTHID	VARCHAR(128)	
CLIENT_PLATFORM	INTEGER	client_platform - 客户机操作平台
CLIENT_PID	INTEGER	client_pid - 客户机进程标识
CLIENT_PRODUCT_ID	VARCHAR(128)	
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - 客户机通信协议
CLIENT_WRKSTNNAME	VARCHAR(128)	
CLIENT_ACCTNG	VARCHAR(200)	
CLIENT_USERID	VARCHAR(64)	
CLIENT_APPLNAME	VARCHAR(128)	
LOCAL_TRANSACTION_ID	VARCHAR(16)	
GLOBAL_TRANSACTION_ID	VARCHAR(40)	
METRICS	BLOB(100K)	包含所有系统级度量值的 XML 文档。

使用统计信息事件监视器来捕获系统监视元素

统计信息事件监视器包含 event_scstats 和 event_wlstats 逻辑数据组中的 details_xml 监视元素。使用此监视元素可捕获关于系统的信息。

监视元素 details_xml 是一个 XML 文档，该文档包含 MON_GET_SERVICE_SUBCLASS_DETAILS 和 MON_GET_WORKLOAD_DETAILS 表函数所报告的所有系统监视元素。系统监视元素是 MON_GET_SERVICE_SUBCLASS_DETAILS 和 MON_GET_WORKLOAD_DETAILS 表函数的 DETAILS 列中报告的详细信息文档的子集。

您可以通过 COLLECT REQUEST METRICS 子句对服务超类控制请求监视元素，也可以通过 mon_req_metrics 数据库配置参数在数据库级别进行控制。仅当处理请求的代理程序所在服务子类的父服务超类已启用请求监视元素收集功能，或者已对整个数据库启用请求监视元素收集功能时，才会对该请求收集这些监视元素。如果已在数据库级别禁用请求监视元素，那么对于服务超类，DETAILS_XML 文档中报告的度量值将停止增大（或者，如果已在激活数据库时禁用请求度量值，那么此度量值将保持为 0）。

文件 sqllib/misc/DB2MonCommon.xsd 包含 DETAILS_XML 列中返回的 XML 文档的模式。顶层元素是 system_metrics。

system_metrics 监视元素的 XML 模式

system_metrics 监视元素包含 MON_GET_SERVICE_SUBCLASS_DETAILS 和 MON_GET_WORKLOAD_DETAILS 表函数所报告的所有系统度量值。

system_metrics

系统级度量值。

表 29. system_metrics

名称	数据类型	最小出现次数	最大出现次数	描述
system_metrics	system_level_metrics	1	1	系统级度量值。

system_level_metrics

此类型定义系统级别的度量值。

表 30. system_level_metrics 属性

名称	数据类型	用法	描述
release	xs:nonNegativeInteger	必需	表示捕获此事件的 DB2 产品的级别。

表 31. system_level_metrics 元素

名称	数据类型	最小出现次数	最大出现次数	描述
WLM_QUEUE_TIME_TOTAL	metric_value_type	1	1	wlm_queue_time_total - 工作负载管理器队列时间总计
WLM_QUEUE_ASSIGNMENTS_TOTAL	metric_value_type	1	1	wlm_queue_assignments_total - 工作负载管理器队列分配总次数
FCM_TQ_RECV_WAIT_TIME	metric_value_type	1	1	fcm_tq_recv_wait_time - FCM 表队列接收等待时间
FCM_MESSAGE_RECV_WAIT_TIME	metric_value_type	1	1	fcm_message_recv_wait_time - 接收 FCM 消息等待时间
FCM_TQ_SEND_WAIT_TIME	metric_value_type	1	1	fcm_tq_send_wait_time - FCM 表队列发送等待时间
FCM_MESSAGE_SEND_WAIT_TIME	metric_value_type	1	1	fcm_message_send_wait_time - 发送 FCM 消息等待时间
AGENT_WAIT_TIME	metric_value_type	1	1	agent_wait_time - 代理程序等待时间
AGENT_WAITS_TOTAL	metric_value_type	1	1	agent_waits_total - 等待代理程序总次数
LOCK_WAIT_TIME	metric_value_type	1	1	lock_wait_time - 等待锁定的时间

表 31. system_level_metrics 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
LOCK_WAITS	metric_value_type	1	1	lock_waits - 等待锁定次数
DIRECT_READ_TIME	metric_value_type	1	1	direct_read_time - 直接读取时间
DIRECT_READ_REQS	metric_value_type	1	1	direct_read_reqs - 直接读请求数
DIRECT_WRITE_TIME	metric_value_type	1	1	direct_write_time - 直接写入时间
DIRECT_WRITE_REQS	metric_value_type	1	1	direct_write_reqs - 直接写请求数
LOG_BUFFER_WAIT_TIME	metric_value_type	1	1	log_buffer_wait_time - 日志缓冲区等待时间
NUM_LOG_BUFFER_FULL	metric_value_type	1	1	num_log_buffer_full - 变满的日志缓冲区的数目
LOG_DISK_WAIT_TIME	metric_value_type	1	1	log_disk_wait_time - 日志磁盘等待时间
LOG_DISK_WAITS_TOTAL	metric_value_type	1	1	log_disk_waits_total - 日志磁盘等待总次数
TCPIP_RECV_WAIT_TIME	metric_value_type	1	1	tcpip_recv_wait_time - TCP/IP 接收等待时间
TCPIP_RECVS_TOTAL	metric_value_type	1	1	tcpip_recvs_total - TCP/IP 接收总次数
CLIENT_IDLE_WAIT_TIME	metric_value_type	1	1	client_idle_wait_time - 客户机空闲等待时间
IPC_RECV_WAIT_TIME	metric_value_type	1	1	ipc_recv_wait_time - 进程间通信接收等待时间
IPC_RECVS_TOTAL	metric_value_type	1	1	ipc_recvs_total - 进程间通信接收总次数
IPC_SEND_WAIT_TIME	metric_value_type	1	1	ipc_send_wait_time - 进程间通信发送等待时间
IPC_SENDS_TOTAL	metric_value_type	1	1	ipc_sends_total - 进程间通信发送总次数
TCPIP_SEND_WAIT_TIME	metric_value_type	1	1	tcpip_send_wait_time - TCP/IP 发送等待时间
TCPIP_SENDS_TOTAL	metric_value_type	1	1	tcpip_sends_total - TCP/IP 发送总次数
POOL_WRITE_TIME	metric_value_type	1	1	pool_write_time - 缓冲池物理写时间总计
POOL_READ_TIME	metric_value_type	1	1	pool_read_time - 缓冲池物理读时间总计
AUDIT_FILE_WRITE_ WAIT_TIME	metric_value_type	1	1	audit_file_write_wait_time - 审计文件写等待时间
AUDIT_FILE_WRITES_TOTAL	metric_value_type	1	1	audit_file_writes_total - 写审计文件总次数
AUDIT_SUBSYSTEM_ WAIT_TIME	metric_value_type	1	1	audit_subsystem_wait_time - 审计子系统等待时间
AUDIT_SUBSYSTEM_ WAITS_TOTAL	metric_value_type	1	1	audit_subsystem_waits_total - 审计子系统等待总次数
DIAGLOG_WRITE_WAIT_TIME	metric_value_type	1	1	diaglog_write_wait_time - 诊断日志文件写等待时间
DIAGLOG_WRITES_TOTAL	metric_value_type	1	1	diaglog_writes_total - 写诊断日志文件总次数
FCM_SEND_WAIT_TIME	metric_value_type	1	1	fcm_send_wait_time - FCM 发送等待时间

表 31. system_level_metrics 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
FCM_RECV_WAIT_TIME	metric_value_type	1	1	fcm_recv_wait_time - FCM 接收等待时间
TOTAL_WAIT_TIME	metric_value_type	1	1	total_wait_time - 等待时间总计
TOTAL_RQST_TIME	metric_value_type	1	1	total_rqst_time - 请求时间总计
RQSTS_COMPLETED_TOTAL	metric_value_type	1	1	rqsts_completed_total - 完成请求总数
TOTAL_APP_RQST_TIME	metric_value_type	1	1	total_app_rqst_time - 应用程序请求时间总计
APP_RQSTS_COMPLETED_TOTAL	metric_value_type	1	1	app_rqsts_completed_total - 完成应用程序请求总数
TOTAL_SECTION_SORT_PROC_TIME	metric_value_type	1	1	total_section_sort_proc_time - 节排序处理时间总计
TOTAL_SECTION_SORT_TIME	metric_value_type	1	1	total_section_sort_time - 节排序时间总计
TOTAL_SECTION_SORTS	metric_value_type	1	1	total_section_sorts - 节排序总次数
ROWS_READ	metric_value_type	1	1	rows_read - 读取的行数
ROWS_MODIFIED	metric_value_type	1	1	rows_modified - 修改的行数
POOL_DATA_L_READS	metric_value_type	1	1	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_INDEX_L_READS	metric_value_type	1	1	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_TEMP_DATA_L_READS	metric_value_type	1	1	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_INDEX_L_READS	metric_value_type	1	1	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_XDA_L_READS	metric_value_type	1	1	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_TEMP_XDA_L_READS	metric_value_type	1	1	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
TOTAL_CPU_TIME	metric_value_type	1	1	total_cpu_time - CPU 时间总计
ACT_COMPLETED_TOTAL	metric_value_type	1	1	act_completed_total - 完成活动总数
POOL_DATA_P_READS	metric_value_type	1	1	pool_data_p_reads - 缓冲池数据物理读取数
POOL_TEMP_DATA_P_READS	metric_value_type	1	1	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_XDA_P_READS	metric_value_type	1	1	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_TEMP_XDA_P_READS	metric_value_type	1	1	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_INDEX_P_READS	metric_value_type	1	1	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_INDEX_P_READS	metric_value_type	1	1	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_DATA_WRITES	metric_value_type	1	1	pool_data_writes - 缓冲池数据写次数
POOL_XDA_WRITES	metric_value_type	1	1	pool_xda_writes - 缓冲池 XDA 数据写次数

表 31. system_level_metrics 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
POOL_INDEX_WRITES	metric_value_type	1	1	pool_index_writes - 缓冲池索引写次数
DIRECT_READS	metric_value_type	1	1	direct_reads - 直接读数据库数目
DIRECT_WRITES	metric_value_type	1	1	direct_writes - 直接写数据库数目
ROWS_RETURNED	metric_value_type	1	1	rows_returned - 返回的行数
DEADLOCKS	metric_value_type	1	1	deadlocks - 检测到的死锁数
LOCK_TIMEOUTS	metric_value_type	1	1	lock_timeouts - 锁定超时次数
LOCK_ESCALS	metric_value_type	1	1	lock_escals - 锁定升级次数
FCM_SENDS_TOTAL	metric_value_type	1	1	fcm_sends_total - FCM 发送总计
FCM_RECVS_TOTAL	metric_value_type	1	1	fcm_recvs_total - FCM 接收总计
FCM_SEND_VOLUME	metric_value_type	1	1	fcm_send_volume - FCM 发送量
FCM_RECV_VOLUME	metric_value_type	1	1	fcm_recv_volume - FCM 接收量
FCM_MESSAGE_SENDS_ TOTAL	metric_value_type	1	1	fcm_message_sends_total - 发送 FCM 消息总数
FCM_MESSAGE_RECVS_ TOTAL	metric_value_type	1	1	fcm_message_recvs_total - 接收 FCM 消息总数
FCM_MESSAGE_SEND_ VOLUME	metric_value_type	1	1	fcm_message_send_volume - 发送 FCM 消息量
FCM_MESSAGE_RECV_ VOLUME	metric_value_type	1	1	fcm_message_recv_volume - 接收 FCM 消息量
FCM_TQ_SENDS_TOTAL	metric_value_type	1	1	fcm_tq_sends_total - FCM 表队列发送总计
FCM_TQ_RECVS_TOTAL	metric_value_type	1	1	fcm_tq_recvs_total - FCM 表队列接收总计
FCM_TQ_SEND_VOLUME	metric_value_type	1	1	fcm_tq_send_volume - FCM 表队列发送量
FCM_TQ_RECV_VOLUME	metric_value_type	1	1	fcm_tq_recv_volume - FCM 表队列接收量
TQ_TOT_SEND_SPILLS	metric_value_type	1	1	tq_tot_send_spills - 溢出表队列缓冲区总数
TCPIP_SEND_VOLUME	metric_value_type	1	1	tcpip_send_volume - TCP/IP 发送量
TCPIP_RECV_VOLUME	metric_value_type	1	1	tcpip_recv_volume - TCP/IP 接收量
IPC_SEND_VOLUME	metric_value_type	1	1	ipc_send_volume - 进程间通信发送量
IPC_RECV_VOLUME	metric_value_type	1	1	ipc_recv_volume - 进程间通信接收量
POST_THRESHOLD_SORTS	metric_value_type	1	1	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	metric_value_type	1	1	post_shrthreshold_sorts - 共享阈值后排序数
SORT_OVERFLOWS	metric_value_type	1	1	sort_overflows - 排序溢出数
AUDIT_EVENTS_TOTAL	metric_value_type	1	1	audit_events_total - 审计事件总数
TOTAL_RQST_MAPPED_IN	metric_value_type	0	1	total_rqst_mapped_in - 映入请求总数
TOTAL_RQST_MAPPED_OUT	metric_value_type	0	1	total_rqst_mapped_out - 映出请求总数

表 31. system_level_metrics 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
ACT_REJECTED_TOTAL	metric_value_type	1	1	act_rejected_total - 被拒绝活动总数
ACT_ABORTED_TOTAL	metric_value_type	1	1	act_aborted_total - 异常终止活动总数
TOTAL_SORTS	metric_value_type	1	1	total_sorts - 排序总数

metric_value_type

度量值时间和计数器的值类型。

使用活动事件监视器来捕获活动监视元素

活动事件监视器包含 event_activity 逻辑数据组中的 details_xml 监视元素。使用此监视元素可捕获关于活动的信息。

监视元素 **details_xml** 返回一个 XML 文档，该文档包含 MON_GET_ACTIVITY_DETAILS 表函数所报告的所有活动监视元素。活动监视元素是 MON_GET_ACTIVITY_DETAILS 表函数的 DETAILS 列中报告的活动详细信息文档的子集。

您可以通过 COLLECT ACTIVITY METRICS 子句对工作负载控制活动监视元素，也可以通过 **mon_act_metrics** 数据库配置参数在数据库级别进行控制。如果提交活动的连接与启用了活动监视元素收集功能的工作负载或数据库相关联，那么将收集这些监视元素。如果未对某个活动收集活动监视元素，那么 DETAILS_XML 将包含空文档。

文件 sqllib/misc/DB2MonCommon.xsd 包含 DETAILS_XML 列中返回的 XML 文档的模式。顶层元素是 activity_metrics。

activity_metrics 监视元素的 XML 模式

activity_metrics 监视元素包含 MON_GET_ACTIVITY_DETAILS 表函数所报告的所有活动度量值。

activty_metrics

活动级别度量值。

表 32. activty_metrics

名称	数据类型	最小出现次数	最大出现次数	描述
activity_metrics	activity_level_metrics	1	1	活动级别度量值。

activity_level_metrics

此类型定义活动级别的度量值。

表 33. activity_level_metrics 属性

名称	数据类型	用法	描述
release	xs:nonNegativeInteger	必需	表示捕获此事件的 DB2 产品的级别。

表 34. activity_level_metrics 元素

名称	数据类型	最小出现次数	最大出现次数	描述
WLM_QUEUE_TIME_TOTAL	metric_value_type	1	1	wlm_queue_time_total - 工作负载管理器队列时间总计
WLM_QUEUE_ ASSIGNMENTS_TOTAL	metric_value_type	1	1	wlm_queue_assignments_total - 工作负载管理器队列分配总次数
FCM_TQ_RECV_WAIT_TIME	metric_value_type	1	1	fcm_tq_recv_wait_time - FCM 表队列接收等待时间
FCM_MESSAGE_RECV_ WAIT_TIME	metric_value_type	1	1	fcm_message_recv_wait_time - 接收 FCM 消息等待时间
FCM_TQ_SEND_WAIT_TIME	metric_value_type	1	1	fcm_tq_send_wait_time - FCM 表队列发送等待时间
FCM_MESSAGE_SEND_ WAIT_TIME	metric_value_type	1	1	fcm_message_send_wait_time - 发送 FCM 消息等待时间
LOCK_WAIT_TIME	metric_value_type	1	1	lock_wait_time - 等待锁定的时间
LOCK_WAITS	metric_value_type	1	1	lock_waits - 等待锁定次数
DIRECT_READ_TIME	metric_value_type	1	1	direct_read_time - 直接读取时间
DIRECT_READ_REQS	metric_value_type	1	1	direct_read_reqs - 直接读请求数
DIRECT_WRITE_TIME	metric_value_type	1	1	direct_write_time - 直接写入时间
DIRECT_WRITE_REQS	metric_value_type	1	1	direct_write_reqs - 直接写请求数
LOG_BUFFER_WAIT_TIME	metric_value_type	1	1	log_buffer_wait_time - 日志缓冲区等待时间
NUM_LOG_BUFFER_FULL	metric_value_type	1	1	num_log_buffer_full - 变满的日志缓冲区的数目
LOG_DISK_WAIT_TIME	metric_value_type	1	1	log_disk_wait_time - 日志磁盘等待时间
LOG_DISK_WAITS_TOTAL	metric_value_type	1	1	log_disk_waits_total - 日志磁盘等待总次数
POOL_WRITE_TIME	metric_value_type	1	1	pool_write_time - 缓冲池物理写时间总计
POOL_READ_TIME	metric_value_type	1	1	pool_read_time - 缓冲池物理读时间总计
AUDIT_FILE_WRITE_ WAIT_TIME	metric_value_type	1	1	audit_file_write_wait_time - 审计文件写等待时间
AUDIT_FILE_WRITES_TOTAL	metric_value_type	1	1	audit_file_writes_total - 写审计文件总次数
AUDIT_SUBSYSTEM_ WAIT_TIME	metric_value_type	1	1	audit_subsystem_wait_time - 审计子系统等待时间
AUDIT_SUBSYSTEM_ WAITS_TOTAL	metric_value_type	1	1	audit_subsystem_waits_total - 审计子系统等待总次数
DIAGLOG_WRITE_WAIT_ TIME	metric_value_type	1	1	diaglog_write_wait_time - 诊断日志文件写等待时间
DIAGLOG_WRITES_TOTAL	metric_value_type	1	1	diaglog_writes_total - 写诊断日志文件总次数
FCM_SEND_WAIT_TIME	metric_value_type	1	1	fcm_send_wait_time - FCM 发送等待时间
FCM_RECV_WAIT_TIME	metric_value_type	1	1	fcm_recv_wait_time - FCM 接收等待时间

表 34. activity_level_metrics 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
TOTAL_ACT_WAIT_TIME	metric_value_type	1	1	total_act_wait_time - 活动等待时间总计
TOTAL_SECTION_SORT_PROC_TIME	metric_value_type	1	1	total_section_sort_proc_time - 节排序处理时间总计
TOTAL_SECTION_SORT_TIME	metric_value_type	1	1	total_section_sort_time - 节排序时间总计
TOTAL_SECTION_SORTS	metric_value_type	1	1	total_section_sorts - 节排序总次数
TOTAL_ACT_TIME	metric_value_type	1	1	total_act_time - 活动时间总计
ROWS_READ	metric_value_type	1	1	rows_read - 读取的行数
ROWS_MODIFIED	metric_value_type	1	1	rows_modified - 修改的行数
POOL_DATA_L_READS	metric_value_type	1	1	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_INDEX_L_READS	metric_value_type	1	1	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_TEMP_DATA_L_READS	metric_value_type	1	1	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_INDEX_L_READS	metric_value_type	1	1	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_XDA_L_READS	metric_value_type	1	1	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_TEMP_XDA_L_READS	metric_value_type	1	1	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
TOTAL_CPU_TIME	metric_value_type	1	1	total_cpu_time - CPU 时间总计
POOL_DATA_P_READS	metric_value_type	1	1	pool_data_p_reads - 缓冲池数据物理读取数
POOL_TEMP_DATA_P_READS	metric_value_type	1	1	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_XDA_P_READS	metric_value_type	1	1	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_TEMP_XDA_P_READS	metric_value_type	1	1	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_INDEX_P_READS	metric_value_type	1	1	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_INDEX_P_READS	metric_value_type	1	1	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_DATA_WRITES	metric_value_type	1	1	pool_data_writes - 缓冲池数据写次数
POOL_XDA_WRITES	metric_value_type	1	1	pool_xda_writes - 缓冲池 XDA 数据写次数
POOL_INDEX_WRITES	metric_value_type	1	1	pool_index_writes - 缓冲池索引写次数
DIRECT_READS	metric_value_type	1	1	direct_reads - 直接读数据库数目
DIRECT_WRITES	metric_value_type	1	1	direct_writes - 直接写数据库数目
ROWS_RETURNED	metric_value_type	1	1	rows_returned - 返回的行数
DEADLOCKS	metric_value_type	1	1	deadlocks - 检测到的死锁数
LOCK_TIMEOUTS	metric_value_type	1	1	lock_timeouts - 锁定超时次数

表 34. activity_level_metrics 元素 (续)

名称	数据类型	最小出现次数	最大出现次数	描述
LOCK_ESCALS	metric_value_type	1	1	lock_escals - 锁定升级次数
FCM_SENDS_TOTAL	metric_value_type	1	1	fcm_sends_total - FCM 发送总计
FCM_RECVS_TOTAL	metric_value_type	1	1	fcm_recvs_total - FCM 接收总计
FCM_SEND_VOLUME	metric_value_type	1	1	fcm_send_volume - FCM 发送量
FCM_RECV_VOLUME	metric_value_type	1	1	fcm_recv_volume - FCM 接收量
FCM_MESSAGE_SENDS_TOTAL	metric_value_type	1	1	fcm_message_sends_total - 发送 FCM 消息总数
FCM_MESSAGE_RECVS_TOTAL	metric_value_type	1	1	fcm_message_recvs_total - 接收 FCM 消息总数
FCM_MESSAGE_SEND_VOLUME	metric_value_type	1	1	fcm_message_send_volume - 发送 FCM 消息量
FCM_MESSAGE_RECV_VOLUME	metric_value_type	1	1	fcm_message_recv_volume - 接收 FCM 消息量
FCM_TQ_SENDS_TOTAL	metric_value_type	1	1	fcm_tq_sends_total - FCM 表队列发送总计
FCM_TQ_RECVS_TOTAL	metric_value_type	1	1	fcm_tq_recvs_total - FCM 表队列接收总计
FCM_TQ_SEND_VOLUME	metric_value_type	1	1	fcm_tq_send_volume - FCM 表队列发送量
FCM_TQ_RECV_VOLUME	metric_value_type	1	1	fcm_tq_recv_volume - FCM 表队列接收量
TQ_TOT_SEND_SPILLS	metric_value_type	1	1	tq_tot_send_spills - 溢出表队列缓冲区总数
POST_THRESHOLD_SORTS	metric_value_type	1	1	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	metric_value_type	1	1	post_shrthreshold_sorts - 共享阈值后排序数
SORT_OVERFLOWS	metric_value_type	1	1	sort_overflows - 排序溢出数
AUDIT_EVENTS_TOTAL	metric_value_type	1	1	audit_events_total - 审计事件总数
TOTAL_SORTS	metric_value_type	1	1	total_sorts - 排序总数

metric_value_type

度量值时间和计数器的值类型。

写表、文件和管道的事件监视器

可以将某些事件监视器配置为，将关于数据库事件的信息写入表、管道或文件。

在指定的事件发生时，事件监视器用来收集有关数据库和所有已连接的应用程序的信息。事件表示数据库活动（如连接、死锁、语句或事务）中的转换。可按想要监视的事件的类型来定义事件监视器。例如，死锁事件监视器等待死锁发生；发生死锁时，它将收集所涉及应用程序和处于争用状态的锁定的信息。

要创建事件监视器，请使用 CREATE EVENT MONITOR SQL 语句。事件监视器仅在活动时才收集事件数据。要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE SQL 语句。事件监视器的状态（活动还是不活动）可通过 SQL 函数 EVENT_MON_STATE 确定。

执行 CREATE EVENT MONITOR SQL 语句时，它创建的事件监视器定义将存储在下列数据库系统目录表中:

- SYSCAT.EVENTMONITORS: 为数据库定义的事件监视器。
- SYSCAT.EVENTS: 对数据库监视的事件。
- SYSCAT.EVENTTABLES: 表事件监视器的目标表。

每个事件监视器有自己的专用逻辑视图，用于查看监视元素中的实例数据。如果释放并重新激活特定事件监视器，那么这些计数器的视图将会复位。只有新激活的事件监视器受到影响; 所有其他事件监视器将继续使用它们的计数器值（及所有新的添加内容）的视图。

事件监视器输出可引导至非分区 SQL 表、文件或命名管道。

注: 缺省情况下，将为每个数据库创建不推荐使用的详细死锁事件监视器 DB2DETAILDEADLOCK，并在激活数据库时启动该监视器。请通过将其删除来避免此事件监视器导致的开销。建议您不要继续使用 DB2DETAILDEADLOCK 监视元素。在将来的发行版中，可能会除去这个不推荐使用的事件监视器。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

收集关于数据库系统事件的信息

在指定的事件发生时，事件监视器用来收集有关数据库和所有已连接的应用程序的信息。使用 SQL 数据定义语言（SQL DDL）语句创建并处理事件监视器和数据库对象之类的内容。

您将需要 SQLADM 或 DBADM 权限来创建和操作事件监视器。

特定类型的事件监视器允许您选择将输出定向到文件、管道或常规表。这里对此类监视器进行描述。其中一些详细信息不适用于将输出定向到非格式化事件表的事件监视器。

下面列示的步骤表示事件监视器的典型生命周期。如果真是如此，也不一定要按显示的顺序执行这些步骤。例如，根据使用情况，事件监视器可能永远不会被删除，甚至不会被释放。但是，事件监视器生命周期中有两项是不变的: 第一步始终是创建事件监视器，而最后一步始终是删除该事件监视器。

1. 第 54 页的『创建事件监视器』。
2. *仅适用于文件和管道事件监视器:*
   - 确保将接收事件记录的目录或命名管道存在。事件监视器本来不会激活。

     在 AIX® 上，可使用 mkfifo 命令来创建命名管道。在 Linux® 和其他 UNIX® 类型（如 Solaris 操作系统）上，使用 pipe() 例程。

     在 Windows® 上，可通过使用 CreateNamedPipe() 例程来创建命名管道。

- *仅适用于管道事件监视器*: 在激活事件监视器前打开命名管道。可使用操作系统功能来完成此操作:
  - 对于 UNIX: `open()`
  - 对于 Windows: `ConnectNamedPipe()`

  还可使用 db2evmon 可执行文件来完成此操作:

  ```
  db2evmon -db databasename
                   -evm eventmonname
  ```

  `databasename` 表示被监视的数据库的名称。

  `evmonname` 表示事件监视器的名称。

3. 激活新创建的事件监视器以使它能够收集信息。

   ```
   SET EVENT MONITOR evmonname STATE 1;
   ```

   如果事件监视器是使用 AUTOSTART 选项创建的，那么将在第一个用户连接至数据库时激活事件监视器。而且一旦显式激活了事件监视器，每次重新激活数据库时它将自动重新启动。在显式释放事件监视器或停止实例之前，重新激活数据库时都会重新启动事件监视器。 启动表事件监视器时，事件监视器将更新 SYSCAT.EVENTMONITORS 目录表的 evmon_activates 列。将记录此更改，所以 DATABASE CONFIGURATION 将显示:

   ```
   数据库是一致的                                  = 否
   ```

   如果事件监视器是使用 AUTOSTART 选项创建的，并且第一个用户连接至数据库后立即断开连接从而使得数据库被释放，那么会生成日志文件。

4. 要查看事件监视器处于活动状态还是不活动状态，在针对表 SYSCAT.EVENTMONITORS 的查询中发出 SQL 函数 EVENT_MON_STATE:

   ```
   SELECT evmonname, EVENT_MON_STATE(evmonname) FROM
                syscat.eventmonitors;
   ```

   将列示所有现有事件监视器的列表及事件监视器状态。返回值 0 指示指定的事件监视器处于不活动状态，而返回值 1 指示事件监视器处于活动状态。

5. 读取事件监视器输出。对于写至表事件监视器，这涉及检查目标表。要从 CLP 访问文件或管道事件监视器数据，请参阅“相关任务: 从命令行格式化文件或管道事件监视器输出”。

6. 为释放或关闭事件监视器，请使用 SET EVENT MONITOR 语句:

   ```
   SET EVENT MONITOR evmonname STATE 0
   ```

   释放事件监视器不会导致删除该监视器。它仍然作为休眠数据库对象存在。释放事件监视器将清空其所有内容。因此，如果重新激活已释放的事件监视器，它将仅包含自重新激活后收集的信息。

   请注意，如果活动事件监视器在数据库停用时处于活动状态，那么队列中储备的活动记录被删除。为了确保获取所有活动事件监视器记录且不会删除这些记录，首先应显式停用该活动事件监视器，然后停用该数据库。当显式停用活动事件监视器时，队列中所有储备的活动记录将得到处理，然后事件监视器停用。

7. 在释放管道事件监视器后，关闭相应的命名管道。在 UNIX 中使用 `close()` 函数，在 Windows 2000 中则使用 `DisconnectNamedPipe()` 函数。

8. 要消除事件监视器对象，请使用 DROP EVENT MONITOR 语句:

   ```
   DROP EVENT MONITOR evmonname
   ```

   只能在事件监视器处于不活动状态时删除事件监视器。
9. 在删除管道事件监视器后，删除相应的命名管道。在 UNIX 中使用 unlink() 函数，在 Windows 中使用 CloseHandle() 函数。 删除写至表事件监视器时，不会删除相关联的目标表。同样，在删除文件事件监视器时，不会删除相关联的文件。

创建事件监视器

事件监视器生命周期中的第一步就是创建事件监视器。在创建事件监视器之前，必须确定要将事件记录发送至 SQL 表或文件或者通过命名管道发送。

您需要 SQLADM 或 DBADM 权限才能创建事件监视器。

对于每个事件记录目标，都将在 CREATE EVENT MONITOR SQL 语句中指定一些特定选项。CREATE EVENT MONITOR 语句的目标表必须是非分区表。在分区数据库中监视事件时也应特别注意。

特定类型的事件监视器允许您选择将输出定向到文件、管道或常规表。这里对此类监视器进行描述。其中一些详细信息不适用于将输出定向到非格式化事件表的事件监视器。

1. 创建表事件监视器。
2. 创建文件事件监视器。
3. 创建管道事件监视器。
4. 为分区数据库创建事件监视器。

一旦创建并激活事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

创建表事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。表事件监视器将事件记录传输至 SQL 表，它提供了文件和管道事件监视器的简单替代品，允许您轻松捕获、分析和管理事件监视数据。对于事件监视器收集的每个事件类型，将对每个关联逻辑数据组创建目标表。

您将需要 SQLADM 或 DBADM 权限来创建表事件监视器。

CREATE EVENT MONITOR 语句的目标表必须是非分区表。

表事件监视器的各种选项将在 CREATE EVENT MONITOR 语句中设置。要进一步获取对写至表事件监视器生成 CREATE EVENT MONITOR SQL 语句的帮助，可使用 db2evtbl 命令。只需要提供事件监视器的名称和期望的事件类型，就会生成 CREATE EVENT MONITOR 语句，完成所有目标表列表。然后可从 CLP 复制生成的语句，进行修改然后执行该语句。

1. 指示会将事件监视器数据收集在一个表（或一组文件）中。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                        WRITE TO TABLE
   ```

   dlmon 是事件监视器的名称。
2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE
```

此事件监视器将监视 CONNECTIONS 和 DEADLOCKS WITH DETAILS 事件类型。假定上述语句由用户“riihi”发出，那么目标表的派生名称和表空间如下所示:

- riihi.connheader_dlmon
- riihi.conn_dlmon
- riihi.connmemuse_dlmon
- riihi.deadlock_dlmon
- riihi.dlconn_dlmon
- riihi.dllock_dlmon
- riihi.control_dlmon

3. 通过调整 BUFFERSIZE 值来指定表事件监视器缓冲区的大小（以 4K 页计）:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8
```

8 是两个事件表缓冲区的组合容量（以 4K 页计）。这加起来高达 32K 缓冲区空间，每个缓冲区 16K。

每个缓冲区的缺省大小为 4 页（分配两个 16K 缓冲区）。最小大小为 1 页。由于缓冲区是根据监视器堆进行分配，所以缓冲区的最大大小受该监视器堆大小限制。由于性能原因，高可用事件监视器的缓冲区应该比不活动事件监视器的缓冲区大。

4. 指示是需要分块事件监视器还是非分块事件监视器。对于分块事件监视器，如果事件缓冲区已满，那么生成事件的每个代理程序将等待事件缓冲区写至表。这可能会导致数据库性能降低，原因是暂挂的代理程序和所有从属代理程序在缓冲区清空之前不能运行。使用 BLOCKED 子句来确保事件数据不会丢失:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 BLOCKED
```

缺省情况下，事件监视器处于受阻状态。

如果数据库性能比收集每个事件记录更重要，那么使用非分块事件监视器。在此情况下，如果事件缓冲区已满，那么生成事件的每个代理程序将不会等待事件缓冲区写至表。因此，非分块事件监视器可能会导致活动频繁的系统上的数据丢失。使用 NONBLOCKED 子句来使事件监视的性能开销降至最低:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
```

5. 指示需要从中收集事件记录的逻辑数据组。事件监视器将每个逻辑数据组中的数据存储在相应的表中。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE CONN, DLCONN, DLLOCK
                    BUFFERSIZE 8 NONBLOCKED
```

将选择 CONN、DLCONN 和 DLLOCK 逻辑数据组。未提到的其他可用逻辑数据组包括 CONNHEADER、DEADLOCK 或 CONTROL。对于 dlmon 事件监视器，将不会存储与 CONNHEADER、DEADLOCK 或 CONTROL 相关的数据。

6. 指示需要从中收集数据的监视元素。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
             WRITE TO TABLE CONN,
             DLCONN (EXCLUDES(agent_id, lock_wait_start_time)),
             DLLOCK (INCLUDES(lock_mode, table_name))
             BUFFERSIZE 8 NONBLOCKED
```

将捕获 CONN 的所有监视元素（这是缺省行为）。对于 DLCONN，将捕获除了 **agent_id** 和 **lock_wait_start_time** 之外的所有监视元素。最后，对于 DLLOCK，**lock_mode** 和 **table_name** 是唯一捕获的监视元素。

7. 提供要创建的表的名称并指定表空间:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO TABLE CONN,
DLCONN (TABLE mydept.dlconnections
EXCLUDES(agent_id, lock_wait_start_time)),
DLLOCK (TABLE dllocks IN mytablespace
INCLUDES(lock_mode, table_name))
BUFFERSIZE 8 NONBLOCKED
```

假定上述语句由用户 riihi 发出，那么目标表的派生名称和表空间如下所示:

- CONN: riihi.conn_dlmon（在缺省表空间上）
- DLCONN: mydept.dlconnections（在缺省表空间上）
- DLLOCK: riihi.dllocks（在 MYTABLESPACE 表空间上）

如果事件监视器定义者具有 USE 特权，那么从 IBMDEFAULTGROUP 指定缺省表空间。如果定义者没有针对此表空间的 USE 特权，那么将指定定义者对其具有 USE 特权的表空间。

8. 指示事件监视器自动释放表空间之前该表空间可以到达的充满程度:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE DLCONN PCTDEACTIVATE 90
                    BUFFERSIZE 8 NONBLOCKED
```

当表空间达到容量的 90% 时，dlmon 事件监视器将自动关闭。PCTDEACTIVATE 子句只能用于 DMS 表空间。如果目标表空间已启用自动调整大小，那么将 PCTDEACTIVATE 子句设置为 100。

9. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器（WLM 事件监视器例外）。
   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                         AUTOSTART NONBLOCKED
     ```

   - 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                         MANUALSTART
     ```

10. 要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

一旦创建并激活表事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

事件监视器表管理

可定义事件监视器以将其事件记录存储在 SQL 表中。为此，将 CREATE EVENT MONITOR 语句与 WRITE TO TABLE 子句配合使用。

在创建“写至表”事件监视器时，数据库将创建*目标表*以存储返回数据的每个逻辑数据组的记录。在缺省情况下，数据库将在事件监视器创建者模式下创建表，并且根据相应的逻辑数据组和事件监视器名称命名这些表。在每个表中，列名必须与它们表示的监视元素名称相匹配。

例如，用户 riihi 将创建捕获 STATEMENTS 事件的事件监视器:

```
CREATE EVENT MONITOR foo FOR STATEMENTS WRITE TO TABLE
```

使用 STATEMENTS 事件类型的事件监视器从 event_connheader、event_stmt 和 event_subsection 逻辑数据组收集数据。数据库创建了下列表:

- riihi.connheader_foo
- riihi.stmt_foo
- riihi.subsection_foo
- riihi.control_foo

除了提供特定于各个事件类型的逻辑数据组的表之外，还将为每个“写至表”事件监视器创建控制表。它在上文用 riihi.control_foo 表示。具体地说，控制表包含来自 event_start、event_db_header（仅适用于 **conn_time** 监视元素）和 event_overflow 逻辑数据组的事件监视器元数据。

每当“写至表”事件监视器激活时，它都会获取每个目标表上的 IN 表锁定，以避免在事件监视器处于活动状态时修改了这些表锁定。在事件监视器处于活动状态时会维护所有表上的表锁定。如果任何目标表上需要互斥存取（例如，在将运行实用程序时），那么会在尝试这些存取前首先停用事件监视器以释放表锁定。

目标表中的每个列名与一个事件监视元素标识相匹配。将忽略没有相应目标表列的事件监视元素。

必须以手动方式来修剪“写至表”事件监视器的目标表，其中包括非格式化事件（UE）表。在高可用系统上，因为事件监视器记录的数据的高容量，事件监视器会迅速填满机器空间。与写至文件或命名管道的事件监视器不同，可将“写至表”事件监视器定义为仅记录特定逻辑数据组或监视元素。此功能允许您仅收集需要的相关数据，从而降低事件监视器生成的数据量。例如，以下语句定义用于捕获 TRANSACTIONS 事件的事件监视器，但它仅从 event_xact 逻辑数据组进行捕获，并且仅包括 **lock_escal** 监视元素:

```
CREATE EVENT MONITOR foo_lite FOR TRANSACTIONS WRITE TO TABLE
                      XACT(INCLUDES(lock_escal))
```

注: 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

存在这样的情况: 让事件监视器的目标表使用缺省表名称驻留在缺省表空间的缺省模式中不太理想。例如，如果预期会有大量监视数据时，您可能想要目标表在它们自己的表空间中。

可在 CREATE EVENT MONITOR 语句中指定模式、表和表空间名。模式名与表名一起提供，形成表的派生名称。

一个目标表只能被一个事件监视器使用。如果发现已经对另一个事件监视器定义了目标表，或者如果因为任何其他原因而不能创建目标表，那么 CREATE EVENT MONITOR 语句会失败。

可使用可选 IN 子句在表名后添加表空间名。与 DB2 自动创建的目标表不同，如果事件监视器定义中包含某个表空间，那么该表空间必须已存在。如果没有指定表空间，那么将指定定义者对其具有 USE 特权的表空间。

在分区数据库环境中，“写至表”事件监视器只有在包含该事件监视器的表空间所在的数据库分区上才处于活动状态。当活动事件监视器的目标表空间在特定数据库分区上不存在时，在该数据库分区上将停用此事件监视器，同时会将错误写至 db2diag 日志文件。

为了在检索事件监视器数据时提高性能，可为事件表创建索引。您也可以添加其他表属性，如触发器、关系完整性和约束。事件监视器会将其忽略。

例如，以下语句使用 event_connheader、event_stmt 和 event_subsection 逻辑数据组以定义用于捕获 STATEMENTS 事件的事件监视器。三个目标表中的每一个都有不同的模式、表和表空间组合:

```
CREATE EVENT MONITOR foo FOR STATEMENTS
WRITE TO TABLE CONNHEADER,
STMT (TABLE mydept.statements),
SUBSECTION (TABLE subsections, IN mytablespace)
```

假定上述语句由用户“riihi”发出，那么目标表的派生名称和表空间如下所示:

- CONNHEADER: riihi.connheader_foo（在缺省表空间上）
- STMT: mydept.statements（在缺省表空间上）
- SUBSECTION: riihi.subsections（在 MYTABLESPACE 表空间上）

如果事件监视器激活时目标表不存在，那么激活将继续进行，并且会忽略本来会插入至目标表的数据。相应的，如果监视元素在目标表中没有专用的列，就会被忽略。

对于活动的“写至表”事件监视器，可能会有存储事件记录的表空间用完容量的风险。为避免 DMS 表空间出现此风险，可定义表空间容量达到某个百分比时事件监视器将会释放。可在 CREATE EVENT MONITOR 语句的 PCTDEACTIVATE 子句中进行声明。

对于 SMS 表空间，此值设置为 100。建议在目标表空间启用了自动调整大小时，将 PCTDEACTIVATE 设置为 100。

在非分区数据库环境中，所有“写至表”事件监视器都会在最后一个应用程序终止（并且数据库未显式激活）时释放。在分区数据库环境中，“写至表”事件监视器将在目录分区释放时释放。

下表提供缺省目标表名，这些表名按对其返回表名的事件类型排序。

表 35. “写至表”事件监视器目标表

事件类型	目标表名	可用信息
DEADLOCKS[1]	CONNHEADER	连接元数据
	DEADLOCK	死锁数据
	DLCONN	死锁涉及的应用程序和锁定
	CONTROL	事件监视器元数据

表 35. “写至表”事件监视器目标表 (续)

事件类型	目标表名	可用信息
DEADLOCKS WITH DETAILS[1]	CONNHEADER	连接元数据
	DEADLOCK	死锁数据
	DLCONN	死锁涉及的应用程序
	DLLOCK	死锁涉及的锁定
	CONTROL	事件监视器元数据
DEADLOCKS WITH DETAILS HISTORY[1]	CONNHEADER	连接元数据
	DEADLOCK	死锁数据
	DLCONN	死锁涉及的应用程序
	DLLOCK	死锁涉及的锁定
	STMTHIST	工作单元中的先前语句列表
	CONTROL	事件监视器元数据
DEADLOCKS WITH DETAILS HISTORY VALUES[1]	CONNHEADER	连接元数据
	DEADLOCK	死锁数据
	DLCONN	死锁涉及的应用程序
	DLLOCK	死锁涉及的锁定
	STMTHIST	工作单元中的先前语句列表
	STMTVALS	STMTHIST 表中的语句的输入数据值
	CONTROL	事件监视器元数据
STATEMENT	CONNHEADER	连接元数据
	STMT	语句数据
	SUBSECTION	特定于子节的语句数据
	CONTROL	事件监视器元数据
TRANSACTIONS[2]	CONNHEADER	连接元数据
	XACT	事务数据
	CONTROL	事件监视器元数据
CONNECTIONS	CONNHEADER	连接元数据
	CONN	连接数据
	CONTROL	事件监视器元数据
	CONNMEMUSE	内存池元数据
DATABASE	DB	数据库管理器数据
	CONTROL	事件监视器元数据
	DBMEMUSE	内存池元数据
BUFFERPOOLS	BUFFERPOOL	缓冲池数据
	CONTROL	事件监视器元数据
TABLESPACES	TABLESPACE	表空间数据
	CONTROL	事件监视器元数据
TABLES	TABLE	表数据
	CONTROL	事件监视器元数据

表 35. “写至表”事件监视器目标表 (续)

事件类型	目标表名	可用信息
ACTIVITIES	ACTIVITY	完成执行的或执行过程中捕获的活动。
	ACTIVITYSTMT	语句活动的语句信息。
	ACTIVITYVALS	具有输入数据的活动的输入数据值。可报告的数据类型不包 括： CLOB、REF、BOOLEAN、STRUCT、DATALINK、 LONG VARGRAPHIC、LONG、XMLLOB 和 DBCLOB。
	CONTROL	事件监视器元数据
STATISTICS	SCSTATS	从在系统中每个服务类、工作类或工作负载内执行的活动计算而来的统计信息。
	WCSTATS	
	WLSTATS	
	HISTOGRAMBIN	
	QSTATS	
	CONTROL	事件监视器元数据
THRESHOLD VIOLATIONS	THRESHOLDVIOLATIONS	违例的阈值及违例时间的列表。
	CONTROL	事件监视器元数据

[1] 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

[2] 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

未对“写至表”事件监视器收集下列逻辑数据组:

- event_log_stream_header
- event_log_header
- event_dbheader（仅收集 **conn_time** 监视元素）

事件监视器表中的每一列的数据类型对应于该列表示的监视元素的数据类型。以下是一组数据类型映射，这些映射使监视元素（在 sqlmon.h 中）的原始系统监视器数据类型与表列的 SQL 数据类型相对应。

表 36. 系统监视器数据类型映射

系统监视器数据类型	SQL 数据类型
SQLM_TYPE_STRING	CHAR[n]、VARCHAR[n] 或 CLOB[n]
SQLM_TYPE_U8BIT 和 SQLM_TYPE_8BIT	SMALLINT、INTEGER 或 BIGINT
SQLM_TYPE_U16BIT 和 SQLM_TYPE_16BIT	SMALLINT、INTEGER 或 BIGINT
SQLM_TYPE_U32BIT 和 SQLM_TYPE_32BIT	INTEGER 或 BIGINT
SQLM_TYPE_U64BIT 和 SQLM_TYPE_64BIT	BIGINT
SQLM_TIMESTAMP	TIMESTAMP
SQLM_TIME	BIGINT
SQLCA: SQLERRMC	VARCHAR[72]
SQLCA: SQLSTATE	CHAR[5]

表 36. 系统监视器数据类型映射 (续)

系统监视器数据类型	SQL 数据类型
SQLCA: SQLWARN	CHAR[11]
SQLCA: 其他字段	INTEGER 或 BIGINT
SQLM_TYPE_HANDLE	BLOB[n]

注:

1. 并非所有列都是空值。
2. 因为带有 CLOB 列的表的性能低于带有 VARCHAR 列的表，所以在指定 stmt evmGroup（或在使用带有详细信息的死锁时指定 dlconn evmGroup）时考虑使用 TRUNC 关键字。
3. SQLM_TYPE_HANDLE 用于表示编译环境句柄对象。

创建文件事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。文件事件监视器将事件记录存储在文件中。文件事件监视器及其选项将由 CREATE EVENT MONITOR 语句定义。

您需要 SQLADM 或 DBADM 权限才能创建文件事件监视器。

文件事件监视器将事件记录流输送至一系列 8 字符编号的文件，这些文件的扩展名为 "EVT"（如 00000000.evt、00000001.evt 和 00000002.evt）。即使数据被分为较小的数据块，也应考虑将数据作为一个逻辑文件（即，数据流的开头是文件 00000000.evt 中的第一个字节；数据流的结尾是文件 nnnnnnnn.evt 的最后一个字节）。事件监视器永远不会让单个事件记录跨越两个文件。

1. 指定会将事件监视器数据收集在一个文件或一组文件中，并提供存储事件文件的目录位置。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO FILE '/tmp/dlevents'
   ```

   dlmon 是事件监视器的名称。

   /tmp/dlevents 是目录路径（在 UNIX 上）的名称，事件监视器会将事件文件写至该目录。

2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents'
   ```

   此事件监视器将监视 CONNECTIONS 和 DEADLOCKS WITH DETAILS 事件类型。

3. 通过调整 BUFFERSIZE 值来指定文件事件监视器缓冲区的大小（以 4K 页计）：

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
   ```

   8 是两个事件文件缓冲区的容量（以 4K 页计）。

   每个缓冲区的缺省大小为 4 页（分配两个 16K 缓冲区）。最小大小为 1 页。由于缓冲区是根据监视器堆进行分配，所以缓冲区的最大大小受该监视器堆大小限制。由于性能原因，高可用事件监视器的缓冲区应该比不活动事件监视器的缓冲区大。

4. 指示是需要分块事件监视器还是非分块事件监视器。对于分块事件监视器，如果事件缓冲区已满，那么生成事件的每个代理程序将等待事件缓冲区写至文件。这可能会导致数据库性能降低，原因是暂挂的代理程序和所有从属代理程序在缓冲区清空之前不能运行。使用 BLOCKED 子句来确保事件数据不会丢失:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    BLOCKED
```

缺省情况下，事件监视器处于受阻状态。如果数据库性能比收集每个事件记录更重要，那么使用非分块事件监视器。在此情况下，如果事件缓冲区已满，那么生成事件的每个代理程序将不会等待事件缓冲区写至文件。因此，非分块事件监视器可能会导致活动频繁的系统上的数据丢失。使用 NONBLOCKED 子句来使事件监视的性能开销降至最低:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED
```

5. 指定可对每个事件监视器收集的事件文件的最大数目。如果达到此限制，那么事件监视器将释放自身。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED MAXFILES 5
```

5 是将要创建的事件文件的最大数目。

还可指定不限制事件监视器可创建的事件文件数:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED MAXFILES NONE
```

6. 指定事件监视器创建的每个事件文件的最大大小（以 4K 页计）。如果达到此限制，那么创建新文件。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED MAXFILES 5 MAXFILESIZE 32
```

32 是事件文件可包含的最大 4K 页数。

此值必须大于 BUFFERSIZE 参数指定的值。还可指定不限制事件文件的大小:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED MAXFILES NONE MAXFILESIZE NONE
```

7. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器（WLM 事件监视器例外）。
   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                         NONBLOCKED AUTOSTART
     ```

   - 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                         NONBLOCKED MANUALSTART
     ```

8. 要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

一旦创建并激活文件事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

事件监视器文件管理

文件事件监视器允许事件监视器将其事件记录存储在文件中。事件监视器的所有输出将存储在 CREATE EVENT MONITOR 语句的 FILE 参数提供的目录中。如果此目录不存在，那么 DB2 将不会创建该目录。在激活监视器之前，该目录必须存在，否则 SET EVENT MONITOR 命令将返回错误。当第一次激活文件事件监视器时，将在此目录中创建控制文件 db2event.ctl。不要除去或修改此文件。

在缺省情况下，事件监视器会将跟踪写至称为 00000000.evt 的单个文件。只要文件系统上还有空间，此文件就会一直增长。如果使用 CREATE EVENT MONITOR 语句的 MAXFILESIZE 参数指定了文件大小限制，那么在文件已满时，输出将自动导向下一个文件。因此，活动文件就是具有最高编号的文件。

还可通过使用 CREATE EVENT MONITOR 语句的 MAXFILES 参数来限制整个事件监视器跟踪的最大大小。当文件数达到 MAXFILES 定义的最大数目时，事件监视器将释放自身，并且会向管理通知日志写入以下消息。

```
DIA1601I 事件监视器 monitor-name 在达到预设的 MAXFILES 和 MAXFILESIZE 限制时释放。
```

可通过除去已满的文件来避免出现此情况。当事件监视器仍在运行时，可以除去活动文件之外的任何事件文件。

如果文件事件监视器用完了磁盘空间，那么在管理通知日志中记录系统错误级别消息后，它会关闭自身。

重新启动文件事件监视器时，它会擦除任何现有数据或追加的新数据。在 CREATE EVENT MONITOR 语句中指定此选项，并可创建 APPEND 监视器或 REPLACE 监视器。APPEND 是缺省选项。APPEND 事件监视器开始在上次使用的文件结尾写入。如果已除去该文件，那么会使用下一个顺序文件编号。重新启动追加事件监视器时，仅生成 start_event。仅对第一次激活生成事件日志头和数据库头。替换（REPLACE）事件监视器总是会删除现有事件文件并开始在 00000000.evt 中写入。

您可能想要在事件监视器活动时处理监视器数据。这是可行的，而且在处理完文件后可以删除它，从而释放空间以供后续监视数据使用。除非停止并重新启动事件监视器，否则不能强制事件监视器切换至下一个文件。它必须也处于 APPEND 方式。为记录在活动文件中处理的事件，可创建一个应用程序，它只记录已处理的上一个记录的文件编号和位置。下一次处理跟踪时，该应用程序只需搜索该文件的位置。

“写至表”和文件事件监视器缓存

对于写表和文件事件监视器而言，事件监视器进程在将它的记录写入文件或表之前，将对这些记录进行缓存。缓冲区满时，将自动写记录。因此，通过指定较大的缓冲区以减少磁盘访问次数，可以改善高吞吐量事件监视器的监视性能。为了强制事件监视器清空其缓冲区，必须释放该事件监视器，或者使用 FLUSH EVENT MONITOR 语句来清空缓冲区。

对于受阻事件监视器来说，当它的两个缓冲区都变满时，它将暂挂正在发送监视器数据的数据库进程。这是为了确保在受阻事件监视器活动期间不会删除任何事件记录。

在将缓冲区内容写入文件或表之前，暂挂的数据库进程以及任何从属数据库进程都无法运行。根据工作负载类型以及 I/O 设备速度的不同，这会对性能产生严重影响。缺省情况下，事件监视器处于受阻状态。

对于非受阻事件监视器来说，如果从代理程序接收监视器数据的速度高于事件监视器写数据的速度，它将删除那些数据。这样就可以避免事件监视操作影响其他数据库活动的性能。

删除了事件记录的事件监视器会生成溢出事件。此事件指定了监视器删除事件的开始时间和停止时间以及在该时间段内删除的事件数。事件监视器有可能对要报告的暂挂溢出终止或释放。如果发生这种情况，就会将以下消息写入管理日志:

```
DIA2503I 事件监视器 monitor-name 在释放时有暂挂的溢出记录。
```

各个事件记录也会发生丢失事件监视数据的情况。如果事件记录长度超过事件缓冲区大小，在缓冲区中装不下的数据就会被截断。例如，如果正在捕获 stmt_text 监视元素并且连接到被监视数据库的应用程序发出了很长的 SQL 语句，就会发生数据截断的情况。如果需要捕获所有事件记录信息，请指定较大的缓冲区。记住，较大的缓冲区会降低写入文件或表的频率。

创建管道事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。管道事件监视器直接将事件记录从事件监视器传输至命名管道。

您将需要 SQLADM 或 DBADM 权限来创建管道事件监视器。

在事件监视器写入事件数据时，将由监视应用程序迅速读取管道中的数据。如果事件监视器无法将数据写至管道（例如，因为管道已满），那么监视器数据将会丢失。

管道事件监视器将使用 CREATE EVENT MONITOR 语句定义。

1. 指示会将事件监视器数据引导至命名管道。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                                          WRITE TO PIPE '/home/riihi/dlevents'
   ```

   dlmon 是事件监视器的名称。

   /home/riihi/dlevents 是命名管道（在 UNIX 上）的名称，事件监视器会将事件记录引导至该命名管道。CREATE EVENT MONITOR 语句支持 UNIX 和 Windows 管道命名语法。

   激活事件监视器时，CREATE EVENT MONITOR 语句中指定的命名管道必须存在并且打开。如果指定事件监视器将自动启动，那么在创建事件监视器之前命名管道必须存在。

2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                                          WRITE TO PIPE '/home/riihi/dlevents'
   ```

   此事件监视器将监视 CONNECTIONS 和 DEADLOCKS WITH DETAILS 事件类型。

3. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器。
   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                                        WRITE TO PIPE '/home/riihi/dlevents'
                                        AUTOSTART
```

- 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                                        WRITE TO PIPE '/home/riihi/dlevents'
                                        MANUALSTART
```

4. 要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

一旦创建并激活管道事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

事件监视器命名管道管理

管道事件监视器允许通过命名管道处理事件监视器数据流。如果需要实时处理事件记录，那么最好使用管道事件监视器。另一个很重要的优点是，应用程序在读完管道时可以忽略不想要的数据，从而可以大幅降低存储器要求。

在 AIX 上，可使用 mkfifo 命令来创建命名管道。在 Linux 和其他 UNIX 类型（如 Solaris 操作系统）上，使用 pipe() 例程。在 Windows 上，可通过使用 CreateNamedPipe() 例程来创建命名管道。

将数据引导至管道时，I/O 总是会分块并且唯一的缓冲将由管道执行。在事件监视器写入事件数据时，将由监视应用程序迅速读取管道中的数据。如果事件监视器无法将数据写至管道（例如，因为管道已满），那么监视器数据将会丢失。

此外，命名管道中必须有足够的空间用来处理入局事件记录。如果应用程序从命名管道读取数据时不够快，管道将填满并溢出。管道缓冲区越小，溢出的机率越大。

当发生管道溢出时，监视器将创建溢出事件记录以指示发生溢出。事件监视器不会关闭，但监视器数据会丢失。如果释放监视器时存在明显的溢出事件记录，那么会记录诊断消息。否则，溢出事件记录可能时将写至管道。

如果操作系统允许定义管道缓冲区的大小，那么使用的管道缓冲区至少应该为 32K。对于高容量事件监视器，应将监视应用程序的进程优先级设置为等于或高于代理进程优先级。

为分区数据库创建事件监视器

在分区数据库系统上创建文件或管道事件监视器时，需要确定想要收集的监视数据的作用域。

开始前

您将需要 SQLADM 或 DBADM 权限来为分区数据库创建事件监视器。

关于此任务

事件监视器使用操作系统进程或线程来写入事件记录。运行此进程或线程的数据库分区称为监视器分区。如果文件和管道事件监视器在监视器分区上以局部方式运行，或者在运行 DB2 数据库管理器的任何分区上以全局方式运行，那么这些事件监视器可充当监视事件。全局事件监视器将在监视分区上写入单个跟踪，它包含所有分区的活动。不管事件监视器是局部的还是全局的，都被称为监视作用域。

监视器分区和监视器作用域都将使用 CREATE EVENT MONITOR 语句指定。

仅当监视器分区处于活动状态时，才能激活事件监视器。如果 SET EVENT MONITOR 语句用于激活事件监视器但监视器分区尚未处于活动状态，那么将在下一次启动监视器分区时激活事件监视器。而且，在显式释放事件监视器或显式释放实例之后，将自动激活事件监视器。例如，在数据库分区 0 上:

```
   db2 connect to sample
db2 create event monitor foo ... on dbpartitionnum 2
db2 set event monitor foo state 1
```

在运行上述命令后，每次数据库 sample 在数据库分区 2 上激活时都将自动激活事件监视器 foo。发出 db2 set event monitor foo state 0 或停止分区 2 之前，会一直进行此自动激活。

局部或全局作用域的注释不适用于写至表事件监视器。激活写至表事件监视器时，事件监视器在所有分区上运行。（更具体地说，事件监视器进程将在属于数据库分区组的分区上运行，而目标表位于这些数据库分区组上。）运行事件监视器进程的每个分区具有同一组目标表。因为是从各个分区的角度表示监视数据，所以这些表中的数据将会有所不同。可通过发出 SQL 语句来访问每个分区的事件监视器目标表中的期望值，以获取来自所有分区的聚集值。

每个目标表的第一列称为 PARTITION_KEY，并且被用作表的分区键。此列的值将被选中，以便每个事件监视器进程将数据插入到运行该进程的数据库分区中；即，插入操作将在运行事件监视器进程的数据库分区本地执行。在任何数据库分区上，PARTITION_KEY 字段都将包含相同的值。这意味着，如果删除数据分区并且重新分布数据，那么被删除数据库分区上的所有数据都将转至另一数据库分区而不是平均分布。因此，在除去数据库分区之前，考虑删除该数据库分区上的所有表行。

此外，可对每个表定义名为 PARTITION_NUMBER 的列。此列包含插入数据的分区的编号。与 PARTITION_KEY 列不同，PARTITION_NUMBER 列不是必需的。

用来定义目标表的表空间必须在所有分区上存在，事件监视器数据将写至这些分区。如果不遵守此规则，那么不会使用事件监视器将记录写至不存在表空间的登录分区。事件仍将写至存在表空间的分区，并且不返回任何错误。此行为允许用户通过创建只在特定分区上存在的表空间，以选择要监视的分区子集。

在写至表事件监视器激活期间，FIRST_CONNECT 和 EVMON_START 的 CONTROL 表行将仅插入至目录数据库分区。这要求目录数据库分区上存在控制表的表空间。如果目录数据库分区上不存在该表空间，那么不会执行这些插入。

如果激活写至表事件监视器时分区尚未激活，那么将在下一次激活该分区时激活事件监视器。

如果添加数据库分区，并且该数据库分区在添加后立即进入联机状态，那么事件监视器将不会立即知道这个新分区。要收集并记录关于新分区的数据，您必须执行下列其中一项操作:

- 对于全局事件监视器，请重新启动事件监视器。
- 对于“写至表”事件监视器，请依次删除、重新创建和重新启动事件监视器。

.

注: 详细的死锁连接事件中的锁定列表将仅包含这样的锁定，这些锁定是应用程序在等待死锁的分区上挂起的。例如，如果涉及死锁的应用程序正在节点 20 上等待锁定，那么只有该应用程序在节点 20 上挂起的锁定才会包括在列表中。

过程

1. 指定要监视的分区。

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                       WRITE TO FILE '/tmp/dlevents'
                       ON PARTITION 3
   ```

   dlmon 表示事件监视器的名称。

   /tmp/dlevents 是目录路径（在 UNIX 上）的名称，事件监视器会将事件文件写至该目录。

   3 表示要监视的分区号。

2. 指定是在局部作用域还是全局作用域收集事件监视器数据。要从所有分区收集事件监视器报告，那么发出以下语句:

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                       WRITE TO FILE '/tmp/dlevents'
                       ON PARTITION 3 GLOBAL
   ```

   只有死锁和与带有详细信息事件监视器的死锁才能定义为 GLOBAL。所有分区会将与死锁有关的事件记录报告至分区 3。

3. 要仅从局部分区收集事件监视器报告，那么发出以下语句:

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                       WRITE TO FILE '/tmp/dlevents'
                       ON PARTITION 3 LOCAL
   ```

   这是分区数据库中的文件和管道事件监视器的缺省行为。对于写至表事件监视器，将忽略 LOCAL 和 GLOBAL 子句。

4. 可复查现有事件监视器的监视器分区和作用域值。为此，请使用以下语句查询 SYSCAT.EVENTMONITORS 表:

   ```
   SELECT EVMONNAME, NODENUM, MONSCOPE FROM SYSCAT.EVENTMONITORS
   ```

结果

一旦创建并激活事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

事件监视器样本输出

从命令行格式化文件或管道事件监视器输出

文件或管道事件监视器的输出是一个逻辑数据分组二进制流。可使用 db2evmon 命令从命令行格式化此数据流。此高效工具从事件监视器的文件或管道读取事件记录，然后将它们写至屏幕（标准输出）。

除非连接至数据库，否则不需要任何权限，如果连接至数据库，那么需要具有下列其中一个权限:

- SYSADM
- SYSCTRL

- SYSMAINT
- DBADM

可通过提供事件文件的路径或提供数据库名称和事件监视器名称，以指示想要格式化的事件监视器输出。要格式化事件监视器输出:

- 指定包含事件监视器文件的目录:

  ```
  db2evmon -path '/tmp/dlevents'
  ```

  /tmp/dlevents 表示（UNIX）路径。
- 指定数据库和事件监视器名称:

  ```
  db2evmon -db 'sample' -evm 'dlmon'
  ```

  sample 表示事件监视器所属的数据库。

  dlmon 表示事件监视器。

事件记录及其相应的应用程序

在带有几百个相连的应用程序的活动数据库的事件跟踪中，跟踪与特定应用程序相关联的事件记录可能会非常麻烦。为提高可跟踪性，每个事件记录包括应用程序句柄和应用程序标识。它们允许您将每个记录与对其生成事件记录的应用程序相关联。

在应用程序有效期间，应用程序句柄（**agent_id**）在系统范围内是唯一的。但是，最终它将被重复使用（将使用 16 位计数器在分区数据库系统上生成此标识，它由协调分区号和 16 位计数器组成）。在大多数情况下，重复使用不存在问题，原因是从跟踪读取记录的应用程序能够检测到已终止的连接。例如，在跟踪中遇到带有已知 agent_ID 的连接头指示带有此 agent_ID 的先前连接已终止。

应用程序标识是一个字符串标识，它包括时间戳记，并且承诺即使在停止并重新启动数据库管理器之后也一直是唯一的。

通过使用写至表事件监视器，查找特定应用程序的事件记录变得特别容易。在事件监视器表中，每行对应一个事件记录，应用程序句柄和应用程序标识是缺省列值。要查找给定应用程序的所有事件记录，只要对与特定应用程序标识对应的所有事件记录发出 SQL SELECT 语句就可以了。

事件监视器自描述数据流

事件监视器的输出是一个逻辑数据分组二进制流，这些分组对于管道和文件事件监视器是完全一样的。可通过使用 db2evmon 命令或开发客户机应用程序来格式化数据流。此数据流显示为自描述格式。第 69 页的图 1 显示数据流的结构，而 第 69 页的表 37 提供可能返回的逻辑数据组和监视元素的一些示例。

注: 在示例和表中，将对标识使用描述性名称。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，**db_event** 在事件监视器输出中显示为 SQLM_ELM_DB_EVENT。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。

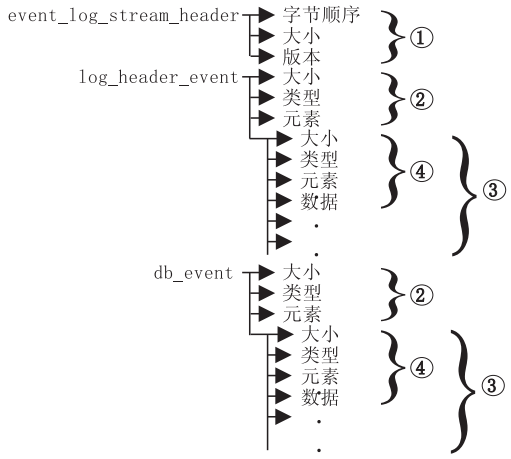


图 1. 事件监视器数据流

1. sqlm_event_log_data_stream_header 的结构与数据流中的其他头不同。版本字段确定是否能作为自描述数据流来处理输出。

   此头的大小和类型与版本 6 之前的事件监视器流的大小和类型相同。这允许应用程序确定事件监视器输出是自描述格式还是版本 6 之前的静态格式。

   注: 将通过从数据流读取 sizeof(sqlm_event_log_data_stream) 字节来抽取此监视元素。
2. 每个逻辑数据组以指示其大小和元素名称的头开始。因为大小元素包含哑元值以保留向后兼容性，所以这不适用于 event_log_stream_header。
3. 头中的大小元素指示该逻辑数据组中的所有数据的大小。
4. 监视元素信息在逻辑数据组头后面，并且也是自描述格式。

表 37. 样本事件数据流

逻辑数据组	数据流	描述
event_log_stream_header		
	sqlm_little_endian	未使用（因为与前发行版的兼容性）。
	200	未使用（因为与前发行版的兼容性）。
	sqlm_dbmon_version9_5	返回数据的数据库管理器的版本。事件监视器以自描述格式写入数据。

表 37. 样本事件数据流 (续)

逻辑数据组	数据流	描述
log_header_event		
	100	逻辑数据组的大小。
	header	指示逻辑数据组的开头。
	log_header	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 32 位数字。
	byte_order	收集的监视元素的名称。
	little_endian	此元素的已收集值。
	2	此监视元素中存储的数据的大小。
	u16bit	监视元素类型 - 不带符号的 16 位数字。
	codepage_id	收集的监视元素的名称。
	850	此元素的已收集值。
db_event		
	100	逻辑数据组的大小。
	header	指示逻辑数据组的开头。
	db_event	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	lock_waits	收集的监视元素的名称。
	2	此元素的已收集值。

event_log_stream_header 标识返回数据的数据库管理器的版本。事件监视器以自描述格式写入它们的数据。与快照监视器不同，事件监视器没有用来返回跟踪总大小的**大小**元素。event_log_stream_header 中显示的数字是为向后兼容性提供的哑元值。写入 event_log_stream_header 时，事件跟踪的总大小未知。通常可读取事件监视器跟踪直到文件或管道末尾。

日志头描述跟踪的特征，它包含各种信息，如在其中收集跟踪的服务器的内存模型（如小尾数法）和数据库的代码页。如果在其中读取跟踪的系统的内存模型与服务器的内存模型不同（例如，如果在 Windows 2000 系统上从 UNIX 服务器读取跟踪），那么您可能必须对数字值执行字节交换。如果配置数据库时使用的语言与在其中读取跟踪的机器使用的语言不同，那么可能还需要进行代码页转换。读取跟踪时，可使用**大小**元素来跳过跟踪中的逻辑数据组。

在系统之间传送事件监视器数据

对于在存储数字值时使用不同约定的系统，在系统间传送事件监视器信息时必须进行转换。UNIX 平台上的信息以小尾数法字节顺序存储，而 Windows 平台上的信息以大尾数法字节顺序存储。如果要在大尾数法平台上读取小尾数法源中的事件监视器数据（或反之），那么需要进行字节转换。

1. 要在逻辑数据组头和监视元素中转换数字值，请使用以下逻辑（用 C 呈示）：

```
#include sqlmon.h
#define SWAP2(s) ((((s) >> 8) &0xFF) | (((s) << 8) &0xFF00))

#define SWAP4(l) ((((l) >> 24) &0xFF) | ((((l) &0xFF0000) >> 8) &0xFF00) \
                      | (((l) &0xFF00) << 8) | ((l) << 24))
```

```
#define SWAP8( where )                                                    \
 {                                                                        \
        sqluint32 temp;                                                       \
        temp = SWAP4(*(sqluint32 *) (where));                                 \
        * (sqluint32 *) (where) = SWAP4(* (((sqluint32 *) (where)) + 1));     \
        * (((sqluint32 *) (where)) + 1) = temp;                               \
 }

int HeaderByteReverse( sqlm_header_info * pHeader)
{   int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

        return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{   int rc = 0;

        sqlm_header_info * pElemHeader = NULL;
        char * pElemData = NULL;
        sqluint32 dataOffset = 0;
        sqluint32 elemDataSize  = 0;
        sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the data stream that are numeric,
        // perform byte reversal.

        while( dataOffset < dataSize)
    {   /* byte reverse the element header */
                pElemHeader = (sqlm_header_info *)
                      ( dataBuf + dataOffset);

                rc = HeaderByteReverse( pElemHeader);
                    if( rc != 0) return rc;
                // Remember the element data's size...it will be byte reversed
                // before we skip to the next element.
        elemDataSize = pElemHeader->size;

                /* byte reverse the element data */
                pElemData = (char *)
                      ( dataBuf + dataOffset + elemHeaderSize);

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {   rc = DataByteReverse( pElemData, pElemHeader->size);
                      if( rc != 0) return rc;
        }
                else
 {   switch( pElemHeader->type)
           {   case SQLM_TYPE_16BIT:
                                case SQLM_TYPE_U16BIT:
                                   *(sqluint16 *)(pElemData) =
                                SWAP2(*(short *)(pElemData));
                                   break;
                                case SQLM_TYPE_32BIT:
                                case SQLM_TYPE_U32BIT:
                                   *(sqluint32 *)(pElemData) =
                                SWAP4(*(sqluint32 *)(pElemData));
                                   break;
                                case SQLM_TYPE_64BIT:
                                case SQLM_TYPE_U64BIT:
                                   SWAP8(pElemData);
                                   break;
                                default:
```

```
                    // Not a numeric type. Do nothing.
                                        break;
                }
            }
                    dataOffset = dataOffset + elemHeaderSize + elemDataSize;
        }

            return 0;
    } /*  end of DataByteReverse */
```

2. 要在逻辑数据组头和监视元素中转换数字值，请使用以下逻辑（用 C 呈示）：

```
#include sqlmon.h
#define SWAP2(s) ((((s) >> 8) &0xFF) | (((s) << 8) &0xFF00))

#define SWAP4(l) ((((l) >> 24) &0xFF) | ((((l) &0xFF0000) >> 8) &0xFF00) \
                      | (((l) &0xFF00) << 8) | ((l) << 24))

#define SWAP8( where )                                                    \
 {                                                                        \
        sqluint32 temp;                                                       \
        temp = SWAP4(*(sqluint32 *) (where));                                 \
        * (sqluint32 *) (where) = SWAP4(* (((sqluint32 *) (where)) + 1));     \
        * (((sqluint32 *) (where)) + 1) = temp;                               \
 }

int HeaderByteReverse( sqlm_header_info * pHeader)
{   int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

        return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{   int rc = 0;

        sqlm_header_info * pElemHeader = NULL;
        char * pElemData = NULL;
        sqluint32 dataOffset = 0;
        sqluint32 elemDataSize  = 0;
        sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the data stream that are numeric,
        // perform byte reversal.

        while( dataOffset < dataSize)
    {   /* byte reverse the element header */
                pElemHeader = (sqlm_header_info *)
                      ( dataBuf + dataOffset);

                rc = HeaderByteReverse( pElemHeader);
                    if( rc != 0) return rc;
                // Remember the element data's size...it will be byte reversed
                // before we skip to the next element.
        elemDataSize = pElemHeader->size;

                /* byte reverse the element data */
                pElemData = (char *)
                      ( dataBuf + dataOffset + elemHeaderSize);

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {   rc = DataByteReverse( pElemData, pElemHeader->size);
                    if( rc != 0) return rc;
        }
                else
```

```
{   switch( pElemHeader->type)
          {   case SQLM_TYPE_16BIT:
                                  case SQLM_TYPE_U16BIT:
                                         *(sqluint16 *)(pElemData) =
                                      SWAP2(*(short *)(pElemData));
                                         break;
                                  case SQLM_TYPE_32BIT:
                                  case SQLM_TYPE_U32BIT:
                                         *(sqluint32 *)(pElemData) =
                                      SWAP4(*(sqluint32 *)(pElemData));
                                         break;
                                  case SQLM_TYPE_64BIT:
                                  case SQLM_TYPE_U64BIT:
                                         SWAP8(pElemData);
                                         break;
                                  default:
                      // Not a numeric type. Do nothing.
                                         break;
              }
         }
               dataOffset = dataOffset + elemHeaderSize + elemDataSize;
   }

        return 0;
} /*  end of DataByteReverse */
```

第 4 章 快照监视器

可使用快照监视器以在特定时间捕获有关数据库和所有已连接应用程序的信息。快照对于确定数据库系统的状态非常有用。如果每隔一定时间间隔获取快照，那么快照在观察趋势和预测潜在问题方面时也很有用。快照监视器中的某些数据获取自系统监视器。系统监视器中的可用数据由系统监视开关确定。

系统监视器仅累积数据库处于活动状态时的信息。如果所有应用程序与数据库断开连接并且数据库释放，那么不再提供该数据库的系统监视器数据。可通过使用 ACTIVATE DATABASE 命令启动数据库或保持与数据库的永久连接，以便让数据库保持活动状态直到获取最终快照。

快照监视需要实例连接。如果没有实例连接，那么创建缺省实例连接。通常，对于应用程序调用第一个数据库系统监视器 API 时，将隐式建立与 DB2INSTANCE 环境变量指定的实例的连接。还可使用 ATTACH TO 命令显式建立连接。一旦连接了应用程序，它调用的所有系统监视器请求都将引导至该实例。这允许客户机只需要连接至远程服务器上的实例就可以监视该远程服务器。

在分区数据库环境中，可在实例的任何分区上获取快照，或者使用单个实例连接获取全局快照。全局快照聚集在每个分区上收集到的数据并返回一组值。

可从 CLP、SQL 表函数捕获快照，也可以通过使用 C 或 C++ 应用程序中的快照监视器 API 来捕获快照。有若干不同快照请求类型可用，每种类型返回特定类型的监视数据。例如，可捕获仅返回缓冲池信息的快照，或者捕获返回数据库管理器信息的快照。在捕获快照前，请考虑是否需要由监视开关控制的监视元素提供的信息。如果特定监视开关处于关闭状态，就不会收集它控制的监视元素。

访问系统监视器数据: SYSMON 权限

作为 SYSMON 数据库管理器级别组的成员的用户有权获取对数据库系统监视器数据的访问权。系统监视器数据是通过使用快照监视器 API、CLP 命令或 SQL 表函数来访问的。

SYSMON 权限组替换 DB2_SNAPSHOT_NOAUTH 注册表变量以使没有系统管理或系统控制权限的用户能够访问数据库系统监视器。

除了 SYSMON 权限之外，访问使用快照监视器的系统监视器数据的唯一方法是使用系统管理或系统控制权限。

属于 SYSMON 组或具有系统管理或系统控制权限的任何用户可以执行下列快照监视器函数:

- CLP 命令:
  - GET DATABASE MANAGER MONITOR SWITCHES
  - GET MONITOR SWITCHES
  - GET SNAPSHOT
  - LIST ACTIVE DATABASES

  - LIST APPLICATIONS
  - LIST DCS APPLICATIONS
  - LIST UTILITIES
  - RESET MONITOR
  - UPDATE MONITOR SWITCHES
- API:
  - db2GetSnapshot - 获取快照
  - db2GetSnapshotSize - 估计 db2GetSnapshot() 输出缓冲区所需的大小
  - db2MonitorSwitches - 获取/更新监视开关
  - db2ResetMonitor - 复位监视器
- 无需预先运行 SYSPROC.SNAP_WRITE_FILE 的快照 SQL 表函数

使用快照管理视图和表函数来捕获数据库系统快照

授权用户可使用快照管理视图或快照表函数来捕获 DB2 实例的监视器信息快照。快照管理视图提供了一种简单的方法，可用来访问已连接数据库的所有数据库分区的数据。快照表函数允许您请求特定数据库分区的数据，全局聚集数据或所有数据库分区中的数据。某些快照表函数允许您请求所有活动数据库中的数据。

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能捕获数据库快照。要获取远程实例的快照，必须先连接至属于该实例的本地数据库。

虽然新的监视器数据可用时在将来发行版中可能需要新的快照表函数，但在新列添加至视图时快照管理视图集合将保持不变，使得长期进行应用程序维护时管理视图成为一个很好的选择。

每个快照视图返回一个表，每个数据库分区每个被监视对象对应一行，每列表示一个监视元素。每个表函数返回一个表，指定分区的每个被监视对象对应一行。返回的表的列名与监视元素名称相关。

例如，SAMPLE 数据库的一般应用程序信息的快照将使用如下 SNAPAPPL 管理视图进行捕获:

```
SELECT * FROM SYSIBMADM.SNAPAPPL
```

您也可以从返回的表中选择个别监视元素。例如，以下语句仅返回 **agent_id** 和 **appl_id** 监视元素:

```
SELECT agent_id, appl_id FROM SYSIBMADM.SNAPAPPL
```

快照管理视图和表函数不能与下列任一项一起使用:

- 监视开关命令/API
- 监视器复位命令/API

此限制包括:

- GET MONITOR SWITCHES
- UPDATE MONITOR SWITCHES
- RESET MONITOR

此限制是由于此类命令使用 INSTANCE ATTACH，而快照表函数使用 DATABASE CONNECT 造成的。

要使用快照管理视图捕获快照:

1. 要使用快照管理视图捕获快照:
   a. 连接到数据库。这可以是实例中任何需要监视的数据库。为了能够使用快照管理视图发出 SQL 查询，必须连接到数据库。
   b. 确定需要捕获的快照的类型。如果想要捕获当前连接的数据库之外的数据库的快照，或者如果想要从单个数据库分区或全局聚集数据中检索数据，那么需要改为使用快照表函数。
   c. 使用适当的快照管理视图发出查询。例如，以下查询将对当前连接的数据库捕获锁定信息的快照:

      ```
      SELECT * FROM SYSIBMADM.SNAPLOCK
      ```

2. 要使用快照表函数捕获快照:
   a. 连接到数据库。这可以是实例中任何需要监视的数据库。为了能够使用快照表函数发出 SQL 查询，必须连接到数据库。
   b. 确定需要捕获的快照的类型。
   c. 使用适当的快照表函数发出查询。例如，以下查询将对当前连接的数据库分区捕获有关 SAMPLE 数据库的锁定信息的快照:

      ```
      SELECT * FROM TABLE(SNAP_GET_LOCK('SAMPLE',-1)) AS SNAPLOCK
      ```

      SQL 表函数有两个输入参数:

      数据库名称
      VARCHAR(255)。如果输入 NULL，就会使用当前连接的数据库的名称。

      分区号 SMALLINT。对于数据库分区号参数，输入对应于需要监视的数据库分区号的整数（0 到 999 之间的值）。要捕获当前连接的数据库分区的快照，请输入值 -1。要捕获全局聚集快照，请输入值 -2。要从所有数据库分区捕获快照，不要对此参数指定值。

      注:

      1) 对于以下快照表函数列表，如果对当前连接的数据库输入 NULL，那么将获取实例中所有数据库的快照信息:
         - SNAP_GET_DB_V95
         - SNAP_GET_DB_MEMORY_POOL
         - SNAP_GET_DETAILLOG_V91
         - SNAP_GET_HADR
         - SNAP_GET_STORAGE_PATHS
         - SNAP_GET_APPL_V95
         - SNAP_GET_APPL_INFO_V95
         - SNAP_GET_AGENT
         - SNAP_GET_AGENT_MEMORY_POOL
         - SNAP_GET_STMT
         - SNAP_GET_SUBSECTION

- SNAP_GET_BP_V95
- SNAP_GET_BP_PART

2) 数据库名称参数不适用于数据库管理器级别快照表函数；那些表函数只有数据库分区号参数。数据库分区号参数是可选的。

使用 SNAP_WRITE_FILE 存储过程将数据库系统快照信息捕获到文件中

通过使用 SNAP_WRITE_FILE 存储过程，可以捕获监视器数据快照并将此信息存储到数据库服务器上的文件中，并且可以允许未拥有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限的用户访问该数据。于是，任何用户都可以发出带有快照表函数的查询以访问这些文件中的快照信息。如果开放对快照监视器数据的访问，所有对快照表函数具有执行特权的用户就能获得一些敏感信息，例如已连接用户列表以及他们对数据库提交的 SQL 语句等。缺省情况下，已将执行快照表函数的特权授予 PUBLIC。（但是，请注意，使用快照监视器表函数并不会泄漏表中的实际数据或用户密码。）

您必须拥有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 SNAP_WRITE_FILE 存储过程来捕获数据库快照。

调用 SNAP_WRITE_FILE 存储过程时，除了标识所要监视的数据库和分区以外，还需要指定*快照请求类型*。每种快照请求类型都确定了所收集的监视器数据的范围。请根据用户需要运行的快照表函数来选择快照请求类型。下表列示了快照表函数及其对应的请求类型。

表 38. 快照请求类型

快照表函数	快照请求类型
SNAP_GET_AGENT	APPL_ALL
SNAP_GET_AGENT_MEMORY_POOL	APPL_ALL
SNAP_GET_APPL_V95	APPL_ALL
SNAP_GET_APPL_INFO_V95	APPL_ALL
SNAP_GET_STMT	APPL_ALL
SNAP_GET_SUBSECTION	APPL_ALL
SNAP_GET_BP_PART	BUFFERPOOLS_ALL
SNAP_GET_BP_V95	BUFFERPOOLS_ALL
SNAP_GET_DB_V95	DBASE_ALL
SNAP_GET_DETAILLOG_V91	DBASE_ALL
SNAP_GET_DB_MEMORY_POOL	DBASE_ALL
SNAP_GET_HADR	DBASE_ALL
SNAP_GET_STORAGE_PATHS	DBASE_ALL
SNAP_GET_DBM_V95	DB2
SNAP_GET_DBM_MEMORY_POOL	DB2
SNAP_GET_FCM	DB2
SNAP_GET_FCM_PART	DB2
SNAP_GET_SWITCHES	DB2
SNAP_GET_DYN_SQL_V95	DYNAMIC_SQL
SNAP_GET_LOCK	DBASE_LOCKS

表 38. 快照请求类型 (续)

快照表函数	快照请求类型
SNAP_GET_LOCKWAIT	APPL_ALL
SNAP_GET_TAB_V91	DBASE_TABLES
SNAP_GET_TAB_REORG	DBASE_TABLES
SNAP_GET_TBSP_V91	DBASE_TABLESPACES
SNAP_GET_TBSP_PART_V91	DBASE_TABLESPACES
SNAP_GET_CONTAINER_V91	DBASE_TABLESPACES
SNAP_GET_TBSP_QUIESCER	DBASE_TABLESPACES
SNAP_GET_TBSP_RANGE	DBASE_TABLESPACES
SNAP_GET_UTIL	DB2
SNAP_GET_UTIL_PROGRESS	DB2

1. 连接到数据库。这可以是实例中任何需要监视的数据库。为了能够调用存储过程，必须连接到数据库。
2. 确定快照请求类型以及需要监视的数据库和分区。
3. 调用 SNAP_WRITE_FILE 存储过程，对快照请求类型参数、数据库参数和分区参数指定适当的设置。例如，以下调用将捕获 SAMPLE 数据库的当前连接分区的应用程序信息快照:

   ```
   CALL SNAP_WRITE_FILE('APPL_ALL','SAMPLE',-1)
   ```

   SNAP_WRITE_FILE 存储过程有 3 个输入参数:

   - 快照请求类型（请参阅第 78 页的表 38，该表提供了快照表函数及其相应请求类型的交叉引用）
   - VARCHAR (128) 数据库名称。如果输入 NULL，就会使用当前连接的数据库的名称。

     注: 此参数不适用于数据库管理器级别快照表函数；那些表函数只有请求类型参数和分区号参数。
   - SMALLINT 分区号（0 到 999 之间的值）。对于分区号参数，输入与所要监视的分区号相对应的整数。要捕获当前连接的分区的快照，请输入值 -1 或 NULL。要捕获全局快照，请输入值 -2。

在将快照数据保存到文件中之后，通过将 (NULL, NULL) 指定为数据库级别表函数的输入值并对数据库管理器级别表函数指定 (NULL)，所有用户都可以发出包含相应快照表函数的查询。他们接收到的监视器数据是从 SNAP_WRITE_FILE 存储过程生成的文件中得来的。

注: 虽然这种方法限制了用户对敏感监视器数据的访问，但这种方法有一些局限性:

- SNAP_WRITE_FILE 文件中的快照监视器数据仅仅是上次调用 SNAP_WRITE_FILE 存储过程时的最新数据。通过定期调用 SNAP_WRITE_FILE 存储过程，可以确保获得最新的快照监视器数据。例如，在 UNIX 系统上，可以设置 cron 作业来完成此操作。
- 发出包含快照表函数的查询的用户不能标识所要监视的数据库或分区。发出 SNAP_WRITE_FILE 调用的用户所标识的数据库名称和分区号确定了快照表函数可访问的文件内容。

- 如果用户发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAP_WRITE_FILE 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

使用 SQL 查询中的快照表函数来访问数据库系统快照（使用文件访问）

对于授权用户调用 SNAP_WRITE_FILE 存储过程的每个请求类型，任何用户都可使用相应快照表函数发出查询。他们接收到的监视器数据是从 SNAP_WRITE_FILE 存储过程生成的文件中检索到的。

对于打算用于访问 SNAP_WRITE_FILE 文件的每个快照表函数，授权用户必须已经使用相应快照请求类型发出了 SNAP_WRITE_FILE 存储过程调用。如果发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAP_WRITE_FILE 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

使用快照表函数从 SNAP_WRITE_FILE 文件访问快照数据的用户不能标识要监视的数据库或分区。如果数据库名称和分区号是通过用户发出 SNAP_WRITE_FILE 调用标识的，那么该数据库名称和分区号将确定 SNAP_WRITE_FILE 文件的内容。SNAP_WRITE_FILE 文件中可用的快照监视器数据仅仅是上次 SNAP_WRITE_FILE 存储过程捕获快照时的最新数据。

1. 连接到数据库。这可以是实例中任何需要监视的数据库。要使用快照表函数发出 SQL 查询，必须连接到数据库。
2. 确定需要捕获的快照的类型。
3. 使用适当的快照表函数发出查询。例如，以下查询将捕获表空间信息的快照。

   ```
   SELECT * FROM TABLE(SNAP_GET_TBSP_V91 (CAST(NULL AS VARCHAR(1)),
                       CAST (NULL AS INTEGER))) AS SNAP_GET_TBSP_V91
   ```

   注：必须对数据库名称和分区号参数输入空值。快照的数据库名称和分区将在 SNAP_WRITE_FILE 存储过程的调用中确定。而且，数据库名称参数不适用于数据库管理器级别快照表函数；那些表函数只有分区号参数。
   每个快照表函数返回带有一行或多行的一个表，每列表示一个监视元素。相应的，监视元素列名与监视元素名称相关。
4. 您也可以从返回的表中选择个别监视元素。例如，以下语句仅返回 **agent_id** 监视元素：

   ```
   SELECT agent_id FROM TABLE(
                              SNAP_GET_APPL_V95(CAST(NULL AS VARCHAR(1)),
                                                CAST (NULL AS INTEGER)))
                              as SNAP_GET_APPL_V95
   ```

快照监视器 SQL 管理视图

提供了许多不同的快照监视器 SQL 管理视图，每个管理视图都返回有关特定数据库系统区域的监视器数据。例如，SYSIBMADM.SNAPBP SQL 管理视图捕获快照或缓冲池信息。下表列示了每个可用的快照监视器管理视图。

表 39. 快照监视器 SQL 管理视图

监视器级别	SQL 管理视图	返回的信息
数据库管理器	SYSIBMADM.SNAPDBM	数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPFCM	关于快速通信管理器（FCM）的数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPFCM_PART	某个分区的关于快速通信管理器（FCM）的数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPSWITCHES	数据库管理器监视开关设置。
数据库管理器	SYSIBMADM.SNAPDBM_MEMORY_POOL	有关内存使用情况的数据库管理器级别信息。
数据库	SYSIBMADM.SNAPDB	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SYSIBMADM.SNAPDB_MEMORY_POOL	仅与 UNIX 平台的内存使用情况有关的数据库级别信息。
数据库	SYSIBMADM.SNAPHADR	有关高可用性灾难恢复的数据库级别信息。
应用程序	SYSIBMADM.SNAPAPPL	有关连接至数据库的每个应用程序的常规应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SYSIBMADM.SNAPAPPL_INFO	有关连接至数据库的每个应用程序的常规应用程序级别标识信息。
应用程序	SYSIBMADM.SNAPLOCKWAIT	有关连接至数据库的应用程序的锁定等待数的应用程序级别信息。
应用程序	SYSIBMADM.SNAPSTMT	有关连接至数据库的应用程序的语句的应用程序级别信息。此信息包括最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SYSIBMADM.SNAPAGENT	有关连接至数据库的应用程序的关联代理程序的应用程序级别信息。
应用程序	SYSIBMADM.SNAPSUBSECTION	有关连接至数据库的应用程序的访问方案子节的应用程序级别信息。
应用程序	SYSIBMADM.SNAPAGENT_MEMORY_POOL	有关代理程序级别的内存使用情况的信息。
表	SYSIBMADM.SNAPTAB	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序*已访问的*每个表在表级别的活动信息。需要表开关。
表	SYSIBMADM.SNAPTAB_REORG	正在重组的数据库中每个表在表级别的重组信息。
锁定	SYSIBMADM.SNAPLOCK	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	SYSIBMADM.SNAPTBSP	有关以下级别的表空间活动信息：数据库级别、与数据库相连接的每个应用程序的应用程序级别以及与数据库相连接的应用程序已访问的每个表空间的表空间级别。需要缓冲池开关。
表空间	SYSIBMADM.SNAPTBSP_PART	表空间配置信息。
表空间	SYSIBMADM.SNAPTBSP_QUIESCER	表空间级别停顿者信息。

表 39. 快照监视器 SQL 管理视图 (续)

监视器级别	SQL 管理视图	返回的信息
表空间	SYSIBMADM.SNAPCONTAINER	表空间级别表空间容器配置信息。
表空间	SYSIBMADM.SNAPTBSP_RANGE	表空间映射的范围信息。
缓冲池	SYSIBMADM.SNAPBP	指定数据库的缓冲池活动计数器。需要缓冲池开关。
缓冲池	SYSIBMADM.SNAPBP_PART	有关针对每个分区计算出来的缓冲区大小和使用情况的信息。
动态 SQL	SYSIBMADM.SNAPDYN_SQL	数据库的 SQL 语句高速缓存中的时间点语句信息。
数据库	SYSIBMADM.SNAPUTIL	关于实用程序的信息。
数据库	SYSIBMADM.SNAPUTIL_PROGRESS	关于实用程序进度的信息。
数据库	SYSIBMADM.SNAPDETAILLOG	有关日志文件的数据库级别信息。
数据库	SYSIBMADM.SNAPSTORAGE_PATHS	返回数据库的自动存储器路径列表，包括每个存储器路径的文件系统信息。

在捕获快照前，请考虑是否需要由监视开关控制的监视元素提供的信息。如果特定监视开关处于关闭状态，就不会收集它控制的监视元素。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

所有快照监视管理视图和关联表函数都使用单独的实例连接，该连接与当前会话使用的连接不同。因此，只有缺省数据库管理器监视开关起作用。不起作用的监视开关包括任何在当前会话或应用程序中动态打开或关闭的开关。

DB2 版本 9.5 还为您提供了一组管理视图，它们不仅返回个别监视元素的值，而且还返回监视任务中通常需要的计算值。例如，SYSIBMADM.BP_HITRATIO 管理视图返回缓冲池命中率计算值，此值由各个监视元素的值累计得出的。

表 40. 快照监视器 SQL 管理公用视图

SQL 管理公用视图	返回的信息
SYSIBMADM.APPLICATIONS	关于已连接的数据库应用程序的信息。
SYSIBMADM.APPL_PERFORMANCE	有关选择的行数与应用程序已读取的行数的比率的信息。
SYSIBMADM.BP_HITRATIO	数据库中的缓冲池命中率，包括命中率总计、数据命中率和索引命中率。
SYSIBMADM.BP_READ_IO	关于缓冲池读性能的信息。
SYSIBMADM.BP_WRITE_IO	关于缓冲池写性能的信息。
SYSIBMADM.CONTAINER_UTILIZATION	关于表空间容器和利用率的信息。
SYSIBMADM.LOCKS_HELD	关于当前挂起的锁定的信息。
ISYSIBMADM.LOCKWAIT	有关代表等待获取锁定的应用程序工作的 DB2 代理程序的信息。
SYSIBMADM.LOG_UTILIZATION	有关当前连接的数据库的日志利用率的信息。
SYSIBMADM.LONG_RUNNING_SQL	有关在当前连接的数据库中花费最长时间运行的 SQL 的信息。
SYSIBMADM.QUERY_PREP_COST	关于准备不同 SQL 语句所需时间的信息。
SYSIBMADM.TBSP_UTILIZATION	表空间配置和利用率信息。
SYSIBMADM.TOP_DYNAMIC_SQL	可按执行次数、平均执行时间、排序数或每个语句的排序数进行排序的顶级动态 SQL 语句数。

对数据库系统快照的 SQL 访问

有两种方法可用来使用快照监视器 SQL 表函数（又称为*快照表函数*）访问快照监视器数据:

- 直接访问
- 文件访问

直接访问

授权用户可使用快照表函数发出查询并接收包含监视数据的结果集。如果使用此方法，那么只有具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限的用户才能访问快照监视器数据。

要使用直接访问来捕获快照信息:

1. 可选: 设置并检查监视开关的状态。
2. 使用 SQL 捕获数据库系统快照。

文件访问

授权用户调用 SNAPSHOT_FILEW 存储过程（标识快照请求类型）及受影响的分区和数据库。然后 SNAPSHOT_FILEW 存储过程会将监视器数据存储在数据库服务器上的文件中。

授权用户可对其调用 SNAPSHOT_FILEW 存储过程的每个请求类型

虽然这是允许所有用户访问快照监视器数据的安全方法，但此方法仍然有一些局限性:

- SNAPSHOT_FILEW 文件中的快照监视器数据仅仅是上次调用 SNAPSHOT_FILEW 存储过程时的最新数据。通过定期调用 SNAPSHOT_FILEW 存储过程，可以确保获得最新的快照监视器数据。例如，在 UNIX 系统上，可以设置 cron 作业来完成此操作。
- 发出包含快照表函数的查询的用户不能标识所要监视的数据库或分区。发出 SNAPSHOT_FILEW 调用的用户所标识的数据库名称和分区号确定了快照表函数可访问的文件内容。
- 如果用户发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAPSHOT_FILEW 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

捕获数据库系统快照信息并保存至文件的 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 用户将执行下列任务。

1. 了解将发出快照请求的用户的需要。 具体地说，确定他们需要的监视器数据、要从中进行收集的数据库以及收集是否限制为特定分区。
2. 可选: 设置并检查监视开关的状态。
3. 将数据库系统快照信息捕获到文件中。

一旦 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 用户完成上述步骤，所有用户都可以使用 SQL 查询中的快照表函数来访问数据库系统快照信息。

从 CLP 捕获数据库快照

可使用 GET SNAPSHOT 命令从 CLP 捕获数据库快照。有若干不同快照请求类型可用，可通过对 GET SNAPSHOT 命令指定特定参数来访问它们。

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能捕获数据库快照。

必须具有实例连接才能捕获数据库快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

1. 可选: 设置并检查监视开关的状态。
2. 从 CLP 发出带有期望参数的 GET SNAPSHOT 命令。在以下示例中，快照将捕获数据库管理器级别信息:

   ```
   db2 get snapshot for dbm
   ```

3. 对于分区数据库系统，可为特定分区捕获专门的数据库快照，或者为所有分区捕获全局的数据库快照。要对特定分区（如分区号 2）上的所有应用程序捕获数据库快照，请发出以下命令:

   ```
   db2 get snapshot for all applications at dbpartitionnum 2
   ```

4. 要对所有分区上的所有应用程序捕获数据库快照，请发出以下命令:

   ```
   db2 get snapshot for all applications global
   ```

   对于分区数据库上的全局快照，将聚集所有分区中的监视器数据。

快照监视器 CLP 命令

下表列示了所有受支持的快照请求类型。对于特定请求类型，仅当相关联的监视开关设置为 ON 时，才返回某些信息。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 41. 快照监视器 CLP 命令

监视器级别	CLP 命令	返回的信息
连接列表	`list applications [show detail]`	当前连接至数据库的所有应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
连接列表	`list applications for database` *dbname* `[show detail]`	当前连接至指定数据库的每个应用程序的标识信息。
连接列表	`list dcs applications`	当前连接至数据库的所有 DCS 应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
数据库管理器	`get snapshot for dbm`	数据库管理器级别信息，包括实例级别监视开关设置。
数据库管理器	`get dbm monitor switches`	实例级别监视开关设置。
数据库	`get snapshot for database on` *dbname*	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`get snapshot for all databases`	在分区上处于活动状态的每个数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`list active databases`	与每个活动数据库的连接的数目。包括使用 ACTIVATE DATABASE 命令启动但没有连接的数据库。
数据库	`get snapshot for dcs database on` *dbname*	特定 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。

表 41. 快照监视器 CLP 命令 (续)

监视器级别	CLP 命令	返回的信息
数据库	`get snapshot for remote database on dbname`	特定联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`get snapshot for all remote databases`	分区上的每个活动联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
应用程序	`get snapshot for application applid appl-id`	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for application agentid appl-handle`	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for applications on dbname`	与分区中数据库相连接的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for all applications`	在分区中处于活动状态的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for dcs application applid appl-id`	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for all dcs applications`	在分区中处于活动状态的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for dcs application agentid appl-handle`	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for dcs applications on dbname`	与分区中数据库相连接的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for remote applications on dbname`	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for all remote applications`	在分区中处于活动状态的每个联合系统应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
表	`get snapshot for tables on dbname`	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序**已访问的**每个表在表级别的活动信息。需要表开关。
锁定	`get snapshot for locks for application applid appl-id`	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	`get snapshot for locks for application agentid appl-handle`	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	`get snapshot for locks on dbname`	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	`get snapshot for tablespaces on dbname`	有关数据库的表空间活动的信息。需要缓冲池开关。还包括有关容器、停顿者和范围的信息。此信息不受开关控制。

表 41. 快照监视器 CLP 命令 (续)

监视器级别	CLP 命令	返回的信息
缓冲池	`get snapshot for all bufferpools`	缓冲池活动计数器。需要缓冲池开关。
缓冲池	`get snapshot for bufferpools on` *dbname*	指定数据库的缓冲池活动计数器。需要缓冲池开关。
动态 SQL	`get snapshot for dynamic sql on` *dbname*	数据库的 SQL 语句高速缓存中的时间点语句信息。该信息也可能来自远程数据源。

从客户机应用程序捕获数据库快照

可使用 C、C++ 或 COBOL 应用程序中的快照监视器 API 来捕获数据库快照。在 C 和 C++ 中，可通过在 db2GetSnapshot() 参数中指定特定参数来访问若干不同快照请求类型。

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 db2MonitorSwitches API。

必须具有实例连接才能捕获数据库快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

1. 可选: 设置并检查监视开关的状态。
2. 包括下列 DB2 库: sqlmon.h 和 db2ApiDf.h。可在 sqllib 中的 include 子目录下找到它们。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

3. 将快照缓冲区单元大小设置为 100 KB。

   ```
   #define SNAPSHOT_BUFFER_UNIT_SZ 102400
   ```

4. 声明 sqlca、sqlma、db2GetSnapshotData 和 sqlm_collected 结构。还应初始化指针以包含快照缓冲区并确定缓冲区大小。

   ```
   struct sqlma *pRequestedDataGroups;
   struct sqlca sqlca;
   memset (&sqlca, '\0', sizeof(struct sqlca));
   struct sqlm_collected collectedData;
   memset (&collectedData, '\0', sizeof(collectedData));
   db2GetSnapshotData getSnapshotParam;
   memset (&getSnapshotParam, '\0', sizeof(getSnapshotParam));

   static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
   sqluint32 outputFormat;
   char *snapshotBuffer;
   ```

5. 初始化 sqlma 结构，并指定要捕获的快照属于数据库管理器级别信息。

   ```
   pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
   memset(pRequestedDataGroups, '\0', SQLMASIZE(1));
   pRequestedDataGroups->obj_num = 1;
   pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
   ```

6. 初始化用来容纳快照输出的缓冲区。

   ```
     snapshotBuffer = (char *)malloc(snapshotBufferSize);
   memset (snapshotBuffer, '\0', snapshotBufferSize);
   ```

7. 使用快照请求类型（来自 sqlma 结构）、缓冲区信息和捕获快照所需的其他信息来填充 db2GetSnapshotData 结构。

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
  getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
  getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_DEFAULT;
```

8. 捕获快照。传递 db2GetSnapshotData 结构和对缓冲区的引用，db2GetSnapshotData 结构包含捕获快照所需的信息，快照输出将引导至该缓冲区。

```
db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. 包括用于处理缓冲区溢出的逻辑。获取快照后，将检查 sqlcode 是否存在缓冲区溢出。如果发生了缓冲区溢出，那么将清除并重新初始化缓冲区，然后再次获取快照。

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
    snapshotBufferSize = snapshotBufferSize +
    SNAPSHOT_BUFFER_UNIT_SZ;
    snapshotBuffer = (char *)malloc(snapshotBufferSize);
    if (snapshotBuffer == NULL)
  {
        printf("\nMemory allocation error.\n");
        return 1;
  }
    getSnapshotParam.iBufferSize = snapshotBufferSize;
    getSnapshotParam.poBuffer = snapshotBuffer;
    db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. 处理快照监视器数据流。
11. 清除缓冲区。

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

快照监视器 API 请求类型

下表列示了所有受支持的快照请求类型。对于特定请求类型，仅当相关联的监视开关设置为 ON 时，才返回某些信息。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 42. 快照监视器 API 请求类型

监视器级别	API 请求类型	返回的信息
连接列表	SQLMA_APPLINFO_ALL	当前连接至数据库的所有应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
连接列表	SQLMA_DBASE_APPLINFO	当前连接至指定数据库的每个应用程序的标识信息。
连接列表	SQLMA_DCS_APPLINFO_ALL	当前连接至数据库的所有 DCS 应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
数据库管理器	SQLMA_DB2	数据库管理器级别信息，包括实例级别监视开关设置。

表 42. 快照监视器 API 请求类型 (续)

监视器级别	API 请求类型	返回的信息
数据库	SQLMA_DBASE	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_ALL	在分区上处于活动状态的每个数据库的数据库级别信息和计数器。与每个活动数据库的连接的数目。包括使用 ACTIVATE DATABASE 命令启动但没有连接的数据库。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DCS_DBASE	特定 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DCS_DBASE_ALL	在分区上处于活动状态的每个 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_REMOTE	特定联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_REMOTE_ALL	分区上的每个活动联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
应用程序	SQLMA_APPL	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_AGENT_ID	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DBASE_APPLS	与分区中数据库相连接的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_APPL_ALL	在分区中处于活动状态的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL_ALL	在分区中处于活动状态的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL_HANDLE	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_DBASE_APPLS	与分区中数据库相连接的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DBASE_APPLS_REMOTE	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_APPL_REMOTE_ALL	在分区中处于活动状态的每个联合系统应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。

表 42. 快照监视器 API 请求类型 (续)

监视器级别	API 请求类型	返回的信息
表	SQLMA_DBASE_TABLES	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序**已访问的**每个表在表级别的活动信息。需要表开关。
锁定	SQLMA_APPL_LOCKS	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	SQLMA_APPL_LOCKS_AGENT_ID	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	SQLMA_DBASE_LOCKS	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	SQLMA_DBASE_TABLESPACES	有关以下级别的表空间活动信息：数据库级别、与数据库相连接的每个应用程序的应用程序级别以及与数据库相连接的应用程序已访问的每个表空间的表空间级别。需要缓冲池开关。
缓冲池	SQLMA_BUFFERPOOLS_ALL	缓冲池活动计数器。需要缓冲池开关。
缓冲池	SQLMA_DBASE_BUFFERPOOLS	指定数据库的缓冲池活动计数器。需要缓冲池开关。
动态 SQL	SQLMA_DYNAMIC_SQL	数据库的 SQL 语句高速缓存中的时间点语句信息。

快照监视器样本输出

为说明快照监视器的特征，下面提供了使用 CLP 获取快照的示例及相应输出。此示例的目的在于获取连接至 SAMPLE 数据库的应用程序挂起的锁定列表。采取的步骤如下所示:

1. 连接至样本数据库:

   ```
   db2 connect to sample
   ```

2. 使用 UPDATE MONITOR SWITCHES 命令将 LOCK 开关设置为 ON，这样就会收集等待锁定所花的时间:

   ```
   db2 update monitor switches using LOCK on
   ```

3. 发出需要针对数据库目录的锁定的命令或语句。在此情况下，我们将声明、打开和访存游标:

   ```
   db2 -c- declare c1 cursor for
                          select * from staff where job='Sales' for update
   db2 -c- open c1
   db2 -c- fetch c1
   ```

4. 使用 GET SNAPSHOT 命令获取数据库锁定快照:

   ```
   db2 get snapshot for locks on sample
   ```

在从 CLP 发出 GET SNAPSHOT 命令后，快照输出将引导至屏幕。

```
                        数据库锁定快照

数据库名称                                  = SAMPLE
数据库路径                                  = C:\DB2\NODE0000\SQL00001\
输入数据库别名                              = SAMPLE
挂起的锁定数                                    = 5
当前连接的应用程序数                        = 1
```

```
当前正在等待锁定的代理程序数            = 0
快照时间戳记                            = 06-05-2002 17:08:25.048027

应用程序句柄                            = 8
应用程序标识                            = *LOCAL.DB2.0098C5210749
序号                                    = 0001
应用程序名称                            = db2bp.exe
连接授权标识                            = DB2ADMIN
应用程序状态                            = UOW 正在等待
状态更改时间                            = 未收集
应用程序代码页                          = 1252
挂起的锁定数                                = 5
总等待时间（ms）                        = 0

锁定列表
  锁定名称                  = 0x020003000500000000000000052
  锁定属性                  = 0x00000000
  释放标志                  = 0x00000001
  锁定计数                  = 1
  挂起计数                  = 0
  锁定对象名                = 5
  对象类型                  = 行
  表空间名称                = USERSPACE1
  表模式                    = DB2ADMIN
  表名                      = STAFF
  方式                      = U

  锁定名称                  = 0x020003000000000000000000054
  锁定属性                  = 0x00000000
  释放标志                  = 0x00000001
  锁定计数                  = 1
  挂起计数                  = 0
  锁定对象名                = 3
  对象类型                  = 表
  表空间名称                = USERSPACE1
  表模式                    = DB2ADMIN
  表名                      = STAFF
  方式                      = IX

  锁定名称                  = 0x010000000100000000100810056
  锁定属性                  = 0x00000000
  释放标志                  = 0x40000000
  锁定计数                  = 1
  挂起计数                  = 0
  锁定对象名                = 0
  对象类型                  = 内部偏差锁定
  方式                      = S

  锁定名称                  = 0x41414141414A48520000000041
  锁定属性                  = 0x00000000
  释放标志                  = 0x40000000
  锁定计数                  = 1
  挂起计数                  = 0
  锁定对象名                = 0
  对象类型                  = 内部计划锁定
  方式                      = S

  锁定名称                  = 0x434F4E544F4B4E310000000041
  锁定属性                  = 0x00000000
  释放标志                  = 0x40000000
  锁定计数                  = 1
  挂起计数                  = 0
  锁定对象名                = 0
  对象类型                  = 内部计划锁定
  方式                      = S
```

可从此快照中看到，当前有一个应用程序连接至 SAMPLE 数据库，并且该应用程序挂起 5 个锁定。

```
挂起的锁定数                                = 5
    当前连接的应用程序数                        = 1
```

注意，应用程序状态成为 UOW 正在锁定的时间（状态更改时间）返回为未收集。这是因为 UOW 开关为 OFF。

锁定快照还将返回直到目前为止连接至此数据库的应用程序等待锁定所花的总时间。

```
总等待时间（ms）                    = 0
```

子节快照

在使用分区间并行性的系统上，SQL 编译器将 SQL 语句的访问方案分为若干子节。每个子节由一个不同的 DB2 代理程序（或 SMP 的代理程序）执行。

通过使用 db2expln 命令，可获取 DB2 代码生成器在编译期间生成的 SQL 语句的访问方案。例如，选择分布在若干分区上的表中的所有行可能导致访问方案有两个子节:

1. 子节 0，协调程序子节，作用是收集其他 DB2 代理程序（子代理程序）访存的行并将其返回至应用程序。
2. 子节 1，作用是执行表扫描并将行返回至协调代理程序。

在此简单示例中，子节 1 将分布在所有数据库分区上。数据库分区组的每个物理分区上都有执行此子节的子代理程序，此表属于该数据库分区组。

数据库系统监视器允许您将运行时信息与访问方案相关联，这是编译时信息。借助分区间并行性，监视器会中断信息并下行至子节级别。例如，当语句监视开关设置为 ON 时，GET SNAPSHOT FOR APPLICATION 将返回在此分区上执行的每个子节的信息以及语句总数。

对应用程序快照返回的子节信息包括:

- 读取/写入的表行数
- CPU 消耗
- 耗用时间
- 发送至其他代理程序和从其他代理程序接收的表队列行数，这些代理程序处理此语句。这允许您通过获取一系列快照来跟踪长时间运行的查询的执行情况。
- 子节状态。如果子节因为等待另一代理程序发送或接收数据而处于 WAIT 状态，那么该信息还会标识阻止子节继续执行的分区。然后可获取这些分区的快照来调查情况。

在每个子节执行完之后，语句事件监视器记录的有关该子节的信息包括: CPU 消耗、总执行时间和若干其他计数器。

分区数据库系统上的全局快照

在分区数据库系统上，可以使用快照监视器来获取当前分区、指定分区或所有分区的快照。对分区数据库的所有分区获取全局快照时，会先聚集数据，然后返回结果。

对不同元素类型聚集数据的方式如下所示:

- **计数器、时间和标尺**

  包含从实例中的每个分区收集的所有可能值的总和。例如，GET SNAPSHOT FOR DATABASE XYZ ON TEST GLOBAL 对分区数据库实例中的所有分区返回从数据库读取的行数（rows_read）。

- **水位标记**

  返回分区数据库系统中的任何分区的最高（高水位）或最低（低水位）值。如果返回的值值得关注，那么可以获取各个分区的快照以确定特定分区是否使用过度或者问题是否为实例范围内的问题。

- **时间戳记**

  设置为连接快照监视器实例代理程序的分区的时间戳记值。注意，所有时间戳记值都在 `timestamp` 监视开关控制之下。

- **信息**

  返回可能妨碍工作的分区的最重要信息。例如，对于元素 `appl_status`，如果一个分区上的状态为“正在执行 UOW”，而另一个分区上的状态为“等待锁定”，那么返回“等待锁定”，原因是这是挂起应用程序的执行的状态。

您也可以复位计数器，设置监视开关，以及检索分区数据库中的个别分区或所有分区的监视开关设置。

注: 获取全局快照时，如果一个或多个分区遇到错误，那么将从成功获取快照的分区收集数据，同时返回一个警告（sqlcode 1629）。如果以全局方式获取或更新监视开关，或者计数器在一个或多个分区上复位失败，那么不会设置这些分区的监视开关或进行数据复位。

快照监视器自描述数据流

在使用 db2GetSnapshot API 捕获快照之后，API 以自描述数据流的形式返回快照输出。第 93 页的图 2 显示数据流的结构，而 第 93 页的表 43 提供可能返回的逻辑数据组和监视元素的一些示例。

注: 描述性名称用于示例和表中的标识。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，COLLECTED 在快照监视器输出中将显示为 SQLM_ELM_COLLECTED。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。

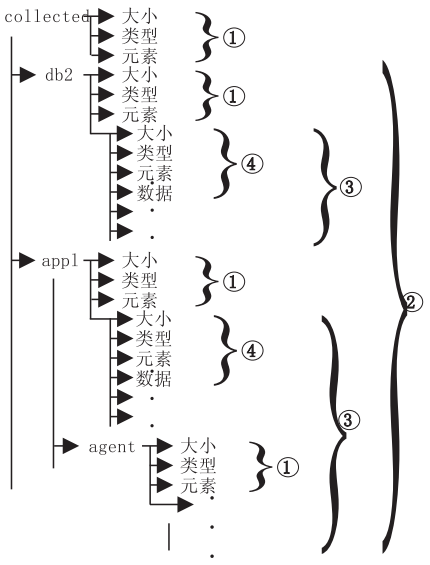


图 2. 快照监视器数据流

1. 每个逻辑数据组以指示其大小和名称的头开始。此大小不包括头本身占用的数据量。
2. 已收集头中的大小返回快照的总大小。
3. 其他头中的大小元素指示该逻辑数据组中的所有数据的大小，包括所有下级分组。
4. 监视元素信息在逻辑数据组头后面，并且也是自描述格式。

表 43. 样本快照数据流

逻辑数据组	数据流	描述
collected		
	1000	快照数据的大小（以字节计）。
	header	指示逻辑数据组的开头。
	collected	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	server_db2_type	收集的监视元素的名称。
	sqlf_nt_server	此元素的已收集值。
	2	此监视元素中存储的数据的大小。
	u16bit	监视元素类型 - 不带符号的 16 位数字。
	node_number	收集的监视元素的名称。
	3	此元素的已收集值。

表 43. 样本快照数据流 (续)

逻辑数据组	数据流	描述
db2		
	200	快照中 DB2 级别数据部分的大小。指示逻辑数据组的开头。
	header	逻辑数据组的名称。
	db2	
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	sort_heap_allocated	收集的监视元素的名称。
	16	此元素的已收集值。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	local_cons	收集的监视元素的名称。
	3	此元素的已收集值。
	. . .	. . .
appl		
	100	快照中的应用程序元素数据的大小。
	header	指示逻辑数据组的开头。
	appl	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	locks_held	收集的监视元素的名称。
	3	此元素的已收集值。
	. . .	. . .
agent		
	50	应用程序结构的代理程序部分的大小。
	header	指示逻辑数据组的开头。
	agent	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 32 位数字。
	agent_pid	收集的监视元素的名称。
	12	此元素的已收集值。
	. . .	. . .

db2GetSnapshot() 例程在用户提供的缓冲区中返回自描述格式快照数据。将在与要捕获的快照类型相关联的逻辑数据分组中返回数据。

快照请求返回的每一项都包含指定大小和类型的字段。该字段可用来对返回的数据进行语法分析。字段的大小还可用来跳过逻辑数据组。例如，要跳过 DB2 记录，需要确定数据流中的字节数。使用以下公式来计算要跳过的字节数:

```
DB2 逻辑数据分组的大小 + sizeof(sqlm_header_info)
```

使用 db2top 以交互方式进行监视的命令

db2top 监视实用程序快速高效地监视复杂的 DB2 环境。它结合来自所有数据库分区的 DB2 快照信息，使用基于文本的用户界面提供正在运行的 DB2 系统的动态实时视图。

以交互方式运行 db2top 时，您可以发出下列命令:

A 监视 HADR 集群中的主数据库或辅助数据库。

a 转至代理程序的应用程序详细信息（或在声明屏幕上限制代理程序）。db2top 命令将提示您输入代理程序标识。

B 显示关键服务器资源的主要使用者（瓶颈分析）。

c 此选项允许您更改屏幕上显示的列的顺序。语法采用下列格式: 1,2,3,...，其中 1,2,3 分别对应于所显示的第 1 列、第 2 列和第 3 列。这些是指定排序条件时要使用的列数。

当使用 c 交换关键字时，将显示屏幕，指定屏幕上显示的列的顺序。屏幕的左侧部分显示缺省顺序和列数；屏幕右侧部分显示当前排序。要更改列的顺序，在屏幕底部文本字段中输入新的列顺序。接着，如左侧显示的那样，输入相对的列位置，用逗号对其分隔。不需要指定所有列。对于后续的 db2top 监视会话，可以通过选择 w 将此列排序保存在 *$DB2TOPRC* 中。您可以进行排序，并选择采用哪种顺序在屏幕上显示列。`.db2toprc` 文件中列排序的有效关键字是:

- sessions=
- tables=
- tablespaces=
- bufferpools=
- dynsql=
- statements=
- locks=
- utilities=
- federation=

b 转至缓冲池屏幕。

C 打开或关闭快照数据收集器。

d 转至数据库屏幕。

D 转至动态 SQL 屏幕。

f 冻结屏幕。

F 在主服务器上监视联合查询。

G 打开或关闭图表。

h 转至帮助屏幕

H 转至历史记录屏幕

i 打开或关闭闲置会话。

k 切换实际值与增量值。

l 转至会话屏幕。

L 允许显示来自 SQL 屏幕的完整查询文本。然后，可以使用 e 或 X 选项来运行常规 DB2 说明。

m 显示内存池。

o 显示会话设置。

p 转至分区屏幕。

P 选择要发出快照的数据库分区。

q 退出 db2top。

R 重置快照数据。

s 转至语句屏幕。

S 运行本机 DB2 快照。

t 转至表空间屏幕。

T 转至表屏幕

u 显示活动的实用程序，并且跨数据库分区将它们聚集起来。

U 转至锁定屏幕。

V 设置缺省说明模式。

w 将会话设置写至 .db2toprc。

W agent_id、os_user、db_user、应用程序或网络名的观看方式。会话快照（选项 1）返回的语句将写至 agent.sql、 os_user-agent.sql、db_user-agent.sql、application-agent.sql 或 netname-agent.sql。当从动态 SQL 屏幕（选项 D）发出时，语句将采用与 db2advis 兼容的格式写至 db2adv.sql。

X 打开或关闭扩展方式。

z|Z 按升序或降序方式进行排序。

/ 将表达式输入至过滤器数据。表达式必须符合正则表达式。您可以采用不同方法过滤每个函数（屏幕）。可对整行应用 regexp 检查。

<|> 移至屏幕的左侧或右侧。

下列切换只适用于应用程序屏幕:

r 返回至上一函数。

R 切换自动刷新。

g 打开或关闭图表。

X 打开或关闭扩展方式。

d 显示代理程序。

要以交互方式启动 db2top，可发出下列命令:

```
db2top -d <database name>
```

当输入

```
db2top -d sample
```

时，将显示下列输出:

```
[\]11:57:10,refresh=2secs(0.000) Inactive,part=[1/1],<instanceName>:sample
[d=Y,a=N,e=N,p=ALL] [qp=off]

[/]: 当旋转时，它表示 db2top 在两个快照之间等待，否则，它表示 db2top 在等待 DB2 的答复
11:57:10: 当前时间
refresh=2secs: 时间间隔
refresh=!secs: 感叹号表示 DB2 处理快照所需的时间超过时间间隔。
在此情况下，db2top 将按 50% 增加时间间隔。
如果由于系统太忙而频繁发生此问题，
那么您可以增加快照时间间隔（选项 I）、
监视单一数据库分区（选项 P）或关闭扩展显示方式（选项 x）
0.000: DB2 内部处理快照所花费的时间
d=Y/N: 增量或累积快照指示器（命令选项 -k 或选项 k）。
a=Y/N: 仅限于活动对象指示器的或所有对象指示器（-a 命令选项集或 i）
e=Y/N: 扩展显示指示器
p=ALL: 所有数据库分区
p=CUR: 当前数据库分区（-P 命令选项，未指定分区数）
p=3: 目标数据库分区数: 例如，3

Inactive: 如果 DB2 没有在运行，那么会显示不活动，否则会显示运行 DB2 的平台
part=[1/1]: 活动数据库分区数与总计数据库分区数。例如，part=[2,3] 表示总共有 3 个
数据库分区，其中有一个数据库分区停机（2 个数据库分区处于活动状态，共有 3 个）
<instanceName>: 实例名
sample: 数据库名称
qp=off/on: 已连接 db2top 的数据库分区的 Query Patroller 指示器（DYNMGMT 数据库配置参数）
```

下列示例演示在分区数据库环境中以交互方式运行 db2top 监视实用程序:

```
db2top -d TEST -n mynode -u user -p passwd -V skm4 -B -i 1
命令参数如下所示:
-d TEST     # 数据库名称
-n mynode   #  节点名
-u user     #  用户标识
-p passwd   #  密码
-V skm4     #  模式名称
-B          #  启用粗体
-i 1        #  屏幕更新时间间隔: 1 秒
```

.db2toprc 配置文件

.db2toprc 配置文件是用户生成的文件，用于在初始化时为 db2top 监视实用程序设置参数。db2top 实用程序将使用用户定义的变量 *$db2topRC* 搜索 .db2toprc 文件的位置。如果该变量尚未设置，那么 db2top 将首先在当前目录中搜索 .db2toprc 文件，然后再在 home 目录中搜索该文件。.db2toprc 文件是用户生成的文件。

环境变量

您可以设置下列环境变量:

- DB2TOPRC

  存储 .db2toprc 文件位置的用户定义的环境变量。例如，在 Linux 上，您可以将 DB2TOPRC 定义为: `export db2topRC="~/db2top"`。

  如果用户未设置该变量，那么 db2top 将首先在当前目录中搜索 .db2toprc 文件，然后再在 home 目录中搜索该文件。

- DB2DBDFT

  此变量指定要用于隐式连接的数据库的数据库别名。当命令行或 .db2toprc 配置文件中未指定数据库名称时，将会使用此变量。

- EDITOR

此系统环境变量指定用于启动文本编辑器的命令，该文本编辑器用于显示说明结果或本机快照。

如果此变量未设置，那么将使用 vi。

结构

此处描述了 .db2toprc 文件中的一些条目。

cpu=command

使用此条目在屏幕输出右侧第二行中显示 CPU 活动的结果。例如:

`cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)0,$14+$15);}'` 将在屏幕右侧显示 `Cpu=2(usr+sys)`。

io=command

使用此条目来指定命令并在屏幕输出左侧的第二行中显示结果。例如:

`io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)0,$10+$11);}'` 将在屏幕左侧显示 `Disk=76(bi+bo)`。

两个命令均作为后台进程运行，并且屏幕上的字段会以异步方式更新。

shell alias=command

使用此 shell 条目来指定用户定义的命令，例如: 当输入 M 时，`shell M=top` 会从 db2top 会话衍生在顶部。

function alias=command

使用此条目来指定用户定义的命令，例如: `function N=netstat` 创建名为 N 的新函数，该函数重复地显示 netstat 的输出。可以存在多个 `function` 条目。您必须将它们置于单独的行。例如:

```
function Q=netstat
function N=df -k
```

sort=command

使用此条目来指定排序顺序，例如: `sort=command` 为此函数创建缺省排序顺序，其中 command 是列数。该排序顺序可以是升序或降序。排序对会话、表、表空间、缓冲池、dynsql、语句、锁、实用程序和联合有效。

样本 .db2toprc 文件

没有缺省 .db2toprc 配置文件。但是，您可以按“W”来为当前设置创建 .db2toprc。使用下列样本 .db2toprc 文件作为参考。注释已添加到所有条目。

```
# db2top 配置文件
# 在 UNIX 上，应该位于 $HOME/<cmdname> .db2toprc</cmdname>
# db2top-1.0a 生成的文件
#
node= # [-n] 节点名
database=sample # [-d] 数据库名称
user= # [-u] 数据库用户
password= # [-p] 用户密码（加密）
schema= # [-V] 说明的缺省模式
interval=2 # [-i] 采样时间间隔
active=OFF # [-a] 仅显示活动会话（打开/关闭）
reset=OFF # [-R] 在启动时重置快照（打开/关闭）
delta=ON # [-k] 切换增量值/累积值的显示（打开/关闭）
gauge=ON # 在会话列表上显示图表（打开/关闭）
colors=ON  # 如果终端支持色彩，那么为 True。如果它可以用色彩显示信息，那么通知 GE_WRS
graphic=ON  # 如果终端支持半图解字符，那么为 True（打开/关闭）。
port= # 用于网络收集的端口
streamsize=size # 每小时的最大收集大小（例如，1024 或 1K : K、M 或 G）
```

```
# 从操作系统获取 cpu 使用信息的命令
cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)0,$14+$15);}'
# 从操作系统获取 IO 使用信息的命令
io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)0,$10+$11);}'
# 会话屏幕中信息的排序
# 会话屏幕的列顺序（选项 l）
sessions=0,1,18,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21,22,23
# 表屏幕的列顺序（选项 T）
tables=0,1,2,4,3,5,6,7
# 表空间屏幕的列顺序（选项 t）。将在列 #22 按升序顺序对显示内容进行排序
tablespaces=0,1,18,2,3,4,5,6,7,8, sort=22a
# 缓冲池屏幕的列顺序（选项 b）
bufferpools=0,1,18,2,3,4,5,6,7,8,9,10
# 动态 SQL 屏幕的列顺序（选项 D）
dynsql=0,1,18,2,3,4,5,6,7,8,9
statements=0,1
locks=0,1
utilities=0 # 包含实用程序屏幕的缺省列和排序顺序
federation=0,2,4  # 包含联合屏幕的缺省列和排序顺序

# 用户定义的命令
shell P=top
function N=date && netstat -t tcp
```

第 5 章 基于开关的监视概念

系统监视开关

系统监视开关用于控制快照监视器和某些事件监视器收集数据的方式。

注: 这些系统监视开关不影响 DB2 版本 9.7 中引入的工作单元事件监视器和锁定事件监视器。

快照监视器和某些事件监视器将报告系统监视器所收集的数据。收集系统监视器数据将引入处理数据库管理器的开销。例如，为了计算 SQL 语句的执行时间，数据库管理器必须对操作系统进行调用，以获取每个语句执行之前和执行之后的时间戳记。这些系统调用类型的成本通常很高。系统监视器导致的另一种形式的开销是增加内存消耗。对于系统监视器跟踪的每个监视元素，数据库管理器将使用内存来存储收集的数据。

为了将维护监视信息所涉及的开销降至最低，监视开关将控制数据库管理器可能进行的高成本信息收集。每个开关只有两个设置: ON 或 OFF。如果监视开关为 OFF，那么受开关控制的监视元素不会收集任何信息。有很大一部分基本监视数据不受开关控制，并且不管开关设置如何，始终会收集这些信息。

每个监视应用程序都有自己的监视开关（和系统监视器数据）逻辑视图。在启动时，每个应用程序将从数据库管理器配置文件中的 dft_monswitches 参数继承其监视开关设置（在实例级别）。监视应用程序可使用 UPDATE MONITOR SWITCHES USING MONSWITCH OFF/ON 命令来更改其监视开关设置。MONSWITCH 参数包含下面的“快照监视开关”表的“监视开关”列中的值。在应用程序级别对开关设置的更改仅影响从中更改开关的应用程序。

可在不停止数据库管理系统的情况下更改实例级别的监视开关。为此，请使用 UPDATE DBM CFG USING DBMSWITCH OFF/ON 命令。DBMSWITCH 参数包含下面的“快照监视开关”表的“DBM 参数”列中的值。开关的动态更新要求执行更新的应用程序显式连接至实例，以使更改动态生效。动态更新不会影响其他现有快照应用程序。新的监视应用程序将继承已更新实例级别监视开关设置。要让现有监视应用程序继承新的缺省监视开关值，它必须终止并重新建立连接。如果在数据库管理器配置文件中更新开关，那么会更新分区数据库中的所有分区的开关。

数据库管理器记录所有快照监视应用程序及其开关设置。如果开关在应用程序的配置中设置为 ON，那么数据库管理器总是会收集该监视器数据。如果之后同一开关在应用程序配置中设置为 OFF，那么只要至少有一个此开关设置为 ON 的应用程序，数据库管理器仍将收集数据。

时间和时间戳记元素的收集由 TIMESTAMP 开关控制。如果将此开关设置为 OFF（在缺省情况下设置为 ON），那么指示在确定与时间或时间戳记相关的监视元素时，数据库管理器将跳过所有时间戳记操作系统调用。在 CPU 利用率达到 100% 时，应将此开关设置为 OFF。当此情况发生时，发出时间戳记导致的性能下降的幅度会急剧增长。对

于可由 TIMESTAMP 开关和另一开关控制的监视元素，如果任一开关设置为 OFF，那么不会收集数据。因此，如果 TIMESTAMP 开关设置为 OFF，那么受另一监视开关控制的整体数据成本会大大降低。

事件监视器不会以与快照监视应用程序相同的方式受监视开关的影响。定义事件监视器时，它会将指定事件类型所需的实例级别监视开关自动设置为 ON。例如，死锁事件监视器会将 LOCK 监视开关自动设置为 ON。在激活事件监视器时，必需的监视开关将设置为 ON。释放事件监视器时，监视开关将设置为 OFF。

事件监视器不会自动设置 TIMESTAMP 监视开关。这是控制所有监视元素的收集的唯一监视开关，这些监视元素属于事件监视器逻辑数据分组。如果 TIMESTAMP 开关设置为 OFF，那么不会收集事件监视器收集的大多数时间戳记和时间监视元素。这些元素仍将写至指定的表、文件或管道，但必须带有值零。

表 44. 快照监视开关

监视开关	DBM 参数	提供的信息
BUFFERPOOL	DFT_MON_BUFPOOL	读写次数，所花时间
LOCK	DFT_MON_LOCK	等待锁定次数，死锁数
SORT	DFT_MON_SORT	使用的堆数，排序性能
STATEMENT	DFT_MON_STMT	启动/停止时间，语句标识
TABLE	DFT_MON_TABLE	活动量度（读/写行数）
UOW	DFT_MON_UOW	开始/结束时间，完成状态
TIMESTAMP	DFT_MON_TIMESTAMP	时间戳记

在捕获快照或使用事件监视器之前，必须确定需要数据库管理器收集的数据。如果想要在快照中收集下列任何特殊类型的数据，那么需要设置适当的监视开关。

- 缓冲池活动信息
- 锁定、锁定等待及与时间有关的锁定信息
- 排序信息
- SQL 语句信息
- 表活动信息
- 时间和时间戳记信息
- 工作单元信息

在缺省情况下，与上述信息类型相对应的开关全部设置为“OFF”，但对应于时间和时间戳记信息的开关在缺省情况下设置为“ON”。

事件监视器仅受时间和时间戳记信息开关影响。所有其他开关设置对事件监视器收集的数据没有影响。

通过 CLP 设置系统监视开关

系统监视开关用于控制系统监视器的数据收集操作。通过将特定监视开关设置为 ON，可以收集特定类型的监视器数据。

执行任何监视开关更新的应用程序必须具有实例连接。必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限中的一种才能使用下列命令:

- UPDATE MONITOR SWITCHES

- GET MONITOR SWITCHES
- GET DATABASE MANAGER MONITOR SWITCHES

必须具有 SYSADM 权限才能使用 UPDATE DBM CFG 命令。

- 要激活任何局部监视开关，请使用 UPDATE MONITOR SWITCHES命令。开关将保持活动状态，直到应用程序（CLP）拆离，或者直到再次使用 UPDATE MONITOR SWITCHES 命令释放它们。以下示例将所有局部监视开关更新为 ON:

```
db2 update monitor switches using BUFFERPOOL on LOCK on
    SORT on STATEMENT on TIMESTAMP on TABLE on UOW on
```

- 要释放任何局部监视开关，请使用 UPDATE MONITOR SWITCHES 命令。以下示例将所有局部监视开关更新为 OFF:

```
db2 update monitor switches using BUFFERPOOL off, LOCK off,
        SORT off, STATEMENT off, TIMESTAMP off, TABLE off, UOW off
```

  以下是在发出上述 UPDATE MONITOR SWITCH 命令后希望看到的输出的示例:

```
          监视器记录开关

数据库分区号 1 的开关列表
缓冲池活动信息（BUFFERPOOL）    = OFF
锁定信息（LOCK）              = OFF
排序信息（SORT）                = OFF
SQL 语句信息（STATEMENT）       = OFF
表活动信息（TABLE）             = OFF
工作单元信息（UOW）             = OFF
获取时间戳记信息（TIMESTAMP）   = OFF
```

- 还可以在数据库管理器级别操作监视开关。这涉及使用 UPDATE DBM CFG 命令在数据库管理器配置文件中更改 dft_monswitches 参数。在以下示例中，除了基本信息之外，仅收集锁定开关控制的信息。

```
db2 update dbm cfg using DFT_MON_LOCK on
```

  每次启动监视应用程序时，它将从数据库管理器继承监视开关设置。对数据库管理器的监视开关设置所作的任何更改不会影响任何正在运行的监视应用程序。监视应用程序必须重新连接至实例以获取对监视开关设置的所有更改。

- 对于分区数据库系统，可专门为特定分区设置监视开关，或者为所有分区设置全局监视开关。
  1. 要为特定分区（如分区号 3）设置监视开关（如 BUFFERPOOL），请发出以下命令:

     ```
     db2 update monitor switches using BUFFERPOOL on
             at dbpartitionnum 3
     ```

  2. 要为所有分区设置监视开关（如 SORT），请发出以下命令:

     ```
     db2 update monitor switches using SORT on global
     ```

- 要检查局部监视开关的状态，请使用 GET MONITOR SWITCHES 命令。

```
db2 get monitor switches
```

- 对于分区数据库系统，可专门查看特定分区的监视开关设置，或者查看所有分区的全局监视开关设置。
  1. 要查看特定分区（如分区号 2）的监视开关设置，请发出以下命令:

     ```
     db2 get monitor switches at dbpartitionnum 2
     ```

  2. 要查看所有分区的监视开关设置，请发出以下命令:

     ```
     db2 get monitor switches global
     ```

- 要在数据库管理器级别（或实例级别）检查监视开关的状态，请使用 GET DATABASE MANAGER MONITOR SWITCHES 命令。此命令将显示要监视的实例的整体开关设置。

  ```
  db2 get database manager monitor switches
  ```

  以下是发出上述命令后希望看到的输出的示例:

  ```
           收集的 DBM 系统监视器信息

  数据库分区号 1 的开关列表
  缓冲池活动信息（BUFFERPOOL）    = OFF
  锁定信息（LOCK）                = ON    10-25-2001 16:04:39
  排序信息（SORT）                = OFF
  SQL 语句信息（STATEMENT）       = OFF
  表活动信息（TABLE）             = OFF
  工作单元信息（UOW）             = OFF
  获取时间戳记信息（TIMESTAMP）   = OFF
  ```

既然已经设置了期望的监视开关并且确认了开关设置，就可以捕获并收集监视器数据了。

通过客户机应用程序设置系统监视开关

系统监视开关用于控制系统监视器的数据收集操作。通过将特定监视开关设置为 ON，可以收集特定类型的监视器数据。

执行任何监视开关更新的应用程序必须具有实例连接。必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 db2MonitorSwitches API。

1. 包括下列 DB2 库: sqlutil.h 和 db2ApiDf.h。可在 sqllib 下的 include 子目录中找到它们。

   ```
   #include <sqlutil.h>
   #include <db2ApiDf.h>
   #include <string.h>
   #include <sqlmon.h>
   ```

2. 将开关列表缓冲区单元大小设置为 1 KB。

   ```
   #define SWITCHES_BUFFER_UNIT_SZ 1024
   ```

3. 初始化 sqlca、db2MonitorSwitches 和 sqlm_recording_group 结构。还应初始化指针以包含开关列表缓冲区并确定缓冲区大小。

   ```
   struct sqlca sqlca;
   memset (&sqlca, '\0', sizeof(struct sqlca));
   db2MonitorSwitchesData switchesData;
   memset (&switchesData, '\0', sizeof(switchesData));
   struct sqlm_recording_group switchesList[SQLM_NUM_GROUPS];
   memset(switchesList, '\0', sizeof(switchesList));
   sqluint32 outputFormat;
   static sqluint32 switchesBufferSize = SWITCHES_BUFFER_UNIT_SZ;
   char *switchesBuffer;
   ```

4. 初始化缓冲区以容纳开关列表输出。

   ```
   switchesBuffer = (char *)malloc(switchesBufferSize);
   memset(switchesBuffer, '\0', switchesBufferSize));
   ```

5. 要更改局部监视开关的状态，请更改 sqlm_recording_group 结构中的元素（按先前步骤中的指示命名为 switchesList）。对于要设置为“ON”的监视开关，参数 input_state 应设置为 SQLM_ON。对于要设置为“OFF”的监视开关，参数 input_state 必须设置为 SQLM_OFF。

```
switchesList[SQLM_UOW_SW].input_state = SQLM_ON;
switchesList[SQLM_STATEMENT_SW].input_state = SQLM_ON;
switchesList[SQLM_TABLE_SW].input_state = SQLM_ON;
switchesList[SQLM_BUFFER_POOL_SW].input_state = SQLM_OFF;
switchesList[SQLM_LOCK_SW].input_state = SQLM_OFF;
switchesList[SQLM_SORT_SW].input_state = SQLM_OFF;
switchesList[SQLM_TIMESTAMP_SW].input_state = SQLM_OFF;
switchesData.piGroupStates = switchesList;
switchesData.poBuffer = switchesBuffer;
switchesData.iVersion = SQLM_DBMON_VERSION9_5;
switchesData.iBufferSize = switchesBufferSize;
switchesData.iReturnData = 0;
switchesData.iNodeNumber = SQLM_CURRENT_NODE;
switchesData.poOutputFormat = &outputFormat;
```

注: 如果 iVersion 小于 SQLM_DBMON_VERSION8，那么 SQLM_TIMESTAMP_SW 不可用。

6. 要提交对开关设置的更改，请调用 db2MonitorSwitches() 函数。将 db2MonitorSwitchesData 结构（在此示例中命名为 switchesData）作为参数传递至 db2MonitorSwitches API。switchesData 以参数的形式包含 sqlm_recording_group 结构。

```
db2MonitorSwitches(db2Version810, &switchesData, &sqlca);
```

7. 从开关列表缓冲区处理开关列表数据流。
8. 清除开关列表缓冲区。

```
free(switchesBuffer);
free(pRequestedDataGroups);
```

既然已经设置了期望的监视开关并且确认了开关设置，就可以捕获并收集监视器数据了。

系统监视开关自描述数据流

在使用 db2MonitorSwitches API 更新或查看当前系统监视开关设置之后，API 将以自描述数据流的形式返回开关设置。第 106 页的图 3 显示可能对分区数据库环境返回的开关列表信息的结构。

注:

1. 描述性名称用于示例和表中的标识。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，db_event 在事件监视器输出中显示为 SQLM_ELM_DB_EVENT。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。
2. 对于全局开关请求，返回的信息的分区顺序在每个开关请求中可能有所不同。在此情况下，分区标识包括在数据流中。

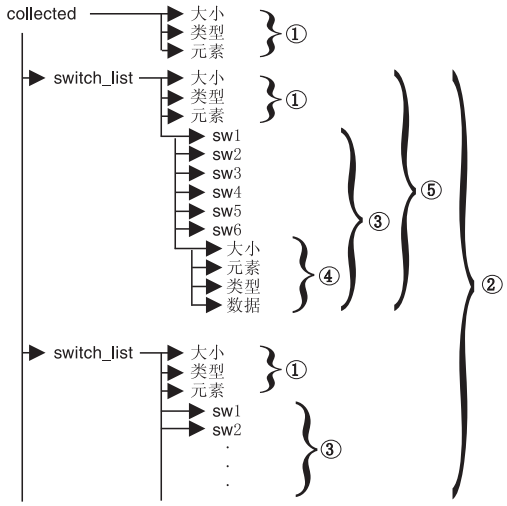


图 3. 开关列表监视器数据流

1. 每个逻辑数据组以指示其大小和名称的头开始。此大小不包括头本身占用的数据量。
2. 已收集头中的大小返回所有分区的所有监视开关列表的总大小。
3. 开关列表头中的大小元素指示该分区的开关数据的大小。
4. 开关信息是自描述格式。
5. 对于非分区数据库，将返回独立分区的开关设置。即只返回一个开关列表。

数据库系统监视器数据结构

系统监视器使用快照监视器和某些事件监视器的接口来收集和存储供您访问的信息。数据库系统监视器将收集的信息存储在称为*监视元素*（先前称为数据元素）的实体中。每个监视元素存储有关数据库系统状态的一个特定方面的信息。此外，监视元素由唯一名称标识并且存储特定类型的信息。

以下是系统监视器使用的元素类型，监视元素在其中存储数据:

计数器 计算活动的发生次数。在监视期间，计数器值会增大。大多数计数器元素可以复位。

标尺 指示某个项的当前值。根据数据库活动的不同，标尺值会增加或减小（例如，挂起的锁定数）。标尺元素不能复位。

水位标记
指示自开始监视以来元素所达到的最高值（最大值）或最低值（最小值）。水位标记元素不能复位。

信息 提供参考类型的监视活动详细信息。这可以包括诸如分区名称、别名和路径详细信息之类的项。信息元素不能复位。

时间戳记
通过提供自 1970 年 1 月 1 日后经历的秒数和微秒数，以指示活动发生的日期和时间。对于快照监视器和事件监视器，时间戳记元素的收集由 TIMESTAMP

监视开关控制。此开关在缺省情况下设置为 ON，如果数据库实例的 CPU 利用率达到 100%，那么考虑性能方面的原因应将其设置为 OFF。时间戳记元素不能复位。

时间戳记元素的值为零意味着“不可用”。如果尝试导入此数据，这样的值将生成超过范围错误（SQL0181）。为避免此错误，在导出数据前将该值更新为有效时间戳记值。

时间 返回执行活动时耗用的秒数和微秒数。对于快照监视器和事件监视器，大多数时间元素的收集由 TIMESTAMP 监视开关控制。此开关在缺省情况下设置为 ON，如果数据库实例的 CPU 利用率达到 100%，那么考虑性能方面的原因应将其设置为 OFF。某些时间元素可以复位。

监视元素收集一个或多个逻辑数据组的数据。逻辑数据组是一组监视元素，它们用来收集特定数据库活动作用域的数据库系统监视信息。监视元素在逻辑数据组中按它们提供的信息级别排序。例如，在进行快照监视时，“总排序时间”监视元素将返回数据库（dbase）、应用程序（appl）和语句（stmt）信息；因此，它将出现在已列示并且用圆括号括起来的每个逻辑数据组中。

尽管许多监视元素同时被快照监视器和事件监视器使用，但每个监视器使用一个不同的逻辑数据组集合。这是因为可对其捕获快照的数据库活动作用域与可对其收集事件数据的数据库活动作用域不同。从实际上说，可从快照监视器访问的完整监视元素集合不同于可从事件监视器访问的那些监视元素。

计数器状态和可视性

系统监视器收集的监视元素包括若干累加计数器。这些计数器将在数据库或数据库管理器操作期间递增，如每次应用程序落实事务时。

将在适用对象可用时初始化计数器。例如，对数据库读取的缓冲池页数（基本监视元素）在数据库激活时设置为零。

某些可以由系统监视器收集的计数器由监视开关控制。如果特定监视开关设置为“OFF”，那么受其控制的监视元素不会收集数据。当监视开关设置为“ON”时，所有关联计数器都将复位为零。

当事件监视器激活时，事件监视器返回的计数器将复位为零。

事件监视器计数表示自下列其中一个起始点之后的计数:

- 对数据库、表空间和表启动事件监视器时。
- 对现有连接启动事件监视器时。
- 应用程序连接，对于在启动监视器之后建立的连接。
- 在启动监视器之后又启动下一个事务（工作单元）或语句时。
- 在启动监视器之后发生死锁时。

每个事件监视器和任意监视应用程序（使用快照监视器 API 的应用程序）有自己的系统监视器数据逻辑视图。这意味着复位或初始化计数器时，将仅影响复位或初始化计数器的事件监视器或应用程序。除非关闭事件监视器然后再打开它，否则事件监视器计数器不能复位。获取快照的应用程序可使用 RESET MONITOR 命令随时复位其计数器视图。

如果在语句启动之后启动语句事件监视器，那么监视器将在启动下一条 SQL 语句时开始收集信息。因此，事件监视器不会返回有关启动监视器时数据库管理器正在执行的语句的信息。对于事务信息也如此。

系统监视器输出: 自描述数据流

除了在屏幕上显示系统监视器数据或将其存储在 SQL 表中之外，还可以开发客户机应用程序来进行处理。系统监视器通过自描述数据流同时对快照监视器和事件监视器返回监视器数据。在快照监视应用程序中，可调用快照 API 以捕获快照，然后直接处理数据流。

处理事件监视器数据的不同之处在于发生数据库事件时事件数据将发送至应用程序。对于管道事件监视器，应用程序将等待事件数据到达，并在事件数据到达时进行处理。对于文件事件监视器，应用程序将对事件文件进行语法分析，因此会分批处理事件记录。

此自描述数据流允许您对返回的数据进行语法分析，一次分析一个元素。这样就可以进行多方面的监视，包括查找有关特定应用程序或特定数据库状态的信息。

返回的监视器数据具有以下格式:

大小 存储在监视元素或逻辑数据分组中的数据的大小（以字节计）。如果是逻辑数据分组，那么此项为逻辑组中的所有数据的大小。例如，数据库逻辑分组（*db*）包含各个监视元素（如 *total_log_used*）及其他逻辑数据分组，如前滚信息信息（*rollforward*）。这不包括“大小”、“类型”和“元素”信息占用的大小。

type 存储在数据中的元素的类型（如可变长度字符串或带符号的 32 位数字值）。元素类型头指的是元素的逻辑数据分组。

元素标识
监视器捕获的监视元素的标识。如果是逻辑数据分组，那么此项为该组的标识（如 *collected*、*dbase* 或 *event_db*）。

数据 监视器对监视元素收集的值。如果是逻辑数据分组，那么该数据由属于它的监视元素组成。

监视元素中的所有时间戳记将以两个不带符号的 4 字节监视元素（秒和微秒）的形式返回。它们表示 GMT 时间 1970 年 1 月 1 日。

监视元素中的字符串的大小元素表示该字符串元素的实际数据大小。因为此字符串不是以 NULL 结束的，所以此大小不包括 NULL 终止符。

监视器数据的内存要求

将从监视器堆分配监视器数据所需的内存。监视器堆大小由 **mon_heap_sz** 数据库配置参数控制。此参数的缺省值为 AUTOMATIC，这表示监视器堆可以根据需要增大，直到达到 instance_memory 限制为止。

如果以手动方式配置 **mon_heap_sz** 参数，请考虑下列因素:

- 监视应用程序的数目
- 事件监视器的数目和特征
- 设置的监视开关

- 数据库活动级别

如果监视器命令失败并且 SQLCODE 为 -973，那么考虑增大 **mon_heap_sz** 参数的值。

以下公式提供监视器堆所需的大致页数:

```
(应用程序使用的内存量                    +
 事件监视器使用的内存量                  +
 监视应用程序使用的内存量                +
 网关应用程序使用的内存量)              / 4096
```

每个应用程序使用的内存量

- 如果 STATEMENT 开关设置为 OFF 或零
- 如果 STATEMENT 开关设置为 ON:
  - 为将要同时运行的每个语句加上 400 字节。(即，应用程序可能具有的打开游标数)。这不是应用程序已经运行的累积语句总数。
  - 如果是分区数据库，那么为每个语句加上以下大小:
    - 200 字节 * (平均子节数)
- 如果应用程序发出 sqleseti() info，那么加上用户标识、应用程序名称、工作站名称和记帐字符串大小。

每个事件监视器使用的内存量

对于每个类型为 ACTIVITIES 的事件监视器:

- 3500 字节
- 如果事件监视器的类型为 TABLES，那么加上 36K * (CPU 核心数 + 1)
- 如果事件监视器的类型为 FILE 或 PIPE，那么加上 2K * (CPU 核心数 + 1)

如果您预计工作量较大，那么为每个记录加上 250 兆字节。否则，根据您估计的工作量加上相应的数目。

对于每个类型为 LOCKING 或 UOW 的事件监视器:

- 3500 字节
- 3K * (CPU 核心数 + 1)

如果您预计工作量较大，那么为每个记录加上 250 兆字节。否则，根据您估计的工作量加上相应的数目。

对于每个具有以下类型的事件监视器:
DATABASE、TABLES、TABLESPACES、BUFFERPOOLS、CONNECTIONS 或 DEADLOCK:

- 4100 字节
- 2 * BUFFERSIZE
- 如果事件监视器写至文件，那么加上 550 字节。
- 如果事件监视器的类型为 DATABASE，那么:
  - 加上 6000 字节
  - 对语句高速缓存中的每个语句加上 100 字节
- 如果事件监视器的类型为 TABLES，那么:

  - 加上 1500 字节
  - 对访问的每个表加上 70 字节
- 如果事件监视器的类型为 TABLESPACES，那么:
  - 加上 450 字节
  - 对每个表空间加上 350 字节
- 如果事件监视器的类型为 BUFFERPOOLS，那么:
  - 加上 450 字节
  - 对每个缓冲池加上 340 字节
- 如果事件监视器的类型为 CONNECTIONS:
  - 加上 1500 字节
  - 对于每个已连接应用程序:
    - 加上 750 字节
  - 记住加上出自第 109 页的『每个应用程序使用的内存量』的值。
- 如果事件监视器的类型为 DEADLOCK:
  - 并且 WITH DETAILS HISTORY 正在运行:
    - 加上 X*475 字节，加上次数为希望运行的并发应用程序的最大数目，其中 X 是应用程序的工作单元中的最大期望语句数。
  - 并且 WITH DETAILS HISTORY VALUES 正在运行:
    - 同时加上 X*Y 字节，加上次数为希望运行的并发应用程序的最大数目，其中 Y 是将绑定至 SQL 语句的参数值的最大期望大小。

每个监视应用程序使用的内存量

- 250 字节
- 对于要复位的每个数据库:
  - 350 字节
  - 对每个远程数据库加上 200 字节。
  - 如果 SORT 开关设置为 ON，那么加上 25 字节。
  - 如果 LOCK 开关设置为 ON，那么加上 25 字节。
  - 如果 TABLE 开关设置为 ON:
    - 加上 600 字节
    - 对访问的每个表加上 75 字节
  - 如果 BUFFERPOOL 开关设置为 ON:
    - 加上 300 字节
    - 对访问的每个表空间加上 250 字节
    - 对访问的每个缓冲池加上 250 字节
  - 如果 STATEMENT 开关设置为 ON:
    - 加上 2100 字节
    - 对每个语句加上 100 字节
  - 对连接至数据库的每个应用程序:
    - 加上 600 字节
    - 对应用程序连接至的每个远程数据库加上 200 字节

- 如果 SORT 开关设置为 ON，那么加上 25 字节
- 如果 LOCK 开关设置为 ON，那么加上 25 字节
- 如果 BUFFERPOOL 开关设置为 ON，那么加上 250 字节

- 对于要复位的每个 DCS 数据库:
  - 对数据库加上 200 字节
  - 对连接至数据库的每个应用程序加上 200 字节
  - 如果 STATEMENT 开关设置为 ON，那么必须复位传输级别数据:
    - 对于每个数据库，对每个传输级别加上 200 字节
    - 对于每个应用程序，对每个传输级别加上 200 字节

网关应用程序使用的内存量

- 对每个主机数据库加上 250 字节（即使所有开关设置为 OFF）
- 对每个应用程序加上 400 字节（即使所有开关设置为 OFF）
- 如果 STATEMENT 开关设置为 ON:
  - 对于每个应用程序，对同时运行的每个语句加上 200 字节（即应用程序可能具有的打开游标数）。这不是应用程序已经运行的累积语句总数。
  - 必须计算传输级别数据:
    - 对于每个数据库，对每个传输级别加上 200 字节
    - 对于每个应用程序，对每个传输级别加上 200 字节
- 如果 UOW 开关设置为 ON:
  - 则对每个应用程序加上 50 字节
- 对于使用 TMDB 的每个应用程序（用于 SYNCPOINT TWOPHASE 活动）:
  - 加上 20 字节并加上 XID 本身的大小
- 对于已发出 sqleseti 以设置客户机名称、应用程序名称、wkstn 或记帐的任何应用程序:
  - 加上 800 字节并加上记帐字符串本身的大小

监视缓冲池活动

数据库服务器执行的所有数据读取和更新操作都是对缓冲池进行的。当应用程序需要数据时，就会将数据从磁盘复制到缓冲池。

下列程序将页放入缓冲池:

- 代理程序。这是同步 I/O。
- I/O 服务器（预取程序）。这是异步 I/O。

下列程序将页从缓冲池写入磁盘:

- 代理程序（同步方式）
- 页清除程序（异步方式）

如果服务器需要读取一页数据，并且该页已在缓冲池中，那么访问该页的速度要比从磁盘读取该页快。如果在缓冲池中能**命中**尽可能多的页，那就最好不过了。避免磁盘 I/O 操作对于提高数据库性能来说十分重要，因此，在调整性能时，正确地配置缓冲池是其中一项最重要的考虑事项。

对于页请求，如果该页已在缓冲池中，数据库管理器就不需要从磁盘装入该页便可以为该请求提供服务，缓冲池命中率指的就是此类情况所占的百分比。缓冲池命中率越高，磁盘 I/O 操作频率就会越低。

缓冲池整体命中率的计算公式如下:

```
    1 - ((pool_data_p_reads + pool_xda_p_reads +
pool_index_p_reads + pool_temp_data_p_reads
+ pool_temp_xda_p_reads + pool_temp_index_p_reads )
          / (pool_data_l_reads + pool_xda_l_reads + pool_index_l_reads +
          pool_temp_data_l_reads + pool_temp_xda_l_reads
    + pool_temp_index_l_reads )) * 100%
```

此计算公式考虑缓冲池高速缓存的所有页（索引和数据）。

另一种方便的方法是使用 BP_HITRATIO 管理视图来监视缓冲池命中率。

对于大型数据库来说，增加缓冲池大小对缓冲池命中率的影响极小。它的数据页数可能非常多，增加缓冲池大小并不会提高命中统计机率。而是，您会发现通过调整索引缓冲池命中率能获得期望的结果。这可以通过两种方法实现:

1. 将数据和索引分到两个不同的缓冲池中并分别对它们进行调整。
2. 使用一个缓冲池，但增加其大小，直到索引命中率不再提高为止。索引缓冲池命中率的计算公式如下:

```
(1 - ((pool_index_p_reads) / (pool_index_l_reads))) * 100%
```

第一种方法通常效率更高，但由于它要求索引和数据在不同的表空间中，所以对于现有数据库来说不能选择此方法。此方法还要求调整两个缓冲池（而不是只调整一个缓冲池），此任务可能更难以完成，当内存有限时尤其如此。

您还应该考虑预取程序对命中率的影响。预取程序将数据页读入缓冲池，即预计应用程序需要这些数据页（异步方式）。在大多数情况下，这些页刚好是在需要它们之前读取的（期望的情况）。但是，预取程序会将不会使用到的页读入缓冲池，从而执行不必要的 I/O 操作。例如，应用程序开始读整个表。这种情况被检测到，预取开始，但应用程序在填满应用程序缓冲区后停止读取。同时，预取操作已经读取了许多其他的页。已经为不会使用到的页执行了 I/O，那些页占用了缓冲池的部分空间。

页清除程序监视缓冲池并以异步方式将页写入磁盘。它们的目标是:

- 确保代理程序在缓冲池中总能找到可用页。如果代理程序在缓冲池中找不到可用页，它就必须自己清除它们，相关应用程序的响应速度就会变慢。
- 加快系统崩溃时的数据库恢复速度。已写入磁盘的页数越多，恢复数据库时必须处理的日志文件记录数就越少。

虽然会将脏页写入磁盘，但除非需要空间来读入新页，否则不会立即从缓冲池中除去这些页。

注: 缓冲池信息通常是在表空间级别收集的，但数据库系统监视器的工具可以提高到缓冲池级别和数据库级别来收集信息。根据分析类型的不同，可能需要在任何或全部这些级别检查此数据。

数据库系统监视器接口

监视任务	API
捕获快照	db2GetSnapshot
转换自描述数据流	db2ConvMonStream
显示数据库系统监视开关	db2MonitorSwitches
估计快照大小	db2GetSnapshotSize
获取/更新监视开关	db2MonitorSwitches
复位监视器计数器	db2ResetMonitor
更新数据库系统监视开关	db2MonitorSwitches

监视任务	CLP 命令
使用 GUI 工具分析事件监视器输出	db2eva
捕获快照	GET SNAPSHOT
显示数据库管理器系统监视开关	GET DATABASE MANAGER MONITOR SWITCHES
显示监视应用程序的监视开关	GET MONITOR SWITCHES
格式化事件监视器跟踪	db2evmon
对写至表 CREATE EVENT MONITOR 语句生成样本 SQL	db2evtbl
列示活动数据库	LIST ACTIVE DATABASES
列示连接至数据库的应用程序	LIST APPLICATIONS
列示 DCS 应用程序	LIST DCS APPLICATIONS
复位监视器计数器	RESET MONITOR
更新数据库系统监视开关	UPDATE MONITOR SWITCHES

监视任务	SQL 语句
激活事件监视器	SET EVENT MONITOR STATE
创建事件监视器	CREATE EVENT MONITOR
释放事件监视器	SET EVENT MONITOR STATE
除去事件监视器	DROP
写入事件监视器值	FLUSH EVENT MONITOR

监视任务	SQL 函数
确定事件监视器的状态	EVENT_MON_STATE 标量函数
获取数据库管理器级别快照	SNAPDBM 管理视图和 SNAP_GET_DBM_V95 表函数
在数据库管理器级别获取当前监视开关设置	SNAPSWITCHES 管理视图和 SNAP_GET_SWITCHES 表函数
获取快速通信管理器快照	SNAPFCM 管理视图和 SNAP_GET_FCM 表函数
获取给定分区的快速通信管理器快照	SNAPFCM_PART 管理视图和 SNAP_GET_FCM_PART 表函数
获取数据库级别快照	SNAPDB 管理视图和 SNAP_GET_DB_V95 表函数
获取应用程序级别快照	SNAPAPPL 管理视图和 SNAP_GET_APPL_V95 表函数
获取应用程序级别快照	SNAPAPPL_INFO 管理视图和 SNAP_GET_APPL_INFO_V95 表函数
获取锁定等待信息的应用程序级别快照	SNAPLOCKWAIT 管理视图和 SNAP_GET_LOCKWAIT 表函数

监视任务	SQL 函数
获取语句信息的应用程序级别快照	SNAPSTMT 管理视图和 SNAP_GET_STMT 表函数
获取代理程序信息的应用程序级别快照	SNAPAGENT 管理视图和 SNAP_GET_AGENT 表函数
获取子节信息的应用程序级别快照	SNAPSUBSECTION 管理视图和 SNAP_GET_SUBSECTION 表函数
获取缓冲池级别快照	SNAPBP 管理视图和 SNAP_GET_BP_V95 表函数
获取表空间级别快照	SNAPTBSP 管理视图和 SNAP_GET_TBSP_V91 表函数
获取配置信息的表空间级别快照	SNAPTBSP_PART 管理视图和 SNAP_GET_TBSP_PART_V91 表函数
获取容器信息的表空间级别快照	SNAPCONTAINER 管理视图和 SNAP_GET_CONTAINER_V91 表函数
获取停顿者信息的表空间级别快照	SNAPTBSP_QUIESCER 管理视图和 SNAP_GET_TBSP_QUIESCER 表函数
获取表空间映射范围的表空间级别快照	SNAPTBSP_RANGE 管理视图和 SNAP_GET_TBSP_RANGE 表函数
获取表级别快照	SNAPTAB 管理视图和 SNAP_GET_TAB_V91 表函数
获取锁定级别快照	SNAPLOCK 管理视图和 SNAP_GET_LOCK 表函数
获取 SQL 语句高速缓存信息的快照	SNAPDYN_SQL 管理视图和 SNAP_GET_DYN_SQL_V95 表函数

第 6 章 不推荐的监视工具

运行状况监视器

监视数据库运行状况

运行状况监视器简介

运行状况监视器是一个服务器端工具，它增添的“按异常情况进行管理”功能是通过不断监视实例和活动数据库的运行状况实现的。运行状况监视器还可以向数据库管理员（DBA）发出有关潜在系统运行状况问题的警报。运行状况监视器以前摄方式检测可能导致硬件故障或不可接受的系统系统性能或功能的问题。运行状况监视器的前摄特征使用户能够在发现的问题影响系统性能之前解决该问题。

运行状况监视器使用运行状况指示器来检查系统状态，以确定是否应该发出警报。可对应警报执行先前配置的操作。运行状况监视器还会将警报记录在管理通知日志中，并通过电子邮件或寻呼机发送通知。这一“按异常情况进行管理”模型对潜在系统运行状况问题生成警报而不需要活动监视，从而让 DBA 从事更有价值的工作。

运行状况监视器定期收集有关系统运行状况的信息，这对整体性能的影响非常小。它不会打开任何快照监视开关来收集信息。

运行状况指示器:

运行状况监视器使用运行状况指示器来评估数据库管理器性能或数据库性能特定方面的运行状况。运行状况指示器测量特定种类数据库对象（例如表空间）某些方面的运行状况。对度量应用条件以确定运行状况。应用的条件视运行状况指示器类型而定。运行状况是否正常是根据生成警报的条件确定的。

运行状况监视器返回三种类型的运行状况指示器:

- **基于阈值**的指示器是对象行为的（连续值范围）统计信息度量。警告阈值和警报阈值定义了正常、警告和警报范围的边界或区域。基于阈值的运行状况指示器有三种有效状态: “正常”、“警告”或“警报”。
- **基于状态**的指示器是有限状态集（包含一个对象的两种或更多种不同状态）度量，该状态集定义了数据库对象或资源的工作是否正常。其中一种状态表示情况正常，所有其他状态都被视为不正常。基于状态的运行状况指示器有两种有效状态: “正常”和“注意”。
- **基于集合状态**的指示器是数据库级度量，它们代表数据库中一个或多个对象的聚集状态。为该集合中的每个对象捕获数据，那些对象中最严重的情况以聚集状态表示。如果该集合中有一个或多个对象处于要求发出警报的状态，运行状况指示器就会显示“注意”状态。基于集合状态的运行状况指示器有两种有效状态: “正常”和“注意”。

在实例级、数据库级、表空间级和表空间容器级都存在运行状况指示器。

可以通过运行状况中心、CLP 或 API 来访问运行状况监视器信息。可以通过这些工具来配置运行状况指示器。

当从正常状态切换到非正常状态，或者运行状况指示器值进入警告或警报区域（基于已定义的阈值边界），就会生成警报。有三种类型的警报：注意、警告和警报。

- 对于量度不同状态的运行状况指示器来说，如果注册了非正常状态，就会发出注意警报。
- 对于量度连续值范围的运行状况指示器来说，阈值定义了正常状态、警告状态和警报状态的边界或区域。例如，如果值进入定义警报区域的阈值范围，就会发出警报类型的警报以指示问题需要立即加以注意。

对于给定的运行状况指示器，运行状况监视器仅在特定报警条件第一次出现时发送通知并运行操作。如果运行状况指示器一直处于特定报警条件下，就不会发送更多通知，也不进一步运行操作。如果运行状况指示器更改了报警条件，或者回到正常状态并重新进入报警条件，就会发送新通知并运行操作。

下表显示了不同刷新时间间隔的运行状况指示器示例以及运行状况监视器对运行状况指示器状态的响应。此示例使用缺省警告阈值和警报阈值，它们分别是 80% 和 90%。

表 45. 不同刷新时间间隔的运行状况指示器条件

刷新时间间隔	**ts.ts_util（表空间利用率）运行状况指示器的值**	**ts.ts_util 运行状况指示器状态**	**运行状况监视器响应**
1	80	警告	发送警告通知，运行警告报警条件的操作
2	81	警告	不发送通知，不运行操作
3	75	正常	不发送通知，不运行操作
4	85	警告	发送警告通知，运行警告报警条件的操作
5	90	警报	发送警报通知，运行警报条件的操作

运行状况指示器处理周期:

下图说明运行状况指示器的求值过程。每次经历运行状况指示器特定的刷新时间间隔时，就会运行这一组步骤。

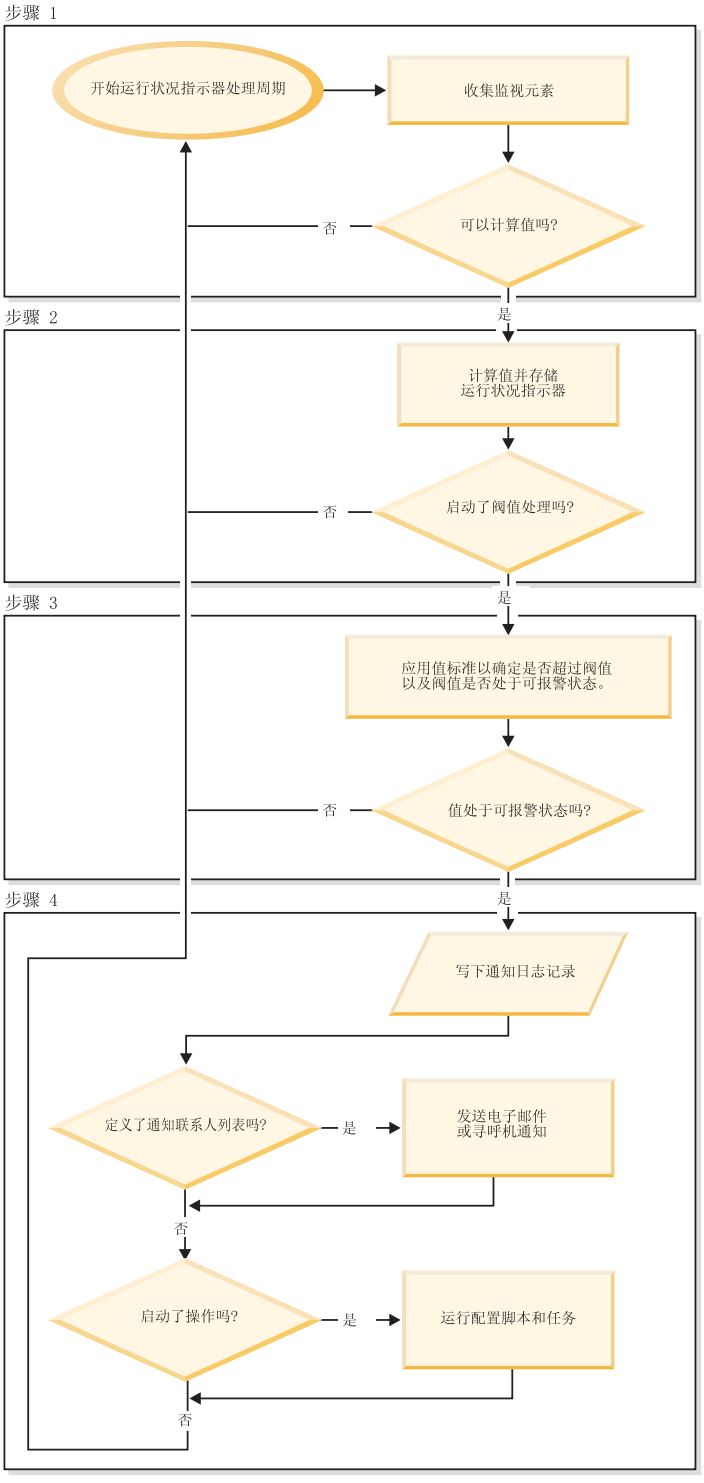


图 4. 运行状况指示器处理周期

注:

1. NOTIFYLEVEL 数据库管理器配置参数控制是将警报通知发送至 DB2 管理通知日志还是发送至所有定义的联系人。警报通知的最低严重性级别需要为 2。要发送警告和注意警报，最低严重性级别需要为 3。

启用运行状况警报通知:

要在生成警报时启用电子邮件或寻呼机通知，必须设置配置参数并指定联系人信息。

DB2 管理服务器（DAS）必须正在联系人列表所在的系统上运行。例如，如果 CONTACT_HOST 配置参数设置为远程系统，那么为了获取要通知有关警报的联系人，DAS 必须在远程系统上运行。

要启用运行状况报警通知:

1. 指定 SMTP_SERVER 参数。 DAS 配置参数 SMTP_SERVER 指定发送电子邮件和寻呼机通知消息时使用的邮件服务器的位置。如果安装 DB2 数据库的系统是作为未授权 SMTP 服务器启用的，那么省略此步骤。
2. 指定 CONTACT_HOST 参数。 DAS 配置参数 CONTACT_HOST 指定本地系统上所有实例的联系人列表的远程位置。通过设置此参数，可在多个系统间共享单个联系人列表。如果想要将联系人列表保留在安装 DB2 数据库的本地系统上，那么省略此步骤。
3. 指定运行状况监视器通知的缺省联系人。为了能够在生成警报时从运行状况监视器获取电子邮件或寻呼机通知，必须指定缺省管理联系人。如果选择不提供此信息，那么不会对报警条件发送通知消息。可在安装期间提供缺省管理联系人信息，也可以将此任务延迟至安装完成之后进行。 如果选择延迟该任务或者想要将更多联系人或组添加至通知列表，那么可以通过 CLP、C API 或运行状况中心来指定联系人:

   - **要使用 CLP 指定联系人**:

     要将电子邮件联系人定义为运行状况监视器通知的缺省联系人，请发出以下命令:

     ```
     DB2 ADD CONTACT contact_name TYPE EMAIL ADDRESS
        email_address DESCRIPTION 'Default Contact'

     DB2 UPDATE NOTIFICATION LIST ADD CONTACT contact_name
     ```

     有关完整的语法详细信息，请参阅 Command Reference。

   - **要使用 C API 指定联系人**:

     下列 C 代码摘录说明如何定义运行状况通知联系人:

     ```
     ...
     #include <db2ApiDf.h>

     SQL_API_RC rc = 0;
     struct db2AddContactData addContactData;
     struct sqlca sqlca;

     char* userid = "myuser";
     char* password = "pwd";
     char* contact = "DBA1";
     char* email = "dba1@mail.com";
     char* desc = "Default contact";

     memset(&addContactData, '\0', sizeof(addContactData));
     ```

```
memset (&sqlca, '\0', sizeof(struct sqlca));
addContactData.piUserid = userid;
addContactData.piPassword = password;
addContactData.piName = contact;
addContactData.iType = DB2CONTACT_EMAIL;
addContactData.piAddress = email;
addContactData.iMaxPageLength = 0;
addContactData.piDescription = desc;

rc = db2AddContact(db2Version810, &addContactData, &sqlca);

if (rc == 0) {
      db2HealthNotificationListUpdate update;
      db2UpdateHealthNotificationListData data;
      db2ContactTypeData contact;

      contact.pName = contact;
      contact.contactType = DB2CONTACT_EMAIL;

      update.iUpdateType = DB2HEALTHNOTIFICATIONLIST_ADD;
      update.piContact = &contact;

      data.iNumUpdates = 1;
      data.piUpdates = &update;

   rc = db2UpdateHealthNotificationList (db2Version810, &data, &ca);
}
...
```

- **要使用运行状况中心指定联系人:**

  a. 右键单击想要对其定义运行状况通知列表的实例。

  b. 单击**配置**，然后单击**警报通知**。“配置运行状况报警通知”窗口将打开。

  c. 如果联系人未出现在窗口左边的**可用**列表中，那么单击**管理联系人**。“联系人”窗口打开，并且带有预选系统名称。

  d. 单击**添加联系人**。“添加联系人”窗口将打开。

  e. 通过提供名称和电子邮件地址来定义联系人。如果指定电子邮件地址是寻呼机，那么选择**将地址用于寻呼机**。

  f. 单击**确定**。

  g. 关闭“联系人”窗口并返回至“配置运行状况报警通知”窗口。新的联系人将出现在**可用联系人**列表中。

  h. 通过单击向右方向按钮，将联系人移至**运行状况通知联系人列表**。

  i. 单击**确定**以将该联系人包括在运行状况通知列表中。

  建议 如果在通知时遇到困难，那么选择运行状况通知联系人列表下面的**故障诊断**。“运行状况报警通知故障诊断”向导将打开。

运行状况中心概述

使用运行状况中心分析和改进 DB2 的运行状况。

以下是定义如何让 DB2 正常工作的条件示例:

- 有足够的资源（如可用内存、表空间容器或记录存储器）来完成任务。
- 资源使用效率较高。

- 执行任务的时间在可接受范围以内，或者任务的执行不会导致性能显著下降。
- 资源或数据库对象不会无限期处于不可用状态。

要点：版本 9.7 中已经不推荐使用“运行状况中心”，在以后的发行版中可能会将其除去。 有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

还可从运行状况中心打开其他中心和工具，以帮助您调查和维护数据库的运行状况。

要在 Intel® 平台上打开运行状况中心，从**开始**菜单单击**开始 → 程序 → IBM DB2 → 监视工具 → 运行状况中心**。

要在 Intel 平台上使用命令行打开运行状况中心，请运行以下命令:

```
db2hc
```

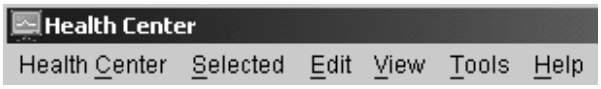
以下列表对可以在运行状况中心中执行的一些关键任务进行了分类:

- 第 118 页的『启用运行状况警报通知』
  - 指定联系人设置和通知配置参数
  - 故障诊断运行状况报警通知
- 第 141 页的『使用运行状况中心配置运行状况指示器』
  - 启用和禁用运行状况指示器求值
  - 更改警报阈值和灵敏度设置
  - 发生警报时运行任务和脚本
- 第 135 页的『使用运行状况中心解决运行状况监视器警报的问题』
  - 使用建议顾问程序来选择和实施建议

运行状况中心界面

运行状况中心界面具有下列元素，可以帮助您确定和解决与系统的整体运行状况有关的问题。

运行状况中心菜单栏



使用菜单栏在“运行状况中心”中处理对象，打开其他管理中心和工具和访问联机帮助。

运行状况中心菜单栏包含下列菜单:

运行状况中心工具栏

使用菜单栏下面的工具栏图标来访问其他中心和工具，以及刷新“运行状况中心”的内容视图。

切换按钮

使用切换按钮来选择“导航”视图中出现的警报状态。每个按钮对应一个数据库对象在视图中显示时需要的最低警报严重性。选择不同的按钮只会影响显示内容，不会影响该对象本身。

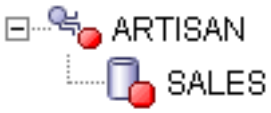
 显示处于警报状态的对象

 显示处于警报和警告状态的对象

 显示处于以下任一警报状态的对象：警报、警告、注意、正常和未监视。

 显示所有对象

导航视图



使用“导航”视图来显示和处理实例和数据库对象。当选择“导航”视图中的对象时，该对象及其所有子代的当前警报将显示在“警报”视图中。要在“导航”视图显示该对象之前更改该对象必须具有的级别，在“导航”视图中右键单击远离列示对象的位置。这将会打开警报级别的弹出菜单。选择想要显示的警报级别。还可通过单击切换按钮来选择要显示的警报级别。

警报视图

使用“警报”视图来显示和处理当前警报。“警报”视图显示在“导航”视图中选择的对象及其子代数据库对象当前存在的警报。例如，如果选择某个实例，那么会显示该实例及其所有数据库和表空间的警报。如果选择某个数据库，那么会显示该数据库及其所有表空间的警报。在“警报”视图中选择并右键单击一个或多个警报以对其调用操作。

警报视图工具栏

使用“警报”视图下面的工具栏来调整“警报”视图中的警报视图以满足您的需要。

调查报警条件:

运行状况监视器

运行状况监视器捕获有关数据库管理器、数据库、表空间和表空间容器的信息。运行状况监视器根据从数据库系统监视元素、操作系统和 DB2 数据库检索到的信息计算运行状况指示器的各项指标。仅当数据库活动时，运行状况监视器才能对数据库及其对

象计算运行状况指示器的各项指标。可通过使用 ACTIVATE DATABASE 命令启动数据库或保持与数据库的永久连接，以便让数据库保持活动状态。

运行状况监视器对每个运行状况指示器保留最大历史记录数。此历史记录存储在 <instance path>\hmonCache 目录中并且在停止运行状况监视器时除去。当达到最大记录数时，运行状况监视器将自动删除过时的历史记录。

运行状况监视器数据可通过运行状况快照访问。每个运行状况快照按最新刷新时间间隔报告每个运行状况指示器的状态。在检测现有数据库运行状况问题和预测数据库环境可能出现的不良运行状况时，快照非常有用。可以使用 C 或 C++ 应用程序中的 API 或者图形管理工具来从 CLP 捕获运行状况快照。

运行状况监视需要实例连接。如果未使用 ATTACH TO 命令建立与实例的连接，那么将创建与本地实例的缺省连接。

在分区数据库环境中，可在实例的任何分区上获取快照，或者使用单个实例连接获取全局快照。全局快照聚集在每个分区上收集到的数据并返回一组值。

用法说明

DB2 数据库的所有版本都支持运行状况监视器。

要从运行状况中心启动或停止运行状况监视器，右键单击运行状况中心导航视图中的某个实例，然后选择“启动运行状况监视器”或“停止运行状况监视器”。

在 Windows 上，DB2 实例的服务需要在具有 SYSADM 权限的帐户下运行。可在 db2icrt 命令上使用 **-u** 选项，或在 Windows 上使用“服务”文件夹然后编辑“登录”属性以使用带有管理员特权的帐户。

运行状况监视器将作为 DB2 受防护方式进程运行。这些进程在 Windows 上显示为 DB2FMP。在其他平台上，运行状况监视器进程显示为 DB2ACD。

DB2 管理服务器必须正在运行状况监视器所在的系统上运行，才能发送通知和运行警报操作。如果使用远程脚本、任务或联系人列表，那么必须同时启动远程系统上的 DB2 管理服务器。

只有创建任务时才需要工具目录数据库。如果不对任何运行状况指示器使用警报任务操作，那么运行状况监视器不需要工具目录数据库。

如果从更新版本的 DB2 数据库系统回退到 DB2 UDB 版本 8.1，那么所作的所有注册表更改将会丢失。注册表复原为 V8.1 HealthRules.reg 文件，该文件包含使用更新注册表文件升级并启动之前存在的设置。

运行状况指示器数据: 运行状况监视器对每个数据库分区上的每个运行状况指示器记录一组数据，包括:

- 运行状况指示器名称
- 值
- 求值时间戳记
- 警报状态
- 公式（如果适用）
- 附加信息（如果适用）

- 最多 10 个最新运行状况指示器求值历史记录。每个历史记录条目捕获导致当前运行状况指示器输出的下列运行状况指示器求值:
  - 值
  - 公式（如果适用）
  - 警报状态
  - 时间戳记

运行状况监视器还会跟踪实例、数据库和表空间级别的最高严重性警报状态。在每一级别，此运行状况指示器表示运行状况指示器在该级别或该级别之下的任何级别存在的最高严重性警报。例如，实例的最高严重性警报状态包括实例、该实例的任何数据库和每个数据库的任何表空间容器上的运行状况指示器。

捕获数据库运行状况快照:

使用 SQL 表函数捕获数据库运行状况快照:

可使用 SQL 表函数来捕获数据库运行状况快照。每个可用的运行状况快照表函数对应一个运行状况快照请求类型。

要使用 SQL 表函数来捕获数据库运行状况快照:

1. 标识计划使用的 SQL 表函数。

   SQL 表函数有两个输入参数:

   - VARCHAR(255)，用于数据库名称
   - INT，用于分区号（0 到 999 之间的值）。输入与要监视的分区号相对应的整数。要捕获当前连接的分区的快照，请输入值 -1。要捕获全局快照，请输入值 -2。

   注: 对于此规则，数据库管理器快照 SQL 表函数是一个例外，原因是它们只有一个参数。单一参数用于分区号。如果对数据库名称参数输入 NULL，那么监视器使用连接定义的数据库，表函数就是通过该连接调用的。
2. 发出 SQL 语句。

   以下示例捕获当前连接的分区的基本运行状况快照，以及连接定义的数据库上的基本运行状况快照（通过该连接调用此表函数）:

   ```
   SELECT * FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1))
                as HEALTH_DB_INFO
   ```

   您也可以从返回的表中选择个别监视元素。返回的表中的每一列对应一个监视元素。相应地，监视元素列名直接对应监视元素名称。以下语句仅返回数据库路径和服务器平台监视元素:

   ```
   SELECT db_path, server_platform
                FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1 ) )
                as HEALTH_DB_INFO
   ```

使用 CLP 捕获数据库运行状况快照:

可从 CLP 使用 GET HEALTH SNAPSHOT 命令来捕获运行状况快照。该命令语法支持检索运行状况监视器监视的不同对象类型的运行状况快照信息。

必须具有实例连接才能捕获运行状况快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

要使用 CLP 捕获数据库运行状况快照

1. 从 CLP 发出带有期望参数的 GET HEALTH SNAPSHOT 命令。

   在以下示例中，将在启动数据库管理器之后立即捕获数据库管理器级别运行状况快照。

   ```
   db2 get health snapshot for dbm
   ```

2. 对于分区数据库系统，可为特定分区捕获专门的数据库快照，或者为所有分区捕获全局的数据库快照。要对特定分区（如分区号 2）上的数据库捕获运行状况快照，请发出以下命令:

   ```
   db2 get health snapshot for db on sample at dbpartitionnum 2
   ```

   要对所有分区上的所有应用程序捕获数据库快照，请发出以下命令:

   ```
   db2 get health snapshot for db on sample global
   ```

   以下命令捕获的运行状况快照带有附加详细信息，包括公式、附加信息和运行状况指示器历史记录:

   ```
   db2 get health snapshot for db on sample show detail
   ```

3. 对于基于集合状态的运行状况指示器，可对所有集合对象捕获数据库快照，而不考虑这些对象的状态。常规 GET HEALTH SNAPSHOT FOR DB 命令返回所有集合对象，这些对象需要针对所有基于集合状态的运行状况指示器的警报。

   要对列示了所有集合对象的数据库捕获运行状况快照，请发出以下命令:

   ```
   db2 get health snapshot for db on sample with full collection
   ```

从客户机应用程序捕获数据库运行状况快照:

可使用 C 或 C++ 应用程序中的快照监视器 API 来捕获运行状况快照。可通过在 db2GetSnapshot API 中指定参数来访问若干不同运行状况快照请求类型。

必须连接至实例才能捕获运行状况快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

1. 将 sqlmon.h 和 db2ApiDf.h DB2 库包括在代码中。这些库可在 sqllib\include 目录中找到。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. 将快照缓冲区单元大小设置为 50 KB。

   ```
   #define SNAPSHOT_BUFFER_UNIT_SZ 51200
   ```

3. 声明 sqlma、sqlca、sqlm_collected 和 db2GetSnapshotData 结构。

   ```
   struct sqlma *pRequestedDataGroups;
   struct sqlca sqlca;
   memset (&sqlca, '\0', sizeof(struct sqlca));
   struct sqlm_collected collectedData;
   memset (&sqlm_collected, '\0', sizeof(struct sqlm_collected));
   db2GetSnapshotData getSnapshotParam;
   memset(&db2GetSnapshotData, '\0', sizeof(db2GetSnapshotData));
   ```

4. 初始化指针以包含快照缓冲区并确定缓冲区大小。

   ```
   static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
   sqluint32 outputFormat;
   char *snapshotBuffer;
   ```

5. 初始化 sqlma 结构，并指定要捕获的快照属于数据库管理器级别信息。

```
pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
memset(&pRequestedDataGroups, '\0', sizeof(struct pRequestedDataGroups));
pRequestedDataGroups->obj_num = 1;
pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
```

6. 初始化缓冲区以容纳快照输出。

```
snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (&snapshotBuffer, '\0', sizeof(snapshotBuffer));
```

7. 使用快照请求类型（来自 sqlma 结构）、缓冲区信息和捕获快照所需的其他信息来填充 db2GetSnapshotData 结构。

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_HEALTH;
```

8. 捕获运行状况快照。传递下列参数:
   - db2GetSnapshotData 结构，它包含捕获快照所需的信息
   - 对缓冲区的引用，快照输出将引导至该缓冲区。

```
db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. 包括用于处理缓冲区溢出的逻辑。获取快照后，将检查 sqlcode 是否存在缓冲区溢出。如果发生了缓冲区溢出，那么将清除并重新初始化缓冲区，然后再次获取快照。

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
  snapshotBufferSize += SNAPSHOT_BUFFER_UNIT_SZ;
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
  if (snapshotBuffer == NULL)
  {
    printf("\nMemory allocation error.\n");
    return;
  }

  getSnapshotParam.iBufferSize = snapshotBufferSize;
  getSnapshotParam.poBuffer = snapshotBuffer;
  db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. 处理快照监视器数据流。参考遵循这些步骤之后的图形就会看到快照监视器数据流。
11. 清除缓冲区。

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

在使用 db2GetSnapshot API 捕获运行状况快照之后，该 API 以自描述数据流的形式返回运行状况快照输出。以下示例显示数据流结构:

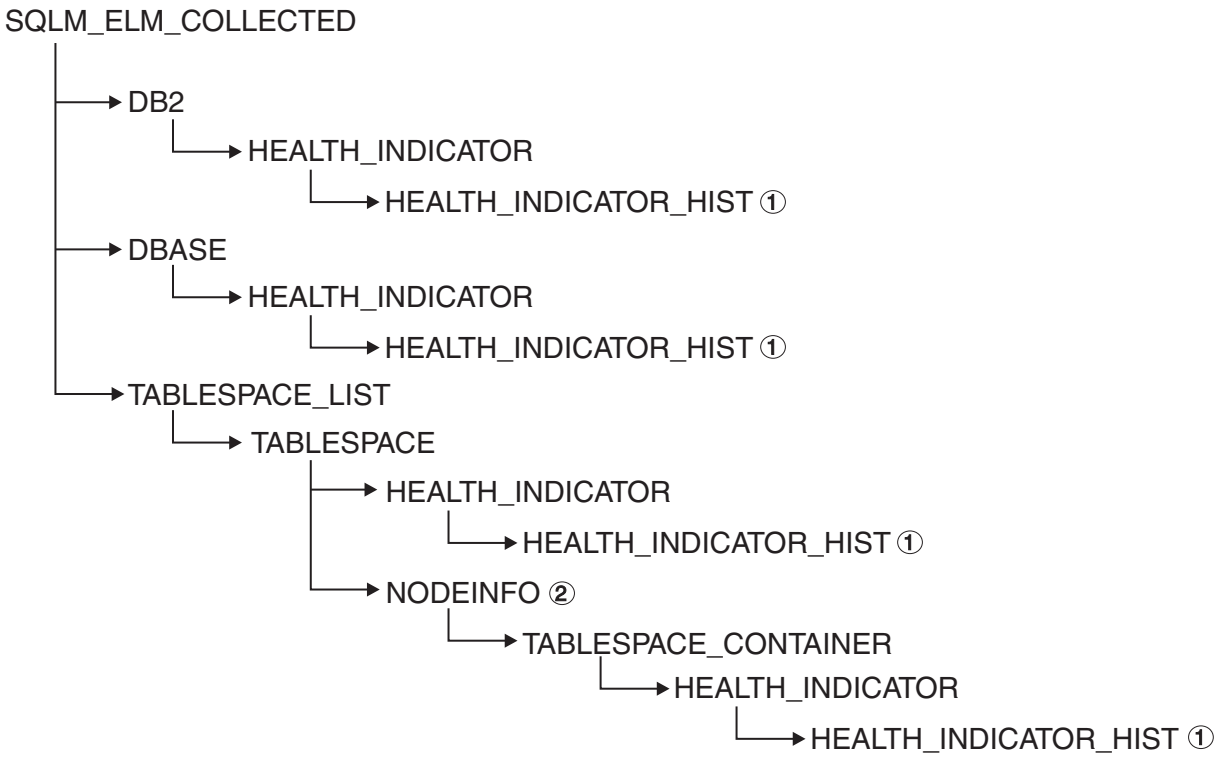


图 5. 运行状况快照自描述数据流

图注:

1. 只有在使用 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类时才可用。
2. 只有在 DB2 企业服务器版中才可用。否则为表空间容器流。

下列层次结构显示运行状况快照自描述数据流中的特定元素。

SQLM_ELM_HI 中的各个元素的层次结构:

```
SQLM_ELM_HI
   SQLM_ELM_HI_ID
   SQLM_ELM_HI_VALUE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
   SQLM_ELM_HI_ALERT_STATE
```

SQLM_ELM_HI_HIST 中的各个元素的层次结构，仅对 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类可用:

```
SQLM_ELM_HI_HIST
   SQLM_ELM_HI_FORMULA
   SQLM_ELM_HI_ADDITIONAL_INFO
      SQLM_ELM_HEALTH_INDICATOR_HIST
         SQLM_ELM_HI_ID
         SQLM_ELM_HI_VALUE
            SQLM_ELM_HI_TIMESTAMP
               SQLM_ELM_SECONDS
               SQLM_ELM_MICROSEC
         SQLM_ELM_HI_ALERT_STATE
         SQLM_ELM_HI_FORMULA
         SQLM_ELM_HI_ADDITIONAL_INFO
```

SQLM_ELM_OBJ_LIST 中的各个元素的层次结构:

```
SQLM_ELM_HI_OBJ_LIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_DETAIL
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

SQLM_ELM_OBJ_LIST_HIST 中的各个元素的层次结构，仅对 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类可用:

```
SQLM_ELM_HI_OBJ_LIST_HIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

运行状况监视器样本输出:

下列示例显示使用 CLP 获取的运行状况快照及其相应输出，并说明运行状况监视器的特征。这些示例的目的是启动数据库管理器后立即检查整体运行状况状态。

1. 使用 GET HEALTH SNAPSHOT 命令获取数据库管理器快照:

   ```
   db2 get health snapshot for dbm
   ```

   在从 CLP 发出 GET HEALTH SNAPSHOT 命令后，快照输出将引导至屏幕。

   ```
   节点名                                  =
   节点类型                                = 带有本地和远程客户机的数据库服务器
   实例名                                  = DB2
   快照时间戳记                            = 11-07-2002 12:43:23.613425

   DB2 实例中的数据库分区数                = 1
   启动数据库管理器时间戳记                = 11-07-2002 12:43:18.000108
   实例最高严重性警报状态                  = 尚未求值

   运行状况指示器:

         尚未求值
   ```

2. 分析输出。从此运行状况快照中，可以看到实例最高严重性警报状态为“Not yet evaluated”。实例处于此状态的原因是运行状况监视器刚刚启动，还没有计算任何运行状况指示器的各项指标。

   如果实例最高严重性警报状态未更改，那么:

   - 检查 HEALTH_MON 数据库管理器配置参数的值以确定运行状况监视器是否已启动。
   - 如果 HEALTH_MON=OFF，那么表示运行状况监视器未启动。要启动运行状况监视器，请发出 UPDATE DBM CFG USING HEALTH_MON ON 命令。
   - 如果 HEALTH_MON=ON，那么连接至实例以激活运行状况监视器。如果存在实例连接，那么可能不能将运行状况监视器装入到内存中。

使用 CLP 获取数据库运行状况快照的另一示例如下所述。

1. 开始之前应确保数据库连接存在并且数据库已停顿。
2. 使用 GET HEALTH SNAPSHOT 命令获取数据库管理器快照:

   ```
   db2 get health snapshot for db on sample
   ```

3. 在从 CLP 发出 GET HEALTH SNAPSHOT 命令后，快照输出将引导至屏幕。

```
                数据库运行状况快照

快照时间戳记                                = 12-09-2002 11:44:37.793184

数据库名称                                  = SAMPLE
数据库路径                                  = E:\DB2\NODE0000\SQL00002\
输入数据库别名                              = SAMPLE
在数据库服务器上运行的操作系统              = NT
数据库的位置                                = 本地
数据库最高严重性警报状态                    = 注意

运行状况指示器:

...
   指示器名称                               = db.log_util
      值                                    = 60
      单位                                  = %
      求值时间戳记                          = 12-09-2002 11:44:00.095000
      警报状态                              = 正常

   指示器名称                               = db.db_op_status
      值                                    = 2
      求值时间戳记                          = 12-09-2002 11:44:00.095000
      警报状态                              = 注意
```

4. 分析输出。

   此运行状况快照显示 *db.db_op_status* 运行状况指示器上存在注意警报。值 2 指示数据库处于停顿状态。

全局运行状况快照:

在分区数据库系统上，可获取当前分区、指定分区或所有分区的运行状况快照。对分区数据库的所有分区获取全局运行状况快照时，会尽可能地先聚集数据，然后返回结果。

运行状况指示器的聚集警报状态相当于所有数据库分区上的最高严重性警报状态。不能聚集各个数据库分区上的附加信息和历史记录数据，因此不会提供它们。运行状况指示器的余下数据的聚集将在下表中作详细描述。

表 46. 运行状况指示器值、时间戳记和公式数据的聚集

运行状况指示器	聚集详细信息
• db2.db2_op_status • db2.sort_privmem_util • db2.mon_heap_util • db.db_op_status • db.sort_shrmem_util • db.spilled_sorts • db.log_util • db.log_fs_util • db.locklist_util • db.apps_waiting_locks • db.db_heap_util • db.db_backup_req • ts.ts_util	从包含最高值的分区获取运行状况指示器值。 从同一分区获取求值时间戳记和公式。
• db.max_sort_shrmem_util • db.pkgcache_hitratio • db.catcache_hitratio • db.shrworkspace_hitratio	从包含最低值的分区获取运行状况指示器值。 从同一分区获取求值时间戳记和公式。
• db.deadlock_rate • db.lock_escal_rate	运行状况指示器值是所有数据库分区上的值的总和。 不能聚集求值时间戳记和公式，所以不提供它们。
• ts.ts_op_status • tsc.tscont_op_status • tsc.tscont_util	不聚集这些运行状况指示器。
• db.hadr_op_status • db.hadr_log_delay	多分区数据库不支持这些运行状况指示器。
• db.tb_reorg_req • db.tb_runstats_req • db.fed_nicknames_op_status • db.fed_servers_op_status	仅在一个分区上对此运行状况指示器求值，所以不需要聚集。从对运行状况指示器求值的分区返回数据。

注: 对单个分区对象获取全局快照时，输出将包括所有属性，这是因为没有要聚集的分区。

运行状况监视器的图形工具: 运行状况中心

运行状况中心是一个图形管理工具，用于支持按异常情况进行管理。对于在客户机上编目的所有 Windows、Linux 及 UNIX 实例和数据库，运行状况中心提供:

- 用于查看所有实例及其数据库的累积警报状态的中央位置
- 用于查看实例和数据库及其子对象的当前警报的图形界面
- 用于访问当前警报的详细信息和建议解决措施的图形界面

要从命令行启动运行状况中心，请输入 db2hc 命令。

要点：版本 9.7 中已经不推荐使用“运行状况中心”，在以后的发行版中可能会将其除去。有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

在 Windows 上，还可通过单击**开始 → 程序 → IBM DB2 →<DB2 copy name>→ 监视工具 → 运行状况中心**从“开始”菜单启动运行状况中心。

运行状况中心的左边面板中是导航树，右边面板中是“警报”视图。“导航”视图的内容将根据在“导航”视图顶部选择的切换按钮进行过滤。

运行状况中心打开时选择了**处于任何警报状态的对象**切换按钮，这有助于标识带有应该解决的当前警报的实例。选择**所有对象**切换按钮后，将显示在客户机上编目的所有的 Windows、Linux 和 UNIX 实例及它们各自的状态。没有图标的实例表示没有正在运行的运行状况监视器，或者是版本 8 以前的实例（缺少对运行状况监视器功能的支持）。

选择实例时，运行状况中心将从运行状况监视器请求所选实例的状态。“警报”视图将填写实例、该实例的任何数据库和每个数据库的任何表空间和表空间容器上的所有当前警报。如果在“导航”视图中展开实例并选择子代数据库对象，那么“警报”视图将被限制为仅显示所选数据库及其任意表空间或表空间容器的警报。

刷新图标在运行状况中心的右上角。单击刷新图标以便立即刷新，或者设置特定刷新时间间隔，这将导致运行状况中心在服务器上查询运行状况监视器的当前状态。此查询不会导致运行状况监视器刷新运行状况指示器求值。每个运行状况指示器都具有定义的刷新时间间隔。仅当经过刷新时间间隔时，才会重新对运行状况指示器的警报状态求值。每次定时刷新或请求刷新运行状况中心时，仅显示运行状况指示器的当前状态。

“警报”视图具有一个功能，它可以定义带有特定定制列和排序顺序的定制视图。运行状况中心中有六种预定义视图可用来定制个人命名和分类方案。可通过使用窗口底部的工具栏或选择**视图**菜单中的**已保存**视图来选择预定义视图。要定义您自己的定制视图，单击窗口底部工具栏上的**视图**按钮，或者使用**视图**菜单。在下一次调用运行状况中心时，记住为显示“警报”视图中的数据而选择的视图。

要获取警报的详细信息，在“警报”视图中选择警报行。使用**已选择**菜单，或右键单击该行并选择**显示详细信息**。详细信息窗口显示警报的详细信息，包括发生警报的对象和分区、公式（如果适用）和运行状况指示器的值。

对于基于阈值的运行状况指示器，将显示用于确定报警条件的阈值。详细信息窗口还将显示运行状况指示器的附加信息。此信息可能包括配置参数的值或提供警报上下文的其他监视器数据。将显示对运行状况指示器的描述，包括运行状况指示器的用途及其作为重要度量属性的原因。

对于基于集合状态的运行状况指示器，集合对象列表将显示在**运行状况指示器警报状态**表的“对象”中。表中将提供对象名、状态、时间戳记和详细信息。

详细信息页上提供了查看历史记录按钮。将从运行状况指示器求值的第二次刷新开始存储运行状况指示器的历史记录。只有在存储历史记录后才会在运行状况中心的“查看历史记录”对话框中显示内容。对于基于集合状态的运行状况指示器，可通过单击历史记录窗口中的**查看集合历史记录**按钮来查看集合对象的历史记录。

运行状况中心状态信标

“运行状况中心状态信标”是可在 DB2 管理工具中启用的可视指示器。当运行状况中心未打开并且您在使用其他 DB2 管理工具时，该信标将向您通知当前警报。该信标用于因为报警条件而提示用户打开运行状况中心。

“运行状况中心状态信标”有两种不同的通知方法。一种通知方法使用弹出消息。另一种通知方法使用显示在打开窗口的状态行右边部分的图形信标。图形信标包括一个按钮，单击该按钮可访问运行状况中心。

两种信标通知方法都可以通过“工具设置”对话框启用。“通过弹出消息通知”方法控制弹出消息通知，而“通过状态行通知”方法控制可视信标。

检索运行状况建议:

使用 SQL 执行的运行状态建议查询:

可借助使用 SYSPROC.HEALTH_HI_REC 存储过程的 SQL 来查询建议。

使用 SYSPROC.HEALTH_HI_REC 存储过程时，将在如下 XML 文档中返回建议:

- 该文档根据 sqllib\misc 目录中的运行状况建议 XML 模式 DB2RecommendationSchema.xsd 进行了格式化。
- 该文档以 UTF-8 进行编码并且包含客户机语言文本。
- 该文档的结构为一组建议集合，其中每个建议集合描述一个要解决的问题（运行状况指示器），并且包含用于解决该运行状况指示器的一个或多个建议。有关可从该文档检索的信息的特定详细信息，请参阅模式定义。

使用 SQL 查询时返回的 XML 建议文档也会提供可通过 CLP 获取的所有信息。

SYSPROC.HEALTH_HI_REC 存储过程采用下列自变量:

- 运行状况指示器
- 运行状况指示器进入警报状态的对象的定义

输出建议文档返回为 BLOB。因此，从命令行使用此存储过程没什么帮助，原因是 CLP 将限制显示的输出量。建议使用高级语言（如 C 或 Java）来调用此存储过程，高级语言能够正确地对返回的 XML 文档进行语法分析以检索任何必需的元素和属性。

使用 CLP 来检索运行状况建议:

可从 CLP 使用 GET RECOMMENDATIONS 命令来检索建议。命令语法支持查询建议以解决特定运行状况警报，如针对特定对象并且目前已进入警报状态的运行状况指示器。

必须具有实例连接才能从运行状况监视器检索建议。如果没有实例连接，那么创建缺省实例连接。要从远程实例上的运行状况监视器获取建议，必须先连接至该实例。从运行状况监视器检索建议不需要任何特权。

命令语法还支持检索给定运行状况指示器的完整建议集合，执行命令时该运行状况指示器不一定要处于警报状态。可在单一分区级别或全局级别查询用于解决特定运行状况指示器上的警报的建议。

查询针对特定对象的运行状况警报的建议时，运行状况监视器将解决特定警报，并且能够在输出的问题部分提供有关要解决的警报的详细信息。

运行状况监视器还可以提供建议的排名，并且在某些情况下，它能够生成一些脚本，执行这些脚本就可以解决警报。此外，如果某些建议不适用于特定问题情况，运行状况监视器可能拒绝并且不显示这些建议。另一方面，如果仅通过运行状况指示器名称查询建议（就像下面的第一个示例那样），那么将总是返回可能建议的总集合。在这样的情况下，CLP 命令仅显示有关用户看到警报时应考虑采用的操作的信息。

使用 GET RECOMMENDATIONS 命令检索建议:

1. 您可以想要发出以下命令以查看完整的操作集合，该操作集合是为了解决 **db.db_op_status** 运行状况指示器上的警报而建议的。

   ```
   db2 get recommendations for health indicator db.db_op_status
   ```

   在此示例中，将对 **db.db_op_status** 运行状况指示器返回完整的建议集合。运行状况指示器不必处于警报状态就可以发出此命令。

   此输出显示针对此运行状况指示器的两个可能建议：取消停顿数据库或调查数据库上的前滚进度。因为该命令将用于查询所有可能的建议而不是询问如何解决特定警报，所以运行状况监视器不能标识此情况下的最佳建议。因此，将返回完整的建议集合。

   ```
   建议:

   建议：调查前滚进度。

   因为管理员提交显式请求，所以前滚正在数据库上进行。您必须等待前滚完成，实例才能
   返回至活动状态。
   执行下列其中一项操作:

   启动 DB2 工具：实用程序状态管理器

   "实用程序状态管理器"允许您监视当前运行的实用程序的进度和更改它们的优先级。
   要打开"实用程序状态管理器":
   1.在"控制中心"中，展开对象树，直到找到想要的数据库为止。
   2.右键单击该数据库并从弹出菜单中单击"管理实用程序"。"实用程序状态管理器"将打开。

   要查看前滚实用程序的进度，右键单击该前滚实用程序并选择"查看进度详细信息"。

   从命令行处理器发出以下示例中显示的命令，以查看前滚实用程序的进度:

      LIST UTILITIES SHOW DETAIL


   建议：取消停顿数据库。

   已通过来自管理员将数据库置于 QUIESCE PENDING 或 QUIESCE 状态。如果您有
   QUIESCE_CONNECT 权限或者是 DBADM 或 SYSADM，那么表示您仍然可以访问该数据库，
   并可正常使用该数据库。对于所有其他用户，不允许与数据库建立新连接，也不能
   启动新的工作单元。而且，根据停顿请求，将允许活动工作单元完成或立即回滚。
   您可发出取消停顿请求以返回活动状态。

   执行下列其中一项操作:

   启动 DB2 工具：控制中心取消停顿数据库

   "控制中心"有一个用于数据库的选项，可用来取消停顿数据库。
   要取消停顿数据库:
   1.在"控制中心"中，展开对象树，直到找到想要的数据库为止。
   2.右键单击该数据库并从弹出菜单中单击"取消停顿"。数据库将取消停顿。
   ```

在"命令行处理器"中发出以下示例中显示的命令:
```
CONNECT TO DATABASE database-alias
UNQUIESCE DATABASE
```

2. 假定您观察到数据库 SAMPLE 的运行状况指示器 **db.db_heap_util** 已进入警报状态，并且您想要确定如何解决该警报。在此情况下，您可能想要解决特定问题，因此您可按以下方式发出 GET RECOMMENDATIONS 命令:

```
db2 get recommendations for health indicator db.db_heap_util
    for database on sample
```

此输出显示问题摘要及用于解决问题的一组建议。运行状况监视器按其首选顺序对这些建议排名。每个建议包含描述和一组操作以指示如何执行建议操作。

问题:

```
    指示器名称       = db.db_heap_util       值   = 42
       求值时间戳记 = 11/25/2003 19:04:54
       警报状态     = 警报
       其他信息     =
```

建议:

建议: 增加数据库堆大小。
排名: 1

充分增大数据库配置参数 dbheap 的大小，以使利用率达到正常操作级别。
要增大该值，请将 dbheap 得新值设置为等于 (pool_cur_size / (4096*U))，
其中 U 是期望得利用率。例如，如果期望的利用率为警告阈值级别的
60%（以前您将其设置为 75%），那么 U = 0.6 * 0.75 = 0.45（或 45%）。
执行下列其中一项操作:

在 DB2 服务器上执行下列脚本（可以使用 EXEC_DB2_CMD 存储过程来完成此任务）:
```
CONNECT TO DATABASE SAMPLE;
UPDATE DB CFG USING DBHEAP 149333;
CONNECT_RESET;
```

启动 DB2 工具: 数据库配置窗口

"数据库配置"窗口可用来查看和更新正在更新数据库配置参数。
要打开"数据库配置"窗口:
1.在"控制中心"中，展开对象树直到找到数据库文件夹为止。
2.单击数据库文件夹。任何现有的数据库都将显示在窗口右边的内容窗格中。
3.在内容窗格中右键单击想要的数据库，然后在弹出菜单中单击"配置参数"。
    "数据库配置"窗口将打开。

建议: 调查数据库堆的内存使用情况。
排名: 2

每个数据库都有一个数据库堆，并且数据库管理器会代表连接至该数据库
的所有应用程序使用该数据库堆。
数据区将根据需要扩展，最高可扩展至 dbheap 指定的最大大小。

有关数据库堆的更多信息，请参阅 DB2 信息中心。

调查一段时间内用于数据库堆的内存量，以确定最适当的数据库堆配置
参数值。
数据库系统监视器会记录用于数据库堆的最大内存量。
执行下列其中一项操作:

启动 DB2 工具: 内存可视化器

内存可视化器用于监视 DB2 实例中的内存分配。它可用来监视整体
内存使用情况，也可以更新各个内存组件的配置参数。

要打开内存可视化器:
1.在"控制中心"中，展开对象树直到找到实例文件夹为止。
2.单击实例文件夹。任何现有的实例都将显示在窗口右边的内容窗格中。
3.在内容窗格中右键单击想要的实例，然后在弹出菜单中单击
"查看内存使用情况"。内存可视化器将打开。

内存可视化器显示数据库管理器内存池的层次列表。数据库堆列示在
每个数据库的数据库管理器组中。在 Windows 上，它列示在
"数据库管理器共享内存"组中。

3. 对于分区数据库系统，可查询针对在特定分区上进入警报状态的运行状况指示器的建议，或者查询针对所有分区的全局建议。以全局方式查询建议时，将返回适用于所有分区上的运行状况指示器的一组建议。例如，如果运行状况指示器在分区 1 和分区 3 上处于警报状态，那么可能返回包含两个脚本的集合，其中每个脚本适用于不同分区。

   以下示例显示如何查询针对特定分区上的运行状况指示器的建议（在此示例中为分区号 2）：

   ```
   db2 get recommendations for health indicator db.db_heap_util
     for database on sample at dbpartitionnum 2
   ```

   以下示例显示如何检索一组建议以解决在若干分区上处于警报状态的运行状况指示器:

   ```
   db2 get recommendations for health indicator db.db_heap_util
     for database on sample global
   ```

使用客户机应用程序检索运行状况建议:

可使用 C 或 C++ 应用程序中的 db2GetRecommendations API 来查询建议。

必须具有实例连接才能捕获运行状况快照。如果没有实例连接，那么创建缺省实例连接。要查询远程实例上的建议，必须先连接至该实例。

使用 db2GetRecommendations API 时，将会在如下 XML 文档中返回建议:

- 该文档根据 SQLLIB 目录的 MISC 子目录中的运行状况建议 XML 模式 DB2RecommendationSchema.xsd 进行了格式化。
- 该文档以 UTF-8 进行编码并且包含客户机语言文本。
- 该文档的结构为一组建议集合，其中每个建议集合描述一个要解决的问题（运行状况指示器），并且包含用于解决该运行状况指示器的一个或多个建议。有关可从该文档检索的信息的特定详细信息，请参阅模式定义。

返回的 XML 建议文档也会提供可通过 CLP 获取的所有信息。

要使用客户机应用程序检索运行状况建议:

1. 包括 sqlmon.h 和 db2ApiDf.h DB2 头文件。可在 sqllib\include 目录中找到这些头文件。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. 声明 sqlca 和 db2GetRecommendationsData 结构。

   ```
   struct sqlca sqlca ;
   db2GetRecommendationsData recData ;

   memset( &sqlca, '\0', sizeof( struct sqlca ) ) ;
   memset( &recData, '\0', sizeof( db2GetRecommendationsData ) ) ;
   ```

3. 使用有关想要对其检索建议的警报的信息来填充 db2GetRecommendationsData 结构。在如下代码摘录中，将针对样本数据库上的 **db2.db_heap_util** 运行状况指示器查询建议。

```
recData.iSchemaVersion = DB2HEALTH_RECSCHEMA_VERSION8_2 ;
recData.iNodeNumber = SQLM_CURRENT_NODE ;
recData.iIndicatorID = SQLM_HI_DATABASE_HEAP_UTILIZATION  ;
recData.iObjType = DB2HEALTH_OBJTYPE_DATABASE ;
recData.piDbName = "SAMPLE" ;
```

4. 调用 db2GetRecommendations API 以检索针对指定数据库上此运行状况指示器的警报的建议。

```
db2GetRecommendations(  db2Version820, &recData, &sqlca ) ;
```

5. 检查 sqlca 中返回的 sqlcode 以了解是否发生任何错误。如果 API 调用成功，那么处理 db2GetRecommendationsData 结构的 poRecommendation 字段中返回的建议 XML 文档。使用您选择的 XML 解析器来抽取必需的元素或属性。请参阅 sqllib\misc 目录中的 DB2RecommendationSchema.xsd XML 模式，以了解有关可从 XML 文档检索的信息的详细信息。
6. 释放 db2GetRecommendations API 分配的所有内存。这将释放 db2GetRecommendationsData 结构的 poRecommendation 字段中返回的建议文档。

```
db2GetRecommendationsFree( db2Version820, &recData, &sqlca );
```

一般来说，当您检测到运行状况指示器进入警报状态时通常会查询建议，所以应将先前的代码与对快照 API 的调用组合在一起来获取运行状况快照。

使用运行状况中心解决运行状况监视器警报的问题:

运行状况中心提供支持以对报警条件检索和实施建议操作。

要使用运行状况中心解决运行状况监视器警报，请执行下列操作:

1. 从运行状况中心的“警报”视图中右键单击想要解决的警报行，然后从弹出菜单中选择建议顾问程序。建议顾问程序打开并以类似详细信息窗口的格式显示警报的详细信息。
2. 遵循建议顾问程序的步骤以选择最适当的建议。建议顾问程序允许您实施建议。

有两种类型的建议：调查和建议。对于这些建议类型，建议顾问程序中支持四种类型的操作:

启动图形管理工具

此选项将启动图形工具，该工具将解决或调查报警条件。该工具将在发生警报的对象的上下文中启动。

更新配置参数

需要更新的配置参数与当前值及建议值列示在一起。建议值可根据需要进行更新。

运行 DB2 命令脚本

建议操作可能需要多个命令。DB2 命令脚本允许运行多个命令以解决报警条件。例如，“需要重组”运行状况指示器提供了运行实用程序的 DB2 命令脚本操作。

实现备用解决方案

如果不能在 DB2 管理工具集中完成该操作，那么将提供指示信息以使用备用方法解决报警条件。

运行状况指示器配置: 在安装期间将提供缺省运行状况监视器配置。这将确保一启动 DB2，运行状况监视器就可以评估数据库环境的运行状况。但是，在对运行状况指示器求值和对警报状态作出反应时，可通过配置特定用户环境来对运行状况监视器的行为作细微的调整。

可在不同级别定义配置。安装 DB2 时为每个运行状况指示器提供了工厂设置的缺省配置。第一次启动运行状况监视器时，工厂设置的副本将提供实例和全局设置的缺省值。

实例设置将应用于实例。全局设置将应用于实例中未定义定制设置的对象，如数据库、表空间和表空间容器。

如果对特定数据库、表空间或表空间容器更新运行状况指示器设置，那么会对更新后的运行状况指示器创建对象设置。对象设置的缺省值是全局设置。

运行状况监视器在对特定数据库、表空间或表空间容器处理运行状况指示器时将检查对象设置。如果特定运行状况指示器的设置未更新，那么会使用缺省全局设置来处理运行状况指示器。当运行状况监视器对实例处理运行状况指示器时，将使用实例设置。

通过使用可对每个运行状况指示器配置的一些属性，可以更改运行状况监视器行为。第一组参数（求值标志、阈值和灵敏度）定义运行状况监视器对运行状况指示器生成警报的条件。第二组参数（操作标志和操作）定义生成警报时运行状况监视器要执行的操作。

求值标志

每个运行状况指示器都有一个求值标志，可用来启用或禁用警报状态的求值。

警告和警报阈值

基于阈值的运行状况指示器包含一些设置，它们用来定义运行状况指示器值的警告和警报区域。可对特定数据库环境修改这些警告和警报阈值。

灵敏度参数

灵敏度参数定义运行状况指示器值必须至少保持警报状态多长时间（以秒计）才会生成警报。与灵敏度值相关联的等待时间从运行状况指示器值进入警报状态期间的第一次刷新时间间隔开始算起。可使用此值来消除因为资源使用中的暂时峰值而生成的错误警报。

考虑使用日志利用率（*db.log_util*）运行状况指示器的示例。假定您每星期复查一次 DB2 通知日志。在第一个星期，*db.log_util* 的某个条目处于警报状态。您想起曾经收到过有关此情况的通知，但在从 CLP 检查警报状态时，运行状况指示器已回复正常状态。第二个星期以后，您在这一个星期的同一时间收到针对同一运行状况指示器的第二个警报通知条目。您在数据库环境中调查两次生成警报的活动，发现有一个应用程序的落实时间很长并且每星期运行一次。此应用程序导致日志利用率在大约 8 到 9 分钟的一小段时间内处于峰值，直到应用程序落实。可查看通知日志的警报通知记录中的历史记录条目，您会发现 *db.log_util* 运行状况指示器每 10 分钟求值一次。因为将生成警报，所以应用程序时间一定会跨越刷新时间间隔。将 *db.log_util* 参数的灵敏度设置为 10 分钟。现在每次 *db.log_util* 的值第一次进入警告或警报阈值区域时，该值必须在该区域中保持至少 10 分钟，才会生成警报。因为应用程序只持续 8 到 9 分钟，所以通知日志中不会再针对此情况记录通知条目。

操作标志

生成警报时的操作运行由操作标志控制。仅当操作标志启用时，才会配置要运行的警报操作。

操作 可将脚本或任务操作配置为在发生警报时运行。对于基于阈值的运行状况指示器，可将操作配置为针对警告或警报阈值运行。对于基于状态的运行状况指示器，可将操作配置为针对任何可能的非正常状态运行。DB2 管理服务器必须正在运行，这些操作才能运行。

下列输入参数将传递至每个操作系统命令脚本:

- <运行状况指示器短名称>
- <对象名>
- <value | state>
- <报警类型>

脚本操作使用操作系统上的缺省解释器。如果想要使用非缺省解释器，那么使用脚本内容在任务中心中创建任务。在多分区环境中，在脚本操作中定义的脚本必须可供所有分区访问。

不能配置运行状况监视器检查每个运行状况指示器的刷新时间间隔。运行状况监视器认为不能配置建议操作。

运行状况监视器配置存储在二进制文件 HealthRules.reg 中:

- 在 Windows 上，HealthRules.reg 存储在 x:\<SQLLIB_PATH>\<INSTANCE_NAME> 中。例如 d:\sqllib\DB2。
- 在 UNIX 上，HealthRules.reg 存储在 ~/<SQLLIB_PATH>/cfg 中。例如，~/home/sqllib/cfg。

可将运行状况监视器配置复制至其他 Linux、UNIX 或 Windows 服务器上的 DB2 V8 实例。可通过将二进制配置文件复制至目标实例上的适当目录位置以完成此复制。

使用 CLP 检索运行状况指示器配置:

GET ALERT CONFIGURATION 命令允许您查看工厂设置和实例设置、全局设置和对象设置。

1. 要查看数据库级别运行状况指示器的全局设置（在运行状况指示器没有定制设置的情况下，这些设置适用于所有数据库），请发出以下命令:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASES
   ```

2. 要查看数据库级别运行状况指示器的全局设置（在运行状况指示器没有定制设置的情况下，这些设置适用于所有数据库），请发出以下命令:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASES
   ```

   每个运行状况指示器设置的输出指示它是否更改了缺省值。在以下输出中，全局设置尚未更新；因此，它们与缺省工厂设置完全相同。要查看数据库级别运行状况指示器的工厂设置，使用 DEFAULT 关键字发出在先前示例中发出的命令:

   ```
                   警报配置

   指示器名称                  = db.db_op_status
   缺省值                      = 是
   类型                        = 基于状态
   灵敏度                      = 0
   公式                        = db.db_status;
   操作                        = 已禁用
   ```

```
        阈值或状态检查                  = 已启用

        指示器名称                      = db.sort_shrmem_util
        缺省值                          = 是
        类型                            = 基于阈值
        警告                            = 70
        警报                            = 85
        单位                            = %
        灵敏度                          = 0
        公式                            = ((db.sort_shrheap_allocated/sheapthres_shr)*100);
    操作                            = 已禁用
        阈值或状态检查                  = 已启用
...
```

3. 要查看 SAMPLE 数据库的定制设置，请发出以下命令:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
   ```

   如果指定的对象上的特定运行状况指示器没有特定设置，那么显示所有数据库的全局设置。要查看特定运行状况指示器的设置，将 USING *health-indicator-name* 子句添加在所有先前示例的前面。

使用 CLP 进行运行状况指示器配置更新: 可对全局设置或特定对象的设置更新特定运行状况指示器的配置。

UPDATE ALERT CONFIGURATION 命令有四个小子句，覆盖了不同的更新选项。每个 UPDATE ALERT CONFIGURATION 命令中只能使用一个小子句。要使用多个选项，必须发出多个 UPDATE ALERT CONFIGURATION 命令。

第一个小子句 SET *parameter-name value* 提供对下列更新的支持:

- 求值标志
- 警告和警报阈值（如果适用）
- 灵敏度标志
- 操作标志

这些设置的参数名分别为:

- THRESHOLDSCHECKED
- WARNING 和 ALARM
- SENSITIVITY
- ACTIONSENABLED

其他三个小子句提供对添加、更新和删除脚本或任务操作的支持。

下列命令更新 SAMPLE 数据库上的 *db.spilled_sorts* 运行状况指示器的基于阈值的配置。该更新会将警告阈值更改为 25，以启用操作和添加脚本操作:

```
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
        SET WARNING 25, ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
        ADD ACTION SCRIPT c:\myscript TYPE OS COMMAND LINE PARAMETERS 'space'
        WORKING DIRECTORY c:\ ON ALARM USER dba1 PASSWORD dba1
```

下列命令更新全局设置的 *ts.ts_util* 运行状况指示器的基于状态的配置。该更新定义任何表空间处于备份暂挂状态时要运行的操作。

```
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    SET ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    ADD ACTION TASK 0.1 ON ATTENTION 32 ON localhost USER dba1 PASSWORD dba1
```

此更新将应用于此运行状况指示器没有定制设置的实例的所有表空间。

将操作添加至运行状况指示器配置时，ON *condition* 子句的选项将基于运行状况指示器的类型:

- 对于基于阈值的运行状况指示器，WARNING 和 ALARM 是有效条件。
- 对于基于状态的运行状况指示器，必须使用 ON ATTENTION *state* 选项。应该使用对运行状况指示器定义的有效数字状态。可在 sqllib\include\sqlmon.h 中找到数据库管理器和数据库操作状态值。表空间和表空间容器操作值列示在 sqllib\include\sqlutil.h 中。注意，不能对处于停机状态的数据库管理器执行这些操作。有关详细信息，请参阅对 db2.db2_op_status 运行状况指示器的描述。

使用 CLP 复位运行状况指示器配置:

CLP 提供对将全局设置复位至工厂设置的支持。特定对象的对象设置还可复位为该对象类型的定制设置。

- 要将 SAMPLE 数据库的对象设置复位至数据库的当前全局设置，请发出以下命令:

  ```
  DB2 RESET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
  ```

- 要将数据库的全局设置复位至工厂设置，请发出以下命令:

  ```
  DB2 RESET ALERT CONFIGURATION FOR DATABASES
  ```

- 要复位特定运行状况指示器的配置，请将 USING *health-indicator-name* 子句添加在所有先前示例的前面。

使用客户机应用程序配置运行状况指示器:

可通过 C 或 C++ 应用程序中的 db2GetAlertCfg、db2UpdateAlertCfg 和 db2ResetAlertCfg API 访问运行状况监视器配置。其中的每个 API 都可访问工厂设置、实例设置、全局设置和对象设置。

必须具有实例连接才能访问运行状况监视器配置。如果没有实例连接，那么创建缺省实例连接。要访问远程实例的运行状况监视器配置，必须先连接至该实例。

db2GetAlertCfgData 结构中的 **objType** 和 **defaultType** 参数组合允许您访问各种级别的运行状况指示器配置。

表 47. objType 和 defaultType 用于访问配置级别的设置

设置	**objType 和 defaultType**
工厂设置	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES |TABLESPACES | CONTAINERS} 和 defaultType = DB2ALERTCFG_DEFAULT
全局设置	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES | TABLESPACES | CONTAINERS} 和 defaultType = DB2ALERTCFG_NOT_DEFAULT 或者 objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} 和 defaultType = DB2ALERTCFG_DEFAULT

表 47. objType 和 defaultType 用于访问配置级别的设置 (续)

设置	objType 和 defaultType
对象设置	objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} 和 defaultType = DB2ALERTCFG_NOT_DEFAULT

1. 要获取 SAMPLE 数据库上的运行状况指示器的特定对象设置:
   a. 包括 sqllib\include 目录中的 db2ApiDf.h DB2 头文件。

      ```
      #include <db2ApiDf.h>
      ```

   b. 声明并初始化 sqlca 和 db2GetAlertCfgData 结构。

      ```
      struct sqlca ca;
      memset (&sqlca, '\0', sizeof(struct sqlca));
      char* objName = NULL;
      char* dbName = "SAMPLE";
      db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASE;
      db2Uint32 defaultType = DB2ALERTCFG_NOT_DEFAULT;

      db2GetAlertCfgData data = {objType, objName, defaultType, dbName, 0, NULL} ;
      ```

   c. 调用 db2GetAlertCfg API。

      ```
      rc = db2GetAlertCfg (db2Version810, &data, &ca);
      ```

   d. 处理返回的配置并释放 API 分配的缓冲区。

      ```
      if (rc >= SQLO_OK) {
        if ((data.ioNumIndicators > 0) && (data.pioIndicators != NULL)) {
          db2GetAlertCfgInd *pIndicators = data.pioIndicators;

          for (db2Uint32 i=0; i  data.ioNumIndicators; i++) {
       //process the entry as necessary using fields defined in db2ApiDf.h
          }
        }

        db2GetAlertCfgFree (db2Version810, &data, &ca);
      }
      ```

2. 下列步骤详细描述更新数据库对象全局设置的 **db.sort_shrmem_util** 运行状况指示器的警报配置的过程，将警告阈值设置为 80 并且添加任务操作 1.1:
   a. 包括 sqllib\include 目录中的 db2ApiDf.h DB2 头文件。

      ```
      #include <db2ApiDf.h>
      ```

   b. 声明并初始化 sqlca 和 db2AlertTaskAction 结构。

      ```
      struct sqlca ca;
      memset (&sqlca, '\0', sizeof(struct sqlca));
      db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASES;

      db2Uint32 taskCondition = DB2ALERTCFG_CONDITION_WARNING;
      char* taskname = "1.1";
      char* hostname = NULL;
      char* userid = "nobody";
      char* password = "nothing";

      db2AlertTaskAction newTask={taskname,taskCondition,userid,password,hostname};
      ```

   c. 声明并初始化 db2UpdateAlertCfgData 结构。

      ```
      struct db2UpdateAlertCfgData setData;

      setData.iObjType = objType;
      setData.piObjName = NULL;
      setData.piDbName = NULL;

      setData.iIndicatorID = 1002;

      setData.iNumIndAttribUpdates = 1;
      setData.piIndAttribUpdates[0].iAttribID = DB2ALERTCFG_WARNING;
      setData.piIndAttribUpdates[0].piAttribValue == 80;

      setData.iNumActionUpdates = 0;
      ```

```
setData.piActionUpdates = NULL;

setData.iNumActionDeletes = 0;
setData.piActionDeletes = NULL;

setData.iNumNewActions = 1;
setData.piNewActions[0].iActionType = DB2ALERTCFG_ACTIONTYPE_TASK;
setData.piNewActions[0].piScriptAttribs = NULL;
setData.piNewActions[0].piTaskAttribs = &newTask;
```

d. 调用 db2UpdateAlertCfg API。

```
rc = db2UpdateAlertCfg(db2Version810, &setData, &ca);
```

3. 下列步骤详细描述复位 SAMPLE 数据库的 MYTS 表空间的定制设置的过程。

a. 包括 sqllib\include 目录中的 db2ApiDf.h DB2 头文件。

```
#include <db2ApiDf.h>
```

b. 声明并初始化 sqlca 和 db2ResetAlertCfgData structures。

```
struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));
char* objName = "MYTS";
char* dbName = "SAMPLE";
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_TABLESPACE;

db2ResetAlertCfgData data = {objType, objName, dbName};
```

c. 调用 db2ResetAlertCfg API。

```
rc = db2ResetAlertCfg (db2Version810, &data, &ca);
```

使用运行状况中心配置运行状况指示器:

运行状况中心提供图形界面以便您查看、更新和复位运行状况指示器配置。运行状况指示器的配置存储在实例的运行状况监视器中。

要定义、更改、启用或禁用运行状况指示器的阈值或灵敏度设置，以及定义、更改、启用或禁用在运行状况指示器发生警报时运行的任务或脚本，必须具有以下其中一种权限:

- SYSADM
- SYSMAINT
- SYSCTRL

可调整某个实例的运行状况指示器设置、实例中包含的数据库对象的全局运行状况指示器设置以及各个数据库对象的运行状况指示器设置。

1. 要使用运行状况中心配置运行状况指示器，请执行下列操作:

a. 选择要配置其运行状况指示器的实例。

b. 从**已选择**菜单或右键单击菜单中，单击**配置**，然后单击**运行状况指示器设置**。“运行状况指示器配置启动板”将打开。

c. 对于每个可更新配置设置级别，启动板上都有一个对应的按钮。选择对应想要查看、更新或复位的配置级别的按钮。每个按钮在所选配置设置级别启动“运行状况指示器配置”窗口。

d. 要更新运行状况指示器设置，在“当前运行状况指示器设置”表中选择对应该运行状况指示器的行。

e. 从**选择**菜单或右键单击菜单中选择**编辑**。 “配置运行状况指示器”笔记本打开并显示以下信息:

- 通过单击**更多信息**来获取对运行状况指示器的描述。
- 可使用**求值**复选框来启用或禁用运行状况指示器的求值。

  注: 还可通过针对当前警报的右键单击菜单选项从“运行状况中心警报视图”对当前警报禁用**求值**标志。在运行状况监视器中再一次刷新运行状况指示器时，此选项将禁用该指示器的求值。在运行状况中心中对某个警报选择**禁用求值**时，运行状况指示器的求值标志将设置为 false，但在发生下列事件之前，不会从“警报”视图中除去该警报:

  – 达到该特定运行状况指示器的运行状况监视器刷新时间间隔
  – 运行状况监视器刷新运行状况指示器求值
  – 运行状况中心刷新其状态视图
- 在“警报”页上，可以对基于阈值的运行状况指示器更新警告和警报阈值。还可以在此页上设置任何运行状况指示器的灵敏度。
- 在“操作”页上，可选择发生警报时要运行的任务或脚本操作。对于基于阈值的运行状况指示器，可将操作配置为针对警告或警报阈值运行，或者对于基于状态的运行状况指示器，可将操作配置为针对任何非正常情况运行。可通过选择或取消选择**启用操作**复选框来启用或禁用操作的执行。要添加、更新或除去任务或脚本操作，请使用**脚本操作**和**任务操作**表旁边的按钮。

2. 要查看实例的工厂运行状况指示器设置:
   a. 在“运行状况指示器配置”启动板中，单击**实例设置**。
   b. 在“实例运行状况指示器配置”窗口中，单击**查看缺省值**。
3. 要查看数据库、表空间或表空间容器的全局运行状况指示器设置，请执行下列操作:
   a. 在“运行状况指示器配置”启动板中，单击**全局设置**。
   b. 在“全局运行状况指示器配置”窗口中选择对象类型。
   c. 要查看这些全局设置的工厂缺省值，请单击**查看缺省值**。
4. 要查看数据库对象的运行状况指示器设置:
   a. 在“运行状况指示器配置”启动板中，单击**对象设置**。
   b. 在“对象运行状况指示器配置”窗口中选择对象。
   c. 要查看此对象类型的全局缺省运行状况指示器设置，请单击**查看缺省值**。

在其中每个窗口中，要将所有已显示运行状况指示器的设置复位为其缺省值，单击**复位为缺省值**。通过在**当前运行状况指示器设置**字段中右键单击想要的一个或多个运行状况指示器，并从弹出菜单中选择**复位为缺省值**，还可以复位各个运行状况指示器。

针对组合状态的运行状况监视器警报操作:

警报操作是在运行状况指示器进入报警状态时运行的任务或脚本。

从 DB2 V9.1 开始，无论其他组合状态如何，每次将表空间的状态设置为单个警报状态时，针对该状态为运行状况指示器 **ts.ts_op_status** 定义的运行状况监视器警报操作都会运行。这使得可以对特定表空间状态运行警报操作，即使该状态与其他状态一起设置。

在以下示例中，即使在表空间状态同时为 QUIESCED:share 和 QUIESCE:update 时，针对注意状态 QUIESCED:share 定义的警报操作 script1 也会运行。

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status set
actionsenabled yes')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention QUIESCED_SHARE on aix1
user guest001 using passw0rd')
```

在以下示例中，当且仅当表空间状态同时为 QUIESCED:share 和 QUIESCED:update 时，使用组合状态（QUIESCED:share + QUIESCED:update = 3）定义的警报操作才执行。

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status set
actionsenabled yes')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention 3 on aix1 user guest001
using passw0rd')
```

从 DB2 V9.1 开始，使用相同操作属性（名称、工作目录、命令行参数、主机、用户和密码）对某个对象定义的运行状况监视器警报操作仅运行一次，即使该操作是针对多个警报状态定义的也是如此。

在以下示例中，对两个不同的警报状态定义了同一操作。即使表空间状态同时为 QUIESCED:share 和 QUIESCED:update，也只能对给定表空间执行一次该操作。

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention QUIESCED_SHARE on aix1
user guest001 using passw0rd')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention QUIESCED_UPDATE on aix1
user guest001 using passw0rd')
```

运行状况指示器

运行状况监视器使用运行状况指示器来评估数据库管理器性能或数据库性能特定方面的运行状况。运行状况指示器测量特定种类数据库对象（例如表空间）某些方面的运行状况。对度量应用条件以确定运行状况。应用的条件视运行状况指示器类型而定。运行状况是否正常是根据生成警报的条件确定的。

运行状况监视器返回三种类型的运行状况指示器:

- **基于阈值**的指示器是对象行为的（连续值范围）统计信息度量。警告阈值和警报阈值定义了正常、警告和警报范围的边界或区域。基于阈值的运行状况指示器有三种有效状态: “正常”、“警告”或“警报”。
- **基于状态**的指示器是有限状态集（包含一个对象的两种或更多种不同状态）度量，该状态集定义了数据库对象或资源的工作是否正常。其中一种状态表示情况正常，所有其他状态都被视为不正常。基于状态的运行状况指示器有两种有效状态: “正常”和“注意”。
- **基于集合状态**的指示器是数据库级度量，它们代表数据库中一个或多个对象的聚集状态。为该集合中的每个对象捕获数据，那些对象中最严重的情况以聚集状态表示。如果该集合中有一个或多个对象处于要求发出警报的状态，运行状况指示器就会显示“注意”状态。基于集合状态的运行状况指示器有两种有效状态: “正常”和“注意”。

在实例级、数据库级、表空间级和表空间容器级都存在运行状况指示器。

可以通过运行状况中心、CLP 或 API 来访问运行状况监视器信息。可以通过这些工具来配置运行状况指示器。

当从正常状态切换到非正常状态，或者运行状况指示器值进入警告或警报区域（基于已定义的阈值边界），就会生成警报。有三种类型的警报：注意、警告和警报。

- 对于量度不同状态的运行状况指示器来说，如果注册了非正常状态，就会发出注意警报。
- 对于量度连续值范围的运行状况指示器来说，阈值定义了正常状态、警告状态和警报状态的边界或区域。例如，如果值进入定义警报区域的阈值范围，就会发出警报类型的警报以指示问题需要立即加以注意。

对于给定的运行状况指示器，运行状况监视器仅在特定报警条件第一次出现时发送通知并运行操作。如果运行状况指示器一直处于特定报警条件下，就不会发送更多通知，也不进一步运行操作。如果运行状况指示器更改了报警条件，或者回到正常状态并重新进入报警条件，就会发送新通知并运行操作。

下表显示了不同刷新时间间隔的运行状况指示器示例以及运行状况监视器对运行状况指示器状态的响应。此示例使用缺省警告阈值和警报阈值，它们分别是 80% 和 90%。

表 48. 不同刷新时间间隔的运行状况指示器条件

刷新时间间隔	ts.ts_util（表空间利用率）运行状况指示器的值	ts.ts_util 运行状况指示器状态	运行状况监视器响应
1	80	警告	发送警告通知，运行警告报警条件的操作
2	81	警告	不发送通知，不运行操作
3	75	正常	不发送通知，不运行操作
4	85	警告	发送警告通知，运行警告报警条件的操作
5	90	警报	发送警报通知，运行警报条件的操作

运行状况监视器接口至逻辑数据组的映射

下表列示了所有受支持的运行状况快照请求类型。

表 49. 运行状况监视器接口至逻辑数据组的映射

API 请求类型	CLP 命令	SQL 表函数	逻辑数据组
SQLMA_DB2	get health snapshot for dbm	HEALTH_DBM_INFO	db2
		HEALTH_DBM_HI	health_indicator
	get health snapshot for dbm show detail	HEALTH_DBM_HI_HIS	health_indicator_history
SQLMA_DBASE	get health snapshot for database on *dbname*	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for database on *dbname* show detail	HEALTH_DB_HI_HIS	health_indicator_history

表 49. 运行状况监视器接口至逻辑数据组的映射 (续)

API 请求类型	CLP 命令	SQL 表函数	逻辑数据组
SQLMA_DBASE with SQLM_HMON_OPT_COLL_FULL in the agent_id	get health snapshot for database on *dbname* with full collection	HEALTH_DB_HIC	health_indicator, hi_obj_list
	get health snapshot for database on *dbname* show detail with full collection	HEALTH_DB_HIC_HIST	health_indicator_history, hi_obj_list
SQLMA_DBASE_ALL	get health snapshot for all databases	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for all databases show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE_TABLESPACES	get health snapshot for tablespaces on *dbname*	HEALTH_TS_INFO	tablespace
		HEALTH_TS_HI	health_indicator
		HEALTH_CONT_INFO	tablespace_container
		HEALTH_CONT_HI	health_indicator
	get health snapshot for tablespaces on *dbname* show detail	HEALTH_TS_HI_HIS	health_indicator_history
		HEALTH_CONT_HI_HIS	health_indicator_history

下图显示逻辑数据分组在运行状况快照数据流中可能出现的顺序。

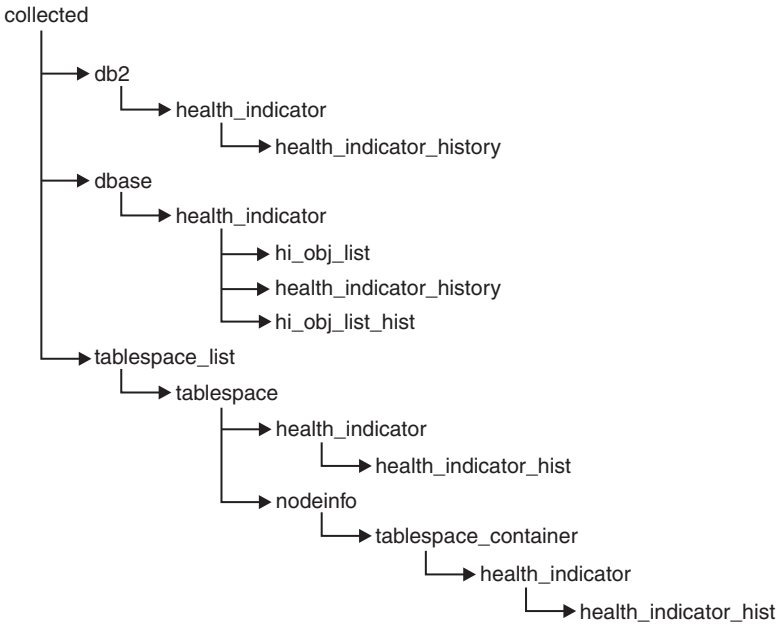


图 6. 运行状况快照逻辑数据分组

运行状况指示器摘要

下表按类别分组列示所有运行状况指示器。

表 50. 数据库自动存储器利用率运行状况指示器

名称	标识	其他信息
数据库自动存储器利用率	db.auto_storage_util	第 149 页的『db.auto_storage_util -“数据库自动存储器利用率”运行状况指示器』

表 51. 表空间存储器运行状况指示器

名称	标识	其他信息
表空间自动调整大小状态	ts.ts_auto_resize_status	第 150 页的『ts.ts_auto_resize_status -“表空间自动调整大小状态”运行状况指示器』
自动调整大小表空间利用率	ts.ts_util_auto_resize	第 150 页的『ts.ts_util_auto_resize -“自动调整表空间大小利用率”运行状况指示器』
表空间利用率	ts.ts_util	第 151 页的『ts.ts_util -“表空间利用率”』
表空间容器利用率	tsc.tscont_util	第 151 页的『tsc.tscont_util -“表空间容器利用率”』
表空间运作状态	ts.ts_op_status	第 152 页的『ts.ts_op_status -“表空间操作状态”』
表空间容器运作状态	tsc.tscont_op_status	第 152 页的『tsc.tscont_op_status -“表空间容器操作状态”』
表空间自动调整大小状态	ts.ts_auto_resize_status	第 150 页的『ts.ts_auto_resize_status -“表空间自动调整大小状态”运行状况指示器』

表 52. 排序运行状况指示器

名称	标识	其他信息
专用排序内存利用率	db2.sort_privmem_util	第 152 页的『db2.sort_privmem_util -“专用排序内存利用率”』
共享排序内存利用率	db.sort_shrmem_util	第 153 页的『db.sort_shrmem_util -“共享排序内存利用率”』
溢出排序百分比	db.spilled_sorts	第 154 页的『db.spilled_sorts -“溢出排序百分比”』
长期共享排序内存利用率	db.max_sort_shrmem_util	第 154 页的『db.max_sort_shrmem_util -“长期共享排序内存利用率”』

表 53. 数据库管理器运行状况指示器

名称	标识	其他信息
实例运作状态	db2.db2_op_status	第 155 页的『db2.db2_op_status -“实例操作状态”』
实例最高严重性警报状态	–	第 155 页的『实例最高级别严重性警报状态』

表 54. 数据库运行状况指示器

名称	标识	其他信息
数据库运作状态	db.db_op_status	第 155 页的『db.db_op_status -“数据库操作状态”』
数据库最高严重性警报状态	–	第 156 页的『“数据库最高级别严重性警报状态”』

表 55. 维护运行状况指示器

名称	标识	其他信息
需要重组	db.tb_reorg_req	第 156 页的『db.tb_reorg_req -“需要重组”』
必需的统计信息收集运行状况指示器	db.tb_runstats_req	第 157 页的『db.tb_runstats_req -“需要收集统计信息”』
必需的数据库备份	db.db_backup_req	第 157 页的『db.db_backup_req -“需要备份数据库”』

表 56. 高可用性灾难恢复运行状况指示器

名称	标识	其他信息
HADR 运作状态运行状况指示器	db.hadr_op_status	第 157 页的『db.hadr_op_status -“HADR 操作状态”』
HADR 日志延迟运行状况指示器	db.hadr_delay	第 158 页的『db.hadr_delay -“HADR 日志延迟”』

表 57. 日志记录运行状况指示器

名称	标识	其他信息
日志利用率	db.log_util	第 158 页的『db.log_util -“日志利用率”』
日志文件系统利用率	db.log_fs_util	第 158 页的『db.log_fs_util -“日志文件系统利用率”』

表 58. 应用程序并发性运行状况指示器

名称	标识	其他信息
死锁率	db.deadlock_rate	第 159 页的『db.deadlock_rate -“死锁率”』
锁定列表利用率	db.locklist_util	第 160 页的『db.locklist_util -“锁定列表利用率”』
锁定升级率	db.lock_escal_rate	第 160 页的『db.lock_escal_rate -“锁定升级率”』
等待锁定的应用程序的百分比	db.apps_waiting_locks	第 161 页的『db.apps_waiting_locks -“等待锁定的应用程序的百分比”』

表 59. 程序包高速缓存、目录高速缓存以及工作空间运行状况指示器

名称	标识	其他信息
目录高速缓存命中率	db.catcache_hitratio	第 161 页的『db.catcache_hitratio -“目录高速缓存命中率”』

表 59. 程序包高速缓存、目录高速缓存以及工作空间运行状况指示器 (续)

名称	标识	其他信息
程序包高速缓存命中率	db.pkgcache_hitratio	第 161 页的『db.pkgcache_hitratio -“程序包高速缓存命中率”』
共享工作空间命中率	db.shrworkspace_hitratio	第 162 页的『db.shrworkspace_hitratio -“共享工作空间命中率”』

表 60. 内存运行状况指示器

名称	标识	其他信息
监视器堆利用率	db2.mon_heap_util	第 162 页的『db2.mon_heap_util -“监视器堆利用率”』
数据库堆利用率	db.db_heap_util	第 162 页的『db.db_heap_util -“数据库堆利用率”』

表 61. 联合运行状况指示器

名称	标识	其他信息
昵称状态	db.fed_nicknames_op_status	第 163 页的『db.fed_nicknames_op_status -“昵称状态”』
数据源服务器状态	db.fed_servers_op_status	第 163 页的『db.fed_servers_op_status -“数据源服务器状态”』

运行状况指示器格式:

对运行状况指示器所收集数据的描述。

运行状况指示器的记录是以如下标准格式描述的:

标识 运行状况指示器的名称。此标识用于 CLP 中的配置。

运行状况监视器级别
运行状况监视器捕获运行状况指示器的级别。

类别 运行状况指示器的类别。

类型 运行状况指示器的类型。该类型有四个可能的值:

- 基于阈值上限，其中警报级数为: 正常，警告，警报
- 基于阈值下限
- 基于状态，其中一种状态表示情况正常，所有其他状态都被视为不正常
- 基于集合状态，其中状态基于集合中对象的状态聚集

单位 运行状况指示器中度量的数据单位，如百分比。此项对基于状态或集合状态的运行状况指示器不适用。

表空间存储器运行状况指示器:

DMS 表空间的运行状况指示器:
此表描述根据表空间的特征来看与 DMS 表空间有关的表空间运行状况指示器:

表 62. DMS 表空间的相关表空间运行状况指示器

表空间特征	定义的最大表空间大小	未定义的最大表空间大小
Automatic resize enabled = Yes	ts.ts_util_auto_resize - 跟踪相对于定义的最大值而言使用的空间在表空间中所占的百分比。警报指示表空间很快就会变满并且需要您的干预。只要将最大大小设置为合理的值（即最大大小指定的空间量存在），它就是此配置最重要的运行状况指示器。 ts.ts_util - 跟踪当前分配的表空间存储器使用情况。警报可能不需要您的干预就能够解决任何问题，原因是表空间在变满时将尝试增加大小。 ts.ts_auto_resize_status - 跟踪调整大小尝试的运行状况。警报指示表空间调整大小失败（即表空间已满）。	ts.ts_util_auto_resize - 不适用。未对表空间大小指定上边界。 ts.ts_util - 跟踪当前分配的表空间存储器使用情况。警报可能不需要您的干预就能够解决任何问题，原因是表空间将尝试增加大小。 ts.ts_auto_resize_status - 跟踪调整大小尝试的运行状况。警报指示表空间调整大小失败（即表空间已满）。 **注**: 如果使用自动存储器定义了 DMS 表空间并且未指定最大大小，那么还应注意 db.auto_storage_util 运行状况指示器。此运行状况指示器跟踪与数据库存储器路径相关联的空间的利用率。当此空间变满时，表空间将无法增长。这可能导致出现表空间已满的情况。
Automatic resize enabled = No	配置无效。最大表空间大小仅对能够自动调整大小的表空间有效。	ts.ts_util_auto_resize - 不适用。表空间将不尝试调整大小。 ts.ts_util - 跟踪当前分配的表空间存储器使用情况。警报指示表空间已满情况并且需要您立即干预。表空间自身不会尝试调整大小。 ts.ts_auto_resize_status - 不适用。表空间将不尝试调整大小。

db.auto_storage_util - "数据库自动存储器利用率"运行状况指示器:

此运行状况指示器跟踪定义的数据库存储器路径的存储器消耗情况。

标识 db.auto_storage_util

运行状况监视器级别
数据库

类别 数据库

类型 基于阈值上限

单位 百分比

创建自动存储器表空间时，将在数据库存储器路径上为这些表空间自动分配容器。如果定义了数据库存储器路径的任何文件系统上没有更多空间，那么自动存储器表空间将无法增加大小，从而可能会变满。

使用以下公式计算指示器:

```
(db.auto_storage_used / db.auto_storage_total) * 100
```

其中:

- *db.auto_storage_used* 是数据库存储器路径中标识的所有物理文件系统中使用的空间总和
- *db.auto_storage_total* 是数据库存储器路径列表中标识的所有物理文件系统中使用的总空间总和

数据库自动存储器路径利用率以数据库存储器路径文件系统上消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

附加信息中对离空间已满所余时间的计算可以预测到消耗所有可用空间所余的时间。

ts.ts_auto_resize_status - "表空间自动调整大小状态"运行状况指示器:

此运行状况指示器标识对能够自动调整大小的 DMS 表空间执行表空间调整大小操作是否成功。当能够自动调整大小的 DMS 表空间未能增加大小时，它会很快变满。此情况可能是因为定义了表空间容器的文件系统上缺少可用空间造成的，也可能是表空间自动调整大小设置造成的。例如，可能已经达到定义的最大大小，或者增加量可能设置得太高而导致余下可用空间无法容纳。

标识 ts.ts_auto_resize_status

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于状态

单位 不适用

ts.ts_util_auto_resize - "自动调整表空间大小利用率"运行状况指示器:

此运行状况指示器跟踪每个 DMS 表空间的存储器消耗情况，这些 DMS 表空间已经定义了最大大小，并且可以自动调整大小。达到最大大小时，那么认为 DMS 表空间已满。

标识 ts.ts_util_auto_resize

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于阈值上限

单位 百分比

使用以下公式计算指示器:

```
((ts.used * ts.page_size) / ts.max_size) * 100
```

其中:

- *ts.used* 是 第 654 页的『 tablespace_used_pages - "表空间中的已使用页数"监视元素 』 的值
- *ts.page_size* 是 第 645 页的『 tablespace_page_size - "表空间页大小"监视元素 』 的值
- *ts.max_size* 是 第 643 页的『 tablespace_max_size - "最大表空间大小" 』 的值

自动调整大小 DMS 表空间利用率是用消耗的最大表空间存储器所占的百分比度量的。高百分比指示表空间接近已满程度。此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

附加信息中对离空间已满所余时间的计算可以预测达到最大大小所余的时间。

ts.ts_util - "表空间利用率":

此运行状况指示器跟踪每个 DMS 表空间的存储器消耗情况。

标识 ts.ts_util

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于阈值上限

单位 百分比

当所有容器已满时，那么认为 DMS 表空间已满。

如果在表空间上启用自动调整大小，那么不会评估此运行状况指示器。相反，对于表空间存储器监视，数据库自动存储器利用率 **db.auto_storage_util** 运行状况指示器和表空间自动调整大小状态 **ts.ts_auto_resize_status** 运行状况指示器是相关的。如果对此表空间定义了最大大小，那么自动调整大小表空间利用率 **ts.ts_util_auto_resize** 运行状况指示器还将可用。如果需要，还可从 TBSP_UTILIZATION 管理视图的 TBSP_UTILIZATION_PERCENT 列检索表空间利用率百分比。

使用以下公式计算指示器:

```
(ts.used / ts.usable) * 100
```

其中:

- *ts.used* 是 第 654 页的『tablespace_used_pages - "表空间中的已使用页数"监视元素』的值
- *ts.usable* 是 第 653 页的『tablespace_usable_pages - "表空间中的可用页数"监视元素』的值

表空间利用率以消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

附加信息中对离空间已满所余时间的计算可以预测到消耗所有可用空间所余的时间。

tsc.tscont_util - "表空间容器利用率":

此运行状况指示器跟踪未使用自动存储器的每个 SMS 表空间的存储器消耗情况。

标识 tsc.tscont_util

运行状况监视器级别
表空间容器

类别 表空间存储器

类型 基于阈值上限

单位 百分比

如果对其定义容器的任何文件系统上都没有更多空间，那么认为 SMS 表空间已满。

如果文件系统上没有可用空间可供扩展 SMS 容器，那么表示关联表空间已满。

可对在用完可用空间的文件系统上定义的每个容器发出警报。

使用以下公式计算指示器:

```
(fs.used / fs.total)*100
```

其中 `fs` 是容器所在的文件系统。

SMS 表空间利用率以消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

附加信息中对离空间已满所余时间的计算可以预测到消耗所有可用空间所余的时间。

ts.ts_op_status - "表空间操作状态":

表空间的状态可以限制可执行的活动或任务。从正常状态切换至另一状态可能生成"注意"警报。

标识 ts.ts_op_status

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于状态

单位 不适用

tsc.tscont_op_status - "表空间容器操作状态":

此运行状况指示器跟踪表空间容器的可访问性。容器的可访问性可以限制可执行的活动或任务。如果容器不可访问，那么可能生成"注意"警报。

标识 tsc.tscont_op_status

运行状况监视器级别
表空间容器

类别 表空间存储器

类型 基于状态

单位 不适用

排序运行状况指示器:

db2.sort_privmem_util - "专用排序内存利用率":

此指示器跟踪专用排序内存的利用率。如果 db2.sort_heap_allocated（系统监视元素）>= *sheapthres*（DBM 配置参数），那么排序可能未成为 *sortheap* 参数定义的已满排序堆，并且可能会生成警报。

标识 db2.sort_privmem_util

运行状况监视器级别
数据库

类别 排序

类型 基于阈值上限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用以下公式计算指示器:

```
(db2.sort_heap_allocated / sheapthres)*100
```

“阈值后排序数”快照监视元素测量超过排序堆阈值后请求堆的排序数。此指示器的值显示在“其他详细信息”中，对此运行状况指示器指示问题的严重程度。

“使用的最大专用排序内存”快照监视元素保留实例的专用排序内存高水位标记。此指示器的值显示在“其他信息”中，指示自上次回收实例后在任一时间点使用的最大专用排序内存量。此值可用来帮助确定 *sheapthres* 的适当值。

db.sort_shrmem_util - “共享排序内存利用率”:

此指示器跟踪共享排序内存的利用率。*sheapthres_shr* 数据库配置参数是硬限制。如果分配接近该限制，那么会生成警报。

标识 db.sort_shrmem_util

运行状况监视器级别
数据库

类别 排序

类型 基于阈值上限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用以下公式计算指示器:

```
(db.sort_shrheap_allocated / sheapthres_shr)*100
```

注意，如果 *sheapthres_shr* 设置为 0，那么 *sheapthres* 充当共享排序堆阈值。

“使用的最大共享排序内存”快照监视元素保留数据库的共享排序内存高水位标记。此指示器的值显示在“其他信息”中，指示自数据库处于活动状态后在任一时间点使用的最大共享排序内存量。此值可用来帮助确定共享排序内存阈值的适当值。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.spilled_sorts - "溢出排序百分比":

溢出至磁盘的排序可能导致严重的性能下降。如果发生这种情况，那么会生成警报。

标识 db.spilled_sorts

运行状况监视器级别
数据库

类别 排序

类型 基于阈值上限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用以下公式计算指示器:

$$(db.sort\_overflows_t - db.sort\_overflows_{t-1}) / (db.total\_sorts_t - db.total\_sorts_{t-1}) * 100$$

其中 *t* 是当前快照，而 *t-1* 是 1 小时以前的快照。系统监视元素 db.sort_overflows（基于 sort_overflows 监视元素）是用完排序堆并且可能需要磁盘空间以供临时存储器使用的排序总数。元素 db.total_sorts（基于 total_sorts 监视元素）是已执行的排序总数。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.max_sort_shrmem_util - "长期共享排序内存利用率":

此指示器跟踪配置过度的共享排序堆，了解能否释放某些资源以用于 DB2 数据库系统中的其他位置。

标识 db.max_sort_shrmem_util

运行状况监视器级别
数据库

类别 排序

类型 基于阈值下限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用百分比很低时，可能会生成警报。

使用以下公式计算指示器:

$$(db.max\_shr\_sort\_mem / sheapthres\_shr)*100$$

系统监视元素 db.max_shr_sort_mem（基于 sort_shrheap_top 监视元素）是共享排序内存使用情况的高水位标记。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

数据库管理器（DBMS）运行状况指示器:

db2.db2_op_status - "实例操作状态":

如果实例状态未对正在执行的活动或任务产生限制，那么认为该实例的运行状况正常。

标识 db2.db2_op_status

运行状况监视器级别
实例

类别 DBMS

类型 基于状态

单位 不适用

状态可以是下列其中一项: "活动"、"停顿暂挂"、"已停顿"或"已关闭"。非"活动"状态可能会生成"注意"警报。

如果运行状况指示器进入"关闭"状态，运行状况监视器就无法执行 db2.db2_op_status 运行状况指示器的操作。例如，当指示器所监视的实例由于显式的停止请求或异常终止而进入不活动状态时，指示器就会进入"关闭"状态。如果要让实例在任何异常终止发生后自动重新启动，那么可以配置故障监视器（db2fm）以保持该实例的高可用性。

实例最高级别严重性警报状态:

此指示器表示要监视的实例的累积警报状态。实例的警报状态是要监视的实例、实例数据库及数据库对象的最高警报状态。

标识 不适用。此运行状况指示器没有配置或建议支持。

运行状况监视器级别
实例

类别 DBMS

类型 基于状态

单位 不适用

警报状态的顺序如下所示:

- 警报
- 警告
- 注意
- 正常

实例的警报状态确定 DB2 数据库系统的整体运行状况。

数据库运行状况指示器:

db.db_op_status - "数据库操作状态":

数据库的状态可以限制可执行的活动或任务。状态可以是下列其中一项: 活动、停顿暂挂、已停顿或已前滚。从活动切换至另一状态可能会生成“注意”警报。

标识 db.db_op_status

运行状况监视器级别
数据库

类别 数据库

类型 基于状态

单位 不适用

“数据库最高级别严重性警报状态”:

此指示器表示要监视的数据库的累积警报状态。数据库的警报状态是数据库及其对象的最高警报状态。

标识 不适用。此运行状况指示器没有配置或建议支持。

运行状况监视器级别
数据库

类别 数据库

类型 基于状态

单位 不适用

警报状态的顺序如下所示:

- 警报
- 警告
- 注意
- 正常

维护运行状况指示器:

db.tb_reorg_req – “需要重组”:

此运行状况指示器跟踪在数据库中重组表或索引的需要。表或对表定义的所有索引需要重组以消除碎片数据。重组将通过压缩信息并重构行或索引数据完成。此操作的结果是性能得到提高并且释放了表或索引中的空间。

标识 db.tb_reorg_req

运行状况监视器级别
数据库

类别 数据库维护

类型 基于集合状态

单位 不适用

通过在自动维护策略中指定要评估的表名，可以过滤此运行状况指示器评估的一组表。可通过使用“自动维护”向导来执行此操作。

可能会生成“注意”警报来指示需要重组。可通过将 AUTO_REORG 数据库配置参数设置为 ON 来自动进行重组。如果已经启用自动重组，那么“注意”警报表示有一个或多个自动重组无法成功完成，或者有需要重组的表，但由于每个数据库分区的表大小超过可以进行脱机重组的最大表大小限定，使重组工作无法执行。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

db.tb_runstats_req - “需要收集统计信息”:

此运行状况指示器跟踪收集数据库中的表及其索引的统计信息的需要。需要收集表和对表定义的所有索引的统计信息以缩短查询执行时间。

标识 db.tb_runstats_req

运行状况监视器级别
数据库

类别 数据库维护

类型 基于集合状态

单位 不适用

此运行状况指示器认为可使用 SQL 查询来限制这些表。附加信息中的作用域显示此查询用于系统表的子查询子句。

可能会生成“注意”警报来指示需要收集统计信息。可通过将 AUTO_RUNSTATS 数据库配置参数设置为 ON 以自动收集统计信息。如果启用了自动统计信息收集，那么“注意”警报将指示未成功完成一个或多个自动统计信息收集。

db.db_backup_req - “需要备份数据库”:

此运行状况指示器跟踪在数据库中执行备份的需要。恢复策略中应包含定期备份以保护数据，从而避免在硬件或软件故障时丢失数据。

标识 db.db_backup_req

运行状况监视器级别
数据库

类别 数据库维护

类型 基于状态

单位 不适用

此运行状况指示器根据上次备份后的经历时间和更改的数据量来确定需要进行数据库备份的时间。

可能会生成“注意”警报来指示需要进行数据库备份。可通过将 AUTO_DB_BACKUP 数据库配置参数设置为 ON 来自动进行数据库备份。如果启用了自动数据库备份，那么“注意”警报将指示未成功完成一个或多个自动数据库备份。

高可用性灾难恢复（HADR）运行状况指示器:

db.hadr_op_status - “HADR 操作状态”:

此运行状况指示器跟踪数据库的高可用性灾难恢复（HADR）操作状态。主服务器与备用服务器之间的状态可能为下列其中一项: 已连接、拥塞或已断开连接。从“已连接”状态切换至另一状态可能生成“注意”警报。

标识 db.hadr_op_status

运行状况监视器级别
数据库

类别 高可用性灾难恢复

类型 基于状态

单位 不适用

db.hadr_delay - “HADR 日志延迟”:

此运行状况指示器跟踪主数据库上的数据更改与备用数据库上的更改复制之间的当前平均延迟（以分钟计）。如果延迟值很高，那么在主数据库发生故障后转移至备用数据库时可能导致数据丢失。如果延迟值很高，还可能意味着因为主数据库先于备用数据库而导致需要接管时所花的停机时间更长。

标识 db.hadr_delay

运行状况监视器级别
数据库

类别 高可用性灾难恢复

类型 基于阈值上限

单位 分钟

日志记录运行状况指示器:

db.log_util - “日志利用率”:

此指示器跟踪在数据库中使用的总活动日志空间量。

标识 db.log_util

运行状况监视器级别
数据库

类别 日志记录

类型 基于阈值上限

单位 百分比

日志利用率以消耗空间的百分比来度量，出现高百分比时可能会生成警报。

使用以下公式计算指示器:

```
(db.total_log_used / (db.total_log_used + db.total_log_available)) * 100
```

与日志有关的数据库配置参数的值显示在附加信息中，这些值显示日志的当前分配。附加信息还包括具有最旧活动事务的应用程序的标识。可强制此应用程序释放日志空间。

db.log_fs_util - “日志文件系统利用率”:

日志文件系统利用率跟踪事务日志所在的文件系统的充满程度。

标识 db.log_fs_util

运行状况监视器级别
数据库

类别 日志记录

类型 基于阈值上限

单位 百分比

如果文件系统上没有空间，那么 DB2 数据库系统可能无法创建新的日志文件。

日志利用率以消耗空间的百分比来度量。如果文件系统中的可用空间量降至最低（即高百分比利用率），那么可能生成警报。

使用以下公式计算此指示器: *(fs.log_fs_used / fs.log_fs_total)*100*，其中 fs 是日志所在的文件系统。

与日志有关的数据库配置参数的值显示在附加信息中，这些值显示日志的当前分配。如果启用了用户出口，那么还会显示其他详细信息。

如果显示在附加详细信息中的“在日志磁盘已满时停止”设置为“是”并且利用率达到 100%，那么应尽快解决所有警报，以限制对不能落实事务的应用程序的影响，直到成功创建日志文件。

应用程序并发性运行状况指示器:

db.deadlock_rate -“死锁率”:

死锁率跟踪死锁出现在数据库上的比率以及应用程序遇到争用问题的等级。

标识 db.deadlock_rate

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 每小时产生的死锁数

死锁可能是由下列情况导致的:

- 数据库发生锁定升级
- 在系统生成的行锁定已足够的情况下应用程序显式锁定了表
- 绑定时应用程序使用了不适当的隔离级别
- 目录表已被锁定以供可重复读
- 应用程序正以不同的顺序获取相同的锁定，从而导致死锁。

使用以下公式计算指示器:

$(db.deadlocks_t - db.deadlocks_{t-1})$

其中 *t* 是当前快照，而 *t-1* 是上一个快照，即距获取当前快照之前 60 分钟时获取的快照。

死锁率越高，可能生成警报的争用等级就越高。

db.locklist_util - "锁定列表利用率":

此指示器跟踪要使用的锁定列表内存量。

标识 db.locklist_util

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 百分比

每个数据库有一个锁定列表，锁定列表包含由同时连接至数据库的所有应用程序挂起的锁定。这是对锁定列表内存设置的限制。一旦达到该限制，就会因为下列情况而使得性能下降:

- 锁定升级将行锁定转换为表锁定，从而降低了数据库中的共享对象的并行性。
- 因为应用程序等待有限数目的表锁定，所以应用程序间会出现更多死锁。因此将回滚事务。

当最大锁定请求数达到对数据库设置的限制时，将对应用程序返回错误。

使用以下公式计算指示器:

```
(db.lock_list_in_use / (locklist * 4096)) * 100
```

利用率以消耗内存的百分比来度量，出现高百分比表示状况不良。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配锁定内存资源。如果对锁定内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.lock_escal_rate - "锁定升级率":

此指示器跟踪锁定已从行锁定升级至表锁定从而影响事务并行性的次数。

标识 db.lock_escal_rate

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 每小时产生的锁定升级数

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量，或者所有应用程序消耗的锁定列表空间达到总锁定列表空间时，锁定将会升级。可用锁定列表空间量由 *maxlocks* 和 *locklist* 数据库配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，应用程序将使用锁定列表中为其他应用程序分配的空间。每个数据库有一个锁定列表，锁定列表包含由同时连接至数据库的所有应用程序挂起的锁定。当整个锁定列表已满时，将发生错误。

使用以下公式计算指示器:

```
(db.lock_escals_t - db.lock_escals_t-1)
```

其中“t”是当前快照，而“t-1”是上一个快照，即距获取当前快照之前 60 分钟时获取的快照。

死锁率越高，可能生成警报的争用等级就越高。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配锁定内存资源。如果对锁定内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.apps_waiting_locks - “等待锁定的应用程序的百分比”:

此指示器度量所有当前执行的等待锁定的应用程序所占的百分比。

标识 db.apps_waiting_locks

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 百分比

高百分比可能指示应用程序遇到并行性问题，这对性能有负面影响。

使用以下公式计算指示器:

```
(db.locks_waiting / db.appls_cur_cons) *100)
```

程序包高速缓存、目录高速缓存以及工作空间运行状况指示器:

db.catcache_hitratio - “目录高速缓存命中率”:

命中率是一个百分比，用于指示目录高速缓存对避免对磁盘上的目录的实际访问所起到的帮助作用。高命中率指示在避免实际磁盘 I/O 访问方面很成功。

标识 db.catcache_hitratio

运行状况监视器级别
数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

使用以下公式计算指示器:

```
(1-(db.cat_cache_inserts/db.cat_cache_lookups))*100
```

db.pkgcache_hitratio - “程序包高速缓存命中率”:

命中率是一个百分比，用于指示程序包高速缓存对避免从系统目录重新装入静态 SQL 的程序包和段以及避免重新编译动态 SQL 语句所起到的帮助作用。高命中率指示在避免这些活动方面很成功。

标识 db.pkgcache_hitratio

运行状况监视器级别
数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

使用以下公式计算指示器:

```
(1-(db.pkg_cache_inserts/db.pkg_cache_lookups))*100
```

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配程序包高速缓存内存资源。如果对程序包高速缓存内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.shrworkspace_hitratio -"共享工作空间命中率":

命中率是一个百分比，用于指示共享 SQL 工作空间对避免初始化要执行的 SQL 语句的各段所起到的帮助作用。高命中率指示在避免此操作方面很成功。

标识 db.shrworkspace_hitratio

运行状况监视器级别
数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

使用以下公式计算指示器:

```
(1-(db.shr_workspace_section_inserts/db.shr_workspace_section_lookups))*100
```

内存运行状况指示器:

db2.mon_heap_util -"监视器堆利用率":

此指示器跟踪基于带有标识 SQLM_HEAP_MONITOR 的内存池的监视器堆内存的消耗。

标识 db2.mon_heap_util

运行状况监视器级别
实例

类别 内存

类型 基于阈值上限

单位 百分比

对内存池标识 SQLM_HEAP_MONITOR 使用以下公式计算利用率:

```
(db2.pool_cur_size / db2.pool_max_size) * 100
```

一旦此百分比达到最大值 100%，监视器操作可能会失败。

db.db_heap_util -"数据库堆利用率":

此指示器跟踪基于带有标识 SQLM_HEAP_DATABASE 的内存池的监视器堆内存的消耗。

标识 db.db_heap_util

运行状况监视器级别
数据库

类别 内存

类型 基于阈值上限

单位 百分比

对内存池标识 SQLM_HEAP_DATABASE 使用以下公式计算利用率:

```
(db.pool_cur_size / db.pool_max_size) * 100
```

一旦此百分比达到最大值 100%，查询和操作可能会因为没有堆可用而失败。

联合运行状况指示器:

db.fed_nicknames_op_status - "昵称状态":

此运行状况指示器检查联合数据库中定义的所有昵称以确定是否存在无效昵称。如果数据源对象已删除或者已更改，又或者如果用户映射不正确，那么昵称可能会无效。

标识 db.fed_nicknames_op_status

运行状况监视器级别
数据库

类别 联合

类型 基于集合状态

单位 不适用

如果在联合数据库中定义的任何昵称无效，那么可能会生成"注意"警报。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

如果此运行状况指示器要检查昵称状态，那么 FEDERATED 数据库管理器参数必须设置为 YES。

db.fed_servers_op_status - "数据源服务器状态":

此运行状况指示器检查联合数据库中定义的所有数据源服务器以确定是否存在不可用的数据源服务器。如果数据源服务器停止、不再存在或者进行了错误的配置，那么数据源服务器可能不可用。

标识 db.fed_servers_op_status

运行状况监视器级别
数据库

类别 联合

类型 基于集合状态

单位 不适用

如果在联合数据库中定义的任何昵称无效，那么可能会生成“注意”警报。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

如果此运行状况指示器要检查数据源服务器状态，那么 FEDERATED 数据库管理器参数必须设置为 YES。

运行状况监视器接口

下表列示 API 的运行状况监视器接口:

注: 在版本 9.7 中，不推荐使用运行状况监视器，因此建议您不要使用这些 API，它们在将来的发行版中可能会被除去。

表 63. 运行状况监视器: API

监视任务	API
捕获运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH 的快照
捕获带有集合对象的完整列表的运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH 并且 agent_id 为 SQLM_HMON_OPT_COLL_FULL 的快照
捕获带有公式、附加信息和历史记录的运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH_WITH_DETAIL 的快照
捕获带有公式、附加信息、历史记录和集合对象的完整列表的运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH_WITH_DETAIL 并且 agent_id 为 SQLM_HMON_OPT_COLL_FULL 的快照
转换自描述数据流	db2ConvMonStream - 转换监视器流
估计运行状况快照的大小	db2GetSnapshotSize - 估计 db2GetSnapshot 输出缓冲区所需的大小

下表列示 CLP 命令的运行状况监视器接口:

注: 在版本 9.7 中，不推荐使用运行状况监视器，因此建议您不要使用这些命令，它们在将来的发行版中可能会被除去。

表 64. 运行状况监视器接口: CLP 命令

监视任务	CLP 命令
捕获运行状况快照	GET HEALTH SNAPSHOT 命令
捕获带有公式、附加信息和历史记录的运行状况快照	GET HEALTH SNAPSHOT WITH DETAILS 命令

下表列示 SQL 函数的运行状况监视器接口:

注: 在版本 9.7 中，不推荐使用运行状况监视器，因此建议您不要使用这些 SQL 函数，它们在将来的发行版中可能会被除去。

表 65. 运行状况监视器接口: SQL 函数

监视任务	SQL 函数
数据库管理器级别运行状况信息快照	HEALTH_DBM_INFO
数据库管理器级别运行状况指示器快照	HEALTH_DBM_HI
数据库管理器级别运行状况指示器历史记录快照	HEALTH_DBM_HI_HIS

表 65. 运行状况监视器接口: SQL 函数 (续)

监视任务	SQL 函数
数据库级别运行状况信息快照	HEALTH_DB_INFO
数据库级别运行状况指示器快照	HEALTH_DB_HI
数据库级别运行状况指示器历史记录快照	HEALTH_DB_HI_HIS
数据库级别运行状况指示器集合快照	HEALTH_DB_HIC
数据库级别运行状况指示器集合历史记录快照	HEALTH_DB_HIC_HIS
表空间级别运行状况信息快照	HEALTH_TBS_INFO
表空间级别运行状况指示器快照	HEALTH_TBS_HI
表空间级别运行状况指示器历史记录快照	HEALTH_TBS_HI_HIS
表空间容器级别运行状况信息快照	HEALTH_CONT_INFO
表空间容器级别运行状况指示器快照	HEALTH_CONT_HI
表空间容器级别运行状况指示器历史记录快照	HEALTH_CONT_HI_HIS

运行状况监视器 SQL 表函数:

下表列示所有快照表函数。每个表函数对应一个运行状况快照请求类型。

注: 在版本 9.7 中，不推荐使用运行状况监视器，因此建议您不要使用这些表函数，它们在将来的发行版中可能会被除去。

表 66. 快照监视器 SQL 表函数

监视器级别	SQL 表函数	返回的信息
数据库管理器	HEALTH_DBM_INFO	有关来自数据库管理器级别的运行状况快照的基本信息
数据库管理器	HEALTH_DBM_HI	来自数据库管理器级别的运行状况指示器信息
数据库管理器	HEALTH_DBM_HI_HIS	来自数据库管理器级别的运行状况指示器历史记录信息
数据库	HEALTH_DB_INFO	有关来自数据库的运行状况快照的基本信息
数据库	HEALTH_DB_HI	来自数据库的运行状况指示器信息
数据库	HEALTH_DB_HI_HIS	来自数据库的运行状况指示器历史记录信息
数据库	HEALTH_DB_HIC	数据库的集合运行状况指示器的集合信息
数据库	HEALTH_DB_HIC_HIS	数据库的集合运行状况指示器的集合历史记录信息
表空间	HEALTH_TBS_INFO	有关数据库的表空间的运行状况快照的基本信息
表空间	HEALTH_TBS_HI	有关数据库的表空间的运行状况指示器信息
表空间	HEALTH_TBS_HI_HIS	有关数据库的表空间的运行状况指示器历史记录信息
表空间	HEALTH_CONT_INFO	有关数据库的容器的运行状况快照的基本信息
表空间	HEALTH_CONT_HI	有关数据库的容器的运行状况指示器信息
表空间	HEALTH_CONT_HI_HIS	有关数据库的容器的运行状况指示器历史记录信息

运行状况监视器 CL 命令:

下表列示了所有受支持的快照请求类型。

表 67. 快照监视器 CLP 命令

监视器级别	CLP 命令	返回的信息
数据库管理器	`get health snapshot for dbm`	数据库管理器级别信息。
数据库	`get health snapshot for all databases`	数据库级别信息。仅当激活数据库后才会返回信息。
数据库	`get health snapshot for database on` *`database-alias`*	数据库级别信息。仅当激活数据库后才会返回信息。
数据库	`get health snapshot for all on` *`database-alias`*	数据库、表空间和表空间容器信息。仅当激活数据库后才会返回信息。
表空间	`get snapshot for tablespaces on` *`database-alias`*	连接到数据库的应用程序已访问的每个表空间的表空间级别信息。并且，返回的信息还包括该表空间中每个表空间容器的运行状况信息。

运行状况监视器 API 请求类型:

下表列示了所有受支持的快照请求类型。

表 68. 快照监视器 API 请求类型

监视器级别	API 请求类型	返回的信息
数据库管理器	SQLMA_DB2	数据库管理器级别信息。
数据库	SQLMA_DBASE_ALL	数据库级别信息。仅当激活数据库后才会返回信息。
数据库	SQLMA_DBASE	数据库级别信息。仅当激活数据库后才会返回信息。
表空间	SQLMA_DBASE_TABLESPACES	连接到数据库的应用程序已访问的每个表空间的表空间级别信息。并且，返回的信息还包括该表空间中每个表空间容器的运行状况信息。

使用内存可视化器

“内存可视化器”帮助数据库管理员监视实例及其所有数据库的内存相关性能。可以查看以分层树结构显示的内存组件的内存利用率的即时可视显示。

要点: 版本 9.7 中已经不推荐使用“内存可视化器”，在以后的发行版中可能会将其除去。有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

要查看内存性能和使用情况绘图及更新“内存可视化器”中的配置参数，必须具有 SYSADM 权限。

可以使用“内存可视化器”来对性能问题进行故障诊断。可更改内存组件的配置参数设置并评估更改的效果。因为内存是按需分配的，所以配置参数会影响 DB2 中的内存使用情况。如果将配置参数的值设置在可接受范围以外，那么会显示错误消息。更改配置参数会直接影响“内存可视化器”，并且新值将集成到下一次刷新周期中。

- 要使用内存可视化器查看内存性能:
  1. 从 Windows **开始**菜单单击**程序 → IBM DB2 → 监视 → 工具 → 内存可视化器**打开“内存可视化器”。 “内存可视化器”实例选择窗口将打开。从**实例名**字段选择实例并单击**确定**。
  2. 展开实例对象树直到分层树中显示数据库及其关联内存组件。内存池的值显示在“内存可视化器”窗口中。
  3. 要显示内存组件的已绘制的图，请使用下列其中一个方法:
     - 在分层树中选择一个组件，然后在“内存可视化器”窗口中单击**显示图形**复选框。
     - 右键单击所选内存组件以显示弹出菜单并选择**显示图形**。
     - 在分层树中选择一个组件，然后从“已选择”菜单的工具栏上选择**显示图形**选项。每个内存组件的已绘制的图数据出现在“内存使用情况绘图”中。
     - 要查看另一内存组件中的数据，从分层树中选择该组件，然后单击**显示图形**复选框。该内存组件的已绘制的数据与其他组件一起出现在“内存使用情况绘图”中。

     该图显示在一段时间内为内存组件收集的数据。每个组件用它们在“内存可视化器”窗口的**绘制图注**字段中显示的颜色和形状来表示。形状按一定时间间隔重复。标签用来标识图形绘制中的组件。

     捕获性能数据的时间显示在图的下面。可以更改该图的时间间隔。

     注: 将新内存组件添加至绘图时，它不会替换先前添加的内存组件。

     水平和垂直滚动条允许您从不同角度查看已绘制的数据。
     - 使用位于图底部的水平滚动条，以查看所选时间段内的内存组件历史数据。按住滑块并沿着图的底边拖动滑块。
     - 使用位于图右边的垂直滚动条，以查看所选组件的内存利用率。按住滑块并拖动滑块以更改视图。

     内存利用率达到新的高度时，垂直滚动条的最大值将更新以反映新值。可将垂直滚动条的最小值设置为 0 以外的值以查看另一范围的池利用率值。
- 可将“内存可视化器”数据文件中的数据装入到新的“内存可视化器”窗口中。此数据可用来针对历史数据比较实例及其所有数据库的性能。要从“内存可视化器”数据文件装入数据，从“内存可视化器”菜单中选择**打开**，然后从“打开对话框”中选择带有扩展名 *.mdf 的数据文件。
- 使用**时间单位**字段在“内存使用情况绘图”窗口中更改时间间隔。图数据的缺省时间间隔以分钟计。可选择以分钟、小时或天为单位的时间间隔。选择后，新的时间间隔将显示在该图的水平范围中并更改水平滚动条的增量移动幅度。
- 要从“内存使用情况绘图”中除去内存组件的已绘制的图，在分层树中选择一个组件并清除“内存可视化器”窗口中的**显示图形**复选框，或者右键单击所选内存组件以显示弹出菜单并取消选择**显示图形**。将从“内存使用情况绘图”窗口中除去该组件的已绘制的数据。表示组件的彩色形状不再显示在“内存可视化器”窗口的**绘制图注**字段中。
- 为帮助您跟踪和创建内存性能的历史记录，可在“内存可视化器”运行时保存内存性能数据，包括已绘制的图。要保存内存性能数据，从“内存可视化器”菜单中选择**保存**或**另存为**，然后选择文件位置及带有扩展名 .mdf 的文件名。

- 要更改内存组件的配置参数设置:
  1. 展开想要的内存池，这样就会看到其配置参数列示在分层树中。
  2. 单击组件以选择该组件，然后单击**参数值**列中的数字。文本框将显示该组件的当前值。在文本框中输入新数字并按 **Enter** 键。新值显示在**参数值**列中的原始值旁边，直到下一次刷新周期可能更新配置参数为止。还可右键单击**参数值**列中对应所选组件的值以显示弹出菜单。单击列的外部以完成更改。 内存组件的新值显示在**参数值**列中的原始值旁边。如果选择查看内存性能图，那么会在视图中看到新值。虽然此更改将立即在“内存可视化器”中生效，但在 DB2 中更新对配置参数的更改时还是会有延迟。可使用弹出菜单中的**复位为缺省值**选项复位配置参数的值。

内存可视化器概述

使用“内存可视化器”来监视实例及其所有数据库的内存相关性能。

要点: 版本 9.7 中已经不推荐使用“内存可视化器”，在以后的发行版中可能会将其除去。 有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

打开“内存可视化器”并在分层树中选择一个或多个内存组件，以在“内存可视化器”窗口中显示对应分配给该组件的内存量及当前内存使用情况的值。“内存可视化器”窗口显示两个数据视图: 树形视图和历史视图。一系列的列显示警报和警告的上限和下限的百分比阈值。这些列还将显示实时内存利用率。

注: “内存可视化器”可用来为版本 8.1 和更新版本的实例提供内存性能数据。

以下列表对可在“内存可视化器”中执行的一些关键任务进行了分类:

- 查看或隐藏有关 DB2 实例及其数据库的所选组件的内存利用率的各列中的数据。
- 查看内存性能数据图。
- 通过更新配置参数来更改各个内存组件的设置。
- 将文件中的性能数据装入到“内存可视化器”窗口中。
- 保存内存性能数据。

“内存可视化器”界面具有下列元素，可使用它们来帮助您监视实例及其所有数据库的内存相关性能。

内存可视化器窗口

“内存可视化器”窗口中的各列显示对应内存组件的性能的值。将显示以下信息:

绘制图注

显示在“内存使用情况绘图”中的所选内存组件或配置参数。按一定时间间隔出现在已绘制的图中的特定形状标识每个组件或参数。

利用率 分配给数据库对象或被数据库对象利用的内存的大小。包括显示利用率和已配置分配的图形柱。该柱的长度是固定的，填充部分以百分比的形式指示利用率。

参数值 配置参数的当前值。

警报上限（%）阈值

生成警报上限的阈值。缺省值为 98%。

警告上限（%）阈值
生成警告上限的阈值。缺省值为 90%。

警报下限（%）阈值
生成警报下限的阈值。缺省值为 2%。

警告下限（%）阈值
生成警告下限的阈值。缺省值为 10%。

图形使用柱

“内存可视化器”窗口中的图形使用柱是内存利用率的可视表示。这些柱可帮助您确定所选内存组件要使用的内存量及该使用量对系统造成的潜在影响。“内存可视化器”还会显示对应于使用情况的百分比值。这两个指示器可帮助您确定是否需要更改组件的配置参数设置或采用另一适当操作。

内存组件

“数据库管理器”在系统上使用以下不同类型的内存：数据库管理器共享内存、数据库全局内存、应用程序全局内存、代理程序/应用程序共享内存和代理程序专用内存。这些内存类型是“内存可视化器”在展开的分层树结构中使用的高级内存组件。

每个高级内存组件下面是用来确定分配或释放内存的方式的其他组件。例如，当数据库管理器启动时，将激活数据库，而应用程序将连接至数据库，或者指定代理程序以便为应用程序工作时，将分配和释放内存。“内存可视化器”使用这些叶级别内存组件来显示在 DB2 实例中分配和使用内存的方式。

分层树结构

“内存可视化器”使用分层树结构来帮助您显示和浏览 DB2 中的内存组件。分层树允许您展开和查看各个内存组件的列、图形显示和图的信息。

树形视图由四个主要内存项类型组成:

DB2实例
当前在系统上运行的实例

数据库 在实例上定义的数据库

高级内存组件
叶级别内存组件的逻辑分组。这些组包括：数据库管理器共享内存、数据库全局内存、代理程序专用内存和代理程序/应用程序共享内存。

叶级别内存组件
“内存可视化器”窗口中显示的内存组件，如缓冲池、排序堆、数据库堆和锁定列表。

树形视图中的图标表示每个内存树项:

- 实例: 
- 数据库: 
- 高级内存分组: 
- 叶级别内存组件: 

如果树项的内存利用率超过阈值，那么彩色指示器会覆盖图标。黄色指示警告条件。红色指示警报条件。

历史视图显示树形视图中选择的内存组件的数据。该数据包括分配的和利用的内存的值，已绘制的图以及运行“内存可视化器”时对配置参数所作的更改。将在“内存可视化器”中的特定时间段内保存该数据。可将内存性能数据保存至“内存可视化器”数据文件，以便跟踪、与其他数据进行比较或进行故障诊断。

内存使用情况绘图

内存使用情况绘图显示所选内存组件在“内存使用情况绘图”中的已绘制的数据。图中的每个组件由一种特定颜色标识，它还会显示在“内存可视化器”窗口的绘制图注列中。该图还会显示对配置参数设置的更改。除了请求更改的时间外，配置参数的初始值和新值设置也将显示在图中。它们将成为可在评估内存性能时使用的历史视图的一部分。

有关更多信息，请参阅第 166 页的『使用内存可视化器』。

活动监视器概述

使用“活动监视器”来监视应用程序性能和并行性、资源消耗以及数据库或数据库分区的 SQL 语句使用情况。

要点: 版本 9.7 中已经不推荐使用“活动监控器”，在以后的发行版中可能会将其除去。有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

活动监视器提供一组基于特定监视器数据子集的预定义报告。这些报告允许您将重点放在监视应用程序性能、应用程序并行性、资源消耗和 SQL 语句使用情况上。“活动监视器”还为大多数报告提供了建议。这些建议可以帮助您诊断发生数据库性能问题的原因，并且调整查询以便最佳地使用数据库资源。

第 171 页的图 7描述使用活动监视器解决问题的过程。

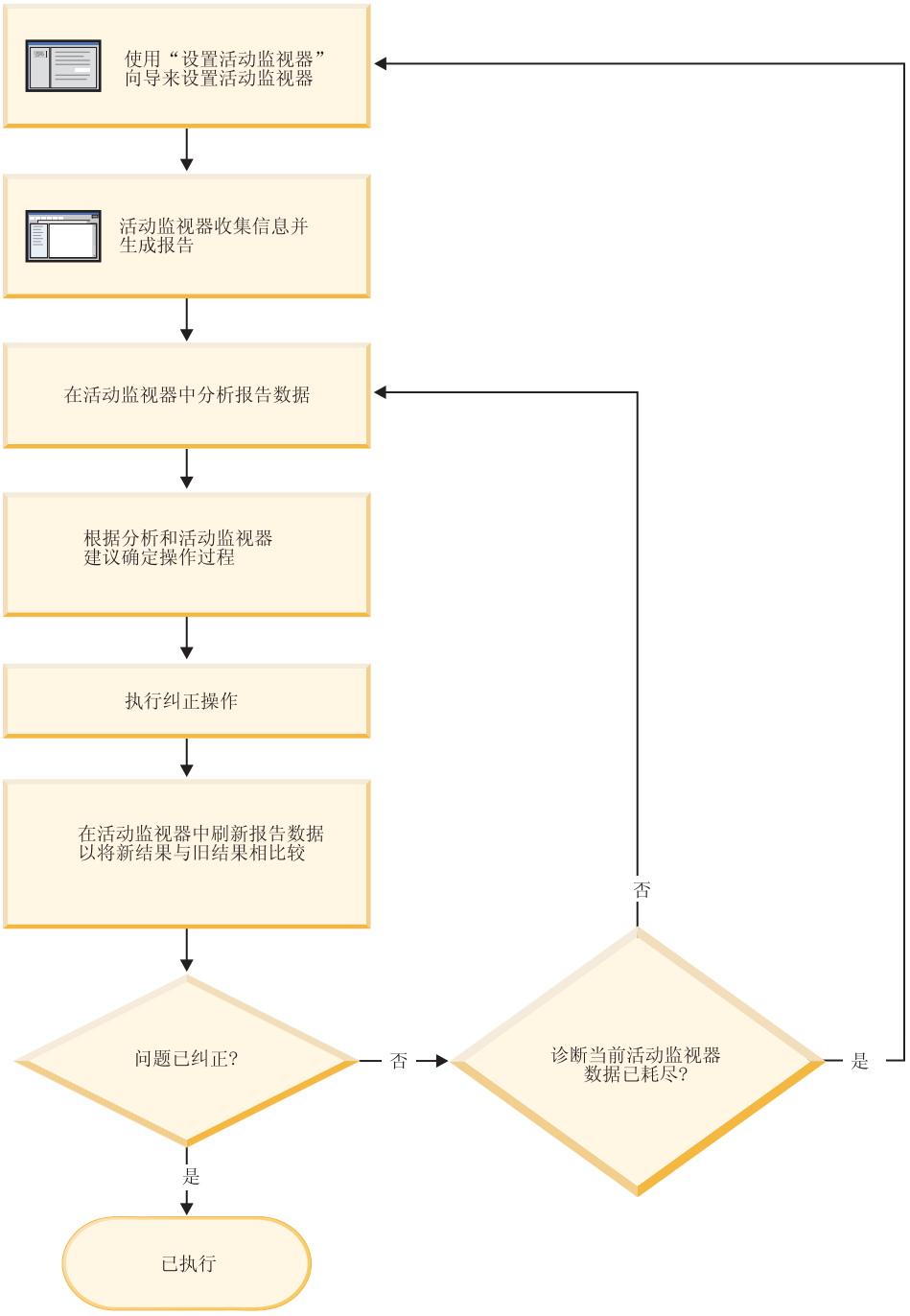

使用“设置活动监视器”向导来设置活动监视器
活动监视器收集信息并生成报告
在活动监视器中分析报告数据
根据分析和活动监视器建议确定操作过程
执行纠正操作
在活动监视器中刷新报告数据以将新结果与旧结果相比较
问题已纠正?
否
诊断当前活动监视器数据已耗尽?
否
是
是
已执行

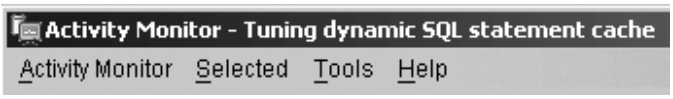
图 7. 活动监视器概述

表 69. 可在活动监视器中执行的任务

活动监视器中的任务	任务的各个方面	调用
事务	查看在所选应用程序上运行的事务。	在**报告数据**窗格中选择一个或多个应用程序。右键单击并选择**显示最新事务**。“应用程序事务”窗口将打开。
语句	查看在所选应用程序上运行的 SQL 语句。	在**报告数据**窗格中选择一个或多个应用程序。右键单击并选择**显示最新语句**。“应用程序语句”窗口将打开。
	查看在所选应用程序上运行的 SQL 语句的文本。	从“应用程序语句”窗口中，右键单击**报告数据**窗格中的语句。选择**显示语句文本**。
应用程序锁定链	查看当前影响所选应用程序的锁定和锁定等待情况。	在**报告数据**窗格中选择应用程序。右键单击并选择**显示锁定链**。“应用程序锁定链”窗口将打开。
	查看有关所选应用程序的信息，您正在查看该应用程序的锁定信息。	从“应用程序锁定链”窗口右键单击应用程序，然后选择**关于**。
	查看有关数据库中所选应用程序挂起的锁定和等待的锁定的信息。	从“应用程序锁定链”窗口右键单击应用程序，然后选择**显示锁定详细信息**。
查看报告数据和建议	查看可帮助您解释报告数据的信息。	在“活动监视器”窗口、“应用程序语句”窗口或“应用程序事务”窗口中，使用**报告**箭头键来选择该报告并单击**报告详细信息**按钮。查看**详细信息**页。
	查看活动监视器提供的建议	在“活动监视器”窗口、“应用程序语句”窗口或“应用程序事务”窗口中，使用**报告**箭头键来选择该报告并单击**报告详细信息**按钮。查看**建议**页。

“活动监视器”界面中有一些元素，可使用这些元素来帮助您组织和解释收集到的监视器数据:

菜单栏



使用菜单栏在“活动监视器”中处理对象，打开其他管理中心工具和访问联机帮助。

活动监视器工具栏

使用工具栏图标来打开 DB2 工具和查看 DB2 信息。

报告数据窗格

Report data

Application Handle (agent ID)	Application Name	Authorization ID	Application ID	Total CPU Time	User CPU Time
18	acmerpt.exe	EDWARDL	*LOCAL.DB2.00...	180259	10014
20	db2cc.exe	DB2ADMIN	*LOCAL.DB2.00...	30042	10014
22	acmefin.exe	FREDS	*LOCAL.DB2.00...	20028	20028
21	db2evm.exe	DB2ADMIN	*LOCAL.DB2.00...	20028	10014
27	acmeacct.exe	ALICET	*LOCAL.DB2.00...	10015	10015

使用**报告数据**窗格来显示和处理“活动监视器”中提供的报告数据。**报告数据**窗格显示构成**报告**字段中选择的报告内容的各项。

报告数据窗格还会提供对其他“活动监视器”窗口的访问。在“活动监视器”中，可从要监视的应用程序向下钻取至各个事务，或向下钻取至这些应用程序要运行的各个 SQL 语句。

报告数据窗格工具栏

使用**报告数据**窗格下面的工具栏来调整**报告数据**窗格中的对象和信息的视图，以满足您的需要。

监视器方案

示例: 使用快照管理视图来标识高成本应用程序

ShopMart 数据库上最近的工作负载增长已开始影响到整体数据库性能。Jessie 是 ShopMart 的 DBA，她尝试使用下列管理视图来标识日常工作负载中较大的资源使用者:

APPLICATION_PERFORMANCE

此视图帮助 Jessie 标识可能正在执行大型表扫描操作的应用程序:

```
connect to shopmart;
select AGENT_ID, ROWS_SELECTED, ROWS_READ from APPLICATION_PERFORMANCE;
```

ROWS_SELECTED 值显示返回给应用程序的行数，ROWS_READ 值显示在基本表中访问的行数。如果选择率很低，那么表示该应用程序可能正在执行表扫描操作，通过创建索引可以避免应用程序执行该操作。Jessie 使用此视图来标识可能有问题的查询，然后，她可以进行进一步调查，即查看 SQL 以确定是否能够减少查询在执行时读取的行数。

LONG_RUNNING_SQL

Jessie 使用 LONG_RUNNING_SQL 管理视图来标识当前正在执行的运行时间最长的查询:

```
connect to shopmart;
select ELAPSED_TIME_MIN, APPL_STATUS, AGENT_ID from long_running_sql
order by ELAPSED_TIME_MIN desc fetch first 5 rows only;
```

通过使用此视图，她可以确定这些查询已运行的时间长度以及这些查询的状态。如果某个查询已执行了很长时间并且正在等待锁，她就可以使用对特定代理程序标识执行查询的 LOCKWAITS 或 LOCK_HELD 管理视图来进行进一步调查。LONG_RUNNING_SQL 视图还会指出正在执行的语句并允许她标识可能有问题的 SQL。

QUERY_PREP_COST

Jessie 使用 QUERY_PREP_COST 来对已确定有问题的查询进行故障诊断。此视图可以指出查询的运行频率以及这些查询中每个查询的平均执行时间:

```
connect to shopmart;
select NUM_EXECUTIONS, AVERAGE_EXECUTION_TIME_S, PREP_TIME_PERCENT from
QUERY_PREP_COST order by NUM_EXECUTIONS desc;
```

PREP_TIME_PERCENT 值向 Jessie 指出准备查询时耗用的时间在查询执行时间中所占的百分比。如果编译和优化查询时耗用的时间几乎与查询的执行时间一样长，那么，Jessie 可以建议该查询的所有者更改用于该查询的优化类。降低优化类可以使该查询更快地完成优化，从而更快地返回结果。但是，如果某个查询需要相当长的时间来进行准备，但要执行数千次（而不必再次进行准备），那么更改优化类并不能提高查询性能。

TOP_DYNAMIC_SQL

Jessie 使用 TOP_DYNAMIC_SQL 视图来标识执行频率最高、运行时间最长和排序次数最多的动态 SQL 语句。有了此信息，Jessie 在进行 SQL 调整工作时就可以把注意力放在代表某些最大资源使用者的查询上。

为了标识运行频率最高的动态 SQL 语句，Jessie 发出以下语句:

```
connect to shopmart;
select * from TOP_DYNAMIC_SQL order by NUM_EXECUTIONS
desc fetch first 5 rows only;
```

此语句返回执行频率最高的 5 个动态 SQL 语句的所有执行时间、排序执行次数和语句文本详细信息。

为了标识执行时间最长的动态 SQL 语句，Jessie 检查 AVERAGE_EXECUTION_TIME_S 值最大的 5 个查询:

```
connect to shopmart;
select * from TOP_DYNAMIC_SQL order by AVERAGE_EXECUTION_TIME_S
desc fetch first 5 rows only;
```

为了查看排序次数最多的动态 SQL 语句的详细信息，Jessie 发出以下语句:

```
connect to shopmart;
select STMT_SORTS, SORTS_PER_EXECUTION, substr(STMT_TEXT,1,60) as STMT_TEXT
from TOP_DYNAMIC_SQL order by STMT_SORTS desc fetch first 5 rows only;
```

示例: 使用管理视图来监视缓冲池效率

John 是一位 DBA，他怀疑 SALES 数据库中应用程序性能不佳的原因是缓冲池的工作效率不高。为了进行调查，他使用 BP_HITRATIO 管理视图来查看缓冲池命中率:

```
connect to SALES;
select BPNAME, TOTAL_HIT_RATIO from BP_HIT_RATIO;
```

John 看到其中一个缓冲池的命中率非常低，这表示从磁盘读取了太多的页（而不是从缓冲池读取那些页）。

然后，他决定使用 BP_READ_IO 管理视图来了解是否需要对预取程序进行调整:

```
connect to SALES;
select BPNAME, PERCENT_SYNC_READS,UNUSED_ASYNC_READS_PERCENT from BP_READ_IO;
```

PERCENT_SYNC_READS 值指示了在未进行预取的情况下以异步方式读取的页所占的百分比。如果数值比较大，那么表示有很大一部分数据是直接从磁盘读取的，这意味

着需要更多的预取程序。UNUSED_ASYNC_READS_PERCENT 值指示以异步方式从磁盘读取但查询从未访问过的页所占的百分比。如果此数值较大，那么表示预取程序过度地读取数据页，从而产生不必要的 I/O。

由于 PERCENT_SYNC_READS 和 UNUSED_ASYNC_READS_PERCENT 值似乎都在可接受的范围内，所以 John 使用 BP_WRITE_IO 管理视图来调查页清除程序为传入数据页清除空间的效率:

```
connect to SALES;
select BPNAME, PERCENT_WRITES_ASYNC from BP_WRITE_IO;
```

PERCENT_WRITES_ASYNC 值指示以异步方式执行的物理写请求所占的百分比。如果此数值较大，那么表示页清除程序能够很好地在传入新数据页请求之前清除缓冲池空间。如果此数值较小，那么表示数据库代理进程执行了大量的物理写操作（在此期间，应用程序将等待数据页被读入缓冲池）。

John 看到 PERCENT_WRITES_ASYNC 值为 25%，这是一个非常小的值，因此他决定为 SALES 数据库配置更多页清除程序以提高异步写比率。在增加页清除程序数之后，他可以再次使用缓冲池管理视图来查看调整效果。

设置活动监视器

要监视应用程序性能和并行性、资源消耗以及数据库或数据库分区的 SQL 语句使用情况，可设置活动监视器。活动监视器提供一组基于特定监视器数据子集的预定义报告。它还提供一些建议，以帮助您诊断数据库性能问题的原因，以及调整查询以优化数据库资源的使用情况。

要点: 版本 9.7 中已经不推荐使用“活动监控器”，在以后的发行版中可能会将其除去。有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

要使用活动监视器:

- 您的服务器上必须有 DB2 UDB 版本 8.2 或更高版本
- 您必须具有 SQLADM 或 DBADM 权限。

打开“设置活动监视器”向导。

- 在“控制中心”中，展开对象树，直到找到想要对其设置活动监视器的实例或数据库。右键单击该对象，然后从弹出菜单中选择“设置活动监视器”。
- 从命令行输入以下命令: db2am。

在“控制中心”中通过上下文帮助工具提供详细信息。

回滚进程的进度监视

如果在事务回滚时获得应用程序快照，那么将在输出中看到回滚监视元素。此信息可用于监视回滚操作的进度。

应用程序快照中提供的信息包括回滚的开始时间、要完成的工作总量和完成的工作。工作度量为字节。

以下是 GET SNAPSHOT FOR ALL APPLICATIONS 命令的输出的示例:

```
                    应用程序快照

应用程序句柄                = 6
应用程序状态                = 回滚活动
    开始时间                = 02/20/2004 12:49:27.713720
    完成工作量              = 1024000 字节
    总工作量                = 4084000 字节

                    应用程序快照

应用程序句柄                = 10
应用程序状态                = 回滚到保存点
    开始时间                = 02/20/2004 12:49:32.832410
    完成工作量              = 102400 字节
    总工作量                = 2048000 字节
```

应用程序状态监视元素中的值指示正在发生的回滚事件的类型:

回滚处于活动状态

这是工作单元回滚: 整个事务的显式（用户调用）或隐式（强制）回滚。

保存点回滚

这是语句或应用程序级别保存点的潜在回滚。嵌套保存点被视为单个单元，使用最外部的保存点。

完成的工作单元显示已回滚的日志流中的相对位置。在处理每个日志记录之后对已完成的工作进行更新。因为日志记录的大小有所不同，所以更新并非均匀执行。

根据需要对事务或保存点回滚的日志流中日志记录的范围来估计“总工作”单元。它不指示需要处理的日志记录字节的精确数目。

使用快照监视器数据来监视分区表的重组

下列信息描述了一些最有用的全局表重组状态监视方法。

没有单独的用于指示分区表整体表重组状态的数据组。分区表使用了数据组织方案，即，表数据根据该表中一个或多个表分区键列中的值分布到多个存储对象（称为数据分区或范围）中。但是，可以根据所重组的各个数据分区数据组中的元素值来推断全局表重组状态。下列信息描述了一些最有用的全局表重组状态监视方法。

确定重组的数据分区数

通过计算表名和模式名相同的表数据监视器数据块数，可以确定表中重组的数据分区总数。此值指示启动了重组的数据分区数。示例 1 和 2 指示正在对 3 个数据分区进行重组。

标识所重组的数据分区

可以根据阶段开始时间（reorg_phase_start）来推断当前正在重组的数据分区。在 SORT/BUILD/REPLACE 阶段，与正在重组的数据分区相对应的监视器数据显示了最新阶段开始时间。在 INDEX_RECREATE 阶段，所有数据分区的阶段开始时间都是相同的。在示例 1 和 2 中，指示了 INDEX_RECREATE 阶段，因此所有数据分区的开始时间都是相同的。

标识索引重建要求

通过获取与任何一个正在重组的数据分区相对应的最大重组阶段数（reorg_max_phase）元素值，可以确定是否需要重建索引。如果 reorg_max_phase 值为 3 或 4，那么表示需要重建索引。示例 1 和 2 报告的 reorg_max_phase 值为 3，即表示需要重建索引。

以下样本输出来自一台 3 节点服务器，该服务器包含一个带有 3 个数据分区的表:

```
CREATE TABLE  sales (c1 INT, c2 INT, c3 INT)
    PARTITION BY RANGE (c1)
                (PART P1 STARTING FROM (1)  ENDING AT (10) IN parttbs,
         PART P2 STARTING FROM (11) ENDING AT (20) IN parttbs,
         PART P3 STARTING FROM (21) ENDING AT (30) IN parttbs)
    DISTRIBUTE BY (c2)
```

执行的语句:

```
REORG TABLE sales ALLOW NO ACCESS ON ALL DBPARTITIONNUMS
```

示例 1:

```
GET SNAPSHOT FOR TABLES ON DPARTDB GLOBAL
```

已将输出修改为仅包括相关表的表信息。

```
            表快照

第一个数据库连接时间戳记              = 06/28/2005 13:46:43.061690
上次复位时间戳记                      = 06/28/2005 13:46:47.440046
快照时间戳记                          = 06/28/2005 13:46:50.964033
数据库名称                            = DPARTDB
数据库路径                            = /work/sales/NODE0000/SQL00001/
输入数据库别名                        = DPARTDB
已访问的表的数目                      = 5

表列表
  表模式                 = NEWTON
  表名                   = SALES
  表类型                 = 用户
  数据分区标识           = 0
  数据对象页             = 3
  读取的行数             = 12
  写入的行数             = 1
  溢出数                 = 0
  重组的页数             = 0
  表重组信息:
   节点号              = 0
   重组类型            =
        回收
        表重组
        不允许访问
        通过表扫描重新集群
        仅重组数据
   重组索引数          = 0
   重组表空间数        = 3
 长临时空间标识        = 3
   开始时间            = 06/28/2005 13:46:49.816883
   重组阶段            = 3 - 索引重建
   最大阶段            = 3
   阶段开始时间        = 06/28/2005 13:46:50.362918
   状态                = 已完成
   当前计数器          = 0
   最大计数器          = 0
   完成                = 0
   结束时间            = 06/28/2005 13:46:50.821244
```

```
表重组信息:
  节点数              = 1
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:49.822701
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.420741
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.899543


表重组信息:
  节点数              = 2
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:49.814813
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.344277
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.803619


表模式                = NEWTON
表名                  = SALES
表类型                = 用户
数据分区标识          = 1
数据对象页            = 3
读取的行数            = 8
写入的行数            = 1
溢出数                = 0
重组的页数            = 0
表重组信息:
  节点号              = 0
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:50.014617
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
```

```
  阶段开始时间      = 06/28/2005 13:46:50.362918
  状态              = 已完成
  当前计数器        = 0
  最大计数器        = 0
  完成              = 0
  结束时间          = 06/28/2005 13:46:50.821244

表重组信息:
  节点数            = 1
  重组类型          =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数        = 0
  重组表空间数      = 3
长临时空间标识      = 3
  开始时间          = 06/28/2005 13:46:50.026278
  重组阶段          = 3 - 索引重建
  最大阶段          = 3
  阶段开始时间      = 06/28/2005 13:46:50.420741
  状态              = 已完成
  当前计数器        = 0
  最大计数器        = 0
  完成              = 0
  结束时间          = 06/28/2005 13:46:50.899543

表重组信息:
  节点数            = 2
  重组类型          =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数        = 0
  重组表空间数      = 3
长临时空间标识      = 3
  开始时间          = 06/28/2005 13:46:50.006392
  重组阶段          = 3 - 索引重建
  最大阶段          = 3
  阶段开始时间      = 06/28/2005 13:46:50.344277
  状态              = 已完成
  当前计数器        = 0
  最大计数器        = 0
  完成              = 0
  结束时间          = 06/28/2005 13:46:50.803619


表模式              = NEWTON
表名                = SALES
表类型              = 用户
数据分区标识        = 2
数据对象页          = 3
读取的行数        = 4
写入的行数          = 1
溢出数              = 0
重组的页数          = 0
表重组信息:
  节点号                          = 0
  重组类型          =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
```

```
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:50.199971
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.362918
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.821244

表重组信息:
  节点数              = 1
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:50.223742
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.420741
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.899543

表重组信息:
  节点数              = 2
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:50.179922
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.344277
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.803619
```

示例 2:

```
GET SNAPSHOT FOR TABLES ON DPARTDB AT DBPARTITIONNUM 2
```

已将输出修改为仅包括相关表的表信息。

```
                表快照

第一个数据库连接时间戳记              = 06/28/2005 13:46:43.617833
上次复位时间戳记                       =
快照时间戳记                          = 06/28/2005 13:46:51.016787
```

```
数据库名称                          = DPARTDB
数据库路径                          = /work/sales/NODE0000/SQL00001/
输入数据库别名                      = DPARTDB
已访问的表的数目                    = 3

表列表
 表模式              = NEWTON
 表名                = SALES
 表类型              = 用户
 数据分区标识        = 0
 数据对象页          = 1
 读取的行数                  = 0
 写入的行数          = 0
 溢出数              = 0
 重组的页数          = 0
 表重组信息:
   节点数            = 2
   重组类型          =
        回收
        表重组
        不允许访问
        通过表扫描重新集群
        仅重组数据
   重组索引数        = 0
   重组表空间数      = 3
 长临时空间标识      = 3
   开始时间          = 06/28/2005 13:46:49.814813
   重组阶段          = 3 - 索引重建
   最大阶段          = 3
   阶段开始时间      = 06/28/2005 13:46:50.344277
   状态              = 已完成
   当前计数器        = 0
   最大计数器        = 0
   完成              = 0
   结束时间          = 06/28/2005 13:46:50.803619


 表模式              = NEWTON
 表名                = SALES
 表类型              = 用户
 数据分区标识        = 1
 数据对象页          = 1
 读取的行数                  = 0
 写入的行数          = 0
 溢出数              = 0
 重组的页数          = 0
 表重组信息:
   节点数            = 2
   重组类型          =
        回收
        表重组
        不允许访问
        通过表扫描重新集群
        仅重组数据
   重组索引数        = 0
   重组表空间数      = 3
 长临时空间标识      = 3
   开始时间          = 06/28/2005 13:46:50.006392
   重组阶段          = 3 - 索引重建
   最大阶段          = 3
   阶段开始时间      = 06/28/2005 13:46:50.344277
   状态              = 已完成
   当前计数器        = 0
   最大计数器        = 0
   完成              = 0
   结束时间          = 06/28/2005 13:46:50.803619
```

```
表模式                = NEWTON
表名                  = SALES
表类型                = 用户
数据分区标识          = 2
数据对象页            = 1
读取的行数          = 4
写入的行数            = 1
溢出数                = 0
重组的页数            = 0
表重组信息:
  节点数              = 2
  重组类型            =
       回收
       表重组
       不允许访问
       通过表扫描重新集群
       仅重组数据
  重组索引数          = 0
  重组表空间数        = 3
长临时空间标识        = 3
  开始时间            = 06/28/2005 13:46:50.179922
  重组阶段            = 3 - 索引重建
  最大阶段            = 3
  阶段开始时间        = 06/28/2005 13:46:50.344277
  状态                = 已完成
  当前计数器          = 0
  最大计数器          = 0
  完成                = 0
  结束时间            = 06/28/2005 13:46:50.803619
```

示例 3:

```
SELECT * FROM SYSIBMADM.SNAPLOCK WHERE tabname = 'SALES';
```

已将输出修改为仅包括相关表的表信息的子集。

```
...   TBSP_NAME TABNAME LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS ...
     --------- ------- ------------------ ---------- -----------
...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
...   -         SALES      TABLE_LOCK      IX         GRNT        ...
...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...
...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
...   -         SALES      TABLE_LOCK      IX         GRNT        ...
...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...
...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
...   -         SALES      TABLE_LOCK      IX         GRNT        ...
...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...

 9 record(s) selected.
```

此查询（已继续）的输出。

```
... LOCK_ESCALATION LOCK_ATTRIBUTES DATA_PARTITION_ID DBPARTITIONNUM
   --------------- --------------- ------------------ --------------
...               0 INSERT          2                  2
...               0 NONE            -                  2
...               0 NONE            2                  2
...               0 INSERT          0                  0
...               0 NONE            -                  0
...               0 NONE            0                  0
...               0 INSERT          1                  1
...               0 NONE            -                  1
...               0 NONE            1                  1
```

示例 4:

```
SELECT * FROM SYSIBMADM.SNAPTAB WHERE tabname = 'SALES';
```

已将输出修改为仅包括相关表的表信息的子集。

```
... TABSCHEMA TABNAME TAB_FILE_ID TAB_TYPE   DATA_OBJECT_PAGES ROWS_WRITTEN ...
... --------- ------- ----------- ---------- ----------------- ------------ ...
... NEWTON    SALES             2 USER_TABLE                 1            1 ...
... NEWTON    SALES             4 USER_TABLE                 1            1 ...
... NEWTON    SALES             3 USER_TABLE                 1            1 ...

 3 record(s) selected.
```

此查询（已继续）的输出。

```
... OVERFLOW_ACCESSES PAGE_REORGS DBPARTITIONNUM TBSP_ID DATA_PARTITION_ID
... ----------------- ----------- -------------- ------- -----------------
...                 0           0              0       3                 0
...                 0           0              2       3                 2
...                 0           0              1       3                 1
```

示例 5:

```
SELECT * FROM SYSIBMADM.SNAPTAB_REORG WHERE tabname = 'SALES';;
```

已将输出修改为仅包括相关表的表信息的子集。

```
    REORG_PHASE   REORG_MAX_PHASE REORG_TYPE                                     ...
     ------------- --------------- ------------------------------------------
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
  INDEX_RECREATE             3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...

 9 record(s) selected.
```

此查询（已继续）的输出。

```
... REORG_STATUS REORG_TBSPC_ID DBPARTITIONNUM DATA_PARTITION_ID
    ------------------------- -------------- -----------------
... COMPLETED                 3              2                 0
... COMPLETED                 3              2                 1
... COMPLETED                 3              2                 2
... COMPLETED                 3              1                 0
... COMPLETED                 3              1                 1
... COMPLETED                 3              1                 2
... COMPLETED                 3              0                 0
... COMPLETED                 3              0                 1
... COMPLETED                 3              0                 2
```

示例 6: 表重组信息包括有关重组操作执行期间回收扩展数据块的信息。以下示例显示了相关的输出。

```
db2 -v "get snapshot for tables on wsdb"

 表重组信息:
   重组类型           =
        回收扩展数据块
        允许写访问
   重组索引数         = 0
```

```
重组表空间数       = 0
开始时间           = 10/22/2008 15:49:35.477532
重组阶段           = 12 - 释放
最大阶段           = 3
```

注: 来自 SQLM_DBMON_VERSION9_7 以前的监视器版本的任何快照请求都不会将任何回收重组状态返回给发出请求的客户机。

具有详细信息历史记录的死锁事件监视器跟踪的不活动语句

运行用于跟踪所有语句（并且可以是数据值）的死锁事件监视器时，如果单个应用程序在一个工作单元中包括数目极多的语句，那么该应用程序可能会耗尽系统监视器堆。如果同时有大量应用程序正在执行，那么也可能会耗尽监视器堆。

为了减少消耗的空间量，当应用程序的不活动语句数达到某个阈值时，该应用程序就会将不活动语句写入事件监视器。写入事件监视器后，就会释放这些不活动语句消耗的内存。此外，如果在任何时间应用程序无法从系统监视器堆中获取内存，那么该应用程序会将其当前不活动的所有语句写入事件监视器，然后再次尝试获取内存。如果第二次尝试失败，那么会记录一条消息，并且会截断应用程序正在处理的 UOW 的语句历史记录列表。

对任何一个应用程序保留的不活动语句数的缺省限制是 250。可通过使用注册表变量 DB2_MAX_INACT_STMTS 指定另一个值来更改此缺省值。用户可能希望选择另一个限制值，以便增加或减少用于不活动语句信息的系统监视器堆空间量。

每次将不活动语句写入事件监视器时，db2diag 日志文件中就会出现一条消息，指示不活动语句已写入事件监视器。每次超过不活动语句数的限制时，db2diag 日志文件中就会出现一条消息，指示已超过不活动语句数的限制。

因为现在应用程序可以将其语句历史记录条目记录在死锁上下文外（当达到上面提及的其中一个阈值时），所以需要一种机制将这些条目与死锁时记录的语句列表关联，以便进行分析。为此，用户可以查找满足以下条件的语句历史记录条目:

- `deadlock_id= 0`
- `participant_no= 0`
- `invocation_id=` 死锁的调用标识
- `application_id=` 参与死锁的应用程序的标识

写入表事件监视器时，还需要检查 evmon_activates 的数目。

注意:

- 对于使用 REOPT ALWAYS 绑定选项编译的 SQL 语句，死锁事件信息中将不会提供任何重新优化编译或语句执行数据值。
- 在协调程序节点中，如果由于前面部分中描述的情况而将不活动语句写入事件监视器，那么写入的所有记录的序列值将会更改，以反映正在处理的当前工作单元。这样做是为了帮助协调此数据与稍后死锁在同一工作单元中生成的任何数据，因为通过搜索 deadlock_id 为 0 的那些记录的序号和应用程序标识信息可以收集所有相关数据。这种更改表明在上一个工作单元中启动并在当前工作单元中仍处于活动状态的语句的工作单元信息不可用，因为其序号将被当前工作单元标识覆盖。远程节点中将不会出现这种行为（也就是说，不会覆盖原始工作单元信息），因此在尝试协调

死锁事件记录与死锁前写入的任何记录时一定要小心，因为当涉及的先前工作单元中有活动的带锁定游标时，序号可能会有所不同。

“Windows 管理规范”（WMI）简介

有一个建立管理基础结构标准的业界开端，它提供结合各种硬件和软件管理系统的信息的一种方法。此开端称为“基于 Web 的企业管理”（WBEM）。WBEM 是基于“公共信息模型”（CIM）模式的，它是由 Desktop Management Task Force（DMTF）派生的业界标准。

“Microsoft® Windows 管理规范”（WMI）实现了受支持的 Windows 平台的 WBEM 开端。WMI 在 Windows 企业网络中非常有用，使用它可以减少维护和管理企业网络组件的成本。WMI 提供:

- Windows 操作、配置和状态的一致模型。
- 允许访问管理信息的 COM API。
- 允许使用其他 Windows 管理服务。
- 一个灵活且可扩展的体系结构，它允许供应商编写其他 WMI 提供程序以支持新设备、应用程序和其他增强功能。
- 用来创建信息的详细查询的“WMI 查询语言”（WQL）。
- 一个 API，管理应用程序开发者可使用它来编写 Visual Basic 或“Windows 脚本编制主机”（WSH）脚本。

WMI 体系结构分为两个部分:

1. 包括“CIM 对象管理器”（CIMOM）的管理基础结构以及用来管理数据的中央存储区（称为 CIMOM 对象库）。CIMOM 允许应用程序以统一的方式访问管理数据。
2. WMI 提供程序。WMI 提供程序是 CIMOM 与受管对象之间的媒介。通过使用 WMI API，WMI 提供程序向 CIMOM 提供来自受管对象的数据、代表管理应用程序处理请求并生成事件通知。

“Windows 管理规范”（WMI）提供程序是标准的 COM 或 DCOM 服务器，它们充当受管对象与“CIM 对象管理器”（CIMOM）之间的介体。如果 CIMOM 接收到管理应用程序对 CIMOM 对象库中不存在的数据或对事件的请求，那么 CIMOM 将请求转发至 WMI 提供程序。WMI 提供程序为特定于其特定域的受管对象提供数据和事件通知。

DB2 数据库系统与 Windows 管理规范集成

“Windows 管理规范”（WMI）可通过 DB2 性能计数器并使用内置 PerfMon 提供程序来访问快照监视器。

WMI 可通过使用内置“注册表”提供程序访问 DB2 概要文件注册表变量。

“WMI 软件开发包”（WMI SDK）包括几个内置提供程序:

- PerfMon 提供程序
- 注册表事件提供程序
- 注册表提供程序
- Windows 事件日志提供程序
- Win32 提供程序

- WDM 提供程序

WMI 可使用内置“Windows 事件日志”提供程序来访问“事件日志”中的 DB2 错误。

DB2 数据库系统具有“DB2 WMI 管理”提供程序和样本 WMI 脚本文件，用于访问下列受管对象:

1. 数据库服务器的实例，包括分发的那些实例。可以执行下列操作:
   - 枚举实例
   - 配置数据库管理器参数
   - 启动/停止/查询 DB2 服务器服务的状态
   - 设置或建立通信
2. 数据库。可以执行下列操作:
   - 枚举数据库
   - 配置数据库参数
   - 创建/删除数据库
   - 备份/复原/前滚数据库

运行 WMI 应用程序之前需要向系统注册 DB2 WMI 提供程序。通过输入下列命令完成注册:

- mofcomp %DB2PATH%\bin\db2wmi.mof

  此命令将 DB2 WMI 模式的定义装入到系统中。
- regsvr %DB2PATH%\bin\db2wmi.dll

  此命令向 Windows 注册 DB2 WMI 提供程序 COM DLL。

在两个命令中，%DB2PATH% 是安装 DB2 的路径。另外，db2wmi.mof 是包含 the DB2 WMI 模式定义的 .MOF 文件。

与 WMI 基础结构集成有几个好处:

1. 您可以在基于 Windows 的环境中使用 WMI 提供的工具很容易地编写脚本来管理 DB2 服务器。提供了样本 Visual Basic（VBS）脚本，可用它来执行简单任务，例如，列示实例、创建和删除数据库以及更新配置参数。样本脚本包括在 DB2 应用程序开发 Windows 版产品中。
2. 可以创建功能强大的管理应用程序，它们使用 WMI 执行许多任务。这些任务可包括:
   - 显示系统信息
   - 监视 DB2 性能
   - 监视 DB2 系统资源消耗情况

   通过此类型的管理应用程序监视系统事件和 DB2 事件，可以更好地管理数据库。
3. 可使用现有 COM 和 Visual Basic 编程知识和技巧。通过提供 COM 或 Visual Basic 接口，程序员可以在开发企业管理应用程序时节省时间。

Windows 性能监视器简介

当使用 DB2 数据库管理器 Windows 版时，可以使用下列工具来监视性能:

- **DB2 性能专家**

  用于多平台的 DB2 性能专家 1.1 版根据与 DB2 数据库性能有关的信息来合并、报告、分析并建议自管理和资源调整更改。

- **DB2 运行状况中心**

  运行状况中心的功能使您能够通过不同方法来处理与性能相关的信息。这些功能在某些方面替代了控制中心的性能监视器功能。

  要点: 版本 9.7 中已经不推荐使用“运行状况中心”，在以后的发行版中可能会将其除去。 有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

- **Windows 性能监视器**

  “Windows 性能监视器”使您能够监视数据库和系统性能，可从向系统注册的任何性能数据提供程序检索信息。Windows 还提供有关计算机运行的所有方面的性能数据，包括下列方面:

  – CPU 使用情况
  – 内存使用率
  – 磁盘活动
  – 网络活动

向 Windows 性能监视器注册 DB2

安装程序自动向“Windows 性能监视器”注册 DB2。

要使“Windows 性能监视器”可访问 DB2 数据库和 DB2® Connect™ 性能信息，必须注册“DB2 Windows 版性能计数器”的 DLL。这也允许任何其他使用 Win32 性能 API 的 Windows 应用程序获取性能数据。要安装和注册“DB2 性能计数器”的 DLL（DB2Perf.DLL）并向“Windows 性能监视器”注册此 DDL，请输入:

```
db2perfi -i
```

注册该 DLL 的同时会在注册表的 services 选项中创建一个新键。其中一个条目给出 DLL 的名称，它提供计数器支持。另外三个条目给出该 DLL 中提供的函数的名称。这些函数包括:

Open 在一个进程中系统首次装入该 DLL 时调用。

Collect
从 DLL 请求性能信息时调用。

Close 卸装 DLL 时调用。

启用对 DB2 性能信息的远程访问

如果 DB2 Windows 版工作站与其他 Windows 计算机联网，那么可使用本节中描述的功能部件。

要从另一台 DB2 Windows 版计算机查看 Windows 性能对象，必须向 DB2 数据库管理器注册管理员用户名和密码。（“Windows 性能监视器”的缺省用户名 SYSTEM 是 DB2 数据库保留字，因此不能使用。）要注册该用户名，请输入:

```
db2perfr -r username password
```

注: 使用的 username 必须符合 DB2 数据库命名规则。

用户名和密码数据保存在注册表内的一个键中，并设置了安全性，它只允许管理员和 SYSTEM 帐户访问。编码该数据，以避免在注册表中存储管理员密码出现安全性问题。

注:

1. 向 DB2 数据库系统注册了用户名和密码组合之后，即使“性能监视器”的本地实例也将使用该用户名和密码显式登录。这就表示，如果向 DB2 数据库系统注册的用户名信息不匹配，那么“性能监视器”的本地会话将不显示 DB2 数据库性能信息。
2. 必须维护该用户名和密码组合，以便与 Windows 安全性数据库中存储的用户名和密码值相匹配。如果在 Windows 安全性数据库中更改了用户名或密码，那么必须复位用于远程性能监视的用户名和密码组合。
3. 要注销，请输入:

   ```
   db2perfr -u <username> <password>
   ```

显示 DB2 数据库和 DB2 Connect 性能值

要使用“性能监视器”显示 DB2 数据库和 DB2 Connect 性能值，只需从**添加到**框中选择您想要显示其值的性能计数器。此框显示性能对象的列表及其性能数据。选择一个对象，查看该对象提供的计数器列表。

一个性能对象也可以有多个实例。例如，LogicalDisk 对象提供诸如“% 磁盘读时间”和“磁盘字节/秒”之类的计数器；它还为计算机上的每个逻辑驱动器提供一个实例，包括“C:”和“D:”。

Windows 性能对象

Windows 提供了下列性能对象:

- **DB2 数据库管理器**

  此对象提供了单个 Windows 实例的常规信息。受到监视的 DB2 数据库实例显示为对象实例。

  由于实际原因和性能原因，每次只能从一个 DB2 数据库实例获取性能信息。“性能监视器”显示的 DB2 数据库实例受“性能监视器”进程中的 db2instance 注册表变量控制。如果您有多个 DB2 数据库实例在同时运行，并且想要查看多个实例的性能信息，对于每个要监视的 DB2 数据库实例，必须将 db2instance 设置为相关值，然后启动一个单独的“性能监视器”会话。

  如果运行的是分区数据库环境，那么一次只能从一个数据库分区服务器获取性能信息。在缺省情况下，显示缺省数据库分区（即带有逻辑端口 0 的数据库分区）的性能信息。要查看另一数据库分区的性能信息，必须启动一个单独的“性能监视器”会话，并将 DB2NODE 环境变量设置为要监视的数据库分区的数据库分区号。

- **DB2 数据库**

此对象提供特定数据库的信息。每个当前活动的数据库都有可用的信息。

- **DB2 应用程序**

  此对象提供特定的 DB2 数据库应用程序的信息。每个当前活动的 DB2 数据库应用程序都有可用的信息。

- **DB2 DCS 数据库**

  此对象提供特定的 DCS 数据库的信息。每个当前活动的数据库都有可用的信息。

- **DB2 DCS 应用程序**

  此对象提供特定的 DB2 DCS 应用程序的信息。每个当前活动的 DB2 DCS 应用程序都有可用的信息。

“Windows 性能监视器”将列示的对象取决于 Windows 计算机上安装了什么内容以及哪些应用程序是活动的。例如，如果安装了 DB2 数据库管理器并且已启动它，将列示“DB2 数据库管理器”对象。如果此计算机上还有一些 DB2 数据库和应用程序当前是活动的，也将列示那些 DB2 数据库和 DB2 应用程序对象。如果将 Windows 系统用作 DB2 Connect 网关，且有一些 DCS 数据库和应用程序当前是活动的，将列示 DB2 DCS 数据库和 DB2 DCS 应用程序对象。

访问远程 DB2 数据库性能信息

前面已讨论过允许远程访问“DB2 性能信息”。在**添加到**框中选择要监视的另一台计算机。这将显示一个列表，列示该计算机上所有可用的性能对象。

为了能够监视远程计算机上的 DB2 性能对象，安装在那台计算机上的 DB2 数据库或 DB2 Connect 代码的级别必须是版本 6 或更高版本。

复位 DB2 性能值

当应用程序调用 DB2 监视器 API 时，由于启动了 DB2 数据库服务器，因此返回的信息通常是累积值。但它经常可用于:

- 复位性能值
- 运行测试
- 再次复位值
- 重新运行测试

要复位数据库性能值，可使用 db2perfc 程序。请输入:

```
db2perfc
```

缺省情况下，此命令将复位所有活动 DB2 数据库的性能值。但也可指定要复位的数据库的列表。还可使用 -d 选项指定应复位 DCS 数据库的性能值。例如:

```
    db2perfc
db2perfc dbalias1 dbalias2 ... dbaliasn

    db2perfc  -d
db2perfc  -d  dbalias1 dbalias2 ... dbaliasn
```

第一个示例复位所有活动 DB2 数据库的性能值。第二个示例复位特定 DB2 数据库的性能值。第三个示例复位所有活动的 DB2 DCS 数据库的性能值。最后一个示例复位特定的 DB2 DCS 数据库的性能值。

db2perfc 程序复位当前访问相关 DB2 数据库服务器实例（即在运行 db2perfc 的会话中的 DB2INSTANCE 中的服务器实例）的数据库性能信息的所有程序的值。

当执行 db2perfc 命令时，调用 db2perfc 也能复位远程访问 DB2 数据库性能信息的人员所见到的值。

注: 有一个 DB2 数据库 API sqlmrset，它允许应用程序复位其在本地（而非全局）看到的特定数据库的值。

不确定事务管理器概述

使用“不确定事务管理器”窗口来处理不确定事务。该窗口列示所选数据库和一个或多个所选分区的所有不确定事务。

要点: 版本 9.7 中已经不推荐使用“不确定事务管理器”，在以后的发行版中可能会将其除去。 有关更多信息，请参阅《DB2 版本 9.7 新增内容》一书中的“已经不推荐使用控制中心工具和 DB2 管理服务器（DAS）”主题。

不确定事务是处于不确定状态的全局事务。当资源所有者（如数据库管理员）不能等待事务管理器执行再同步操作时，DB2 将提供数据库管理员可对不确定事务执行的试探操作。例如，如果通信线路中断，并且不确定事务正在安排需要的资源（如针对表和索引的锁定、日志空间和事务使用的存储器），那么可能会出现这种情况。

虽然由事务管理器启动再同步操作比较好，但有时可能必须对不确定事务执行试探操作。在这些情况下，使用试探操作时应特别谨慎，不到万不得已不要使用，并且应遵循这些准则。

- 事务标识的 *gtrid* 部分是参与全局事务的其他资源管理器（RM）中的全局事务标识。
- 使用您掌握的有关该应用程序和操作环境的知识，来确定其他参与的资源管理器，
- 如果事务管理器为 CICS®，并且唯一资源管理器为 CICS 资源，那么执行试探回滚。
- 如果事务管理器不是 CICS，那么使用它来确定与不确定事务具有相同 *gtrid* 的事务的状态。
- 如果至少有一个资源管理器已经落实或回滚，那么执行试探落实或回滚。
- 如果所有事务处于已编译状态，那么执行试探回滚。
- 如果至少有一个资源管理器不可用，那么执行试探回滚。

要在 Intel 平台上打开不确定事务管理器，从**开始**菜单单击**开始 -> 程序 ->IBM DB2 -> 监视工具 -> 不确定事务管理器**。

要在 UNIX 或 Intel 上使用命令行打开不确定事务管理器，请运行以下命令:

```
db2indbt
```

可对不确定事务执行试探操作:

- 遗忘

  它允许资源管理器通过除去日志记录并释放日志页来擦除试探性完成的事务的痕迹。试探性完成的事务就是试探性落实或回滚的事务。对于所选数据库及一个或多个所选分区，可对试探性落实或回滚的事务使用遗忘操作。要遗忘不确定事务，选择数据库和分区，然后右键单击状态为**已落实**或**已回滚**的事务，然后从弹出菜单中选择**遗忘**。将显示确认消息。

- 落实

  这会落实准备好落实的不确定事务。如果操作成功，事务的状态将变为试探性落实。要落实不确定事务，选择数据库和分区，然后右键单击状态为**不确定**或**落实确认已丢失**的事务，然后从弹出菜单中选择**落实**。将显示确认消息。
- 回滚

  这会回滚已准备好的不确定事务。如果操作成功，事务的状态将变为试探性回滚。要回滚不确定事务，选择数据库和分区，然后右键单击状态为**不确定**或**已结束**的事务，然后从弹出菜单中选择**回滚**。将显示确认消息。

要对不确定事务执行这些操作，必须具有 SYSADM 或 DBADM 权限。

“不确定事务管理器”窗口中的各列将提供指定视图，可使用这些视图来以不同方式组织和显示不确定事务。以下列表描述界面中的各列:

状态 事务的不确定状态包括: 已落实（c）、已结束（e）、不确定（i）、落实确认已丢失（m）和已回滚（r）:

已落实 处于此状态的事务已试探性落实。

已结束 处于此状态的事务可能已超时。

不确定 处于此状态的事务等待落实或回滚。

落实确认已丢失
事务管理器在落实事务之前等待接收确认。

已回滚 处于此状态的事务已试探性回滚。

时间戳记
事务进入已编译（不确定）状态时服务器上的时间戳记。该时间是客户机的本地时间。

事务标识
事务管理器为唯一标识全局事务而指定的 XA 标识。

Application ID
数据库管理器为此事务指定的应用程序标识。

Authorization ID
运行该事务的用户的标识。

序号 数据库管理器指定为应用程序标识扩展的序号。

分区 不确定事务所在的分区。

发起方 指示是 XA 还是 DB2 在分区数据库环境中发起事务。

日志满载
指示此事务是否导致日志满载情况。

类型 显示数据库在每个不确定事务中的角色的类型信息。

- **TM** 指示不确定事务将该数据库用作事务管理器数据库。
- **RM** 指示不确定事务将该数据库用作资源管理器。这表示它是参与事务的数据库之一，但不是事务管理器数据库。

第 2 部分 监视元素

第 7 章 监视器表函数中报告的监视元素

DB2 版本 9.7 引入了多个通过新增的监视器表函数报告的监视元素。

这些监视元素将提供关于系统处理、活动和数据对象（例如表、表空间、表空间容器和缓冲池）的信息。

- 第 331 页的『client_wrkstnname －“客户机工作站名称”监视元素』
- 第 334 页的『comp_env_desc －“编译环境”监视元素』
- 第 337 页的『concurrentdbcoordactivities_db_threshold_id －“并发数据库协调程序活动数据库阈值标识”监视元素』
- 第 337 页的『concurrentdbcoordactivities_db_threshold_queued －“已由并发数据库协调程序活动数据库阈值排队”监视元素』
- 第 337 页的『concurrentdbcoordactivities_db_threshold_value －“并发数据库协调程序活动数据库阈值”监视元素』
- 第 338 页的『concurrentdbcoordactivities_db_threshold_violated －“违反并发数据库协调程序活动数据库阈值”监视元素』
- 第 338 页的『concurrentdbcoordactivities_subclass_threshold_id －“并发数据库协调程序活动服务子类阈值标识”监视元素』
- 第 338 页的『concurrentdbcoordactivities_subclass_ threshold_queued －“已由并发数据库协调程序活动服务子类阈值排队”监视元素』
- 第 339 页的『concurrentdbcoordactivities_subclass_ threshold_value －“并发数据库协调程序活动服务子类阈值”监视元素』
- 第 339 页的『concurrentdbcoordactivities_subclass_ threshold_violated －“违反并发数据库协调程序活动服务子类阈值”监视元素』
- 第 340 页的『concurrentdbcoordactivities_superclass_ threshold_id －“并发数据库协调程序活动服务超类阈值标识”监视元素』
- 第 340 页的『concurrentdbcoordactivities_superclass_ threshold_queued －“已由并发数据库协调程序活动服务超类阈值排队”监视元素』
- 第 340 页的『concurrentdbcoordactivities_superclass_ threshold_value －“并发数据库协调程序活动服务超类阈值”监视元素』
- 第 341 页的『concurrentdbcoordactivities_superclass_ threshold_violated －“违反并发数据库协调程序活动服务超类阈值”监视元素』
- 第 341 页的『concurrentdbcoordactivities_work_action_set_ threshold_id －“并发数据库协调程序活动工作操作集阈值标识”监视元素』
- 第 341 页的『concurrentdbcoordactivities_work_action_set_ threshold_queued －“已由并发数据库协调程序活动工作操作集阈值排队”监视元素』
- 第 342 页的『concurrentdbcoordactivities_work_action_set_ threshold_value －“并发数据库协调程序活动工作操作集阈值”监视元素』
- 第 342 页的『concurrentdbcoordactivities_work_action_set_ threshold_violated －“违反并发数据库协调程序活动工作操作集阈值”监视元素』
- 第 344 页的『container_accessible －“容器可访问”监视元素』
- 第 345 页的『container_id －“容器标识”监视元素』
- 第 345 页的『container_name －“容器名称”监视元素』
- 第 346 页的『container_stripe_set －“容器分割集”监视元素』
- 第 346 页的『container_total_pages －“容器中的总页数”监视元素』
- 第 346 页的『container_type －“容器类型”监视元素』
- 第 347 页的『container_usable_pages －“容器中的可用页数”监视元素』
- 第 351 页的『coord_member －“协调程序成员”监视元素』

- 第 355 页的『cputime_threshold_id -“CPU 时间阈值标识”监视元素』
- 第 355 页的『cputime_threshold_value -“CPU 时间阈值”监视元素』
- 第 356 页的『cputime_threshold_violated -“违反 CPU 时间阈值”监视元素』
- 第 356 页的『cputimeinsc_threshold_id -“服务类中 CPU 时间阈值标识”监视元素』
- 第 356 页的『cputimeinsc_threshold_value -“服务类中 CPU 时间阈值”监视元素』
- 第 357 页的『cputimeinsc_threshold_violated -“违反服务类中 CPU 时间阈值”监视元素』
- 第 359 页的『current_extent -“当前正在移动的扩展数据块”监视元素』
- 第 360 页的『data_partition_id -“数据分区标识”监视元素』
- 第 365 页的『db_storage_path_state -“存储器路径状态”监视元素』
- 第 365 页的『db_storage_path_with_dpe -“包含数据库分区表达式的存储器路径”监视元素』
- 第 365 页的『db_work_action_set_id -“数据库工作操作集标识”监视元素』
- 第 366 页的『db_work_class_id -“数据库工作类标识”监视元素』
- 第 369 页的『deadlocks -“检测到的死锁数”监视元素』
- 第 370 页的『del_keys_cleaned -“清除伪删除键数目”监视元素』
- 第 372 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
- 第 373 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』
- 第 373 页的『direct_read_reqs -“直接读请求数”监视元素』
- 第 375 页的『direct_read_time -“直接读时间”监视元素』
- 第 376 页的『direct_reads -“直接读数据库数目”监视元素』
- 第 378 页的『direct_write_reqs -“直接写请求数”监视元素』
- 第 380 页的『direct_write_time -“直接写时间”监视元素』
- 第 381 页的『direct_writes -“直接写数据库数目”监视元素』
- 第 384 页的『eff_stmt_text -“有效语句文本”监视元素』
- 第 385 页的『effective_isolation -“有效隔离级别”监视元素』
- 第 385 页的『effective_lock_timeout -“有效锁定超时”监视元素』
- 第 385 页的『effective_query_degree -“有效查询并行度”监视元素』
- 第 386 页的『empty_pages_deleted -“删除的空页数”监视元素』
- 第 387 页的『empty_pages_reused -“复用的空页数”监视元素』
- 第 387 页的『entry_time -“进入时间”监视元素』
- 第 387 页的『estimatedsqlcost_threshold_id -“估计 SQL 成本阈值标识”监视元素』
- 第 387 页的『estimatedsqlcost_threshold_value -“估计 SQL 成本阈值”监视元素』
- 第 388 页的『estimatedsqlcost_threshold_violated -“违反估计 SQL 成本阈值”监视元素』
- 第 389 页的『executable_id -“可执行文件标识”监视元素』
- 第 391 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』
- 第 392 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』
- 第 393 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』
- 第 393 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』

- 第 394 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』
- 第 395 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』
- 第 395 页的『fcm_recv_volume -“FCM 接收量”监视元素』
- 第 396 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』
- 第 397 页的『fcm_recvs_total -“FCM 接收总计”监视元素』
- 第 398 页的『fcm_send_volume -“FCM 发送量”监视元素』
- 第 399 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』
- 第 400 页的『fcm_sends_total -“FCM 发送总计”监视元素』
- 第 401 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』
- 第 402 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』
- 第 403 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』
- 第 404 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』
- 第 405 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』
- 第 405 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』
- 第 407 页的『files_closed -“关闭数据库文件数”监视元素』
- 第 408 页的『fs_caching -“文件系统高速缓存”监视元素』
- 第 409 页的『fs_id -“唯一文件系统标识号”监视元素』
- 第 409 页的『fs_total_size -“文件系统总大小”监视元素』
- 第 410 页的『fs_used_size -“文件系统中的已用空间量”监视元素』
- 第 426 页的『iid -“索引标识”监视元素』
- 第 427 页的『include_col_updates -“更新包括列次数”监视元素』
- 第 428 页的『index_only_scans -“纯索引扫描次数”监视元素』
- 第 428 页的『index_scans -“索引扫描次数”监视元素』
- 第 428 页的『index_tbsp_id -“索引表空间标识”监视元素』
- 第 429 页的『insert_timestamp -“语句插入时间戳记”监视元素』
- 第 432 页的『int_node_splits -“中间节点分割次数”监视元素』
- 第 435 页的『invocation_id -“调用标识”监视元素』
- 第 435 页的『ipc_recv_volume -“进程间通信接收量”监视元素』
- 第 436 页的『ipc_recv_wait_time -“进程间通信接收等待时间”监视元素』
- 第 436 页的『ipc_recvs_total -“进程间通信接收总次数”监视元素』
- 第 437 页的『ipc_send_volume -“进程间通信发送量”监视元素』
- 第 438 页的『ipc_send_wait_time -“进程间通信发送等待时间”监视元素』
- 第 439 页的『ipc_sends_total -“进程间通信发送总次数”监视元素』
- 第 440 页的『key_updates -“更新键次数”监视元素』
- 第 441 页的『last_extent -“移动的最后一个扩展数据块”监视元素』
- 第 441 页的『last_reference_time -“上次引用时间”监视元素』
- 第 444 页的『local_start_time -“本地开始时间”监视元素』
- 第 446 页的『lock_escals -“锁定升级次数”监视元素』
- 第 454 页的『lock_timeouts -“锁定超时次数”监视元素』
- 第 456 页的『lock_wait_time -“等待锁定时间”监视元素』

- 第 458 页的『lock_waits -“等待锁定次数”监视元素』
- 第 461 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』
- 第 462 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』
- 第 463 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』
- 第 467 页的『long_tbsp_id -“长表空间标识”监视元素』
- 第 479 页的『member -“数据库成员”监视元素』
- 第 481 页的『nesting_level -“嵌套级别”监视元素』
- 第 482 页的『nleaf -“叶子页数”监视元素』
- 第 482 页的『nlevels -“索引层数”监视元素』
- 第 483 页的『nonboundary_leaf_node_splits -“非边界叶节点分割次数”监视元素』
- 第 484 页的『num_executions -“语句执行次数”监视元素』
- 第 485 页的『num_exec_with_metrics -“在收集度量值情况下的执行次数”监视元素』
- 第 485 页的『num_extents_left -“尚未处理的扩展数据块数”监视元素』
- 第 485 页的『num_extents_moved -“移动的扩展数据块数”监视元素』
- 第 486 页的『num_log_buffer_full -“日志缓冲区变满次数”监视元素』
- 第 489 页的『num_remaps -“重新映射次数”监视元素』
- 第 497 页的『overflow_accesses -“访问溢出记录次数”监视元素』
- 第 498 页的『overflow_creates -“创建溢出行数”监视元素』
- 第 498 页的『package_name -“程序包名”监视元素』
- 第 498 页的『package_schema -“程序包模式”监视元素』
- 第 499 页的『package_version_id -“程序包版本”监视元素』
- 第 499 页的『page_allocations -“分配页数”监视元素』
- 第 500 页的『pages_from_block_ios -“块 I/O 读取总页数”监视元素』
- 第 501 页的『pages_from_vectored_ios -“向量 I/O 读取总页数”监视元素』
- 第 501 页的『pages_merged -“合并页数”监视元素』
- 第 501 页的『pages_read -“读取页数”监视元素』
- 第 502 页的『pages_written -“写入页数”监视元素』
- 第 502 页的『parent_activity_id -“父活动标识”监视元素』
- 第 502 页的『parent_uow_id -“父工作单元标识”监视元素』
- 第 509 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』
- 第 510 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』
- 第 511 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』
- 第 511 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』
- 第 512 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』
- 第 513 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』
- 第 515 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』
- 第 515 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』
- 第 516 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』
- 第 518 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
- 第 520 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

- 第 521 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
- 第 523 页的『pool_drty_pg_steal_clns -“触发缓冲池牺牲页清除程序次数”监视元素』
- 第 524 页的『pool_drty_pg_thrsh_clns -“触发缓冲池阈值清除程序次数”监视元素』
- 第 526 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
- 第 528 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
- 第 529 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
- 第 531 页的『pool_lsn_gap_clns -“触发缓冲池日志空间清除程序次数”监视元素』
- 第 532 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』
- 第 533 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
- 第 535 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
- 第 537 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
- 第 539 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
- 第 540 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
- 第 542 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
- 第 544 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
- 第 546 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
- 第 547 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
- 第 549 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
- 第 551 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
- 第 553 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』
- 第 555 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』
- 第 557 页的『prep_time -“编译时间”监视元素』
- 第 563 页的『pseudo_deletes -“伪删除数”监视元素』
- 第 564 页的『pseudo_empty_pages -“伪空页数”监视元素』
- 第 564 页的『qp_query_id -“Query Patroller 查询标识”监视元素』
- 第 565 页的『query_cost_estimate -“查询估计成本”监视元素』
- 第 570 页的『reclaimable_space_enabled -“已启用可回收空间指示器”监视元素』
- 第 581 页的『root_node_splits -“根节点分割次数”监视元素』
- 第 582 页的『routine_id -“例程标识”监视元素』
- 第 582 页的『rows_deleted -“删除行数”监视元素』
- 第 583 页的『rows_inserted -“插入行数”监视元素』
- 第 584 页的『rows_modified -“修改的行数”监视元素』
- 第 585 页的『rows_read -“读取行数”监视元素』
- 第 586 页的『rows_returned -“返回的行数”监视元素』
- 第 589 页的『rows_updated -“更新行数”监视元素』
- 第 590 页的『rqsts_completed_total -“完成请求总数”监视元素』
- 第 591 页的『sc_work_action_set_id -“服务类工作操作集标识”监视元素』
- 第 591 页的『sc_work_class_id -“服务类工作类标识”监视元素』
- 第 593 页的『section_number -“节号”监视元素』
- 第 594 页的『section_type -“节类型指示器”监视元素』

- 第 599 页的『service_class_id -“服务类标识”监视元素』
- 第 600 页的『service_subclass_name -“服务子类名”监视元素』
- 第 600 页的『service_superclass_name -“服务超类名”监视元素』
- 第 605 页的『sort_overflows -“排序溢出数”监视元素』
- 第 610 页的『sqlrowsread_threshold_id -“读取 SQL 行数阈值标识”监视元素』
- 第 610 页的『sqlrowsread_threshold_value -“读取 SQL 行数阈值”监视元素』
- 第 610 页的『sqlrowsread_threshold_violated -“违反读取 SQL 行数阈值”监视元素』
- 第 611 页的『sqlrowsreadinsc_threshold_id -“服务类中读取 SQL 行数阈值标识”监视元素』
- 第 611 页的『sqlrowsreadinsc_threshold_value -“服务类中读取 SQL 行数阈值”监视元素』
- 第 611 页的『sqlrowsreadinsc_threshold_violated -“违反服务类中读取 SQL 行数阈值”监视元素』
- 第 612 页的『sqlrowsreturned_threshold_id -“返回所读取 SQL 行数阈值标识”监视元素』
- 第 612 页的『sqlrowsreturned_threshold_value -“返回所读取 SQL 行数阈值”监视元素』
- 第 612 页的『sqlrowsreturned_threshold_violated -“违反返回所读取 SQL 行数阈值”监视元素』
- 第 613 页的『sqltempspace_threshold_id -“SQL 临时空间阈值标识”监视元素』
- 第 613 页的『sqltempspace_threshold_value -“SQL 临时空间阈值”监视元素』
- 第 613 页的『sqltempspace_threshold_violated -“违反 SQL 临时空间阈值”监视元素』
- 第 624 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”』
- 第 627 页的『stmt_text -“SQL 语句文本”监视元素』
- 第 634 页的『tab_file_id -“表文件标识”监视元素』
- 第 635 页的『tab_type -“表类型”监视元素』
- 第 635 页的『table_file_id -“表文件标识”监视元素』
- 第 635 页的『table_name -“表名”监视元素』
- 第 636 页的『table_scans -“表扫描次数”监视元素』
- 第 637 页的『table_schema -“表模式名”监视元素』
- 第 638 页的『table_type -“表类型”监视元素』
- 第 638 页的『tablespace_auto_resize_enabled -“允许自动调整表空间大小”监视元素』
- 第 639 页的『tablespace_content_type -“表空间内容类型”监视元素』
- 第 639 页的『tablespace_cur_pool_id -“当前使用的缓冲池”监视元素』
- 第 640 页的『tablespace_extent_size -“表空间扩展数据块大小”监视元素』
- 第 640 页的『tablespace_free_pages -“表空间中的空闲页数”监视元素』
- 第 641 页的『tablespace_id -“表空间标识”监视元素』
- 第 643 页的『tablespace_name -“表空间名称”监视元素』
- 第 644 页的『tablespace_next_pool_id -“下次启动时使用的缓冲池”监视元素』
- 第 645 页的『tablespace_page_size -“表空间页大小”监视元素』
- 第 646 页的『tablespace_page_top -“表空间高水位标记”监视元素』

- 第 646 页的『tablespace_paths_dropped -“表空间正在使用已删除的路径”监视元素』
- 第 646 页的『tablespace_pending_free_pages -“表空间中的暂挂可用页数”监视元素』
- 第 647 页的『tablespace_prefetch_size -“表空间预取大小”监视元素』
- 第 648 页的『tablespace_rebalancer_mode -“重新平衡程序方式”监视元素』
- 第 650 页的『tablespace_state -“表空间状态”监视元素』
- 第 652 页的『tablespace_total_pages -“表空间中的总页数”监视元素』
- 第 653 页的『tablespace_type -“表空间类型”监视元素』
- 第 653 页的『tablespace_usable_pages -“表空间中的可用页数”监视元素』
- 第 654 页的『tablespace_used_pages -“表空间中的已使用页数”监视元素』
- 第 654 页的『tablespace_using_auto_storage -“已对表空间启用自动存储器”监视元素』
- 第 654 页的『tbsp_max_page_top -“最大表空间页号高水位标记”监视元素』
- 第 655 页的『tcpip_recv_volume -“TCP/IP 接收量”监视元素』
- 第 656 页的『tcpip_recv_wait_time -“TCP/IP 接收等待时间”监视元素』
- 第 656 页的『tcpip_recvs_total -“TCP/IP 接收总次数”监视元素』
- 第 657 页的『tcpip_send_volume -“TCP/IP 发送量”监视元素』
- 第 658 页的『tcpip_send_wait_time -“TCP/IP 发送等待时间”监视元素』
- 第 659 页的『tcpip_sends_total -“TCP/IP 发送总次数”监视元素』
- 第 665 页的『total_act_time -“活动时间总计”监视元素』
- 第 665 页的『total_act_wait_time -“活动等待时间总计”监视元素』
- 第 666 页的『total_app_rqst_time -“应用程序请求时间总计”监视元素』
- 第 668 页的『total_cpu_time -“CPU 时间总计”监视元素』
- 第 669 页的『total_move_time -“扩展数据块移动时间总计”监视元素』
- 第 671 页的『total_rqst_mapped_in -“映入请求总数”监视元素』
- 第 672 页的『total_rqst_mapped_out -“映出请求总数”监视元素』
- 第 672 页的『total_rqst_time -“请求时间总计”监视元素』
- 第 673 页的『total_section_sorts -“节排序总次数”监视元素』
- 第 674 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
- 第 675 页的『total_section_sort_time -“节排序时间总计”监视元素』
- 第 677 页的『total_sorts -“排序总数”监视元素』
- 第 680 页的『total_wait_time -“等待时间总计”监视元素』
- 第 686 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』
- 第 688 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』
- 第 690 页的『uow_id -“工作单元标识”监视元素』
- 第 694 页的『utility_id -“实用程序标识”』
- 第 696 页的『valid -“节有效性指示器”监视元素』
- 第 696 页的『vectored_ios -“向量 I/O 请求数”监视元素』
- 第 697 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』
- 第 698 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』
- 第 700 页的『workload_id -“工作负载标识”监视元素』

- 第 701 页的『workload_name －“工作负载名称”监视元素』
- 第 702 页的『workload_occurrence_id －“工作负载项标识”监视元素』
- 第 702 页的『workload_occurrence_state －“工作负载实例状态”监视元素』

第 8 章 请求监视元素

使用请求监视元素来监视数据库系统，尤其是数据服务器为了处理应用程序请求而完成的工作量。

请求是对数据库代理程序发出的伪指令，用于执行某些需要耗用数据库资源的工作。请求的来源包括:

- 由外部应用程序直接发出的伪指令，例如 OPEN 或 EXECUTE 伪指令。这些请求被称为“应用程序请求”。
- 协调代理程序向同一个或另一个数据库成员上的子代理程序发出的伪指令。
- 由另一个数据库成员上的代理程序发出的伪指令。

请求监视元素用于度量数据库服务器处理不同类型的请求时完成的工作量，其中包括整体系统处理、与特定类型的处理相关的请求以及与特定数据服务器环境相关的请求。

用于度量整体系统处理信息的一些典型监视元素如下所示:

- **rqsts_completed_total** 监视元素用于度量系统已完成的请求数。
- **total_rqst_time** 监视元素用于度量数据服务器中的请求所耗用的时间，其中包括等待时间和处理时间。
- **total_wait_time** 监视元素用于度量整体等待时间。
- **total_cpu_time** 监视元素用于度量 CPU 使用时间。

用于度量客户机/服务器处理信息的一些典型监视元素如下所示:

- **client_idle_wait_time** 监视元素用于度量等待下一个来自已打开连接的请求时耗用的时间。
- **tcpip_recv_volume** 监视元素用于度量数据服务器通过 TCP/IP 从客户机接收的数据量。

用于度量常用数据服务器处理操作的一些典型监视元素如下所示:

- **pool_data_l_reads** 是其中一个用于提供缓冲池资源使用情况信息的监视元素。
- **pool_read_time** 是其中一个用于提供 I/O 处理信息的监视元素。
- **lock_wait_time** 是其中一个用于提供锁定信息的监视元素。
- **total_section_sorts** 是其中一个用于提供排序信息的监视元素。

用于监视与所选类型的数据服务器环境相关的处理的一些典型监视元素如下所示:

- **fcm_recv_wait_time** 是其中一个用于测量快速通信管理器（FCM）处理的监视元素。
- **wlm_queue_time_total** 是其中一个用于测量工作负载管理控制操作的监视元素。

使用表函数来访问请求度量值

可以使用下列表函数来访问请求度量值:

- MON_GET_SERVICE_SUBCLASS 和 MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_WORKLOAD 和 MON_GET_WORKLOAD_DETAILS

- MON_GET_CONNECTION 和 MON_GET_CONNECTION_DETAILS
- MON_GET_UNIT_OF_WORK 和 MON_GET_UNIT_OF_WORK_DETAILS

这组监视表函数中的每个表函数都有两种格式，其中一种格式的名称以“DETAILS”结尾。未以“DETAILS”结尾的函数提供了用于返回最常用数据的 SQL 关系接口。另一个函数以基于 XML 的方式来访问监视数据并返回一组更详尽的数据。

这组表函数使您能够侧重于特定聚集级别的请求度量值。您可以选择表函数，以便侧重于您在给定情况下关注的部分系统工作负载（或者系统工作负载的聚集）。所有这些表函数都包括一组公共的请求度量值监视元素。每个表函数都可以返回几项并非所有表函数都返回的附加详细信息。

在不存在用户定义的工作负载或服务类的数据库中，数据库管理器执行的所有用户工作都在缺省用户工作负载和用户服务类中发生。对每个服务类（或工作负载）都返回数据的表函数将返回单一服务类（或工作负载）的数据，该服务类（或工作负载）代表整个数据库的用户工作负载的处理。

在存在用户定义的工作负载和服务类的数据库中，对每个服务类（或工作负载）都返回数据的表函数使您能够对每个服务类（或工作负载）的处理进行比较。通过使用 SQL，可以对所有服务类（或工作负载）的值进行求和以获取一个监视元素的值，该监视元素代表整个数据库的用户工作负载的处理。

使用事件监视器来访问请求度量值

请求度量值由下列事件监视器报告:

- 统计信息事件监视器 – 请求度量值是此事件监视器所报告的多种信息的其中一种。
- UoW 事件监视器 – 此事件监视器所报告的字段与 MON_GET_UNIT_OF_WORK 表函数所报告的字段类似或等同。

第 9 章 活动监视元素

活动监视元素是请求监视元素的子集。您可以使用活动度量值来监视与执行活动（尤其是为了执行 SQL 语句节而完成的处理）相关的数据服务器处理子集。

请求监视元素用于监视数据服务器为了处理应用程序请求而执行的全部工作。活动监视元素用于监视为了执行 SQL 语句节而完成的工作，其中包括锁定、排序和行处理。

要访问活动监视元素的当前值，请使用下列表函数:

MON_GET_ACTIVITY_DETAILS

返回关于进行中的一项或多项活动的详细信息。请在输入参数中指定您所关注的活动。返回的数据包括活动度量值监视元素、许多其他监视元素以及语句文本。数据以 XML 格式返回。

MON_GET_PKG_CACHE_STMT

返回数据库程序包高速缓存中某些或全部 SQL 语句节的详细信息，这些语句既包括静态 SQL 语句也包括动态 SQL 语句。返回的数据包括针对该节被添加到程序包高速缓存后全部各次执行进行聚集的活动度量值监视元素。数据以关系格式返回。

使用活动事件监视器来访问关于活动的历史数据。此监视器将捕获有关每个活动的每次执行的数据。此活动事件监视器与 MON_GET_ACTIVITY_DETAILS 表函数捕获相同的活动监视元素。它还捕获一些附加信息。

第 10 章 数据对象监视元素

数据对象监视元素提供关于对特定数据对象（其中包括表、索引、缓冲池、表空间和容器）执行的操作的信息。

每种数据对象类型都有一组可以监视的监视元素。例如，缓冲池的元素可用于计算缓冲池命中率。

请使用下列表函数来访问数据对象监视元素的当前值。这些监视器表函数以关系格式返回数据:

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER
- MON_GET_TABLE
- MON_GET_INDEX

第 11 章 工作单元事件监视器所报告的监视元素

工作单元事件监视器将报告下列监视元素。

第 12 章 锁定事件监视器所报告的监视元素

锁定事件监视器将报告下列监视元素。

- 第 620 页的『stmt_invocation_id -“语句调用标识”监视元素』
- 第 621 页的『stmt_lock_timeout -“语句锁定超时”监视元素』
- 第 622 页的『stmt_nest_level -“语句嵌套级别”监视元素』
- 第 623 页的『stmt_operation/operation -“语句操作”监视元素』
- 第 624 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”』
- 第 625 页的『stmt_query_id -“语句查询标识”监视元素』
- 第 625 页的『stmt_source_id -“语句源标识”』
- 第 627 页的『stmt_text -“SQL 语句文本”监视元素』
- 第 628 页的『stmt_type -“语句类型”监视元素』
- 第 629 页的『stmt_value_data -“值数据”』
- 第 630 页的『stmt_value_index -“值索引”』
- 第 630 页的『stmt_value_isnull -“包含空值”监视元素』
- 第 631 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』
- 第 631 页的『stmt_value_type -“值类型”监视元素』
- 第 635 页的『table_name -“表名”监视元素』
- 第 637 页的『table_schema -“表模式名”监视元素』
- 第 643 页的『tablespace_name -“表空间名称”监视元素』
- 第 690 页的『uow_id -“工作单元标识”监视元素』
- 第 700 页的『workload_id -“工作负载标识”监视元素』
- 第 701 页的『workload_name -“工作负载名称”监视元素』

第 13 章 等待时间监视元素

确定性能下降或瓶颈位置的一种有效方法是，确定请求在请求处理期间是否以及在什么位置耗费时间等待。等待时间监视元素用于度量等待特定实体或资源时耗用的时间。

例如，性能下降问题的一种常见原因是，请求在等待特定锁定时耗用时间过多。要确定延迟性质，您可以检查 **lock_wait_time** 监视元素的值。

每个等待时间监视元素还属于请求监视元素集、活动监视元素集或数据对象监视元素集。

表 70. 等待时间监视元素

类别	子类别	等待时间监视元素
高级别等待时间	DB2 处理（其中包括执行活动）期间的等待时间	第 680 页的『total_wait_time -“等待时间总计”监视元素』
	仅执行活动期间的等待时间	第 665 页的『total_act_wait_time -“活动等待时间总计”监视元素』
I/O	缓冲池 I/O	第 533 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』 第 546 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
	用于 LOB 的直接 I/O	第 375 页的『direct_read_time -“直接读时间”监视元素』 第 380 页的『direct_write_time -“直接写时间”监视元素』
	事务日志记录 I/O	第 461 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』
	诊断消息日志记录	第 372 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
锁定	-	第 456 页的『lock_wait_time -“等待锁定时间”监视元素』
连接	代理程序等待时间	第 291 页的『agent_wait_time -“代理程序等待时间”监视元素』
审计	-	第 310 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』 第 312 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』

表 70. 等待时间监视元素 (续)

类别	子类别	等待时间监视元素
FCM	整体 FCM（请求消息和表队列数据）	第 399 页的『fcm_send_wait_time - "FCM 发送等待时间"监视元素』 第 396 页的『fcm_recv_wait_time - "FCM 接收等待时间"监视元素』
	与请求消息相关的 FCM	第 394 页的『fcm_message_send_wait_time -"发送 FCM 消息等待时间"监视元素』 第 392 页的『fcm_message_recv_wait_time -"接收 FCM 消息等待时间"监视元素』
	与表队列相关的 FCM	第 405 页的『fcm_tq_send_wait_time -"FCM 表队列发送等待时间"监视元素』 第 402 页的『fcm_tq_recv_wait_time -"FCM 表队列接收等待时间"监视元素』
工作负载管理器控制操作	由于超出并行阈值而进行的排队	第 698 页的『wlm_queue_time_total -"工作负载管理器队列时间总计"监视元素』
客户机/服务器处理	等待来自活动连接的下一个请求时耗用的时间	第 328 页的『client_idle_wait_time -"客户机空闲等待时间"监视元素』
	远程客户机网络（TCPIP）	第 656 页的『tcpip_recv_wait_time - "TCP/IP 接收等待时间"监视元素』 第 658 页的『tcpip_send_wait_time - "TCP/IP 发送等待时间"监视元素』
	本地客户机通信（IPC）	第 436 页的『ipc_recv_wait_time -"进程间通信接收等待时间"监视元素』 第 438 页的『ipc_send_wait_time -"进程间通信发送等待时间"监视元素』

可以通过下列接口来获得等待时间元素:

- MON_GET_SERVICE_SUBCLASS 表函数
- MON_GET_SERVICE_SUBCLASS_DETAILS 表函数
- MON_GET_WORKLOAD 表函数
- MON_GET_WORKLOAD_DETAILS 表函数
- MON_GET_CONNECTION 表函数
- MON_GET_CONNECTION_DETAILS 表函数
- MON_GET_UNIT_OF_WORK 表函数
- MON_GET_UNIT_OF_WORK_DETAILS 表函数
- MON_GET_PKG_CACHE_STMT 表函数

- MON_GET_ACTIVITY_DETAILS 表函数
- 统计信息事件监视器（event_wlstats 和 event_scstats 逻辑组中的 DETAILS_XML 元素）
- 活动事件监视器（event_activity 逻辑组中的 DETAILS_XML 元素）
- 工作单元事件监视器

并非所有等待时间元素都通过所有接口报告。例如，**client_idle_wait_time** 监视元素仅适用于系统级接口，例如 MON_GET_SERVICE_SUBCLASS 表函数。数据对象监视器表函数不报告此元素。

请参阅每个监视元素的描述，以获取报告该元素的接口的列表。

第 14 章 逻辑数据组

快照监视器接口至逻辑数据组的映射

下表列示用来访问快照监视器数据的一些方法。所有快照监视器数据都存储在监视元素中，这些监视元素按逻辑数据组分类。每个 API 请求类型、CLP 命令和 SQL 管理视图仅从所有逻辑数据组的某个子集捕获监视器数据。

此表中列示的每个 API 请求类型、CLP 命令和 SQL 管理视图返回最右列中列示的逻辑数据组中的监视元素。

注:

1. 有一些 API 请求类型和 CLP 命令没有相应的 SQL 管理视图。对于其他 API 请求类型和 CLP 命令，各个 SQL 管理视图捕获关联逻辑数据组的子集。
2. 仅当关联监视开关设置为 ON 时，才返回某些监视元素。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 71. 快照监视器接口至逻辑数据组的映射

db2GetSnapshot API 请求类型	**CLP 命令**	**SQL 管理视图**	**逻辑数据组**
SQLMA_APPLINFO_ALL	list applications [show detail]	APPLICATIONS	appl_info
SQLMA_DBASE_APPLINFO	list applications for database *dbname* [show detail]	APPLICATIONS	appl_info
SQLMA_DCS_APPLINFO_ALL	list dcs applications [show detail]		dcs_appl_info
SQLMA_DB2	get snapshot for dbm	SNAPDBM	db2
		SNAPFCM	fcm
		SNAPFCMPART	fcm_node
		SNAPUTIL	utility_info
		SNAPUTIL_PROGRESS	progress 和 progress_info
		SNAPDBM_MEMORY_POOL	memory_pool
	get dbm monitor switches	SNAPSWITCHES	switch_list
SQLMA_DBASE	get snapshot for database on *dbname*	SNAPDB	dbase
		SNAPDETAILLOG	detail_log
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool

表 71. 快照监视器接口至逻辑数据组的映射 (续)

db2GetSnapshot API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_DBASE_ALL	get snapshot for all databases	SNAPDB	dbase
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
	list active databases		dbase
SQLMA_DCS_DBASE	get snapshot for dcs database on *dbname*		dcs_dbase 和 stmt_transmissions
SQLMA_DCS_DBASE_ALL	get snapshot for all dcs databases		dcs_dbase 和 stmt_transmissions
SQLMA_DBASE_REMOTE	get snapshot for remote database on *dbname*		dbase_remote
SQLMA_DBASE_REMOTE_ALL	get snapshot for all remote databases		dbase_remote
SQLMA_APPL	get snapshot for application applid *appl-id*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_AGENT_ID	get snapshot for application agentid *appl-handle*	SNAPAGENT	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DBASE_APPLS	get snapshot for applications on *dbname*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool

表 71. 快照监视器接口至逻辑数据组的映射 (续)

db2GetSnapshot API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_APPL_ALL	get snapshot for all applications	SNAPAPPL	appl
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTATEMENT	stmt
		SNAPAGENT	agent
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DCS_APPL	get snapshot for dcs application applid *appl-id*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_APPL_ALL	get snapshot for all dcs applications		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_APPL_HANDLE	get snapshot for dcs application agentid *appl-handle*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_DBASE_APPLS	get snapshot for dcs applications on *dbname*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DBASE_APPLS_REMOTE	get snapshot for remote applications on *dbname*		dbase_appl
SQLMA_APPL_REMOTE_ALL	get snapshot for all remote applications		dbase_appl
SQLMA_DBASE_TABLES	get snapshot for tables on *dbname*	SNAPTAB	表
		SNAPTAB_REORG	table_reorg
			table_list
SQLMA_APPL_LOCKS	get snapshot for locks for application applid *appl-id*	SNAPLOCK、SNAPAPPL 和 SNAPLOCKWAIT	appl_lock_list、lock_wait 和 lock
SQLMA_APPL_LOCKS_AGENT_ID	get snapshot for locks for application agentid *appl-handle*	SNAPLOCK、SNAPAPPL 和 SNAPLOCKWAIT	appl_lock_list、lock_wait 和 lock
SQLMA_DBASE_LOCKS	get snapshot for locks on *dbname*	SNAPLOCK	appl_lock_list 和 lock
		SNAPLOCK 和 SNAPLOCKWAIT	db_lock_list 和 lock_wait

表 71. 快照监视器接口至逻辑数据组的映射 (续)

db2GetSnapshot API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_DBASE_TABLESPACES	get snapshot for tablespaces on *dbname*	SNAPTBSP	tablespace
		SNAPTBSPPART	tablespace 和 tablespace_nodeinfo
		SNAPTBSP_QUIESCER	tablespace_quiescer 和 tablespace_nodeinfo
		SNAPCONTAINER	tablespace_container 和 tablespace_nodeinfo
		SNAPTBSP_RANGE	tablespace_ranges 和 tablespace_nodeinfo
			tablespace_list 和 tablespace_nodeinfo
SQLMA_BUFFERPOOLS_ALL	get snapshot for all bufferpools	SNAPBP	bufferpool
SQLMA_DBASE_BUFFERPOOLS	get snapshot for bufferpools on *dbname*	SNAPBP	bufferpool
SQLMA_DYNAMIC_SQL	get snapshot for dynamic sql on *dbname*	SNAPDYN_SQL	dynsql
			dynsql_list

下图显示逻辑数据分组在快照数据流中可能出现的顺序。

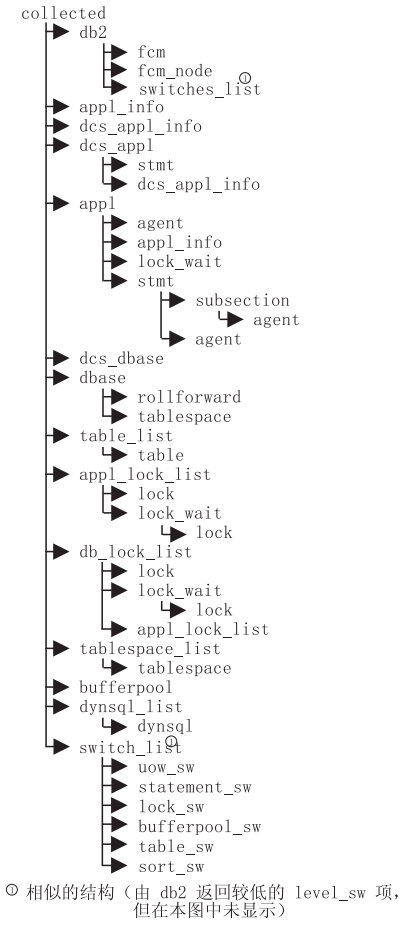


图 8. 数据流层次结构

注: 时间可能作为任何逻辑数据分组的一部分返回。

快照监视器逻辑数据组和监视元素

以下各部分列示快照监视可能返回的逻辑数据分组和监视元素。

- 第 224 页的『代理程序逻辑数据组』
- 第 224 页的『应用程序逻辑数据组』
- 第 227 页的『appl_id_info 逻辑数据组』
- 第 228 页的『appl_info 逻辑数据组』
- 第 228 页的『appl_lock_list 逻辑数据组』
- 第 229 页的『appl_remote 逻辑数据组』
- 第 229 页的『bufferpool 逻辑数据组』
- 第 231 页的『bufferpool_nodeinfo 逻辑数据组』
- 第 231 页的『collected 逻辑数据组』
- 第 231 页的『db2 逻辑数据组』
- 第 232 页的『db_lock_list 逻辑数据组』
- 第 232 页的『dbase 逻辑数据组』

- 第 236 页的『dbase_remote 逻辑数据组』
- 第 237 页的『db_storage_group 逻辑数据组』
- 第 237 页的『dcs_appl 逻辑数据组』
- 第 239 页的『dcs_appl_info 逻辑数据组』
- 第 240 页的『dcs_dbase 逻辑数据组』
- 第 242 页的『dcs_stmt 逻辑数据组』
- 第 242 页的『detail_log 逻辑数据组』
- 第 242 页的『dynsql 逻辑数据组』
- 第 243 页的『dynsql_list 逻辑数据组』
- 第 243 页的『fcm 逻辑数据组』
- 第 243 页的『fcm_node 逻辑数据组』
- 第 244 页的『hadr 逻辑数据组』
- 第 244 页的『lock 逻辑数据组』
- 第 245 页的『lock_wait 逻辑数据组』
- 第 245 页的『memory_pool 逻辑数据组』
- 第 245 页的『progress 逻辑数据组』
- 第 245 页的『progress_list 逻辑数据组』
- 第 245 页的『rollforward 逻辑数据组』
- 第 246 页的『stmt 逻辑数据组』
- 第 247 页的『stmt_transmissions 逻辑数据组』
- 第 248 页的『subsection 逻辑数据组』
- 第 249 页的『table 逻辑数据组』
- 第 249 页的『table_list 逻辑数据组』
- 第 249 页的『table_reorg 逻辑数据组』
- 第 250 页的『tablespace 逻辑数据组』
- 第 251 页的『tablespace_container 逻辑数据组』
- 第 251 页的『tablespace_list 逻辑数据组』
- 第 252 页的『tablespace_nodeinfo 逻辑数据组』
- 第 252 页的『tablespace_quiescer 逻辑数据组』
- 第 253 页的『tablespace_range 逻辑数据组』
- 第 253 页的『utility_info 逻辑数据组』

代理程序逻辑数据组

第 289 页的『agent_pid -“引擎可分派单元（EDU）标识”监视元素』

第 454 页的『lock_timeout_val -“锁定超时值”监视元素』

应用程序逻辑数据组

第 278 页的『acc_curs_blk -“接受的块游标请求数”』

第 290 页的『agent_sys_cpu_time -“代理程序使用的系统 CPU 时间”』

第 291 页的『agent_usr_cpu_time -“代理程序使用的用户 CPU 时间”』

第 295 页的『agents_stolen -“失窃代理程序数”』

第 299 页的『appl_con_time -“连接请求启动时间戳记”』
第 302 页的『appl_idle_time -“应用程序空闲时间”』
第 303 页的『appl_priority -“应用程序代理程序优先级”』
第 303 页的『appl_priority_type -“应用程序优先级类型”』
第 308 页的『associated_agents_top -“最大关联代理程序数”』
第 314 页的『authority_bitmap -“用户权限级别”监视元素』
第 315 页的『authority_lvl -“用户权限级别”监视元素』
第 316 页的『binds_precompiles -“尝试的绑定次数/预编译次数”』
第 322 页的『cat_cache_inserts -“目录高速缓存插入数”』
第 322 页的『cat_cache_lookups -“目录高速缓存查询数”』
第 323 页的『cat_cache_overflows -“目录高速缓存溢出数”』
第 333 页的『commit_sql_stmts -“尝试的落实语句数”』
第 342 页的『conn_complete_time -“连接请求完成时间戳记”』
第 367 页的『ddl_sql_stmts -“数据定义语言（DDL）SQL 语句数”』
第 369 页的『deadlocks -“检测到的死锁数”监视元素』
第 373 页的『direct_read_reqs -“直接读请求数”监视元素』
第 375 页的『direct_read_time -“直接读时间”监视元素』
第 376 页的『direct_reads -“直接读数据库数目”监视元素』
第 378 页的『direct_write_reqs -“直接写请求数”监视元素』
第 380 页的『direct_write_time -“直接写时间”监视元素』
第 381 页的『direct_writes -“直接写数据库数目”监视元素』
第 384 页的『dynamic_sql_stmts -“尝试的动态 SQL 语句数”』
第 390 页的『failed_sql_stmts -“失败的语句操作”』
第 422 页的『hash_join_overflows -“散列连接溢出数”』
第 423 页的『hash_join_small_overflows -“散列连接小溢出数”』
第 427 页的『inbound_comm_address -“入站通信地址”』
第 430 页的『int_auto_rebinds -“内部自动重新绑定次数”』
第 430 页的『int_commits -“内部落实数”』
第 431 页的『int_deadlock_rollbacks -“死锁导致的内部回滚数”』
第 432 页的『int_rollbacks -“内部回滚数”』
第 433 页的『int_rows_deleted -“删除的内部行数”』
第 434 页的『int_rows_inserted -“插入的内部行数”』
第 434 页的『int_rows_updated -“更新的内部行数”』
第 441 页的『last_reset -“最后复位时间戳记”』
第 446 页的『lock_escalation -“锁定升级”监视元素』
第 454 页的『lock_timeout_val -“锁定超时值”监视元素』
第 454 页的『lock_timeouts -“锁定超时次数”监视元素』
第 456 页的『lock_wait_time -“等待锁定时间”监视元素』
第 458 页的『lock_waits -“等待锁定次数”监视元素』
第 459 页的『locks_held -“挂起的锁定数”』

第 589 页的『rows_updated -“更新行数”监视元素』
第 589 页的『rows_written -“写入的行数”』
第 594 页的『select_sql_stmts -“执行的 Select SQL 语句数”』
第 601 页的『shr_workspace_num_overflows -“共享工作空间溢出数”』
第 602 页的『shr_workspace_section_inserts -“共享工作空间节插入数”』
第 603 页的『shr_workspace_section_lookups -“共享工作空间节查询数”』
第 603 页的『shr_workspace_size_top -“最大共享工作空间大小”』
第 605 页的『sort_overflows -“排序溢出数”监视元素』
第 609 页的『sql_reqs_since_commit -“上次落实后的 SQL 请求数”』
第 616 页的『static_sql_stmts -“尝试的静态 SQL 语句数”』
第 669 页的『total_hash_joins -“散列连接总数”』
第 670 页的『total_hash_loops -“总散列循环数”』
第 671 页的『total_olap_funcs -“OLAP 函数总数”监视元素』
第 676 页的『total_sort_time -“排序时间总计”监视元素』
第 677 页的『total_sorts -“排序总数”监视元素』
第 687 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』
第 688 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』
第 689 页的『uow_comp_status -“工作单元完成状态”』
第 689 页的『uow_elapsed_time -“最新工作单元耗用时间”』
第 690 页的『uow_lock_wait_time -“工作单元等待锁定的总时间”监视元素』
第 690 页的『uow_log_space_used -“使用的工作单元日志空间”』
第 691 页的『uow_start_time -“工作单元开始时间戳记”』
第 692 页的『uow_stop_time -“工作单元停止时间戳记”监视元素』
第 703 页的『x_lock_escals -“互斥锁定升级数”』
第 704 页的『xquery_stmts -“尝试的 XQuery 语句数”』

appl_id_info 逻辑数据组

第 288 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 299 页的『appl_id -“应用程序标识”』
第 302 页的『appl_name -“应用程序名称”监视元素』
第 305 页的『appl_status -“应用程序状态”』
第 313 页的『auth_id -“授权标识”』
第 327 页的『client_db_alias -“应用程序使用的数据库别名”』
第 329 页的『client_prdid -“客户机产品和版本标识”监视元素』
第 332 页的『codepage_id -“应用程序使用的代码页标识”』
第 363 页的『db_name -“数据库名称”』
第 363 页的『db_path -“数据库路径”』
第 428 页的『input_db_alias -“输入数据库别名”』
第 595 页的『sequence_no -“序号”监视元素』
第 618 页的『status_change_time -“应用程序状态更改时间”』

appl_info 逻辑数据组

第 288 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 299 页的『appl_id -“应用程序标识”』
第 302 页的『appl_name -“应用程序名称”监视元素』
第 304 页的『appl_section_inserts -“节插入数”监视元素』
第 304 页的『appl_section_lookups -“节查询数”』
第 305 页的『appl_status -“应用程序状态”』
第 313 页的『auth_id -“授权标识”』
第 314 页的『authority_bitmap -“用户权限级别”监视元素』
第 315 页的『authority_lvl -“用户权限级别”监视元素』
第 327 页的『client_db_alias -“应用程序使用的数据库别名”』
第 329 页的『client_pid -“客户机进程标识”』
第 329 页的『client_platform -“客户机操作平台”』
第 329 页的『client_prdid -“客户机产品和版本标识”监视元素』
第 330 页的『client_protocol -“客户机通信协议”』
第 332 页的『codepage_id -“应用程序使用的代码页标识”』
第 352 页的『coord_agent_pid -“协调代理程序标识”监视元素』
第 353 页的『coord_node -“协调节点”』
第 354 页的『corr_token -“DRDA 关联标记”』
第 363 页的『db_name -“数据库名称”』
第 363 页的『db_path -“数据库路径”』
第 390 页的『execution_id -“用户登录标识”』
第 428 页的『input_db_alias -“输入数据库别名”』
第 439 页的『is_system_appl -“是系统应用程序”监视元素』
第 484 页的『num_assoc_agents -“关联代理程序数”』
第 595 页的『sequence_no -“序号”监视元素』
第 601 页的『session_auth_id -“会话授权标识”监视元素』
第 618 页的『status_change_time -“应用程序状态更改时间”』
第 660 页的『territory_code -“数据库地域代码”』
第 681 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』
第 682 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』
第 682 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』
第 683 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』
第 700 页的『workload_id -“工作负载标识”监视元素』

appl_lock_list 逻辑数据组

第 288 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 299 页的『appl_id -“应用程序标识”』
第 302 页的『appl_name -“应用程序名称”监视元素』
第 305 页的『appl_status -“应用程序状态”』

第 313 页的『auth_id -“授权标识”』

第 327 页的『client_db_alias -“应用程序使用的数据库别名”』

第 332 页的『codepage_id -“应用程序使用的代码页标识”』

第 456 页的『lock_wait_time -“等待锁定时间”监视元素』

第 459 页的『locks_held -“挂起的锁定数”』

第 461 页的『locks_waiting -“等待锁定的当前代理程序数”』

第 595 页的『sequence_no -“序号”监视元素』

第 601 页的『session_auth_id -“会话授权标识”监视元素』

第 618 页的『status_change_time -“应用程序状态更改时间”』

appl_remote 逻辑数据组

第 333 页的『commit_sql_stmts -“尝试的落实语句数”』

第 357 页的『create_nickname -“创建昵称数”』

第 357 页的『create_nickname_time -“创建昵称响应时间”』

第 361 页的『datasource_name -“数据源名称”』

第 363 页的『db_name -“数据库名称”』

第 371 页的『delete_sql_stmts -“删除数”』

第 371 页的『delete_time -“删除响应时间”』

第 390 页的『failed_sql_stmts -“失败的语句操作”』

第 429 页的『insert_sql_stmts -“插入数”』

第 429 页的『insert_time -“插入响应时间”』

第 505 页的『passthru_time -“传递时间”』

第 505 页的『passthrus -“传递数”』

第 572 页的『remote_lock_time -“远程锁定时间”』

第 572 页的『remote_locks -“远程锁定”』

第 579 页的『rollback_sql_stmts -“尝试的回滚语句数”』

第 582 页的『rows_deleted -“删除行数”监视元素』

第 583 页的『rows_inserted -“插入行数”监视元素』

第 588 页的『rows_selected -“选择的行数”』

第 589 页的『rows_updated -“更新行数”监视元素』

第 594 页的『select_sql_stmts -“执行的 Select SQL 语句数”』

第 595 页的『select_time -“查询响应时间”』

第 607 页的『sp_rows_selected -“存储过程返回的行数”』

第 632 页的『stored_proc_time -“存储过程时间”』

第 632 页的『stored_procs -“存储过程”』

第 693 页的『update_sql_stmts -“更新数”』

第 693 页的『update_time -“更新响应时间”』

bufferpool 逻辑数据组

第 317 页的『block_ios -“块 I/O 请求数”监视元素』

第 319 页的『bp_id -“缓冲池标识”监视元素』

第 319 页的『bp_name －“缓冲池名称”监视元素』
第 363 页的『db_name －“数据库名称”』
第 363 页的『db_path －“数据库路径”』
第 373 页的『direct_read_reqs －“直接读请求数”监视元素』
第 375 页的『direct_read_time －“直接读时间”监视元素』
第 376 页的『direct_reads －“直接读数据库数目”监视元素』
第 378 页的『direct_write_reqs －“直接写请求数”监视元素』
第 380 页的『direct_write_time －“直接写时间”监视元素』
第 381 页的『direct_writes －“直接写数据库数目”监视元素』
第 407 页的『files_closed －“关闭数据库文件数”监视元素』
第 428 页的『input_db_alias －“输入数据库别名”』
第 500 页的『pages_from_block_ios －“块 I/O 读取总页数”监视元素』
第 501 页的『pages_from_vectored_ios －“向量 I/O 读取总页数”监视元素』
第 509 页的『pool_async_data_read_reqs －“缓冲池异步读请求数”监视元素』
第 510 页的『pool_async_data_reads －“缓冲池异步数据读次数”监视元素』
第 511 页的『pool_async_data_writes －“缓冲池异步数据写次数”监视元素』
第 511 页的『pool_async_index_read_reqs －“缓冲池异步索引读请求数”监视元素』
第 512 页的『pool_async_index_reads －“缓冲池异步索引读取数”监视元素』
第 513 页的『pool_async_index_writes －“缓冲池异步索引写次数”监视元素』
第 514 页的『pool_async_read_time －“缓冲池异步读取时间”』
第 514 页的『pool_async_write_time －“缓冲池异步写入时间”』
第 515 页的『pool_async_xda_read_reqs －“缓冲池异步 XDA 读请求数”监视元素』
第 515 页的『pool_async_xda_reads －“缓冲池异步 XDA 数据读取数”监视元素』
第 516 页的『pool_async_xda_writes －“缓冲池异步 XDA 数据写次数”监视元素』
第 518 页的『pool_data_l_reads －“缓冲池数据逻辑读取数”监视元素』
第 520 页的『pool_data_p_reads －“缓冲池数据物理读取数”监视元素』
第 521 页的『pool_data_writes －“缓冲池数据写次数”监视元素』
第 526 页的『pool_index_l_reads －“缓冲池索引逻辑读取数”监视元素』
第 528 页的『pool_index_p_reads －“缓冲池索引物理读取数”监视元素』
第 529 页的『pool_index_writes －“缓冲池索引写次数”监视元素』
第 532 页的『pool_no_victim_buffer －“缓冲池无牺牲缓冲区次数”监视元素』
第 533 页的『pool_read_time －“缓冲池物理读时间总计”监视元素』
第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』
第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』
第 539 页的『pool_temp_index_l_reads －“缓冲池临时索引逻辑读取数”监视元素』
第 540 页的『pool_temp_index_p_reads －“缓冲池临时索引物理读取数”监视元素』
第 542 页的『pool_temp_xda_l_reads －“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 544 页的『pool_temp_xda_p_reads －“缓冲池临时 XDA 数据物理读取数”监视元素』

第 546 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 547 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 549 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 551 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 696 页的『vectored_ios -“向量 I/O 请求数”监视元素』

bufferpool_nodeinfo 逻辑数据组

第 319 页的『bp_cur_buffsz -“缓冲池的当前大小”』

第 320 页的『bp_new_buffsz -“新的缓冲池大小”』

第 320 页的『bp_pages_left_to_remove -“要除去的余下页数”』

第 320 页的『bp_tbsp_use_count -“映射至缓冲池的表空间数”』

第 482 页的『node_number -“节点号”』

collected 逻辑数据组

第 482 页的『node_number -“节点号”』

第 596 页的『server_db2_type -“受监视的（服务器）节点上的数据库管理器类型”』

第 597 页的『server_instance_name -“服务器实例名称”』

第 597 页的『server_prdid -“服务器产品/版本标识”』

第 598 页的『server_version -“服务器版本”』

第 664 页的『time_stamp -“快照时间”』

第 664 页的『time_zone_disp -“时区偏移”』

db2 逻辑数据组

第 293 页的『agents_created_empty_pool -“由于空的代理程序池而创建的代理程序数”』

第 294 页的『agents_from_pool -“从池中分配的代理程序数”』

第 294 页的『agents_registered -“已注册的代理程序数”』

第 294 页的『agents_registered_top -“已注册的最大代理程序数”』

第 295 页的『agents_stolen -“失窃代理程序数”』

第 296 页的『agents_waiting_on_token -“正在等待令牌的代理程序数”』

第 296 页的『agents_waiting_top -“正在等待的最大代理程序数”监视元素』

第 333 页的『comm_private_mem -“已落实的专用内存”』

第 335 页的『con_local_dbases -“带有当前连接的本地数据库”』

第 353 页的『coord_agents_top -“最大协调代理程序数”』

第 361 页的『db2start_time -“启动数据库管理器时间戳记”』

第 364 页的『db_status -“数据库状态”』

第 412 页的『gw_cons_wait_client -“等待客户机发送请求的连接数”』

第 412 页的『gw_cons_wait_host -“等待主机应答的连接数”』

第 412 页的『gw_cur_cons -“DB2 Connect 的当前连接数”』

第 413 页的『gw_total_cons -“对 DB2 Connect 尝试连接的总数”』

第 426 页的『idle_agents -“空闲代理程序数”』

第 441 页的『last_reset -“最后复位时间戳记”』

第 443 页的『local_cons -“本地连接数”』

第 443 页的『local_cons_in_exec -“数据库管理器中正在执行的本地连接数”』

第 467 页的『max_agent_overflows -“最大代理程序溢出次数”』

第 486 页的『num_gw_conn_switches -“连接交换次数”』

第 489 页的『num_nodes_in_db2_instance -“分区中的节点数”』

第 505 页的『piped_sorts_accepted -“接受的管道排序数”』

第 506 页的『piped_sorts_requested -“请求的管道排序数”』

第 554 页的『post_threshold_hash_joins -“散列连接阈值”』

第 555 页的『post_threshold_olap_funcs -“OLAP 函数阈值”监视元素』

第 555 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』

第 561 页的『product_name -“产品名称”』

第 571 页的『rem_cons_in -“与数据库管理器的远程连接数”』

第 571 页的『rem_cons_in_exec -“数据库管理器中正在执行的远程连接数”』

第 599 页的『service_level -“服务级别”』

第 604 页的『smallest_log_avail_node -“带有最少可用日志空间的节点”』

第 604 页的『sort_heap_allocated -“分配的总排序堆”』

第 605 页的『sort_heap_top -“排序专用堆高水位标记”』

db_lock_list 逻辑数据组

第 308 页的『appls_cur_cons -“当前连接的应用程序数”』

第 363 页的『db_name -“数据库名称”』

第 363 页的『db_path -“数据库路径”』

第 428 页的『input_db_alias -“输入数据库别名”』

第 459 页的『locks_held -“挂起的锁定数”』

第 461 页的『locks_waiting -“等待锁定的当前代理程序数”』

dbase 逻辑数据组

第 283 页的『active_hash_joins -“活动散列连接数”』

第 283 页的『active_olap_funcs -“活动 OLAP 函数”监视元素』

第 284 页的『active_sorts -“活动排序次数”』

第 295 页的『agents_top -“创建的代理程序数”』

第 301 页的『appl_id_oldest_xact -“带有最旧事务的应用程序”』

第 304 页的『appl_section_inserts -“节插入数”监视元素』

第 304 页的『appl_section_lookups -“节查询数”』

第 308 页的『appls_cur_cons -“当前连接的应用程序数”』

第 308 页的『appls_in_db2 -“数据库中当前执行的应用程序数”』

第 309 页的『async_runstats -“异步 RUNSTATS 请求总数”监视元素』

第 316 页的『binds_precompiles -“尝试的绑定次数/预编译次数”』

第 318 页的『blocks_pending_cleanup -“暂挂清除已转出块”监视元素』

第 322 页的『cat_cache_inserts -“目录高速缓存插入数”』

第 322 页的『cat_cache_lookups -“目录高速缓存查询数”』

第 323 页的『cat_cache_overflows －“目录高速缓存溢出数”』
第 324 页的『cat_cache_size_top －“目录高速缓存高水位标记”监视元素』
第 324 页的『catalog_node －“目录节点号”』
第 325 页的『catalog_node_name －“目录节点网络名”』
第 333 页的『commit_sql_stmts －“尝试的落实语句数”』
第 344 页的『connections_top －“最大并行连接数”』
第 353 页的『coord_agents_top －“最大协调代理程序数”』
第 362 页的『db_conn_time －“数据库激活时间戳记”监视元素』
第 362 页的『db_heap_top －“分配的最大数据库堆”』
第 362 页的『db_location －“数据库位置”』
第 363 页的『db_name －“数据库名称”』
第 363 页的『db_path －“数据库路径”』
第 364 页的『db_status －“数据库状态”』
第 367 页的『ddl_sql_stmts －“数据定义语言（DDL）SQL 语句数”』
第 369 页的『deadlocks －“检测到的死锁数”监视元素』
第 373 页的『direct_read_reqs －“直接读请求数”监视元素』
第 375 页的『direct_read_time －“直接读时间”监视元素』
第 376 页的『direct_reads －“直接读数据库数目”监视元素』
第 378 页的『direct_write_reqs －“直接写请求数”监视元素』
第 380 页的『direct_write_time －“直接写时间”监视元素』
第 381 页的『direct_writes －“直接写数据库数目”监视元素』
第 384 页的『dynamic_sql_stmts －“尝试的动态 SQL 语句数”』
第 390 页的『failed_sql_stmts －“失败的语句操作”』
第 407 页的『files_closed －“关闭数据库文件数”监视元素』
第 422 页的『hash_join_overflows －“散列连接溢出数”』
第 423 页的『hash_join_small_overflows －“散列连接小溢出数”』
第 428 页的『input_db_alias －“输入数据库别名”』
第 430 页的『int_auto_rebinds －“内部自动重新绑定次数”』
第 430 页的『int_commits －“内部落实数”』
第 431 页的『int_deadlock_rollbacks －“死锁导致的内部回滚数”』
第 432 页的『int_rollbacks －“内部回滚数”』
第 433 页的『int_rows_deleted －“删除的内部行数”』
第 434 页的『int_rows_inserted －“插入的内部行数”』
第 434 页的『int_rows_updated －“更新的内部行数”』
第 440 页的『last_backup －“上次备份时间戳记”』
第 441 页的『last_reset －“最后复位时间戳记”』
第 446 页的『lock_escals －“锁定升级次数”监视元素』
第 448 页的『lock_list_in_use －“使用中的锁定列表内存总量”』
第 454 页的『lock_timeouts －“锁定超时次数”监视元素』
第 456 页的『lock_wait_time －“等待锁定时间”监视元素』

第 529 页的『pool_index_writes －“缓冲池索引写次数”监视元素』
第 531 页的『pool_lsn_gap_clns －“触发缓冲池日志空间清除程序次数”监视元素』
第 532 页的『pool_no_victim_buffer －“缓冲池无牺牲缓冲区次数”监视元素』
第 533 页的『pool_read_time －“缓冲池物理读时间总计”监视元素』
第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』
第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』
第 539 页的『pool_temp_index_l_reads －“缓冲池临时索引逻辑读取数”监视元素』
第 540 页的『pool_temp_index_p_reads －“缓冲池临时索引物理读取数”监视元素』
第 542 页的『pool_temp_xda_l_reads －“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 544 页的『pool_temp_xda_p_reads －“缓冲池临时 XDA 数据物理读取数”监视元素』
第 546 页的『pool_write_time －“缓冲池物理写时间总计”监视元素』
第 547 页的『pool_xda_l_reads －“缓冲池 XDA 数据逻辑读取数”监视元素』
第 549 页的『pool_xda_p_reads －“缓冲池 XDA 数据物理读取数”监视元素』
第 551 页的『pool_xda_writes －“缓冲池 XDA 数据写次数”监视元素』
第 553 页的『post_shrthreshold_hash_joins －“阈值后散列连接数”』
第 553 页的『post_shrthreshold_sorts －“共享阈值后排序数”监视元素』
第 558 页的『priv_workspace_num_overflows －“专用工作空间溢出数”』
第 559 页的『priv_workspace_section_inserts －“专用工作空间节插入数”』
第 559 页的『priv_workspace_section_lookups －“专用工作空间节查询数”』
第 560 页的『priv_workspace_size_top －“最大专用工作空间大小”』
第 579 页的『rollback_sql_stmts －“尝试的回滚语句数”』
第 582 页的『rows_deleted －“删除行数”监视元素』
第 583 页的『rows_inserted －“插入行数”监视元素』
第 585 页的『rows_read －“读取行数”监视元素』
第 588 页的『rows_selected －“选择的行数”』
第 589 页的『rows_updated －“更新行数”监视元素』
第 592 页的『sec_log_used_top －“使用的最大辅助日志空间”』
第 592 页的『sec_logs_allocated －“当前分配的辅助日志数”』
第 594 页的『select_sql_stmts －“执行的 Select SQL 语句数”』
第 597 页的『server_platform －“服务器操作系统”』
第 601 页的『shr_workspace_num_overflows －“共享工作空间溢出数”』
第 602 页的『shr_workspace_section_inserts －“共享工作空间节插入数”』
第 603 页的『shr_workspace_section_lookups －“共享工作空间节查询数”』
第 603 页的『shr_workspace_size_top －“最大共享工作空间大小”』
第 604 页的『sort_heap_allocated －“分配的总排序堆”』
第 605 页的『sort_overflows －“排序溢出数”监视元素』
第 606 页的『sort_shrheap_allocated －“对当前分配的共享堆进行排序”』
第 607 页的『sort_shrheap_top －“排序共享堆高水位标记”』

第 616 页的『static_sql_stmts －“尝试的静态 SQL 语句数”』
第 617 页的『stats_cache_size －“统计信息高速缓存大小”监视元素』
第 617 页的『stats_fabricate_time －“产生统计信息的活动所花的总时间”监视元素』
第 618 页的『stats_fabrications －“产生统计信息的次数”监视元素』
第 633 页的『sync_runstats －“同步 RUNSTATS 活动总数”监视元素』
第 633 页的『sync_runstats_time －“同步 RUNSTATS 活动所花的总时间”监视元素』
第 665 页的『tot_log_used_top －“使用的最大总日志空间”』
第 667 页的『total_cons －“数据库激活以后的连接数”』
第 669 页的『total_hash_joins －“散列连接总数”』
第 670 页的『total_hash_loops －“总散列循环数”』
第 670 页的『total_log_available －“可用的总日志量”』
第 671 页的『total_log_used －“使用的总日志空间”』
第 671 页的『total_olap_funcs －“OLAP 函数总数”监视元素』
第 673 页的『total_sec_cons －“辅助连接数”』
第 676 页的『total_sort_time －“排序时间总计”监视元素』
第 677 页的『total_sorts －“排序总数”监视元素』
第 687 页的『uid_sql_stmts －“执行的 Update/Insert/Delete SQL 语句数”』
第 688 页的『unread_prefetch_pages －“未读取的预取页数”监视元素』
第 703 页的『x_lock_escals －“互斥锁定升级数”』
第 704 页的『xquery_stmts －“尝试的 XQuery 语句数”』

dbase_remote 逻辑数据组

第 333 页的『commit_sql_stmts －“尝试的落实语句数”』
第 357 页的『create_nickname －“创建昵称数”』
第 357 页的『create_nickname_time －“创建昵称响应时间”』
第 361 页的『datasource_name －“数据源名称”』
第 363 页的『db_name －“数据库名称”』
第 371 页的『delete_sql_stmts －“删除数”』
第 371 页的『delete_time －“删除响应时间”』
第 383 页的『disconnects －“断开连接次数”』
第 390 页的『failed_sql_stmts －“失败的语句操作”』
第 429 页的『insert_sql_stmts －“插入数”』
第 429 页的『insert_time －“插入响应时间”』
第 505 页的『passthru_time －“传递时间”』
第 505 页的『passthrus －“传递数”』
第 572 页的『remote_lock_time －“远程锁定时间”』
第 572 页的『remote_locks －“远程锁定”』
第 579 页的『rollback_sql_stmts －“尝试的回滚语句数”』
第 582 页的『rows_deleted －“删除行数”监视元素』
第 583 页的『rows_inserted －“插入行数”监视元素』

第 588 页的『rows_selected －“选择的行数”』

第 589 页的『rows_updated －“更新行数”监视元素』

第 594 页的『select_sql_stmts －“执行的 Select SQL 语句数”』

第 595 页的『select_time －“查询响应时间”』

第 607 页的『sp_rows_selected －“存储过程返回的行数”』

第 632 页的『stored_proc_time －“存储过程时间”』

第 632 页的『stored_procs －“存储过程”』

第 667 页的『total_cons －“数据库激活以后的连接数”』

第 693 页的『update_sql_stmts －“更新数”』

第 693 页的『update_time －“更新响应时间”』

db_storage_group 逻辑数据组

第 364 页的『db_storage_path －“自动存储器路径”监视元素』

第 409 页的『fs_id －“唯一文件系统标识号”监视元素』

第 409 页的『fs_total_size －“文件系统总大小”监视元素』

第 410 页的『fs_type －“文件系统类型”』

第 410 页的『fs_used_size －“文件系统中的已用空间量”监视元素』

第 482 页的『node_number －“节点号”』

第 631 页的『sto_path_free_sz －“自动存储器路径可用空间量”』

dcs_appl 逻辑数据组

第 302 页的『appl_idle_time －“应用程序空闲时间”』

第 333 页的『commit_sql_stmts －“尝试的落实语句数”』

第 386 页的『elapsed_exec_time －“语句执行耗用时间”』

第 390 页的『failed_sql_stmts －“失败的语句操作”』

第 411 页的『gw_con_time －“DB2 Connect 网关首次启动的连接”』

第 413 页的『gw_exec_time －“DB2 Connect 网关处理所耗用的时间”』

第 425 页的『host_response_time －“主机响应时间”』

第 426 页的『inbound_bytes_received －“接收的入站字节数”』

第 427 页的『inbound_bytes_sent －“发送的入站字节数”』

第 441 页的『last_reset －“最后复位时间戳记”』

第 467 页的『max_data_received_1024 －“接收的出站字节数在 513 到 1024 字节之间的语句数”』

第 468 页的『max_data_received_128 －“接收的出站字节数在 1 到 128 字节之间的语句数”』

第 468 页的『max_data_received_16384 －“接收的出站字节数在 8193 到 16384 字节之间的语句数”』

第 469 页的『max_data_received_2048 －“接收的出站字节数在 1025 到 2048 字节之间的语句数”』

第 469 页的『max_data_received_256 －“接收的出站字节数在 129 到 256 字节之间的语句数”』

第 470 页的『max_data_received_31999 －“接收的出站字节数在 16385 到 31999 字节之间的语句数”监视元素』

第 470 页的『max_data_received_4096 －“接收的出站字节数在 2049 到 4096 字节之间的语句数”』

第 470 页的『max_data_received_512 －“接收的出站字节数在 257 到 512 字节之间的语句数”』

第 471 页的『max_data_received_64000 －“接收的出站字节数在 32000 到 64000 字节之间的语句数”监视元素』

第 471 页的『max_data_received_8192 －“接收的出站字节数在 4097 到 8192 字节之间的语句”』

第 472 页的『max_data_received_gt64000 －“接收的出站字节数高于 64000 的语句数”』

第 472 页的『max_data_sent_1024 －“发送的出站字节数在 513 到 1024 字节之间的语句数”』

第 472 页的『max_data_sent_128 －“发送的出站字节数在 1 到 128 字节之间的语句数”』

第 473 页的『max_data_sent_16384 －“发送的出站字节数在 8193 到 16384 字节之间的语句数”』

第 473 页的『max_data_sent_2048 －“发送的出站字节数在 1025 到 2048 字节之间的语句数”』

第 474 页的『max_data_sent_256 －“发送的出站字节数在 129 到 256 字节之间的语句数”』

第 474 页的『max_data_sent_31999 －“发送的出站字节数在 16385 到 31999 字节之间的语句数”』

第 475 页的『max_data_sent_4096 －“发送的出站字节数在 2049 到 4096 字节之间的语句数”』

第 475 页的『max_data_sent_512 －“发送的出站字节数在 257 到 512 字节之间的语句数”』

第 476 页的『max_data_sent_64000 －“发送的出站字节数在 32000 到 64000 字节之间的语句数”』

第 476 页的『max_data_sent_8192 －“发送的出站字节数在 4097 到 8192 字节之间的语句数”』

第 476 页的『max_data_sent_gt64000 －“发送的出站字节数高于 64000 的语句数”』

第 477 页的『max_network_time_100_ms －“网络时间在 16 到 100 毫秒之间的语句数”』

第 477 页的『max_network_time_16_ms －“网络时间在 4 到 16 毫秒之间的语句数”』

第 478 页的『max_network_time_1_ms －“网络时间最多为 1 毫秒的语句数”』

第 478 页的『max_network_time_4_ms －“网络时间在 1 到 4 毫秒之间的语句数”』

第 478 页的『max_network_time_500_ms －“网络时间在 100 到 500 毫秒之间的语句数”』

第 479 页的『max_network_time_gt500_ms －“网络时间大于 500 毫秒的语句数”』

第 481 页的『network_time_bottom －“语句的最短网络时间”』

第 482 页的『network_time_top -“语句的最长网络时间”』
第 492 页的『open_cursors -“打开的游标数”』
第 494 页的『outbound_bytes_received -“接收的出站字节数”』
第 495 页的『outbound_bytes_sent -“发送的出站字节数”』
第 558 页的『prev_uow_stop_time -“上一个工作单元完成时间戳记”』
第 579 页的『rollback_sql_stmts -“尝试的回滚语句数”』
第 588 页的『rows_selected -“选择的行数”』
第 609 页的『sql_stmts -“尝试的 SQL 语句数”』
第 681 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』
第 682 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』
第 682 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』
第 683 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』
第 689 页的『uow_comp_status -“工作单元完成状态”』
第 689 页的『uow_elapsed_time -“最新工作单元耗用时间”』
第 691 页的『uow_start_time -“工作单元开始时间戳记”』
第 692 页的『uow_stop_time -“工作单元停止时间戳记”监视元素』
第 704 页的『xid -“事务标识”』

dcs_appl_info 逻辑数据组

第 288 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 290 页的『agent_status -“DCS 应用程序代理程序数”』
第 299 页的『appl_id -“应用程序标识”』
第 302 页的『appl_name -“应用程序名称”监视元素』
第 313 页的『auth_id -“授权标识”』
第 329 页的『client_pid -“客户机进程标识”』
第 329 页的『client_platform -“客户机操作平台”』
第 329 页的『client_prdid -“客户机产品和版本标识”监视元素』
第 330 页的『client_protocol -“客户机通信协议”』
第 332 页的『codepage_id -“应用程序使用的代码页标识”』
第 366 页的『dcs_appl_status -“DCS 应用程序状态”』
第 367 页的『dcs_db_name -“DCS 数据库名称”』
第 390 页的『execution_id -“用户登录标识”』
第 413 页的『gw_db_alias -“网关上的数据库别名”』
第 424 页的『host_ccsid -“主机编码字符集标识”』
第 425 页的『host_db_name -“主机数据库名称”』
第 425 页的『host_prdid -“主机产品/版本标识”』
第 427 页的『inbound_comm_address -“入站通信地址”』
第 494 页的『outbound_appl_id -“出站应用程序标识”』
第 496 页的『outbound_comm_address -“出站通信地址”』
第 496 页的『outbound_comm_protocol -“出站通信协议”』

第 497 页的『outbound_sequence_no -“出站序号”』

第 595 页的『sequence_no -“序号”监视元素』

第 618 页的『status_change_time -“应用程序状态更改时间”』

dcs_dbase 逻辑数据组

第 333 页的『commit_sql_stmts -“尝试的落实语句数”』

第 334 页的『con_elapsed_time -“最新连接耗用时间”』

第 335 页的『con_response_time -“连接的最新响应时间”』

第 367 页的『dcs_db_name -“DCS 数据库名称”』

第 386 页的『elapsed_exec_time -“语句执行耗用时间”』

第 390 页的『failed_sql_stmts -“失败的语句操作”』

第 410 页的『gw_comm_error_time -“通信错误时间”』

第 411 页的『gw_comm_errors -“通信错误”』

第 411 页的『gw_con_time -“DB2 Connect 网关首次启动的连接”』

第 411 页的『gw_connections_top -“与主机数据库的最大并行连接数”』

第 412 页的『gw_cons_wait_client -“等待客户机发送请求的连接数”』

第 412 页的『gw_cons_wait_host -“等待主机应答的连接数”』

第 412 页的『gw_cur_cons -“DB2 Connect 的当前连接数”』

第 413 页的『gw_total_cons -“对 DB2 Connect 尝试连接的总数”』

第 425 页的『host_db_name -“主机数据库名称”』

第 425 页的『host_response_time -“主机响应时间”』

第 426 页的『inbound_bytes_received -“接收的入站字节数”』

第 441 页的『last_reset -“最后复位时间戳记”』

第 467 页的『max_data_received_1024 -“接收的出站字节数在 513 到 1024 字节之间的语句数”』

第 468 页的『max_data_received_128 -“接收的出站字节数在 1 到 128 字节之间的语句数”』

第 468 页的『max_data_received_16384 -“接收的出站字节数在 8193 到 16384 字节之间的语句数”』

第 469 页的『max_data_received_2048 -“接收的出站字节数在 1025 到 2048 字节之间的语句数”』

第 469 页的『max_data_received_256 -“接收的出站字节数在 129 到 256 字节之间的语句数”』

第 470 页的『max_data_received_31999 -“接收的出站字节数在 16385 到 31999 字节之间的语句数”监视元素』

第 470 页的『max_data_received_4096 -“接收的出站字节数在 2049 到 4096 字节之间的语句数”』

第 470 页的『max_data_received_512 -“接收的出站字节数在 257 到 512 字节之间的语句数”』

第 471 页的『max_data_received_64000 -“接收的出站字节数在 32000 到 64000 字节之间的语句数”监视元素』

第 471 页的『max_data_received_8192 －“接收的出站字节数在 4097 到 8192 字节之间的语句”』

第 472 页的『max_data_received_gt64000 －“接收的出站字节数高于 64000 的语句数”』

第 472 页的『max_data_sent_1024 －“发送的出站字节数在 513 到 1024 字节之间的语句数”』

第 472 页的『max_data_sent_128 －“发送的出站字节数在 1 到 128 字节之间的语句数”』

第 473 页的『max_data_sent_16384 －“发送的出站字节数在 8193 到 16384 字节之间的语句数”』

第 473 页的『max_data_sent_2048 －“发送的出站字节数在 1025 到 2048 字节之间的语句数”』

第 474 页的『max_data_sent_256 －“发送的出站字节数在 129 到 256 字节之间的语句数”』

第 474 页的『max_data_sent_31999 －“发送的出站字节数在 16385 到 31999 字节之间的语句数”』

第 475 页的『max_data_sent_4096 －“发送的出站字节数在 2049 到 4096 字节之间的语句数”』

第 475 页的『max_data_sent_512 －“发送的出站字节数在 257 到 512 字节之间的语句数”』

第 476 页的『max_data_sent_64000 －“发送的出站字节数在 32000 到 64000 字节之间的语句数”』

第 476 页的『max_data_sent_8192 －“发送的出站字节数在 4097 到 8192 字节之间的语句数”』

第 476 页的『max_data_sent_gt64000 －“发送的出站字节数高于 64000 的语句数”』

第 477 页的『max_network_time_100_ms －“网络时间在 16 到 100 毫秒之间的语句数”』

第 477 页的『max_network_time_16_ms －“网络时间在 4 到 16 毫秒之间的语句数”』

第 478 页的『max_network_time_1_ms －“网络时间最多为 1 毫秒的语句数”』

第 478 页的『max_network_time_4_ms －“网络时间在 1 到 4 毫秒之间的语句数”』

第 478 页的『max_network_time_500_ms －“网络时间在 100 到 500 毫秒之间的语句数”』

第 479 页的『max_network_time_gt500_ms －“网络时间大于 500 毫秒的语句数”』

第 481 页的『network_time_bottom －“语句的最短网络时间”』

第 482 页的『network_time_top －“语句的最长网络时间”』

第 495 页的『outbound_bytes_sent －“发送的出站字节数”』

第 579 页的『rollback_sql_stmts －“尝试的回滚语句数”』

第 588 页的『rows_selected －“选择的行数”』

第 609 页的『sql_stmts －“尝试的 SQL 语句数”』

dcs_stmt 逻辑数据组

第 318 页的『blocking_cursor -“分块游标”』
第 358 页的『creator -“应用程序创建者”』
第 386 页的『elapsed_exec_time -“语句执行耗用时间”』
第 406 页的『fetch_count -“成功的访存数”』
第 413 页的『gw_exec_time -“DB2 Connect 网关处理所耗用的时间”』
第 425 页的『host_response_time -“主机响应时间”』
第 426 页的『inbound_bytes_received -“接收的入站字节数”』
第 427 页的『inbound_bytes_sent -“发送的入站字节数”』
第 490 页的『num_transmissions -“传输次数”』
第 490 页的『num_transmissions_group -“传输组数目”』
第 494 页的『outbound_bytes_received -“接收的出站字节数”』
第 495 页的『outbound_bytes_sent -“发送的出站字节数”』
第 498 页的『package_name -“程序包名”监视元素』
第 564 页的『query_card_estimate -“行查询数估计”』
第 565 页的『query_cost_estimate -“查询估计成本”监视元素』
第 593 页的『section_number -“节号”监视元素』
第 619 页的『stmt_elapsed_time -“最新语句耗用时间”』
第 623 页的『stmt_operation/operation -“语句操作”监视元素』
第 626 页的『stmt_start -“语句操作开始时间戳记”』
第 626 页的『stmt_stop -“语句操作停止时间戳记”』
第 627 页的『stmt_text -“SQL 语句文本”监视元素』

detail_log 逻辑数据组

第 358 页的『current_active_log -“当前活动日志文件编号”』
第 359 页的『current_archive_log -“当前归档日志文件编号”』
第 407 页的『first_active_log -“第一个活动日志文件编号”』
第 440 页的『last_active_log -“最后一个活动日志文件编号”』
第 482 页的『node_number -“节点号”』

dynsql 逻辑数据组

第 406 页的『fetch_count -“成功的访存数”』
第 429 页的『insert_timestamp -“语句插入时间戳记”监视元素』
第 433 页的『int_rows_deleted -“删除的内部行数”』
第 434 页的『int_rows_inserted -“插入的内部行数”』
第 434 页的『int_rows_updated -“更新的内部行数”』
第 484 页的『num_compilations -“语句编译次数”』
第 484 页的『num_executions -“语句执行次数”监视元素』
第 518 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 520 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 526 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 528 页的『pool_index_p_reads －“缓冲池索引物理读取数”监视元素』

第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』

第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』

第 539 页的『pool_temp_index_l_reads －“缓冲池临时索引逻辑读取数”监视元素』

第 540 页的『pool_temp_index_p_reads －“缓冲池临时索引物理读取数”监视元素』

第 542 页的『pool_temp_xda_l_reads －“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 544 页的『pool_temp_xda_p_reads －“缓冲池临时 XDA 数据物理读取数”监视元素』

第 547 页的『pool_xda_l_reads －“缓冲池 XDA 数据逻辑读取数”监视元素』

第 549 页的『pool_xda_p_reads －“缓冲池 XDA 数据物理读取数”监视元素』

第 557 页的『prep_time_best －“语句最短编译时间”监视元素』

第 557 页的『prep_time_worst －“语句最长编译时间”监视元素』

第 585 页的『rows_read －“读取行数”监视元素』

第 589 页的『rows_written －“写入的行数”』

第 605 页的『sort_overflows －“排序溢出数”监视元素』

第 617 页的『stats_fabricate_time －“产生统计信息的活动所花的总时间”监视元素』

第 624 页的『stmt_pkgcache_id －“语句程序包高速缓存标识”』

第 625 页的『stmt_sorts －“语句排序数”』

第 627 页的『stmt_text －“SQL 语句文本”监视元素』

第 633 页的『sync_runstats_time －“同步 RUNSTATS 活动所花的总时间”监视元素』

第 669 页的『total_exec_time －“执行语句所耗用的时间”』

第 676 页的『total_sort_time －“排序时间总计”监视元素』

第 678 页的『total_sys_cpu_time －“语句的系统 CPU 时间总计”监视元素』

第 680 页的『total_usr_cpu_time －“语句的用户 CPU 时间总计”监视元素』

dynsql_list 逻辑数据组

第 363 页的『db_name －“数据库名称”』

第 363 页的『db_path －“数据库路径”』

fcm 逻辑数据组

第 321 页的『buff_free －“当前可用的 FCM 缓冲区数”』

第 321 页的『buff_free_bottom －“最少可用 FCM 缓冲区数”』

第 325 页的『ch_free －“当前可用的通道数”』

第 325 页的『ch_free_bottom －“最低可用通道数”』

fcm_node 逻辑数据组

第 343 页的『connection_status －“连接状态”』

第 482 页的『node_number －“节点号”』

第 667 页的『total_buffers_rcvd －“接收到的 FCM 缓冲区总数”』

第 667 页的『total_buffers_sent －“发送的 FCM 缓冲区总数”』

hadr 逻辑数据组

lock 逻辑数据组

lock_wait 逻辑数据组

第 289 页的『agent_id_holding_lock -“挂起锁定的代理程序标识”』

第 301 页的『appl_id_holding_lk -“挂起锁定的应用程序标识”』

第 360 页的『data_partition_id -“数据分区标识”监视元素』

第 444 页的『lock_attributes -“锁定属性”监视元素』

第 445 页的『lock_current_mode -“转换前的原始锁定方式”』

第 446 页的『lock_escalation -“锁定升级”监视元素』

第 449 页的『lock_mode -“锁定方式”监视元素』

第 450 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 450 页的『lock_name -“锁定名称”监视元素』

第 452 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 452 页的『lock_release_flags -“锁定释放标志”监视元素』

第 456 页的『lock_wait_start_time -“锁定等待开始时间戳记”』

第 482 页的『node_number -“节点号”』

第 614 页的『ss_number -“子节号”』

第 635 页的『table_name -“表名”监视元素』

第 637 页的『table_schema -“表模式名”监视元素』

第 643 页的『tablespace_name -“表空间名称”监视元素』

memory_pool 逻辑数据组

第 482 页的『node_number -“节点号”』

第 517 页的『pool_config_size -“内存池的已配置大小”』

第 518 页的『pool_cur_size -“内存池的当前大小”』

第 525 页的『pool_id -“内存池标识”』

第 535 页的『pool_secondary_id -“内存池辅助标记”』

第 545 页的『pool_watermark -“内存池水位标记”』

progress 逻辑数据组

第 561 页的『progress_completed_units -“完成的进度工作单元数”』

第 561 页的『progress_description -“进度描述”』

第 562 页的『progress_seq_num -“进度序号”』

第 562 页的『progress_start_time -“进度开始时间”』

第 563 页的『progress_total_units -“进度工作单元总数”』

第 563 页的『progress_work_metric -“进度工作度量”』

progress_list 逻辑数据组

第 562 页的『progress_list_attr -“当前进度列表属性”』

第 562 页的『progress_list_cur_seq_num -“当前进度列表序号”』

rollforward 逻辑数据组

第 482 页的『node_number -“节点号”』

第 578 页的『rf_log_num -“正在前滚的日志”』

第 578 页的『rf_status -“日志阶段”』

第 579 页的『rf_timestamp -“前滚时间戳记”』

第 579 页的『rf_type -“前滚类型“』

第 687 页的『ts_name -“正在前滚的表空间”监视元素』

stmt 逻辑数据组

第 295 页的『agents_top -“创建的代理程序数”』

第 318 页的『blocking_cursor -“分块游标”』

第 344 页的『consistency_token -“程序包一致性标记”监视元素』

第 358 页的『creator -“应用程序创建者”』

第 359 页的『cursor_name -“游标名称”』

第 370 页的『degree_parallelism -“并行度”』

第 406 页的『fetch_count -“成功的访存数”』

第 433 页的『int_rows_deleted -“删除的内部行数”』

第 434 页的『int_rows_inserted -“插入的内部行数”』

第 434 页的『int_rows_updated -“更新的内部行数”』

第 483 页的『num_agents -“正在处理语句的代理程序数”』

第 498 页的『package_name -“程序包名”监视元素』

第 499 页的『package_version_id -“程序包版本”监视元素』

第 518 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 520 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 526 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 528 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 535 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 537 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 539 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 540 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 542 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 544 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 547 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 549 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 564 页的『query_card_estimate -“行查询数估计”』

第 565 页的『query_cost_estimate -“查询估计成本”监视元素』

第 585 页的『rows_read -“读取行数”监视元素』

第 589 页的『rows_written -“写入的行数”』

第 593 页的『section_number -“节号”监视元素』

第 605 页的『sort_overflows -“排序溢出数”监视元素』

第 619 页的『stmt_elapsed_time -“最新语句耗用时间”』

第 622 页的『stmt_node_number -“语句节点”』

第 623 页的『stmt_operation/operation -“语句操作”监视元素』

第 625 页的『stmt_sorts -“语句排序数”』

第 626 页的『stmt_start -“语句操作开始时间戳记”』

第 626 页的『stmt_stop -“语句操作停止时间戳记”』

第 627 页的『stmt_sys_cpu_time -“语句使用的系统 CPU 时间”』

第 627 页的『stmt_text -“SQL 语句文本”监视元素』

第 628 页的『stmt_type -“语句类型”监视元素』

第 629 页的『stmt_usr_cpu_time -“语句使用的用户 CPU 时间”』

第 676 页的『total_sort_time -“排序时间总计”监视元素』

stmt_transmissions 逻辑数据组

第 386 页的『elapsed_exec_time -“语句执行耗用时间”』

第 425 页的『host_response_time -“主机响应时间”』

第 467 页的『max_data_received_1024 -“接收的出站字节数在 513 到 1024 字节之间的语句数”』

第 468 页的『max_data_received_128 -“接收的出站字节数在 1 到 128 字节之间的语句数”』

第 468 页的『max_data_received_16384 -“接收的出站字节数在 8193 到 16384 字节之间的语句数”』

第 469 页的『max_data_received_2048 -“接收的出站字节数在 1025 到 2048 字节之间的语句数”』

第 469 页的『max_data_received_256 -“接收的出站字节数在 129 到 256 字节之间的语句数”』

第 470 页的『max_data_received_31999 -“接收的出站字节数在 16385 到 31999 字节之间的语句数”监视元素』

第 470 页的『max_data_received_4096 -“接收的出站字节数在 2049 到 4096 字节之间的语句数”』

第 470 页的『max_data_received_512 -“接收的出站字节数在 257 到 512 字节之间的语句数”』

第 471 页的『max_data_received_64000 -“接收的出站字节数在 32000 到 64000 字节之间的语句数”监视元素』

第 471 页的『max_data_received_8192 -“接收的出站字节数在 4097 到 8192 字节之间的语句”』

第 472 页的『max_data_received_gt64000 -“接收的出站字节数高于 64000 的语句数”』

第 472 页的『max_data_sent_1024 -“发送的出站字节数在 513 到 1024 字节之间的语句数”』

第 472 页的『max_data_sent_128 -“发送的出站字节数在 1 到 128 字节之间的语句数”』

第 473 页的『max_data_sent_16384 -“发送的出站字节数在 8193 到 16384 字节之间的语句数”』

第 473 页的『max_data_sent_2048 -“发送的出站字节数在 1025 到 2048 字节之间的语句数”』

第 474 页的『max_data_sent_256 －“发送的出站字节数在 129 到 256 字节之间的语句数”』

第 474 页的『max_data_sent_31999 －“发送的出站字节数在 16385 到 31999 字节之间的语句数”』

第 475 页的『max_data_sent_4096 －“发送的出站字节数在 2049 到 4096 字节之间的语句数”』

第 475 页的『max_data_sent_512 －“发送的出站字节数在 257 到 512 字节之间的语句数”』

第 476 页的『max_data_sent_64000 －“发送的出站字节数在 32000 到 64000 字节之间的语句数”』

第 476 页的『max_data_sent_8192 －“发送的出站字节数在 4097 到 8192 字节之间的语句数”』

第 476 页的『max_data_sent_gt64000 －“发送的出站字节数高于 64000 的语句数”』

第 477 页的『max_network_time_100_ms －“网络时间在 16 到 100 毫秒之间的语句数”』

第 477 页的『max_network_time_16_ms －“网络时间在 4 到 16 毫秒之间的语句数”』

第 478 页的『max_network_time_1_ms －“网络时间最多为 1 毫秒的语句数”』

第 478 页的『max_network_time_4_ms －“网络时间在 1 到 4 毫秒之间的语句数”』

第 478 页的『max_network_time_500_ms －“网络时间在 100 到 500 毫秒之间的语句数”』

第 479 页的『max_network_time_gt500_ms －“网络时间大于 500 毫秒的语句数”』

第 481 页的『network_time_bottom －“语句的最短网络时间”』

第 482 页的『network_time_top －“语句的最长网络时间”』

第 494 页的『outbound_bytes_received －“接收的出站字节数”』

第 495 页的『outbound_bytes_received_bottom －“接收的最小出站字节数”』

第 495 页的『outbound_bytes_received_top －“接收的最大出站字节数”』

第 495 页的『outbound_bytes_sent －“发送的出站字节数”』

第 496 页的『outbound_bytes_sent_bottom －“发送的最小出站字节数”』

第 496 页的『outbound_bytes_sent_top －“发送的最大出站字节数”』

第 608 页的『sql_chains －“尝试的 SQL 链数”』

第 609 页的『sql_stmts －“尝试的 SQL 语句数”』

subsection 逻辑数据组

第 585 页的『rows_read －“读取行数”监视元素』

第 589 页的『rows_written －“写入的行数”』

第 613 页的『ss_exec_time －“子节执行耗用时间”』

第 614 页的『ss_node_number －“子节节点号”』

第 614 页的『ss_number －“子节号”』

第 614 页的『ss_status －“子节状态”』

第 615 页的『ss_sys_cpu_time －“子节使用的系统 CPU 时间”』

第 615 页的『ss_usr_cpu_time －“子节使用的用户 CPU 时间”』

第 683 页的『tq_cur_send_spills －“当前溢出的表队列缓冲区数”监视元素』

第 684 页的『tq_id_waiting_on －“在表队列的节点上等待”』

第 684 页的『tq_max_send_spills －“最大表队列缓冲区溢出数”』

第 684 页的『tq_node_waited_for －“在表队列上等待节点”』

第 685 页的『tq_rows_read －“从表队列读取的行数”』

第 685 页的『tq_rows_written －“写至表队列的行数”』

第 686 页的『tq_tot_send_spills －“溢出表队列缓冲区总数”监视元素』

第 687 页的『tq_wait_for_any －“在表队列上等待发送任何节点”』

table 逻辑数据组

第 360 页的『data_object_pages －“数据对象页数”』

第 360 页的『data_partition_id －“数据分区标识”监视元素』

第 427 页的『index_object_pages －“索引对象页数”』

第 442 页的『lob_object_pages －“LOB 对象页数”』

第 466 页的『long_object_pages －“长对象页数”』

第 497 页的『overflow_accesses －“访问溢出记录次数”监视元素』

第 500 页的『page_reorgs －“页重组”』

第 585 页的『rows_read －“读取行数”监视元素』

第 589 页的『rows_written －“写入的行数”』

第 635 页的『table_file_id －“表文件标识”监视元素』

第 635 页的『table_name －“表名”监视元素』

第 637 页的『table_schema －“表模式名”监视元素』

第 638 页的『table_type －“表类型”监视元素』

第 641 页的『tablespace_id －“表空间标识”监视元素』

第 704 页的『xda_object_pages －“XDA 对象页数”』

table_list 逻辑数据组

第 362 页的『db_conn_time －“数据库激活时间戳记”监视元素』

第 363 页的『db_name －“数据库名称”』

第 363 页的『db_path －“数据库路径”』

第 428 页的『input_db_alias －“输入数据库别名”』

第 441 页的『last_reset －“最后复位时间戳记”』

table_reorg 逻辑数据组

第 360 页的『data_partition_id －“数据分区标识”监视元素』

第 572 页的『reorg_completion －“重组完成标志”』

第 573 页的『reorg_current_counter －“重组进度”』

第 573 页的『reorg_end －“表重组结束时间”』

第 573 页的『reorg_index_id －“用于重组表的索引”』

第 574 页的『reorg_max_counter －“重组总量”』

第 574 页的『reorg_max_phase －“最大重组阶段”』

第 574 页的『reorg_phase -“表重组阶段”监视元素』

第 575 页的『reorg_phase_start -“重组阶段开始时间”』

第 575 页的『reorg_rows_compressed -“压缩行数”』

第 575 页的『reorg_rows_rejected_for_compression -“拒绝压缩行数”』

第 575 页的『reorg_start -“表重组开始时间”』

第 576 页的『reorg_status -“表重组状态”』

第 576 页的『reorg_tbspc_id -“用来重组表或数据分区的表空间”』

第 576 页的『reorg_type -“表重组属性”』

第 577 页的『reorg_xml_regions_compressed -“已压缩的 XML 区域数”监视元素』

第 577 页的『reorg_xml_regions_rejected_for_compression -“拒绝压缩的 XML 区域数”监视元素』

tablespace 逻辑数据组

第 373 页的『direct_read_reqs -“直接读请求数”监视元素』

第 375 页的『direct_read_time -“直接读时间”监视元素』

第 376 页的『direct_reads -“直接读数据库数目”监视元素』

第 378 页的『direct_write_reqs -“直接写请求数”监视元素』

第 380 页的『direct_write_time -“直接写时间”监视元素』

第 381 页的『direct_writes -“直接写数据库数目”监视元素』

第 407 页的『files_closed -“关闭数据库文件数”监视元素』

第 408 页的『fs_caching -“文件系统高速缓存”监视元素』

第 509 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』

第 510 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』

第 511 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』

第 511 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』

第 512 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』

第 513 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』

第 514 页的『pool_async_read_time -“缓冲池异步读取时间”』

第 514 页的『pool_async_write_time -“缓冲池异步写入时间”』

第 515 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』

第 515 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』

第 516 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』

第 518 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 520 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 521 页的『pool_data_writes -“缓冲池数据写次数”监视元素』

第 526 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 528 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 529 页的『pool_index_writes -“缓冲池索引写次数”监视元素』

第 532 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』

第 533 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』

第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』

第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』

第 539 页的『pool_temp_index_l_reads －“缓冲池临时索引逻辑读取数”监视元素』

第 540 页的『pool_temp_index_p_reads －“缓冲池临时索引物理读取数”监视元素』

第 542 页的『pool_temp_xda_l_reads －“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 544 页的『pool_temp_xda_p_reads －“缓冲池临时 XDA 数据物理读取数”监视元素』

第 546 页的『pool_write_time －“缓冲池物理写时间总计”监视元素』

第 547 页的『pool_xda_l_reads －“缓冲池 XDA 数据逻辑读取数”监视元素』

第 549 页的『pool_xda_p_reads －“缓冲池 XDA 数据物理读取数”监视元素』

第 551 页的『pool_xda_writes －“缓冲池 XDA 数据写次数”监视元素』

第 638 页的『tablespace_auto_resize_enabled －“允许自动调整表空间大小”监视元素』

第 639 页的『tablespace_content_type －“表空间内容类型”监视元素』

第 639 页的『tablespace_cur_pool_id －“当前使用的缓冲池”监视元素』

第 640 页的『tablespace_extent_size －“表空间扩展数据块大小”监视元素』

第 641 页的『tablespace_id －“表空间标识”监视元素』

第 643 页的『tablespace_name －“表空间名称”监视元素』

第 644 页的『tablespace_next_pool_id －“下次启动时使用的缓冲池”监视元素』

第 645 页的『tablespace_page_size －“表空间页大小”监视元素』

第 647 页的『tablespace_prefetch_size －“表空间预取大小”监视元素』

第 648 页的『tablespace_rebalancer_mode －“重新平衡程序方式”监视元素』

第 653 页的『tablespace_type －“表空间类型”监视元素』

第 654 页的『tablespace_using_auto_storage －“已对表空间启用自动存储器”监视元素』

tablespace_container 逻辑数据组

第 344 页的『container_accessible －“容器可访问”监视元素』

第 345 页的『container_id －“容器标识”监视元素』

第 345 页的『container_name －“容器名称”监视元素』

第 346 页的『container_stripe_set －“容器分割集”监视元素』

第 346 页的『container_total_pages －“容器中的总页数”监视元素』

第 346 页的『container_type －“容器类型”监视元素』

第 347 页的『container_usable_pages －“容器中的可用页数”监视元素』

tablespace_list 逻辑数据组

第 362 页的『db_conn_time －“数据库激活时间戳记”监视元素』

第 363 页的『db_name －“数据库名称”』

第 363 页的『db_path －“数据库路径”』

第 428 页的『input_db_alias －“输入数据库别名”』

第 441 页的『last_reset －“最后复位时间戳记”』

tablespace_nodeinfo 逻辑数据组

tablespace_quiescer 逻辑数据组

tablespace_range 逻辑数据组

第 568 页的『range_adjustment －“范围调整”』

第 568 页的『range_container_id －“范围容器”』

第 568 页的『range_end_stripe －“结束分割区”』

第 569 页的『range_max_extent －“范围中的最大扩展数据块”』

第 569 页的『range_max_page_number －“范围中的最大页”』

第 569 页的『range_num_containers －“范围中的容器数”』

第 569 页的『range_number －“范围编号”』

第 569 页的『range_offset －“范围偏移”』

第 570 页的『range_start_stripe －“起始分割区”』

第 570 页的『range_stripe_set_number －“分割集编号”』

utility_info 逻辑数据组

第 482 页的『node_number －“节点号”』

第 694 页的『utility_dbname －“实用程序操作的数据库”』

第 694 页的『utility_description －“实用程序描述”』

第 694 页的『utility_id －“实用程序标识”』

第 695 页的『utility_invoker_type －“实用程序调用者类型”』

第 695 页的『utility_priority －“实用程序优先级”』

第 695 页的『utility_start_time －“实用程序启动时间”』

第 695 页的『utility_state －“实用程序状态”』

第 696 页的『utility_type －“实用程序类型”』

事件类型至逻辑数据组的映射

对于文件、管道和表事件监视器，事件监视器输出由一个有序的逻辑数据分组序列组成。不管事件监视器类型如何，输出记录总是包含相同的起始逻辑数据组。它们为逻辑数据组设置了框架，这些逻辑数据组的存在与否取决于事件监视器记录的事件类型。

对于文件和管道事件监视器，可能会对任何连接生成事件记录，并且事件记录可能会因此以混合顺序出现在流中。这意味着您可能获得连接 1 的事务事件，之后紧跟连接 2 的连接事件。但是，属于单个连接或单个事件的记录将以逻辑顺序出现。例如，语句记录（语句结尾）总是在事务记录（UOW 结尾）之前，如果有。同样，死锁事件总是在死锁中涉及的每个连接的死锁连接事件记录之前。**应用程序标识**或**应用程序句柄（agent_id）**可用来将记录与连接相匹配。

通常会对与数据库的每个连接写入连接头事件。对于带有详细信息的死锁事件监视器，仅当发生死锁时才将它们写入。在此情况下，将仅对死锁参与者（而不是与数据库的所有连接）写入连接头事件。

逻辑数据分组将按以下四个不同级别排序：监视器、序言、内容和结尾。下面详细描述每个级别，包括对应的事件类型和逻辑数据组。

监视器

将对所有事件监视器生成监视器级别的信息。它包含事件监视器元数据。

表 72. 事件监视器数据流：监视器部分

事件类型	逻辑数据组	可用信息
监视器级别	event_log_stream_header	标识事件监视器的版本级别和字节顺序。应用程序可使用此头来确定是否能够处理 evmon 输出流。

序言

在激活事件监视器时将生成序言信息。

表 73. 事件监视器数据流：序言部分

事件类型	逻辑数据组	可用信息
日志头	event_log_header	跟踪的特征，如服务器类型和内存布局。
数据库头	event_db_header	数据库名称、路径和激活时间。
事件监视器启动	event_start	启动或重新启动监视器的时间。
连接头	event_connheader	每个当前事件对应一个连接头，包括连接时间和应用程序名称。仅对连接、语句、事务和死锁事件监视器生成事件连接头。带有详细信息的死锁事件监视器仅在发生死锁时生成连接头。

内容

特定于事件监视器的指定事件类型的信息出现在内容部分。

表 74. 事件监视器数据流：内容部分

事件类型	逻辑数据组	可用信息
语句事件	event_stmt	语句级别数据，包括动态语句的文本。语句事件监视器不记录访存。
子节事件	event_subsection	子节级别数据。
事务事件[1]	event_xact	事务级别数据。
连接事件	event_conn	连接级别数据。
死锁事件	event_deadlock	死锁级别数据。
死锁的连接事件	event_dlconn	死锁涉及的每个连接对应一个死锁的连接事件，包括涉及的应用程序和处于争用状态的锁定。
带有详细信息的死锁的连接事件	event_detailed_dlconn，lock	死锁涉及的每个连接对应一个带有详细信息的死锁的连接事件，包括涉及的应用程序、处于争用状态的锁定、当前语句信息和应用程序争用挂起的其他锁定。

表 74. 事件监视器数据流：内容部分 (续)

事件类型	逻辑数据组	可用信息
溢出	event_overflow	丢失的记录的个数 – 在写程序不能与（非分块）事件监视器保持一致时生成。
带有详细信息的死锁的历史纪录[2]	event_stmt_history	列示死锁涉及的任何工作单元中执行的语句列表。
带有详细信息的死锁的历史记录值[2]	event_data_value	event_stmt_history 列表中的语句的参数标记。
活动	event_activity	系统上已完成执行的或在完成前捕获的活动列表。
	event_activitystmt	活动类型为语句时，有关活动正在执行的语句的信息。
	event_activityvals	用作每个 SQL 语句活动的输入变量的数据值。这些数据值不包括 LOB 数据、长数据或结构化类型数据。
统计信息	event_scstats	从在系统中每个服务类、工作类或工作负载内执行的活动计算出来的统计信息，以及从阈值队列计算出来的统计信息。
	event_wcstats	
	event_wlstats	
	event_qstats	
	event_histogrambin	
阈值违例	event_threshold_violations	标识阈值违例及违例时间的信息。

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

2 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

结尾

结尾信息将在数据库释放时（最后一个应用程序断开连接后）生成:

表 75. 事件监视器数据流：结尾部分

事件类型	逻辑数据组	可用信息
数据库事件	event_db	数据库管理器级别数据。
缓冲池事件	event_bufferpool	缓冲池级别数据。
表空间事件	event_tablespace	表空间级别数据。
表事件	event_table	表级别数据。

事件监视器逻辑数据组和监视元素

下表列示事件监视可能返回的逻辑数据分组和监视元素。

- 『event_activity 逻辑数据组』
- 第 258 页的『event_activitystmt 逻辑数据组』
- 第 259 页的『event_activityvals 逻辑数据组』
- 第 259 页的『event_bufferpool 逻辑数据组』
- 第 260 页的『event_conn 逻辑数据组』
- 第 262 页的『event_connheader 逻辑数据组』
- 第 262 页的『event_connmemuse 逻辑数据组』
- 第 262 页的『event_data_value 逻辑数据组』
- 第 263 页的『event_db 逻辑数据组』
- 第 266 页的『event_dbheader 逻辑数据组』
- 第 266 页的『event_dbmemuse 逻辑数据组』
- 第 266 页的『event_deadlock 逻辑数据组』
- 第 266 页的『event_detailed_dlconn 逻辑数据组』
- 第 267 页的『event_dlconn 逻辑数据组』
- 第 268 页的『event_histogrambin 逻辑数据组』
- 第 268 页的『event_log_header 逻辑数据组』
- 第 269 页的『event_overflow 逻辑数据组』
- 第 269 页的『event_qstats 逻辑数据组』
- 第 269 页的『event_scstats 逻辑数据组』
- 第 270 页的『event_start 逻辑数据组』
- 第 270 页的『event_stmt 逻辑数据组』
- 第 271 页的『event_stmt_history 逻辑数据组』
- 第 272 页的『event_subsection 逻辑数据组』
- 第 272 页的『event_table 逻辑数据组』
- 第 272 页的『event_tablespace 逻辑数据组』
- 第 273 页的『event_thresholdviolations 逻辑数据组』
- 第 274 页的『event_wlstats 逻辑数据组』
- 第 274 页的『event_wcstats 逻辑数据组』
- 第 275 页的『event_xact 逻辑数据组』
- 第 275 页的『lock 逻辑数据组』
- 第 276 页的『sqlca 逻辑数据组』

event_activity 逻辑数据组

第 280 页的『act_exec_time -“活动执行时间”监视元素』

第 283 页的『activate_timestamp -“激活时间戳记”监视元素』

第 284 页的『activity_id -“活动标识”监视元素』

第 285 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 286 页的『activity_type －“活动类型”监视元素』

第 287 页的『address － 从中发起连接的 IP 地址』

第 288 页的『agent_id －“应用程序句柄（代理程序标识）”监视元素』

第 299 页的『appl_id －“应用程序标识”』

第 302 页的『appl_name －“应用程序名称”监视元素』

第 308 页的『arm_correlator －“应用程序响应测量相关因子”监视元素』

第 353 页的『coord_partition_num －“协调程序分区号”监视元素』

第 365 页的『db_work_action_set_id －“数据库工作操作集标识”监视元素』

第 366 页的『db_work_class_id －“数据库工作类标识”监视元素』

details_xml（此 XML 文档包含 MON_GET_ACTIVITY_DETAILS 表函数输出的 DETAILS 列中报告的所有监视元素。）

第 502 页的『parent_activity_id －“父活动标识”监视元素』

第 502 页的『parent_uow_id －“父工作单元标识”监视元素』

第 503 页的『partial_record －“部分记录”监视元素』

第 518 页的『pool_data_l_reads －“缓冲池数据逻辑读取数”监视元素』

第 520 页的『pool_data_p_reads －“缓冲池数据物理读取数”监视元素』

第 489 页的『num_remaps －“重新映射次数”监视元素』

第 591 页的『sc_work_action_set_id －“服务类工作操作集标识”监视元素』

第 591 页的『sc_work_class_id －“服务类工作类标识”监视元素』

第 600 页的『service_subclass_name －“服务子类名”监视元素』

第 600 页的『service_superclass_name －“服务超类名”监视元素』

第 601 页的『session_auth_id －“会话授权标识”监视元素』

第 605 页的『sort_overflows －“排序溢出数”监视元素』

第 609 页的『sqlca －“SQL 通信区（SQLCA）”』

第 663 页的『time_completed －“完成时间”监视元素』

第 663 页的『time_created －“创建时间”监视元素』

第 664 页的『time_started －“开始时间”监视元素』

第 676 页的『total_sort_time －“排序时间总计”监视元素』

第 677 页的『total_sorts －“排序总数”监视元素』

第 681 页的『tpmon_acc_str －“TP 监视器客户机记帐字符串”监视元素』

第 682 页的『tpmon_client_app －“TP 监视器客户机应用程序名称”监视元素』

第 682 页的『tpmon_client_userid －“TP 监视器客户机用户标识”监视元素』

第 683 页的『tpmon_client_wkstn －“TP 监视器客户机工作站名称”监视元素』

第 690 页的『uow_id －“工作单元标识”监视元素』

第 700 页的『workload_id －“工作负载标识”监视元素』

第 702 页的『workload_occurrence_id －“工作负载项标识”监视元素』

第 526 页的『pool_index_l_reads －“缓冲池索引逻辑读取数”监视元素』

第 528 页的『pool_index_p_reads －“缓冲池索引物理读取数”监视元素』

第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』

第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』

第 539 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 540 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 542 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 544 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 547 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 549 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 557 页的『prep_time -“编译时间”监视元素』

第 564 页的『query_card_estimate -“行查询数估计”』

第 565 页的『query_cost_estimate -“查询估计成本”监视元素』

第 583 页的『rows_fetched -“访存的行数”监视元素』

第 584 页的『rows_modified -“修改的行数”监视元素』

第 586 页的『rows_returned -“返回的行数”监视元素』

第 634 页的『system_cpu_time -“系统 CPU 时间”』

第 694 页的『user_cpu_time -“用户 CPU 时间”』

event_activitystmt 逻辑数据组

第 283 页的『activate_timestamp -“激活时间戳记”监视元素』

第 284 页的『activity_id -“活动标识”监视元素』

第 285 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 299 页的『appl_id -“应用程序标识”』

第 334 页的『comp_env_desc -“编译环境”监视元素』

第 358 页的『creator -“应用程序创建者”』

第 384 页的『eff_stmt_text -“有效语句文本”监视元素』

第 389 页的『executable_id -“可执行文件标识”监视元素』

第 498 页的『package_name -“程序包名”监视元素』

第 499 页的『package_version_id -“程序包版本”监视元素』

第 592 页的『section_env -“节环境”监视元素』

第 593 页的『section_number -“节号”监视元素』

第 619 页的『stmt_first_use_time -“语句第一次使用时间”』

第 621 页的『stmt_isolation -“语句隔离”』

第 621 页的『stmt_last_use_time -“语句上一次使用时间”监视元素』

第 621 页的『stmt_lock_timeout -“语句锁定超时”监视元素』

第 622 页的『stmt_nest_level -“语句嵌套级别”监视元素』

第 624 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”』

第 625 页的『stmt_query_id -“语句查询标识”监视元素』

第 625 页的『stmt_source_id -“语句源标识”』

第 627 页的『stmt_text -“SQL 语句文本”监视元素』

第 628 页的『stmt_type -“语句类型”监视元素』

第 690 页的『uow_id -“工作单元标识”监视元素』

event_activityvals 逻辑数据组

第 283 页的『activate_timestamp -“激活时间戳记”监视元素』
第 284 页的『activity_id -“活动标识”监视元素』
第 285 页的『activity_secondary_id -“活动辅助标识”监视元素』
第 299 页的『appl_id -“应用程序标识”』
第 629 页的『stmt_value_data -“值数据”』
第 630 页的『stmt_value_index -“值索引”』
第 630 页的『stmt_value_isnull -“包含空值”监视元素』
第 631 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』
第 631 页的『stmt_value_type -“值类型”监视元素』
第 690 页的『uow_id -“工作单元标识”监视元素』

event_bufferpool 逻辑数据组

第 319 页的『bp_id -“缓冲池标识”监视元素』
第 319 页的『bp_name -“缓冲池名称”监视元素』
第 363 页的『db_name -“数据库名称”』
第 363 页的『db_path -“数据库路径”』
第 373 页的『direct_read_reqs -“直接读请求数”监视元素』
第 375 页的『direct_read_time -“直接读时间”监视元素』
第 376 页的『direct_reads -“直接读数据库数目”监视元素』
第 378 页的『direct_write_reqs -“直接写请求数”监视元素』
第 380 页的『direct_write_time -“直接写时间”监视元素』
第 381 页的『direct_writes -“直接写数据库数目”监视元素』
第 388 页的『event_time -“事件时间”』
第 388 页的『evmon_activates -“事件监视器激活数”』
第 390 页的『evmon_flushes -“事件监视器清空数”』
第 407 页的『files_closed -“关闭数据库文件数”监视元素』
第 503 页的『partial_record -“部分记录”监视元素』
第 509 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』
第 510 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』
第 511 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』
第 512 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』
第 513 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』
第 514 页的『pool_async_read_time -“缓冲池异步读取时间”』
第 514 页的『pool_async_write_time -“缓冲池异步写入时间”』
第 518 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 520 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 521 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 526 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 528 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

event_conn 逻辑数据组

第 454 页的『lock_timeouts －“锁定超时次数”监视元素』
第 456 页的『lock_wait_time －“等待锁定时间”监视元素』
第 458 页的『lock_waits －“等待锁定次数”监视元素』
第 491 页的『olap_func_overflows －“OLAP 函数溢出次数”监视元素』
第 503 页的『partial_record －“部分记录”监视元素』
第 434 页的『int_rows_updated －“更新的内部行数”』
第 506 页的『pkg_cache_inserts －“程序包高速缓存插入数”』
第 507 页的『pkg_cache_lookups －“程序包高速缓存查询数”』
第 518 页的『pool_data_l_reads －“缓冲池数据逻辑读取数”监视元素』
第 520 页的『pool_data_p_reads －“缓冲池数据物理读取数”监视元素』
第 521 页的『pool_data_writes －“缓冲池数据写次数”监视元素』
第 526 页的『pool_index_l_reads －“缓冲池索引逻辑读取数”监视元素』
第 528 页的『pool_index_p_reads －“缓冲池索引物理读取数”监视元素』
第 529 页的『pool_index_writes －“缓冲池索引写次数”监视元素』
第 533 页的『pool_read_time －“缓冲池物理读时间总计”监视元素』
第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』
第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』
第 539 页的『pool_temp_index_l_reads －“缓冲池临时索引逻辑读取数”监视元素』
第 540 页的『pool_temp_index_p_reads －“缓冲池临时索引物理读取数”监视元素』
第 546 页的『pool_write_time －“缓冲池物理写时间总计”监视元素』
第 556 页的『prefetch_wait_time －“等待预取的时间”监视元素』
第 558 页的『priv_workspace_num_overflows －“专用工作空间溢出数”』
第 559 页的『priv_workspace_section_inserts －“专用工作空间节插入数”』
第 559 页的『priv_workspace_section_lookups －“专用工作空间节查询数”』
第 560 页的『priv_workspace_size_top －“最大专用工作空间大小”』
第 570 页的『rej_curs_blk －“拒绝的块游标请求数”』
第 579 页的『rollback_sql_stmts －“尝试的回滚语句数”』
第 585 页的『rows_read －“读取行数”监视元素』
第 588 页的『rows_selected －“选择的行数”』
第 589 页的『rows_written －“写入的行数”』
第 594 页的『select_sql_stmts －“执行的 Select SQL 语句数”』
第 595 页的『sequence_no －“序号”监视元素』
第 601 页的『shr_workspace_num_overflows －“共享工作空间溢出数”』
第 602 页的『shr_workspace_section_inserts －“共享工作空间节插入数”』
第 603 页的『shr_workspace_section_lookups －“共享工作空间节查询数”』
第 603 页的『shr_workspace_size_top －“最大共享工作空间大小”』
第 605 页的『sort_overflows －“排序溢出数”监视元素』
第 616 页的『static_sql_stmts －“尝试的静态 SQL 语句数”』
第 634 页的『system_cpu_time －“系统 CPU 时间”』
第 669 页的『total_hash_joins －“散列连接总数”』

第 670 页的『total_hash_loops -“总散列循环数”』

第 671 页的『total_olap_funcs -“OLAP 函数总数”监视元素』

第 673 页的『total_sec_cons -“辅助连接数”』

第 676 页的『total_sort_time -“排序时间总计”监视元素』

第 677 页的『total_sorts -“排序总数”监视元素』

第 687 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』

第 688 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』

第 694 页的『user_cpu_time -“用户 CPU 时间”』

第 703 页的『x_lock_escals -“互斥锁定升级数”』

第 704 页的『xquery_stmts -“尝试的 XQuery 语句数”』

event_connheader 逻辑数据组

第 288 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 299 页的『appl_id -“应用程序标识”』

第 302 页的『appl_name -“应用程序名称”监视元素』

第 313 页的『auth_id -“授权标识”』

第 327 页的『client_db_alias -“应用程序使用的数据库别名”』

第 329 页的『client_pid -“客户机进程标识”』

第 329 页的『client_platform -“客户机操作平台”』

第 329 页的『client_prdid -“客户机产品和版本标识”监视元素』

第 330 页的『client_protocol -“客户机通信协议”』

第 332 页的『codepage_id -“应用程序使用的代码页标识”』

第 343 页的『conn_time -“数据库连接时间”监视元素』

第 354 页的『corr_token -“DRDA 关联标记”』

第 390 页的『execution_id -“用户登录标识”』

第 482 页的『node_number -“节点号”』

第 595 页的『sequence_no -“序号”监视元素』

第 660 页的『territory_code -“数据库地域代码”』

event_connmemuse 逻辑数据组

第 482 页的『node_number -“节点号”』

第 517 页的『pool_config_size -“内存池的已配置大小”』

第 518 页的『pool_cur_size -“内存池的当前大小”』

第 525 页的『pool_id -“内存池标识”』

第 535 页的『pool_secondary_id -“内存池辅助标记”』

第 545 页的『pool_watermark -“内存池水位标记”』

event_data_value 逻辑数据组

第 368 页的『deadlock_id -“死锁事件标识”』

第 368 页的『deadlock_node -“发生死锁的分区号”』

第 388 页的『evmon_activates -“事件监视器激活数”』

第 504 页的『participant_no -“死锁参与者”』

第 619 页的『stmt_history_id -“语句历史记录标识”』

第 629 页的『stmt_value_data -“值数据”』

第 630 页的『stmt_value_index -“值索引”』

第 630 页的『stmt_value_isnull -“包含空值”监视元素』

第 631 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 631 页的『stmt_value_type -“值类型”监视元素』

event_db 逻辑数据组

第 283 页的『active_hash_joins -“活动散列连接数”』

第 304 页的『appl_section_inserts -“节插入数”监视元素』

第 304 页的『appl_section_lookups -“节查询数”』

第 309 页的『async_runstats -“异步 RUNSTATS 请求总数”监视元素』

第 316 页的『binds_precompiles -“尝试的绑定次数/预编译次数”』

第 318 页的『blocks_pending_cleanup -“暂挂清除已转出块”监视元素』

第 322 页的『cat_cache_inserts -“目录高速缓存插入数”』

第 322 页的『cat_cache_lookups -“目录高速缓存查询数”』

第 323 页的『cat_cache_overflows -“目录高速缓存溢出数”』

第 324 页的『cat_cache_size_top -“目录高速缓存高水位标记”监视元素』

第 324 页的『catalog_node -“目录节点号”』

第 325 页的『catalog_node_name -“目录节点网络名”』

第 333 页的『commit_sql_stmts -“尝试的落实语句数”』

第 344 页的『connections_top -“最大并行连接数”』

第 362 页的『db_heap_top -“分配的最大数据库堆”』

第 367 页的『ddl_sql_stmts -“数据定义语言（DDL）SQL 语句数”』

第 369 页的『deadlocks -“检测到的死锁数”监视元素』

第 373 页的『direct_read_reqs -“直接读请求数”监视元素』

第 375 页的『direct_read_time -“直接读时间”监视元素』

第 376 页的『direct_reads -“直接读数据库数目”监视元素』

第 378 页的『direct_write_reqs -“直接写请求数”监视元素』

第 380 页的『direct_write_time -“直接写时间”监视元素』

第 381 页的『direct_writes -“直接写数据库数目”监视元素』

第 383 页的『disconn_time -“数据库释放时间戳记”』

第 384 页的『dynamic_sql_stmts -“尝试的动态 SQL 语句数”』

第 388 页的『evmon_activates -“事件监视器激活数”』

第 390 页的『evmon_flushes -“事件监视器清空数”』

第 390 页的『failed_sql_stmts -“失败的语句操作”』

第 407 页的『files_closed -“关闭数据库文件数”监视元素』

第 422 页的『hash_join_overflows -“散列连接溢出数”』

第 423 页的『hash_join_small_overflows -“散列连接小溢出数”』

第 529 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 531 页的『pool_lsn_gap_clns -“触发缓冲池日志空间清除程序次数”监视元素』
第 532 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』
第 533 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 535 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 537 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 539 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 540 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 546 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
第 553 页的『post_shrthreshold_hash_joins -“阈值后散列连接数”』
第 553 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』
第 556 页的『prefetch_wait_time -“等待预取的时间”监视元素』
第 558 页的『priv_workspace_num_overflows -“专用工作空间溢出数”』
第 559 页的『priv_workspace_section_inserts -“专用工作空间节插入数”』
第 559 页的『priv_workspace_section_lookups -“专用工作空间节查询数”』
第 560 页的『priv_workspace_size_top -“最大专用工作空间大小”』
第 579 页的『rollback_sql_stmts -“尝试的回滚语句数”』
第 582 页的『rows_deleted -“删除行数”监视元素』
第 583 页的『rows_inserted -“插入行数”监视元素』
第 585 页的『rows_read -“读取行数”监视元素』
第 588 页的『rows_selected -“选择的行数”』
第 589 页的『rows_updated -“更新行数”监视元素』
第 592 页的『sec_log_used_top -“使用的最大辅助日志空间”』
第 594 页的『select_sql_stmts -“执行的 Select SQL 语句数”』
第 597 页的『server_platform -“服务器操作系统”』
第 601 页的『shr_workspace_num_overflows -“共享工作空间溢出数”』
第 602 页的『shr_workspace_section_inserts -“共享工作空间节插入数”』
第 603 页的『shr_workspace_section_lookups -“共享工作空间节查询数”』
第 603 页的『shr_workspace_size_top -“最大共享工作空间大小”』
第 605 页的『sort_overflows -“排序溢出数”监视元素』
第 616 页的『static_sql_stmts -“尝试的静态 SQL 语句数”』
第 617 页的『stats_cache_size -“统计信息高速缓存大小”监视元素』
第 617 页的『stats_fabricate_time -“产生统计信息的活动所花的总时间”监视元素』
第 618 页的『stats_fabrications -“产生统计信息的次数”监视元素』
第 633 页的『sync_runstats -“同步 RUNSTATS 活动总数”监视元素』
第 633 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』
第 665 页的『tot_log_used_top -“使用的最大总日志空间”』
第 667 页的『total_cons -“数据库激活以后的连接数”』
第 669 页的『total_hash_joins -“散列连接总数”』
第 670 页的『total_hash_loops -“总散列循环数”』

第 671 页的『total_olap_funcs －“OLAP 函数总数”监视元素』

第 676 页的『total_sort_time －“排序时间总计”监视元素』

第 677 页的『total_sorts －“排序总数”监视元素』

第 687 页的『uid_sql_stmts －“执行的 Update/Insert/Delete SQL 语句数”』

第 688 页的『unread_prefetch_pages －“未读取的预取页数”监视元素』

第 703 页的『x_lock_escals －“互斥锁定升级数”』

第 704 页的『xquery_stmts －“尝试的 XQuery 语句数”』

event_dbheader 逻辑数据组

第 343 页的『conn_time －“数据库连接时间”监视元素』

第 363 页的『db_name －“数据库名称”』

第 363 页的『db_path －“数据库路径”』

event_dbmemuse 逻辑数据组

第 482 页的『node_number －“节点号”』

第 517 页的『pool_config_size －“内存池的已配置大小”』

第 518 页的『pool_cur_size －“内存池的当前大小”』

第 525 页的『pool_id －“内存池标识”』

第 545 页的『pool_watermark －“内存池水位标记”』

event_deadlock 逻辑数据组

第 368 页的『deadlock_id －“死锁事件标识”』

第 368 页的『deadlock_node －“发生死锁的分区号”』

第 383 页的『dl_conns －“死锁中涉及的连接数”监视元素』

第 388 页的『evmon_activates －“事件监视器激活数”』

第 580 页的『rolled_back_agent_id －“回滚的代理程序”』

第 580 页的『rolled_back_appl_id －“回滚的应用程序”』

第 581 页的『rolled_back_participant_no －“回滚的应用程序参与者”监视元素』

第 581 页的『rolled_back_sequence_no －“回滚的序号”』

第 616 页的『start_time －“事件启动时间”』

event_detailed_dlconn 逻辑数据组

第 288 页的『agent_id －“应用程序句柄（代理程序标识）”监视元素』

第 299 页的『appl_id －“应用程序标识”』

第 301 页的『appl_id_holding_lk －“挂起锁定的应用程序标识”』

第 318 页的『blocking_cursor －“分块游标”』

第 344 页的『consistency_token －“程序包一致性标记”监视元素』

第 358 页的『creator －“应用程序创建者”』

第 359 页的『cursor_name －“游标名称”』

第 360 页的『data_partition_id －“数据分区标识”监视元素』

第 368 页的『deadlock_id －“死锁事件标识”』

第 368 页的『deadlock_node －“发生死锁的分区号”』

第 388 页的『evmon_activates -“事件监视器激活数”』
第 446 页的『lock_escalation -“锁定升级”监视元素』
第 449 页的『lock_mode -“锁定方式”监视元素』
第 450 页的『lock_mode_requested -“请求的锁定方式”监视元素』
第 451 页的『lock_node -“锁定节点”』
第 451 页的『lock_object_name -“锁定对象名称”』
第 452 页的『lock_object_type -“等待的锁定对象类型”监视元素』
第 456 页的『lock_wait_start_time -“锁定等待开始时间戳记”』
第 459 页的『locks_held -“挂起的锁定数”』
第 460 页的『locks_in_list -“报告的锁定数”』
第 498 页的『package_name -“程序包名”监视元素』
第 499 页的『package_version_id -“程序包版本”监视元素』
第 504 页的『participant_no -“死锁参与者”』
第 504 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』
第 593 页的『section_number -“节号”监视元素』
第 595 页的『sequence_no -“序号”监视元素』
第 596 页的『sequence_no_holding_lk -“挂起锁定的序号”』
第 616 页的『start_time -“事件启动时间”』
第 623 页的『stmt_operation/operation -“语句操作”监视元素』
第 627 页的『stmt_text -“SQL 语句文本”监视元素』
第 628 页的『stmt_type -“语句类型”监视元素』
第 635 页的『table_name -“表名”监视元素』
第 637 页的『table_schema -“表模式名”监视元素』
第 643 页的『tablespace_name -“表空间名称”监视元素』

event_dlconn 逻辑数据组

第 288 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 299 页的『appl_id -“应用程序标识”』
第 301 页的『appl_id_holding_lk -“挂起锁定的应用程序标识”』
第 360 页的『data_partition_id -“数据分区标识”监视元素』
第 368 页的『deadlock_id -“死锁事件标识”』
第 368 页的『deadlock_node -“发生死锁的分区号”』
第 388 页的『evmon_activates -“事件监视器激活数”』
第 444 页的『lock_attributes -“锁定属性”监视元素』
第 445 页的『lock_count -“锁定计数”监视元素』
第 445 页的『lock_current_mode -“转换前的原始锁定方式”』
第 446 页的『lock_escalation -“锁定升级”监视元素』
第 448 页的『lock_hold_count -“锁定挂起计数”监视元素』
第 449 页的『lock_mode -“锁定方式”监视元素』
第 450 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 450 页的『lock_name -“锁定名称”监视元素』
第 451 页的『lock_node -“锁定节点”』
第 451 页的『lock_object_name -“锁定对象名称”』
第 452 页的『lock_object_type -“等待的锁定对象类型”监视元素』
第 452 页的『lock_release_flags -“锁定释放标志”监视元素』
第 456 页的『lock_wait_start_time -“锁定等待开始时间戳记”』
第 504 页的『participant_no -“死锁参与者”』
第 504 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』
第 595 页的『sequence_no -“序号”监视元素』
第 596 页的『sequence_no_holding_lk -“挂起锁定的序号”』
第 616 页的『start_time -“事件启动时间”』
第 635 页的『table_name -“表名”监视元素』
第 637 页的『table_schema -“表模式名”监视元素』
第 643 页的『tablespace_name -“表空间名称”监视元素』
第 681 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』
第 682 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』
第 682 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』
第 683 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

event_histogrambin 逻辑数据组

第 316 页的『bin_id -“直方图条形标识”监视元素』
第 319 页的『bottom -“直方图类别底部”监视元素』
第 423 页的『histogram_type -“直方图类型”监视元素』
第 491 页的『number_in_bin -“条形中的数目”监视元素』
第 599 页的『service_class_id -“服务类标识”监视元素』
第 616 页的『statistics_timestamp -“统计信息时间戳记”监视元素』
第 664 页的『top -“直方图类别顶部”监视元素』
第 699 页的『work_action_set_id -“工作操作集标识”监视元素』
第 700 页的『work_class_id -“工作类标识”监视元素』
第 700 页的『workload_id -“工作负载标识”监视元素』

event_log_header 逻辑数据组

第 321 页的『byte_order -“事件数据的字节顺序”』
第 332 页的『codepage_id -“应用程序使用的代码页标识”』
第 388 页的『event_monitor_name -“事件监视器名称”』
第 489 页的『num_nodes_in_db2_instance -“分区中的节点数”』
第 597 页的『server_instance_name -“服务器实例名称”』
第 597 页的『server_prdid -“服务器产品/版本标识”』
第 660 页的『territory_code -“数据库地域代码”』
第 697 页的『version -“监视器数据版本”』

event_overflow 逻辑数据组

第 355 页的『count －“事件监视器溢出数”』

第 408 页的『first_overflow_time －“第一次事件溢出时间”』

第 441 页的『last_overflow_time －“最后一次事件溢出时间”』

第 482 页的『node_number －“节点号”』

event_qstats 逻辑数据组

第 442 页的『last_wlm_reset －“最后一次复位时间”监视元素』

第 565 页的『queue_assignments_total －“队列分配总次数”监视元素』

第 566 页的『queue_size_top －“队列大小顶部”监视元素』

第 566 页的『queue_time_total －“总队列时间”监视元素』

第 600 页的『service_subclass_name －“服务子类名”监视元素』

第 600 页的『service_superclass_name －“服务超类名”监视元素』

第 616 页的『statistics_timestamp －“统计信息时间戳记”监视元素』

第 661 页的『threshold_domain －“阈值域”监视元素』

第 662 页的『threshold_name －“阈值名称”监视元素』

第 662 页的『threshold_predicate －“阈值谓词”监视元素』

第 662 页的『thresholdid －“阈值标识”监视元素』

第 699 页的『work_action_set_name －“工作操作集名称”监视元素』

第 700 页的『work_class_name －“工作类名”监视元素』

event_scstats 逻辑数据组

第 280 页的『act_cpu_time_top －“活动 CPU 时间顶部”监视元素』

第 282 页的『act_remapped_in －“重新映入的活动数”监视元素』

第 282 页的『act_remapped_out －“重新映出的活动数”监视元素』

第 282 页的『act_rows_read_top －“活动读取行数顶部”监视元素』

第 296 页的『agg_temp_tablespace_top －“聚集临时表空间顶部”监视元素』

第 335 页的『concurrent_act_top －“并行活动顶部”监视元素』

第 336 页的『concurrent_wlo_top －“并行工作负载项顶部”监视元素』

第 336 页的『concurrent_connection_top －“并行连接顶部”监视元素』

第 347 页的『coord_act_aborted_total －“异常终止的协调程序活动总数”监视元素』

第 348 页的『coord_act_completed_total －“完成的协调程序活动总数”监视元素』

第 348 页的『coord_act_est_cost_avg －“平均协调程序活动估计成本”监视元素』

第 349 页的『coord_act_exec_time_avg －“平均协调程序活动执行时间”监视元素』

第 349 页的『coord_act_interarrival_time_avg －“平均协调程序活动到达时间”监视元素』

第 350 页的『coord_act_lifetime_avg －“平均协调程序活动生存期”监视元素』

第 351 页的『coord_act_lifetime_top －“协调程序活动生存期项部”监视元素』

第 351 页的『coord_act_queue_time_avg －“平均协调程序活动队列时间”监视元素』

第 352 页的『coord_act_rejected_total －“被拒绝的协调程序活动总数”监视元素』

第 354 页的『cost_estimate_top －“成本估计顶部”监视元素』

details_xml（此 XML 文档包含 MON_GET_SERVICE_SUBCLASS_DETAILS 表函数输出的 DETAILS 列中报告的所有监视元素。）

第 442 页的『last_wlm_reset －“最后一次复位时间”监视元素』

第 578 页的『request_exec_time_avg －“平均请求执行时间”监视元素』

第 588 页的『rows_returned_top －“最高实际返回行数”监视元素』

第 599 页的『service_class_id －“服务类标识”监视元素』

第 600 页的『service_subclass_name －“服务子类名”监视元素』

第 600 页的『service_superclass_name －“服务超类名”监视元素』

第 616 页的『statistics_timestamp －“统计信息时间戳记”监视元素』

第 660 页的『temp_tablespace_top －“临时表空间顶部”监视元素』

第 693 页的『uow_total_time_top －“UOW 时间总计顶部”监视元素』

event_start 逻辑数据组

第 616 页的『start_time －“事件启动时间”』

event_stmt 逻辑数据组

第 288 页的『agent_id －“应用程序句柄（代理程序标识）”监视元素』

第 295 页的『agents_top －“创建的代理程序数”』

第 299 页的『appl_id －“应用程序标识”』

第 318 页的『blocking_cursor －“分块游标”』

第 344 页的『consistency_token －“程序包一致性标记”监视元素』

第 358 页的『creator －“应用程序创建者”』

第 359 页的『cursor_name －“游标名称”』

第 406 页的『fetch_count －“成功的访存数”』

第 433 页的『int_rows_deleted －“删除的内部行数”』

第 434 页的『int_rows_inserted －“插入的内部行数”』

第 434 页的『int_rows_updated －“更新的内部行数”』

第 498 页的『package_name －“程序包名”监视元素』

第 499 页的『package_version_id －“程序包版本”监视元素』

第 503 页的『partial_record －“部分记录”监视元素』

第 518 页的『pool_data_l_reads －“缓冲池数据逻辑读取数”监视元素』

第 520 页的『pool_data_p_reads －“缓冲池数据物理读取数”监视元素』

第 526 页的『pool_index_l_reads －“缓冲池索引逻辑读取数”监视元素』

第 528 页的『pool_index_p_reads －“缓冲池索引物理读取数”监视元素』

第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』

第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』

第 539 页的『pool_temp_index_l_reads －“缓冲池临时索引逻辑读取数”监视元素』

第 540 页的『pool_temp_index_p_reads －“缓冲池临时索引物理读取数”监视元素』

第 585 页的『rows_read －“读取行数”监视元素』

第 589 页的『rows_written －“写入的行数”』

第 593 页的『section_number －“节号”监视元素』

第 595 页的『sequence_no -“序号”监视元素』
第 605 页的『sort_overflows -“排序溢出数”监视元素』
第 608 页的『sql_req_id -“SQL 语句的请求标识”』
第 609 页的『sqlca -“SQL 通信区（SQLCA）”』
第 616 页的『start_time -“事件启动时间”』
第 617 页的『stats_fabricate_time -“产生统计信息的活动所花的总时间”监视元素』
第 623 页的『stmt_operation/operation -“语句操作”监视元素』
第 627 页的『stmt_text -“SQL 语句文本”监视元素』
第 628 页的『stmt_type -“语句类型”监视元素』
第 632 页的『stop_time -“事件停止时间”』
第 633 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』
第 634 页的『system_cpu_time -“系统 CPU 时间”』
第 676 页的『total_sort_time -“排序时间总计”监视元素』
第 677 页的『total_sorts -“排序总数”监视元素』
第 694 页的『user_cpu_time -“用户 CPU 时间”』

event_stmt_history 逻辑数据组

第 334 页的『comp_env_desc -“编译环境”监视元素』
第 358 页的『creator -“应用程序创建者”』
第 368 页的『deadlock_id -“死锁事件标识”』
第 368 页的『deadlock_node -“发生死锁的分区号”』
第 388 页的『evmon_activates -“事件监视器激活数”』
第 498 页的『package_name -“程序包名”监视元素』
第 499 页的『package_version_id -“程序包版本”监视元素』
第 504 页的『participant_no -“死锁参与者”』
第 593 页的『section_number -“节号”监视元素』
第 595 页的『sequence_no -“序号”监视元素』
第 619 页的『stmt_first_use_time -“语句第一次使用时间”』
第 619 页的『stmt_history_id -“语句历史记录标识”』
第 620 页的『stmt_invocation_id -“语句调用标识”监视元素』
第 621 页的『stmt_isolation -“语句隔离”』
第 621 页的『stmt_last_use_time -“语句上一次使用时间”监视元素』
第 621 页的『stmt_lock_timeout -“语句锁定超时”监视元素』
第 622 页的『stmt_nest_level -“语句嵌套级别”监视元素』
第 624 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”』
第 625 页的『stmt_query_id -“语句查询标识”监视元素』
第 625 页的『stmt_source_id -“语句源标识”』
第 627 页的『stmt_text -“SQL 语句文本”监视元素』
第 628 页的『stmt_type -“语句类型”监视元素』

event_subsection 逻辑数据组

event_table 逻辑数据组

event_tablespace 逻辑数据组

第 390 页的『evmon_flushes －“事件监视器清空数”』
第 407 页的『files_closed －“关闭数据库文件数”监视元素』
第 503 页的『partial_record －“部分记录”监视元素』
第 509 页的『pool_async_data_read_reqs －“缓冲池异步读请求数”监视元素』
第 510 页的『pool_async_data_reads －“缓冲池异步数据读次数”监视元素』
第 511 页的『pool_async_data_writes －“缓冲池异步数据写次数”监视元素』
第 511 页的『pool_async_index_read_reqs －“缓冲池异步索引读请求数”监视元素』
第 512 页的『pool_async_index_reads －“缓冲池异步索引读取数”监视元素』
第 513 页的『pool_async_index_writes －“缓冲池异步索引写次数”监视元素』
第 514 页的『pool_async_read_time －“缓冲池异步读取时间”』
第 514 页的『pool_async_write_time －“缓冲池异步写入时间”』
第 518 页的『pool_data_l_reads －“缓冲池数据逻辑读取数”监视元素』
第 520 页的『pool_data_p_reads －“缓冲池数据物理读取数”监视元素』
第 521 页的『pool_data_writes －“缓冲池数据写次数”监视元素』
第 526 页的『pool_index_l_reads －“缓冲池索引逻辑读取数”监视元素』
第 528 页的『pool_index_p_reads －“缓冲池索引物理读取数”监视元素』
第 529 页的『pool_index_writes －“缓冲池索引写次数”监视元素』
第 532 页的『pool_no_victim_buffer －“缓冲池无牺牲缓冲区次数”监视元素』
第 533 页的『pool_read_time －“缓冲池物理读时间总计”监视元素』
第 535 页的『pool_temp_data_l_reads －“缓冲池临时数据逻辑读取数”监视元素』
第 537 页的『pool_temp_data_p_reads －“缓冲池临时数据物理读取数”监视元素』
第 539 页的『pool_temp_index_l_reads －“缓冲池临时索引逻辑读取数”监视元素』
第 540 页的『pool_temp_index_p_reads －“缓冲池临时索引物理读取数”监视元素』
第 546 页的『pool_write_time －“缓冲池物理写时间总计”监视元素』
第 643 页的『tablespace_name －“表空间名称”监视元素』

event_thresholdviolations 逻辑数据组

第 283 页的『activate_timestamp －“激活时间戳记”监视元素』
第 284 页的『activity_collected －“收集的活动”监视元素』
第 284 页的『activity_id －“活动标识”监视元素』
第 288 页的『agent_id －“应用程序句柄（代理程序标识）”监视元素』
第 299 页的『appl_id －“应用程序标识”』
第 353 页的『coord_partition_num －“协调程序分区号”监视元素』
第 371 页的『destination_service_class_id －“目标服务类标识”监视元素』
第 607 页的『source_service_class_id －“源服务类标识”监视元素』
第 660 页的『threshold_action －“阈值操作”监视元素』
第 661 页的『threshold_maxvalue －“阈值最大值”监视元素』
第 662 页的『threshold_predicate －“阈值谓词”监视元素』
第 662 页的『threshold_queuesize －“阈值队列大小”监视元素』
第 662 页的『thresholdid －“阈值标识”监视元素』

第 663 页的『time_of_violation -“违例时间”监视元素』

第 690 页的『uow_id -“工作单元标识”监视元素』

event_wlstats 逻辑数据组

第 280 页的『act_cpu_time_top -“活动 CPU 时间顶部”监视元素』

第 282 页的『act_rows_read_top -“活动读取行数顶部”监视元素』

第 336 页的『concurrent_wlo_act_top -“并行 WLO 活动顶部”监视元素』

第 336 页的『concurrent_wlo_top -“并行工作负载项顶部”监视元素』

第 347 页的『coord_act_aborted_total -“异常终止的协调程序活动总数”监视元素』

第 348 页的『coord_act_completed_total -“完成的协调程序活动总数”监视元素』

第 348 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 349 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 349 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 350 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 351 页的『coord_act_lifetime_top -“协调程序活动生存期项部”监视元素』

第 351 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 352 页的『coord_act_rejected_total -“被拒绝的协调程序活动总数”监视元素』

第 354 页的『cost_estimate_top -“成本估计顶部”监视元素』

details_xml（此 XML 文档包含 MON_GET_WORKLOAD_DETAILS 表函数输出的 DETAILS 列中报告的所有监视元素。）

第 442 页的『last_wlm_reset -“最后一次复位时间”监视元素』

第 458 页的『lock_wait_time_top -“锁定等待时间顶部”监视元素』

第 588 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 616 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 693 页的『uow_total_time_top -“UOW 时间总计顶部”监视元素』

第 660 页的『temp_tablespace_top -“临时表空间顶部”监视元素』

第 699 页的『wlo_completed_total -“完成的工作负载项总数”监视元素』

第 700 页的『workload_id -“工作负载标识”监视元素』

第 701 页的『workload_name -“工作负载名称”监视元素』

event_wcstats 逻辑数据组

第 280 页的『act_cpu_time_top -“活动 CPU 时间顶部”监视元素』

第 282 页的『act_rows_read_top -“活动读取行数顶部”监视元素』

第 283 页的『act_total -“活动总数”监视元素』

第 348 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 349 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 349 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 350 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 351 页的『coord_act_lifetime_top -“协调程序活动生存期项部”监视元素』

第 351 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 354 页的『cost_estimate_top -“成本估计顶部”监视元素』

第 442 页的『last_wlm_reset -“最后一次复位时间”监视元素』

第 588 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 616 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 660 页的『temp_tablespace_top -“临时表空间顶部”监视元素』

第 699 页的『work_action_set_id -“工作操作集标识”监视元素』

第 699 页的『work_action_set_name -“工作操作集名称”监视元素』

第 700 页的『work_class_id -“工作类标识”监视元素』

第 700 页的『work_class_name -“工作类名”监视元素』

event_xact 逻辑数据组

第 288 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 299 页的『appl_id -“应用程序标识”』

第 446 页的『lock_escals -“锁定升级次数”监视元素』

第 456 页的『lock_wait_time -“等待锁定时间”监视元素』

第 460 页的『locks_held_top -“挂起的最大锁定数”』

第 503 页的『partial_record -“部分记录”监视元素』

第 558 页的『prev_uow_stop_time -“上一个工作单元完成时间戳记”』

第 585 页的『rows_read -“读取行数”监视元素』

第 589 页的『rows_written -“写入的行数”』

第 595 页的『sequence_no -“序号”监视元素』

第 634 页的『system_cpu_time -“系统 CPU 时间”』

第 681 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 682 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 682 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 683 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

第 690 页的『uow_log_space_used -“使用的工作单元日志空间”』

第 691 页的『uow_start_time -“工作单元开始时间戳记”』

第 692 页的『uow_status -“工作单元状态”』

第 692 页的『uow_stop_time -“工作单元停止时间戳记”监视元素』

第 694 页的『user_cpu_time -“用户 CPU 时间”』

第 703 页的『x_lock_escals -“互斥锁定升级数”』

lock 逻辑数据组

第 360 页的『data_partition_id -“数据分区标识”监视元素』

第 444 页的『lock_attributes -“锁定属性”监视元素』

第 445 页的『lock_count -“锁定计数”监视元素』

第 445 页的『lock_current_mode -“转换前的原始锁定方式”』

第 446 页的『lock_escalation -“锁定升级”监视元素』

第 448 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 449 页的『lock_mode -“锁定方式”监视元素』

第 450 页的『lock_name -“锁定名称”监视元素』

第 451 页的『lock_object_name -“锁定对象名称”』

第 452 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 452 页的『lock_release_flags -“锁定释放标志”监视元素』

第 453 页的『lock_status -“锁定状态”监视元素』

第 482 页的『node_number -“节点号”』

第 635 页的『table_file_id -“表文件标识”监视元素』

第 635 页的『table_name -“表名”监视元素』

第 637 页的『table_schema -“表模式名”监视元素』

第 643 页的『tablespace_name -“表空间名称”监视元素』

sqlca 逻辑数据组

sqlcabc

sqlcaid

sqlcode

sqlerrd

sqlerrmc

sqlerrml

sqlerrp

sqlstate

sqlwarn

受 COLLECT ACTIVITY DATA 设置影响的逻辑数据组

下表显示为不同的 COLLECT ACTIVITY DATA 选项指定所有类型的 WLM 对象时收集的逻辑数据组，这些对象包括“服务子类”、“工作负载”、“工作类”（通过“工作操作”）和“阈值”。

表 76. COLLECT ACTIVITY DATA 设置

COLLECT ACTIVITY DATA 的设置	收集的逻辑数据组
NONE	无
WITHOUT DETAILS	event_activity
WITH DETAILS	event_activity event_activitystmt
WITH DETAILS AND VALUES	event_activity event_activitystmt event_activityvals

第 15 章 数据库系统监视元素

对监视元素所收集数据的描述。

系统监视器返回的监视元素分为下列几个类别:

- **标识**，用于要监视的数据库管理器、应用程序或数据库连接。
- 这些数据主要用于帮助您**配置**系统。
- 各个级别的数据库**活动**，包括数据库、应用程序、表或语句。此信息可用于活动监视、问题确定和性能分析。此信息还可用于配置。
- 有关 **DB2 Connect** 应用程序的信息。包括在网关上运行的 DCS 应用程序、要执行的 SQL 语句和数据库连接。
- 有关**联合数据库系统**的信息。这包括有关由在 DB2 联合系统中运行的应用程序对数据源的总体访问的信息，以及由在联合服务器实例中运行的给定应用程序对数据源的访问的信息。

监视元素是以如下标准格式描述的:

元素标识

元素的名称。如果直接对数据流进行语法分析，那么元素标识将是大写的，并且加上 SQLM_ELM_ 前缀。

元素类型

监视元素返回的信息类型。例如，db2start_time 监视元素返回时间戳记。

快照监视信息

如果监视元素返回快照监视信息，将显示带有下列字段的表。

- *快照级别*: 快照监视器可捕获的信息的级别。例如，appl_status 监视元素返回应用程序级别和锁定级别的信息。
- *逻辑数据分组*: 返回捕获的快照信息的逻辑数据组。如果直接对数据流进行语法分析，那么逻辑数据组标识将是大写的，并且加上 SQLM_ELM_ 前缀。例如，appl_status 监视元素返回有关 appl_id_info 分组和 appl_lock_list 分组的信息。
- *监视开关*: 为获取此信息而必须设置的系统监视开关。如果开关为“基本”，那么总是收集有关该监视元素的数据。

事件监视信息

如果监视元素是由事件监视器收集的，将显示带有下列字段的表。

- *事件类型*: 事件监视器可收集的信息的级别。必须使用此事件类型来创建事件监视器以收集此信息。例如，将为 CONNECTIONS 事件监视器收集 appl_status 监视元素。
- *逻辑数据分组*: 在其中返回捕获的事件信息的逻辑数据组。如果直接对数据流进行语法分析，那么逻辑数据组标识将是大写的，并且加上 SQLM_ELM_ 前缀。例如，appl_status 监视元素返回有关 event_conn 分组的信息。
- *监视开关*: 为获取此信息而必须设置的系统监视开关。对于事件监视器，TIMESTAMP 开关是唯一能够限制事件数据收集的监视开关。如果此字段显示为虚线，那么总是收集有关该监视元素的数据。

用法 有关在监视数据库系统时能够如何使用监视元素收集的信息的信息。

acc_curs_blk – “接受的块游标请求数”

接受 I/O 块请求的次数。

元素标识

acc_curs_blk

元素类型

计数器

表 77. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 78. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 可将此元素与 *rej_curs_blk* 一起使用来计算接受的和/或拒绝的分块请求百分比。

有关如何使用此信息来调整配置参数的建议，请参阅 *rej_curs_blk*。

act_aborted_total – “异常终止活动总数”监视元素

在任何嵌套级别以出错情况完成的协调程序活动的总数。对于服务类而言，如果在活动异常终止前通过 REMAP ACTIVITY 操作将其重新映射到另一个服务子类，那么此活动将仅计入在其中异常终止此活动的子类的总数。

表 79. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 – 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 – 获取工作负载度量值	REQUEST METRICS BASE

表 79. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 80. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来了解系统上的活动是否成功完成。活动可能因取消、错误或反应性阈值而异常终止。

act_completed_total - "完成活动总数"监视元素

在任何嵌套级别成功完成的协调程序活动的总数。对于服务类而言，如果在活动完成前通过 REMAP ACTIVITY 操作将其重新映射到另一个子类，那么此活动将仅计入在其中完成此活动的子类的总数。

表 81. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 82. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定系统中的活动的吞吐量。

act_cpu_time_top －“活动 CPU 时间顶部”监视元素

服务类、工作负载或工作类中所有嵌套级别的活动所使用的处理器时间的高水位标记。

当此活动运行所在的服务类或工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素将返回 -1。仅当启用了请求度量值时，活动才会影响此高水位标记。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更新完成该活动的服务子类的 act_cpu_time_top 高水位标记（如果达到新的高水位标记的话）。该活动所映射到但未在其中完成该活动的其他服务子类的 act_cpu_time_top 高水位标记不受影响。

表 83. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此元素来确定在收集时间间隔内，分区中服务类、工作负载或工作类的活动所使用的最大处理器时间量。

act_exec_time －“活动执行时间”监视元素

在此分区执行所花的时间（以毫秒计）。对于游标来说，执行时间是打开、访存和关闭的综合时间。游标闲置的时间不会计入执行时间。对于例程来说，执行时间是指例程调用开始到结束这段时间。在例程结束之后由例程将其保持为打开状态以返回结果集的任何游标的生存期不会计入例程执行时间。对于其他所有活动来说，执行时间为开始时间与停止时间之间的时间差。在所有情况下，执行时间均不包括初始化或排队所花的时间。

表 84. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

可单独使用此元素来了解 DB2 在每个分区上执行活动所消耗的时间。还可以将此元素与协调程序分区上的 **time_started** 和 **time_completed** 监视元素配合使用来计算游标活动的空闲时间。您可以使用以下公式:

```
Cursor idle time = (time_completed - time_started) - act_exec_time
```

act_rejected_total - "被拒绝活动总数"监视元素

在任何嵌套级别由于被拒绝而未被允许执行的协调程序活动的总数。

表 85. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 86. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来帮助确定导致无法执行活动的预测性阈值或工作操作是否有效以及它们的限制是否过于严格。

act_remapped_in - "重新映入的活动数"监视元素

自从上次复位之后重新映射到此服务子类的活动数。

表 87. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此计数来确定是否已按预期将活动重新映射到此服务子类。

act_remapped_out - "重新映出的活动数"监视元素

自从上次复位之后从此服务子类中向外重新映射的活动数。

表 88. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此计数来确定是否已按预期从此服务子类中向外重新映射活动。

act_rows_read_top - "活动读取行数顶部"监视元素

服务类、工作负载或工作类中所有嵌套级别的活动所读取的行数的高水位标记。

当此活动运行所在的服务类或工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素将返回 -1。仅当启用了请求度量值时，活动才会影响此高水位标记。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更新完成该活动的服务子类的 act_rows_read_top 高水位标记（如果达到新的高水位标记的话）。该活动所映射到但未在其中完成该活动的服务子类的 act_rows_read_top 高水位标记不受影响。

表 89. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此元素来确定在收集时间间隔内，分区中服务类、工作负载或工作类的活动所读取的最大行数。

act_total －“活动总数”监视元素

自最后一次复位以后在任何嵌套级别处应用与指定工作类相对应的工作操作的活动总数。

表 90. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wcstats	-

用法

每次活动应用一个或多个与工作类相关联的工作操作时，均会更新工作类的计数器。此计数器通过使用 **act_total** 监视元素来展示。计数器可用于判断工作操作集的效用（例如，判断有多少活动已应用操作）。另外，它还可用于了解系统上不同类别的活动。

activate_timestamp －“激活时间戳记”监视元素

激活事件监视器的时间。

表 91. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
活动	event_activitystmt	-
活动	event_activityvals	-
阈值违例	event_thresholdviolations	-

用法

使用此元素以使上述事件类型返回的信息相关。

active_hash_joins －“活动散列连接数”

当前正在运行并消耗内存的散列连接的总数。

表 92. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

active_olap_funcs －“活动 OLAP 函数”监视元素

当前正在运行并消耗排序堆内存的 OLAP 函数总数。

表 93. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

可将快照监视的计数器复位。

active_sorts - "活动排序次数"

数据库中当前分配了排序堆的排序数。

表 94. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 将此值与 *sort_heap_allocated* 一起使用来确定每个排序使用的平均排序堆空间。如果 *sortheap* 配置参数实际上大于使用的平均排序堆，那么您可以降低此参数的值。

此值包括用于相关操作期间创建的临时表的排序堆。

activity_collected - "收集的活动"监视元素

此元素指示是否对违例的阈值收集活动事件监视记录。

表 95. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来确定是否要将违反阈值的活动的活动事件写入活动事件监视器。

当活动完成或异常终止且活动事件监视器处于活动状态时，如果此监视元素的值为"Y"，那么将收集违反此阈值的活动。如果此监视元素的值为"N"，那么将不收集。

activity_id - "活动标识"监视元素

用于唯一地标识给定工作单元中某个应用程序的活动的计数器。

表 96. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 97. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

表 97. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
活动	event_activitystmt	-
活动	event_activityvals	-
阈值违例	event_thresholdviolations	-

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

要在活动的工作单元外部唯一地标识该活动，请将 **activity_id** 和 **uow_id** 的组合与下列其中一项配合使用: **appl_id** 或 **agent_id**。

activity_secondary_id - "活动辅助标识"监视元素

此元素的值在每次对同一活动写入活动记录时增加。例如，如果调用 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 过程后写入活动记录一次，活动结束时再写入一次，那么第一个记录的元素的值为 0，第二个记录的元素值为 1。

表 98. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
活动	event_activitystmt	-
活动	event_activityvals	-

用法

当有关相同活动的信息被多次写入活动事件监视器时，将此元素与 **activity_id**、**uow_id** 和 **appl_id** 监视元素配合使用，以唯一地标识活动记录。

例如，在以下情况下，会将有关活动的信息发送至活动事件监视器两次:

- 在活动运行时，使用 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 存储过程来捕获有关活动的信息
- 因为对与活动关联的服务类指定了 COLLECT ACTIVITY DATA 子句，所以在完成活动时收集有关活动的信息

activity_state - "活动状态"监视元素

活动的当前状态。

表 99. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 DETAILS 中报告）	始终收集

用法

使用此监视元素来确定此活动当前正在执行的操作（例如，此活动是停滞在队列中还是正在等待来自客户机的输入）。可能的值包括:

- CANCEL_PENDING
- EXECUTING
- IDLE
- INITIALIZING
- QP_CANCEL_PENDING
- QP_QUEUED
- QUEUED
- TERMINATING
- UNKNOWN

activity_type - “活动类型”监视元素

活动的类型。

表 100. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 101. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

可能的值包括:

- LOAD
- READ_DML
- WRITE_DML
- DDL
- CALL
- OTHER

activitytotaltime_threshold_id - “活动时间总计阈值标识”监视元素

应用于此活动的 ACTIVITYTOTALTIME 阈值的标识。

表 102. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 ACTIVITYTOTALTIME 阈值（如果有的话）。

activitytotaltime_threshold_value - "活动时间总计阈值"监视元素

通过将 ACTIVITYTOTALTIME 阈值持续时间与活动进入时间相加计算而得的时间戳记。达到此时间戳记时，如果此活动仍在执行，那么将违反该阈值。

表 103. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 ACTIVITYTOTALTIME 阈值（如果有的话）。

activitytotaltime_threshold_violated - "违反活动时间总计阈值"监视元素

此监视元素返回"Yes"表明此活动已违反 ACTIVITYTOTALTIME 阈值。"No"表明此活动尚未违反该阈值。

表 104. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 ACTIVITYTOTALTIME 阈值。

address - 从中发起连接的 IP 地址

从中发起活动连接的 IP 地址。

表 105. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

使用此元素来确定从中发起活动连接的 IP 地址。显示的安全域名将转换为 IP 地址。

agent_id - “应用程序句柄（代理程序标识）”监视元素

应用程序在系统范围内的唯一标识。在单分区数据库中，此标识由一个 16 位的计数器构成。在多分区数据库中，此标识由协调分区号与 16 位计数器的并置构成。并且，对于应用程序从中建立辅助连接的每个分区，此标识都相同。

表 106. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	始终收集
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 107. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 108. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-
连接	event_connheader	-
语句	event_stmt	-
语句	event_subsection	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-
阈值违例	event_thresholdviolations	-
活动	event_activity	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除

去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

应用程序句柄，也称为代理程序标识，可用来唯一标识活动应用程序。

注: 根据您使用的 DB2 的版本，**agent_id** 监视元素会有不同的行为。从版本为 SQLM_DBMON_VERSION1 或 SQLM_DBMON_VERSION2 的 DB2 获取快照并发送至 DB2（版本 5 或更高版本）数据库时，返回的 **agent_id** 不能用作应用程序标识，而是作为应用程序提供服务的代理程序的 **agent_pid**。在这些情况下，仍然会返回 **agent_id** 以便与先前发行版兼容，但 DB2 数据库服务器内部不再将该值识别为 **agent_id**。

此值可用作需要代理程序标识的 GET SNAPSHOT 命令的输入或者需要应用程序句柄的监视器表函数的输入。

读取事件跟踪时，它可用来将事件记录与给定应用程序相匹配。

它还可用作 FORCE APPLICATION 命令或 API 的输入。在多节点系统上，可从应用程序具有连接的任何节点发出此命令。它的影响是全局性的。

agent_id_holding_lock －“挂起锁定的代理程序标识”

代理程序的应用程序句柄，该代理程序持有该应用程序正在等待的锁定。必须开启锁定监视器组，才能获取此信息。

表 109. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock_wait	锁定

用法 此元素可帮助您确定存在资源争用的应用程序。

如果此元素为 0（零）并且应用程序正在等待锁定，那么这指示锁定被不确定事务所挂起。可使用 appl_id_holding_lk 或 命令行处理器 LIST INDOUBT TRANSACTIONS 命令（当事务变得不确定时，它将显示处理该事务的 CICS 代理程序的应用程序标识）来确定不确定事务，然后对其执行落实或回滚操作。

注意，多个应用程序可挂起针对此应用程序正在等待的对象的共享锁定。有关该应用程序挂起的锁定类型的信息，请参阅 lock_mode。如果正在获取应用程序快照，那么将只返回对该对象挂起锁定的其中一个代理程序标识。如果正在获取锁定快照，那么将标识对该对象挂起锁定的所有代理程序标识。

agent_pid －“引擎可分派单元（EDU）标识”监视元素

代理程序的引擎可分派单元（EDU）的唯一标识。除在 Linux 操作系统上以外，EDU 标识与线程标识映射。在 Linux 操作系统上，EDU 标识是 DB2 生成的唯一标识。

表 110. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	agent	语句

表 111. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

用法

可以使用此元素来将数据库系统监视器信息链接至其他诊断信息源，如系统跟踪。还可使用它来监视为数据库应用程序工作的代理程序使用系统资源的方式。

agent_status - "DCS 应用程序代理程序数"

在连接集中器环境中，此值显示当前具有关联代理程序的应用程序。

元素标识

agent_status

元素类型

信息

表 112. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 值包括:

- SQLM_AGENT_ASSOCIATED

  代表此应用程序工作的代理程序与其相关联。

- SQLM_AGENT_NOT_ASSOCIATED

  代表此应用程序工作的代理程序不再与其相关联并且正被另一应用程序使用。没有关联代理程序的应用程序下一次完成工作时，将重新关联代理程序。

agent_sys_cpu_time - "代理程序使用的系统 CPU 时间"

数据库管理器代理进程使用的总系统 CPU 时间（以秒和微秒计）。

元素标识

agent_sys_cpu_time

元素类型

时间

表 113. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

对于应用程序级别的快照监视，可复位此计数器。不能在其他级别复位此计数器。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此元素包括花费在 SQL 和非 SQL 语句上的 CPU 时间，同时包括花费在所有不受防护的用户定义的函数（UDF）上的 CPU 时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

agent_usr_cpu_time - "代理程序使用的用户 CPU 时间"

数据库管理器代理进程使用的总 CPU 时间（以秒和微秒计）。

元素标识

agent_usr_cpu_time

元素类型

时间

表 114. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

可将快照监视的计数器复位。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您标识消耗大量 CPU 的应用程序或查询。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将返回为 0。

agent_wait_time - "代理程序等待时间"监视元素

在集中器配置中，已排队的应用程序在等待代理程序时耗用的时间。此值以毫秒计。

表 115. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 115. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 116. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

agent_wait_time 监视元素可用于帮助评估系统在集中器环境中的运行效率。如果代理程序等待时间相对于 **total_request_time** 监视元素的值而言过长，那么表明请求在进行排队以等待代理程序时耗用时间过多，即，表明发生下列一种或多种情况:

- 所配置的 **max_coordagents** 配置参数相对于工作负载而言太小。您可能需要增大 **max_coordagents** 配置参数的值，或者需要增大 **max_coordagents** 配置参数相对于 **max_connections** 配置参数的比率（如果这两个参数都设置为 AUTOMATIC 的话），以确保提供足够的协调代理程序为应用程序请求及时地提供服务。
- 落实工作负载的频率不够高。为了使集中器高效地工作，应用程序应该相对频繁地发出落实请求，从而确保它们的代理程序可以被释放以便为其他应用程序的请求提供服务。如果应用程序执行落实操作的频率不够高，那么可能需要配置相对较大的协调代理程序数目，以便缩短等待代理程序变为可用时耗用的时间。

agent_waits_total - “等待代理程序总次数”监视元素

在集中器配置中，应用程序被迫等待代理程序被分配的次数。

表 117. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 118. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过将此元素与 **agent_wait_time** 监视元素配合使用，可以确定集中器环境中应用程序请求等待代理程序时的平均耗用时间。

agents_created_empty_pool - “由于空的代理程序池而创建的代理程序数”

由于空的代理程序池而创建的代理程序数。它包括在 DB2 启动时创建的代理程序数（*num_initagents*）。

表 119. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 与 agents_from_pool 一起使用来计算

由于空的代理程序池而创建的代理程序数 / 从池中分配的代理程序数

有关使用此元素的信息，请参阅 agents_from_pool。

agents_from_pool －“从池中分配的代理程序数”

从代理程序池中分配的代理程序数。

表 120. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此元素可与 **agents_created_empty_pool** 监视元素配合使用以确定因为池是空的而必须创建代理程序的频率。

以下比率

由于空的代理程序池而创建的代理程序数 / 从池中分配的代理程序数

可用来帮助为 **num_poolagents** 配置参数设置适当的值。

对于大多数用户而言，缺省值 100 与 AUTOMATIC 将确保最优性能。

此比率可能随工作负载而有所波动。当系统上的活动量低时，可能会出现额外的代理程序创建和终止。而当系统上的活动量高时，将出现更多代理程序复用。比率低表示代理程序复用量高，这种情况一般出现在活动量高的系统上。比率高表示发生的代理程序创建量比复用量高。如果有问题，请增加 **num_poolagents** 配置参数的值以降低比率。但是，这样会导致系统产生额外的资源消耗。

agents_registered －“已注册的代理程序数”

正在监视的数据库管理器实例中的已注册代理程序的数目（协调代理程序和子代理程序）。

表 121. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

使用此元素来帮助评估 **max_coordagents** 和 **max_connections** 配置参数的设置以及查询内并行性的设置。

agents_registered_top －“已注册的最大代理程序数”

数据库管理器自启动后曾经同时注册的代理程序（协调代理程序和子代理程序）的最大数目。

表 122. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

可以使用此元素来帮助您评估 **max_coordagents** 和 **max_connections** 配置参数的设置以及查询内并行性的设置。

agents_registered 监视元素将记录捕获快照时注册的代理程序数。

agents_stolen -"失窃代理程序数"

在数据库管理器快照级别，此监视元素表示与应用程序关联的重新分配到其他应用程序工作的空闲代理数。在应用程序快照级别，此监视元素表示与其他应用程序关联的重新分配到此应用程序工作的空闲代理数。

表 123. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
应用程序	appl	基本

可将快照监视的计数器复位。

用法

缺省情况下，**num_poolagents** 配置参数设置为 AUTOMATIC。这意味着 DB2 自动管理空闲代理的组合，包括对与其他应用程序关联的空闲代理分配工作。

agents_top -"创建的代理程序数"

在应用程序级别，此项是执行语句时使用的代理程序的最大数目。在数据库级别，此项是用于所有应用程序的代理程序的最大数目。

元素标识

agents_top

元素类型

水位标记

表 124. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句
应用程序	stmt	语句

用法 指示查询内并行性实现情况的指示符。

agents_waiting_on_token －“正在等待令牌的代理程序数”

等待令牌以便在数据库管理器中执行事务的代理程序数。

注: 从 DB2 版本 9.5 开始，不推荐使用 **agents_waiting_on_token** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 125. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

可使用此元素来帮助您评估 **maxcagents** 配置参数的设置。

每个应用程序都有一个专用协调代理程序来处理数据库管理器中的数据库请求。每个代理程序都必须获取令牌才能执行事务。可以执行 数据库管理器 事务的最大代理程序数由配置参数 **maxcagents** 限制。

agents_waiting_top －“正在等待的最大代理程序数”监视元素

自数据库管理器启动后曾经同时等待令牌的代理程序的最大数目。

注: 从 DB2 版本 9.5 开始，不推荐使用 **agents_waiting_top** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 126. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

使用此元素来帮助您评估 **maxcagents** 配置参数的设置。

获取快照时等待令牌的代理程序数由 **agents_waiting_on_token** 监视元素将记录。

如果 **maxcagents** 参数设置为其缺省值（-1），那么不应有任何代理程序等待令牌并且此监视元素的值应该为零。

agg_temp_tablespace_top －“聚集临时表空间顶部”监视元素

服务类中所有嵌套级别的 DML 活动聚集临时表空间使用量的高水位标记（以 KB 计）。此聚集值是通过对服务子类中所有活动的临时表空间使用量进行求和计算而得的，此高水位标记表示此聚集值在上次复位之后达到的最大值。当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素将返回 -1。必须至少对此记录所属子类的超类中的一个服务子类定义并启用 AGGSQLTEMPSPACE 阈值，否则将返回 0 值。

表 127. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此元素来确定在收集时间间隔内，服务子类分区中达到的最高聚集 DML 活动系统临时表空间使用量。

aggsqltempspace_threshold_id -“阈值 SQL 临时空间阈值标识”监视元素

应用于此活动的 AGGSQLTEMPSPACE 阈值的数字标识。

表 128. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 AGGSQLTEMPSPACE 阈值（如果有的话）。

aggsqltempspace_threshold_value -“AggSQL 临时空间阈值”监视元素

应用于此活动的 AGGSQLTEMPSPACE 阈值的上限。

表 129. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 AGGSQLTEMPSPACE 阈值（如果有的话）。

aggsqltempspace_threshold_violated -“违反 AggSQL 临时空间阈值”监视元素

当这个可选的监视元素设置为“Yes”时，表明此活动已违反对其应用的 AGGSQLTEMPSPACE 阈值。“No”表明此活动尚未违反该阈值。

表 130. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 AGGSQLTEMPSPACE 阈值。

app_rqsts_completed_total - “完成应用程序请求总数”监视元素

协调程序所执行的外部（应用程序）请求的总数。对于服务子类而言，将仅对完成此应用程序请求的子类更新此监视元素。

表 131. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 132. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定应用程序向系统提交的请求数。

appl_con_time – “连接请求启动时间戳记”

应用程序启动连接请求的日期和时间。

元素标识

appl_con_time

元素类型

时间戳记

表 133. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

用法 使用此元素来确定应用程序启动连接至数据库的请求的时间。

appl_id – “应用程序标识”

此标识是在应用程序连接至数据库管理器上的数据库或 DB2 Connect 接收到连接至 DRDA® 数据库的请求时生成的。

表 134. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本
锁定	appl_lock_list	基本

表 135. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-
连接	event_conn	-
连接	event_connheader	-
语句	event_stmt	-
事务[1]	event_xact	-
死锁[2]	event_dlconn	-
带有详细信息的死锁[2]	event_detailed_dlconn	-
活动	event_activitystmt	-
活动	event_activity	-
活动	event_activityvals	-
阈值违例	event_thresholdviolations	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

2 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此标识在客户机和服务器上都是已知的，所以可使用它来使应用程序的客户机部分与服务器部分相关。对于 DB2 Connect 应用程序，那么还需要使用 **outbound_appl_id** 监视元素来使应用程序的客户机部分与服务器部分相关。

此标识在网络上是唯一的。应用程序标识有不同的格式，这取决于运行数据库管理器和/或 DB2 Connect 的客户机与服务器之间的通信协议。每种格式由用逗号隔开的三个部分构成。

1. TCP/IP

   格式 `IPAddr.Port.Application instance`

   IPv4

   示例

   ```
   G91A3955.F33A.02DD18143340
   ```

   详细信息

   在 IPv4 中，TCP/IP 生成的应用程序标识由三个部分组成。第一个部分包含 IP 地址。它表示为 32 位数字，最大显示为 8 位十六进制字符。第二部分包含端口号，表示为 4 位十六进制字符。第三部分包含此应用程序的实例的唯一标识。

   注: 当 IP 地址或端口号的十六进制版本以 0 至 9 开头，那么它们将分别转换为 G 至 P。例如，“0”映射为“G”，“1”映射为“H”，以此类推。

   IP 地址 AC10150C.NA04.006D07064947 表示为如下所示:

   - IP 地址仍为 AC10150C，它将转换为 172.16.21.12。
   - 端口号为 NA04。第一个字符为“N”，它将映射为“7”。因此，端口号的十六进制为 7A04，它将转换为十进制格式 31236。

   IPv6

   示例

   ```
   1111:2222:3333:4444:5555:6666:
   7777:8888.65535.0123456789AB
   ```

   详细信息

   在 IPv6 中，TCP/IP 生成的应用程序标识由三部分组成。第一部分包含 IP 地址，它是格式为 a:b:c:d:e:f:g:h 的 39 位可读地址，a 至 h 中的每一项为 4 位十六进制数字。第二部分为可读的 5 字节端口号。第三部分是此应用程序的实例的唯一时间戳记标识。

2. 本地应用程序

   格式 `*LOCAL.DB2 instance.Application instance`

示例

```
*LOCAL.DB2INST1.930131235945
```

详细信息

为本地应用程序生成的应用程序标识是通过并置字符串 *LOCAL、DB2 实例的名称和此应用程序的实例的唯一标识构成的。

对于多数据库分区实例，LOCAL 将替换为 Nx，其中 x 是客户机用来连接至数据库的分区号。例如，*N2.DB2INST1.0B5A12222841。

使用 **client_protocol** 监视元素来确定连接使用的通信协议，并因此确定 **appl_id** 监视元素的格式。

appl_id_holding_lk - "挂起锁定的应用程序标识"

已经持有所需的对象锁定的应用程序的标识。

元素标识

appl_id_holding_lk

元素类型

信息

表 136. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock_wait	锁定

表 137. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

用法 此元素可帮助您确定存在资源争用的应用程序。具体地说，它可以帮助您标识挂起锁定的应用程序句柄（代理程序标识）和表标识。注意，可使用 LIST APPLICATIONS 命令来获取有关带有代理程序标识的应用程序标识的信息。但是，最好在获取快照时收集此类型的信息，原因是应用程序在运行 LIST APPLICATIONS 命令之前结束时此项将变得不可用。

注意，多个应用程序可挂起某个对象的共享锁定，此应用程序正在等待对该对象获取锁定。有关该应用程序挂起的锁定类型的信息，请参阅 lock_mode。如果正在获取应用程序快照，那么将只返回对该对象挂起锁定的其中一个应用程序标识。如果正在获取锁定快照，那么将返回对该对象挂起锁定的所有应用程序标识。

appl_id_oldest_xact - "带有最旧事务的应用程序"

具有最旧事务的应用程序的标识（对应于应用程序快照中的 *agent_id* 值）。

元素标识

appl_id_oldest_xact

元素类型

信息

表 138. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 此元素可帮助您确定具有最旧活动事务的应用程序。可强制此应用程序释放日志空间。如果它占用了大量日志空间，那么应检查应用程序以确定是否可以修改它以提高执行落实操作的频率。

有时没有事务停止记录或者最旧的事务没有应用程序标识（例如，不确定事务或不活动事务）。在这类情况下，数据流中不会返回此应用程序的标识。

appl_idle_time - "应用程序空闲时间"

应用程序对服务器发出任何请求后经历的秒数。这包括未终止事务的应用程序，如未发出落实或回滚的应用程序。

元素标识

appl_idle_time

元素类型

信息

表 139. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句
DCS 应用程序	dcs_appl	语句

用法 此信息可用来实现强制用户空闲指定秒数的应用程序。

appl_name - "应用程序名称"监视元素

正在客户机上运行的并且数据库或 DB2 Connect 服务器所知道的应用程序的名称。

表 140. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 141. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-
连接	event_connheader	-
活动	event_activity	-

用法

此元素可与 **appl_id** 配合使用以使数据项与应用程序相关。

在客户机/服务器环境中建立数据库连接时此名称将从客户机传递至服务器。CLI 应用程序可通过对 SQLSetConnectAttr 的调用来设置 SQL_ATTR_INFO_PROGRAMNAME 属性。如果 SQL_ATTR_INFO_PROGRAMNAME 是在建立与服务器的连接前设置的，那么指定的值将覆盖实际客户机应用程序名称并且成为 **appl_name** 监视元素中显示的值。

如果客户机应用程序代码页与运行 数据库系统监视器 的代码页不同，可使用 **codepage_id** 来帮助转换 **appl_name**。

appl_priority - "应用程序代理程序优先级"

为此应用程序工作的代理程序的优先级。

元素标识

appl_priority

元素类型

信息

表 142. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 143. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 可使用此元素来检查应用程序是否以期望的优先级运行。应用程序优先级可由管理员设置。可通过控制器实用程序（**db2gov**）来更改优先级。

DB2 使用控制器来监视和更改对数据库运行的应用程序的行为。此信息用来调度应用程序和平衡系统资源。

控制器守护程序通过获取快照来收集有关应用程序的统计信息。它将对照管理对该数据库运行的应用程序的规则来检查这些统计信息。如果控制器检测到规则违例，那么采取适当的操作。这些规则和操作是由您在控制器配置文件中指定的。

如果与某个规则相关联的操作将更改应用程序的优先级，那么控制器将在检测到违例的分区中更改代理程序的优先级。

appl_priority_type - "应用程序优先级类型"

代表应用程序工作的代理程序的操作系统优先级类型。

元素标识

appl_priority_type

元素类型

信息

表 144. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 145. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 动态优先级由操作系统根据使用情况重新计算。静态优先级不会更改。

appl_section_inserts – “节插入数”监视元素

应用程序从其共享 SQL 工作空间插入 SQL 节的次数。

表 146. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 147. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法

任何可执行节的工作副本都存储在共享 SQL 工作空间中。这是出现副本不可用并且必须插入的情况的计数。

appl_section_lookups – “节查询数”

应用程序从其共享 SQL 工作空间查询 SQL 节的次数。

表 148. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 149. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

表 149. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法

每个代理程序都可以访问保留任何可执行节的工作副本的共享 SQL 工作空间。此计数器指示应用程序的代理程序访问 SQL 工作区的次数。

appl_status –“应用程序状态”

应用程序的当前状态。

表 150. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本

表 151. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
连接	event_conn	-

用法

此元素可以帮助您诊断潜在的应用程序问题。下表列示了此字段的值。

API 常量	描述
SQLM_AUTONOMOUS_WAIT	**自主等待**: 应用程序等待自主例程完成。
SQLM_BACKUP	**正在备份数据库**: 应用程序正在执行数据库备份。
SQLM_COMMIT_ACT	**正在落实**: 工作单元正在落实对其数据库所作的更改。
SQLM_COMP	**正在编译**: 数据库管理器正在为应用程序编译 SQL 语句或者预编译方案。
SQLM_CONNECTED	**数据库连接已完成**: 应用程序已启动数据库连接请求，并且该请求已完成。
SQLM_CONNECTPEND	**数据库连接暂挂**: 应用程序已启动数据库连接请求，但该请求尚未完成。
SQLM_CREATE_DB	**正在创建数据库**: 代理程序已启动了数据库创建请求，该请求尚未完成。
SQLM_DECOUPLED	**已经与代理程序解耦**: 当前没有与应用程序相关联的代理程序。这是一种正常状态。当连接集中器处于启用状态时，没有专用的协调代理程序，因此可以在协调程序分区中将应用程序解耦。在非集中器环境中，由于始终有专用的协调代理程序，所以无法在协调程序分区中将应用程序解耦。

API 常量	描述
SQLM_DISCONNECTPEND	**数据库断开连接暂挂**：应用程序已启动数据库断开连接命令，但该命令尚未完成执行。可能是应用程序未显式执行数据库断开连接命令。如果应用程序在未执行断开连接命令的情况下结束，数据库管理器就会断开与数据库的连接。
SQLM_INTR	**请求已中断**：正在处理请求中断。
SQLM_IOERROR_WAIT	**等待以禁用表空间**：应用程序检测到 I/O 错误，并且正在尝试禁用特定表空间。应用程序必须先等待对该表空间执行的所有其他活动事务完成，然后才能禁用该表空间。
SQLM_LOAD	**数据快速装入**：应用程序正在执行“快速装入”以便将数据装入到数据库中。
SQLM_LOCKWAIT	**等待锁定**：工作单元正在等待锁定。获取锁定之后，状态将恢复为其先前值。
SQLM_QUIESCE_TABLESPACE	**正在停顿表空间**：应用程序正在执行停顿表空间请求。
SQLM_RECOMP	**正在重新编译**：数据库管理器正在为应用程序重新编译（即，重新绑定）方案。
SQLM_REMOTE_RQST	**联合请求暂挂**：应用程序正在等待来自联合数据源的结果。
SQLM_RESTART	**正在重新启动数据库**：应用程序正在重新启动数据库以执行崩溃恢复。
SQLM_RESTORE	**正在恢复数据库**：应用程序正在将备份映像恢复到数据库。
SQLM_ROLLBACK_ACT	**正在回滚**：工作单元正在回滚对其数据库所作的更改。
SQLM_ROLLBACK_TO_SAVEPOINT	**回滚到保存点**：应用程序正在回滚到保存点。
SQLM_TEND	**事务已结束**：该工作单元是已结束但尚未进入两阶段落实协议准备阶段的全局事务的一部分。
SQLM_THABRT	**事务已试探性回滚**：该工作单元是已试探性回滚的全局事务的一部分。
SQLM_THCOMT	**事务已试探性落实**：该工作单元是已试探性落实的全局事务的一部分。
SQLM_TPREP	**事务已准备好**：该工作单元是已进入两阶段落实协议准备阶段的全局事务的一部分。
SQLM_UNLOAD	**数据快速卸装**：应用程序正在执行“快速卸装”以便从数据库中卸装数据。
SQLM_UOWEXEC	**工作单元正在执行**：数据库管理器正在代表工作单元执行请求。
SQLM_UOWWAIT	**工作单元正在等待**：数据库管理器代表应用程序中的工作单元正在等待。此状态通常表示系统正在执行应用程序代码。
SQLM_WAITFOR_REMOTE	**暂挂远程请求**：应用程序正在等待来自分区数据库实例中的远程分区的响应。

application_handle – “应用程序句柄”监视元素

应用程序在系统范围内的唯一标识。在单分区数据库中，此标识由一个 16 位的计数器构成。在多分区数据库中，此标识由协调分区号与 16 位计数器的并置构成。并且，对于应用程序从中建立辅助连接的每个分区，此标识都相同。

表 152. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 – 获取连接度量值	始终收集
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 153. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 154. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-
连接	event_connheader	-
语句	event_stmt	-
语句	event_subsection	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-
阈值违例	event_thresholdviolations	-
活动	event_activity	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此监视元素是 **agent_id** 监视元素的别名。

appls_cur_cons - "当前连接的应用程序数"

指示当前连接至数据库的应用程序数。

表 155. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
锁定	db_lock_list	基本

用法 可使用此元素来帮助您了解数据库内的活动级别以及正在使用的系统资源量。

它可帮助您调整 *maxappls* 和 *max_coordagents* 配置参数的设置。例如，它的值总是与 *maxappls* 相同，您可能想要提高 *maxappls* 的值。有关更多信息，请参阅 *rem_cons_in* 和 *local_cons* 监视元素。

appls_in_db2 - "数据库中当前执行的应用程序数"

指示当前连接至数据库并且数据库管理器正对其处理请求的应用程序的数目。

表 156. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

arm_correlator - "应用程序响应测量相关因子"监视元素

符合应用程序响应测量（ARM）标准的事务标识。

表 157. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

如果与此活动关联的应用程序也支持应用程序响应测量（ARM）标准，那么可以使用此元素来将活动事件监视器收集的活动链接至此类应用程序。

associated_agents_top - "最大关联代理程序数"

与此应用程序相关联的最大子代理程序数。

表 158. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

async_runstats - "异步 RUNSTATS 请求总数"监视元素

实时统计信息收集对数据库中所有应用程序执行的成功异步 RUNSTATS 活动总数。所有数据库分区报告的值将汇总合计。

表 159. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句

可将快照监视的计数器复位。

表 160. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

使用此元素来确定实时统计信息收集执行的成功异步 RUNSTATS 活动数。此值经常更改。为更好地了解系统使用情况，请长期在特定时间间隔捕获快照。与 **sync_runstats** 和 **stats_fabrications** 监视元素配合使用时，此元素可帮助您跟踪与实时统计信息收集相关的不同统计信息收集活动类型并分析它们对性能的影响。

audit_events_total - "审计事件总数"监视元素

所生成的审计事件的总数。

表 161. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 162. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 162. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

audit_file_write_wait_time - "审计文件写等待时间"监视元素

等待写审计记录时耗用的时间。此值以毫秒计。

表 163. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 164. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定代理程序等待以同步方式打开审计事件并将其写入磁盘时耗用的时间。

在典型情况下，每次将只有一个代理程序尝试打开审计日志文件，这是因为，其他代理程序在打开该文件前将等待访问公共审计子系统。因此，此等待时间通常代表等待

操作系统将该文件写入磁盘时耗用的时间。审计实用程序可能会在执行期间锁定审计日志文件，这将导致代理程序打开并写审计日志文件时的等待时间超出正常情况。如果已启用异步审计功能，那么大于异步审计缓冲区大小的审计事件将被直接写入磁盘而不是写入缓冲区，这将导致等待时间延长。

除特殊的审计实用程序情况以外，等待时间取决于磁盘速度以及操作系统将数据写入磁盘的及时性。对于给定的应用程序和审计配置，要缩短此等待时间，您可以调整操作系统或者使用更快的磁盘。

audit_file_writes_total - "写审计文件总次数"监视元素

代理程序被迫等待将审计事件直接写入磁盘的总次数。

表 165. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 166. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过将此监视元素与 **audit_file_write_wait_time** 监视元素配合使用，可以确定应用程序请求等待以同步方式打开审计事件并将其写入磁盘时的平均耗用时间。

audit_subsystem_wait_time – “审计子系统等待时间”监视元素

等待审计缓冲区空间时耗用的时间。当审计缓冲区已满，并且代理程序必须等待审计守护程序将缓冲区内容写入磁盘时，就会发生这种等待情况。此值以毫秒计。

表 167. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 – 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 168. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定代理程序等待访问公共审计子系统时耗用的时间，在这段时间内，公共审计子系统正忙于为其他代理程序处理事件。

审计子系统的某些公共部分每次只能由一个代理程序访问。此监视元素的值指示代理程序访问公共审计子系统时必须等待的时间。这包括填充当前异步缓冲区的代理程序在等待审计守护程序将先前异步缓冲区写入磁盘完成时耗用的时间。其他正在等待写审计日志文件或者正在等待发出审计守护程序请求的代理程序也已访问公共审计子系统，并且其等待时间也将在此值中反映。

如果正在使用异步审计功能，那么可以通过更改 **audit_buf_sz** 配置参数的值来缩短此等待时间。您可以不断增大 **audit_buf_sz** 配置参数的值，直到进一步增大此值不再能够缩短公共审计子系统等待时间为止。在这个点，异步缓冲区大小已足以保证守护程序能够在下一个缓冲区变满前将一个完整的缓冲区写入磁盘，因此守护程序不再是瓶

颈。如果必须将 **audit_buf_sz** 配置参数的值增大到发生系统故障可能会导致过多审计记录丢失的程度，那么可以通过调整操作系统或使用更高速的磁盘来缩短等待时间。如果有必要进一步缩短等待时间，请使用审计策略来减少所生成的审计事件的数目。

audit_subsystem_waits_total - "审计子系统等待总次数"监视元素

审计子系统等待缓冲区写操作完成的次数。

表 169. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 170. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定代理程序在访问公共审计子系统时被迫等待的总次数。在生成一个审计事件时，可能不需要访问公共审计子系统，也可以需要访问该子系统一次或多次以记录该事件。请使用 **audit_events_total** 监视元素来确定所生成的审计事件的准确数目。

auth_id - "授权标识"

调用受监视应用程序的用户的授权标识。在 DB2 Connect 网关节点上，这是用户在主机上的授权标识。

表 171. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 172. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-
连接	event_connheader	-

用法

在显式的可信连接中，当您切换用户时，**auth_id** 值不会立即更改。而是在您切换用户之后首次访问数据库时，才会更新 **auth_id**。这是因为切换用户操作始终会影响到后续操作。

可使用此元素来确定调用该应用程序的人员。

authority_bitmap －“用户权限级别”监视元素

对用户及用户所属的组授予的权限。这些权限包括授予特定角色的权限，该角色是授予该用户及其所属组的角色。对用户或授予用户的角色授予的权限被视为用户权限。对用户所属组或授予用户所属组的角色授予的权限被视为组权限。

表 173. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	appl_info	基本

表 174. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法

authority_bitmap 监视元素的格式为数组格式。每个数组元素为一个字符，表示是否对用户标识授予了特定权限以及用户获得该权限的方式。

各个数组元素通过在 sql.h 文件中定义的下标值来创建下标。authority_bitmap 数组中的下标值称为*权限下标*。例如，SQL_DBAUTH_SYSADM 是确定用户是否拥有 SYSADM 权限的下标。

authority_bitmap 数组中由权限下标标识的一个元素的值表示授权标识是否拥有该权限。要确定授权标识如何拥有权限，对由授权下标标识的每个数组元素使用来自 sql.h 的下列定义:

SQL_AUTH_ORIGIN_USER
如果此位为 on，那么表示该授权标识拥有授予该用户或其角色的权限。

SQL_AUTH_ORIGIN_GROUP
如果此位为 on，那么表示该授权标识拥有授予该用户或其角色的权限。

例如，要确定用户是否具有 DBADM 权限，验证以下值:

```
authority_bitmap[SQL_DBAUTH_DBADM]
```

要确定用户是否直接拥有 DBADM 权限，验证:

```
authority_bitmap[SQL_DBAUTH_DBADM] & SQL_AUTH_ORIGIN_USER
```

authority_lvl －“用户权限级别”监视元素

授予应用程序的最高权限级别。

注: 从 DB2 数据库版本 9.5 开始，不推荐使用 authority_lvl 监视元素。请改为使用 authority_bitmap 监视元素。请参阅第 314 页的『authority_bitmap －“用户权限级别”监视元素』。

表 175. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	appl_info	基本

表 176. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 直接或间接授权进行应用程序允许的操作。

下面的定义来自 sql.h，可用来确定显式授予用户的权限:

- SQL_SYSADM
- SQL_DBADM
- SQL_CREATETAB
- SQL_BINDADD
- SQL_CONNECT
- SQL_CREATE_EXT_RT
- SQL_CREATE_NOT_FENC
- SQL_SYSCTRL
- SQL_SYSMAINT

下面的定义来自 sql.h，可用来确定从组或公用继承的间接权限:

- SQL_SYSADM_GRP
- SQL_DBADM_GRP
- SQL_CREATETAB_GRP
- SQL_BINDADD_GRP

- SQL_CONNECT_GRP
- SQL_CREATE_EXT_RT_GRP
- SQL_CREATE_NOT_FENC_GRP
- SQL_SYSCTRL_GRP
- SQL_SYSMAINT_GRP

auto_storage_hybrid - "混合自动存储器表空间指示器"监视元素

如果表空间是包含一些非自动存储器容器的自动存储器表空间，那么此监视元素将返回值 1。否则，它将返回值 0。

表 177. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

混合自动存储器表空间是已使用 ALTER TABLESPACE 命令进行转换以便由自动存储器管理，但尚未进行重新平衡的表空间。此表空间仍包含非自动存储器容器。对此表空间进行重新平衡之后，它将只包含自动存储器容器，并且不再被视为混合表空间。

automatic - "自动调整缓冲池"监视元素

指示是否已对特定缓冲池启用自调整功能。如果已对此缓冲池启用自调整功能，那么此元素设置为 1，否则设置为 0。

表 178. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

bin_id - "直方图条形标识"监视元素

直方图条形标识。**bin_id** 在直方图内是唯一的。

表 179. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

使用此元素来区分同一直方图内的条形。

binds_precompiles - "尝试的绑定次数/预编译次数"

尝试的绑定次数和预编译次数。

元素标识

binds_precompiles

元素类型

计数器

表 180. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 181. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来了解数据库管理器内的当前活动级别。

此值不包括 *int_auto_rebinds* 的计数，但它包括因为 REBIND PACKAGE 命令而产生的绑定次数。

block_ios - "块 I/O 请求数"监视元素

块 I/O 请求的数目。更具体地说，就是 DB2 在缓冲池的块区域中执行顺序页预取的次数。

表 182. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 183. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

用法

如果启用了基于块的缓冲池，那么此监视元素将报告执行块 I/O 的频率。否则，此监视元素将返回 0。在使用基于块的缓冲池时，只有在顺序预取期间才监视块 I/O 请求的数目。

如果已启用基于块的缓冲池，并且此数目很低或者接近于向量 I/O 数（**vectored_ios** 监视元素的值），那么请考虑更改块大小。此状态可能指示下列其中一项:

- 一个或多个与缓冲池绑定的表空间的扩展数据块大小小于对缓冲池指定的块大小。
- 预取请求中请求的某些页已存在于缓冲池的页区域中。

预取程度允许在每个缓冲池块中浪费一些页，但如果浪费的页数过多，那么预取程序将决定在缓冲池的页区域中执行向量 I/O。

为了更好地利用基于块的缓冲池提供的顺序预取性能改进，应对块大小选择适当的值。但是，因为带有不同扩展数据块大小的多个表空间可能与同一个基于块的缓冲池绑定，所以这一点可能比较难以做到。为了获取最佳性能，建议将具有相同扩展数据块大小的表空间与一个基于块的缓冲池绑定，该缓冲池的块大小等于扩展数据块大小。如果表空间的扩展数据块大小大于块大小，那么可以获得较好的性能，扩展数据块大小小于块大小时情况则相反。

例如，如果扩展数据块大小为 2 而块大小为 8，那么将使用向量 I/O 而不是块 I/O（块 I/O 会浪费 6 页）。将块大小降低至 2 将解决此问题。

blocking_cursor - "分块游标"

此元素指示要执行的语句是否在使用分块游标。

元素标识

blocking_cursor

元素类型

信息

表 184. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 185. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

用法 对查询的数据传输使用分块可以改进性能。用于查询的 SQL 会影响分块的使用并且可能需要一些修改。

blocks_pending_cleanup - "暂挂清除已转出块"监视元素

数据库中继滚出删除后暂挂异步清除的 MDC 表块数。

表 186. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-
数据库	event_db	-

用法

使用此元素来确定删除延迟清除滚出后，未作为可用存储释放回系统的 MDC 表块数。

bottom - "直方图类别底部"监视元素

直方图类别范围的底部（该范围不含该底部值）。此监视元素的值也是上一直方图类别（如果有）的范围的包含顶端。

表 187. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

将此元素与相应的 **top** 元素配合使用来确定直方图内的类别范围。

boundary_leaf_node_splits - "边界叶节点分割次数"监视元素

插入操作期间分割边界叶节点的次数。

表 188. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

bp_cur_buffsz - "缓冲池的当前大小"

当前缓冲池大小。

表 189. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

bp_id - "缓冲池标识"监视元素

此元素包含正在监视的缓冲池的缓冲池标识。

表 190. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	基本

bp_name - "缓冲池名称"监视元素

缓冲池的名称。

表 191. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 192. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	基本

用法 每个数据库都至少需要一个缓冲池。根据您的需要，可以选择对单个数据库创建若干个大小不同的缓冲池。CREATE、ALTER 和 DROP BUFFERPOOL 语句允许您创建、更改或除去缓冲池。

创建新数据库后，它将具有缺省缓冲池 IBMDEFAULTBP，其大小将由平台确定。它还会具有一组系统缓冲池，每个系统缓冲池对应不同页大小:

- IBMSYSTEMBP4K
- IBMSYSTEMBP8K
- IBMSYSTEMBP16K
- IBMSYSTEMBP32K

不能更改这些系统缓冲池。

bp_new_buffsz - “新的缓冲池大小”

一旦重新启动数据库后缓冲池将更改至的大小。当以 DEFERRED 方式执行 ALTER BUFFERPOOL 语句时，在停止并重新启动数据库之前，缓冲池大小不会更改。

表 193. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

bp_pages_left_to_remove - “要除去的余下页数”

在完成缓冲池调整大小之前，缓冲池中要除去的余下页数。此项仅适用于以 IMMEDIATE 方式执行的 ALTER BUFFERPOOL 语句调用的缓冲池调整大小操作。

表 194. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

bp_tbsp_use_count - “映射至缓冲池的表空间数”

使用此缓冲池的表空间数。

表 195. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

buff_free – “当前可用的 FCM 缓冲区数”

此元素指示当前可用的 FCM 缓冲区数。

元素标识

buff_free

元素类型

标尺

表 196. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法 将当前可用的 FCM 缓冲区数与 *fcm_num_buffers* 配置参数配合使用来确定当前 FCM 缓冲池利用率。可使用此信息来调整 *fcm_num_buffers*。

buff_free_bottom – “最少可用 FCM 缓冲区数”

处理期间达到的最低可用 FCM 缓冲区数。

表 197. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法 将此元素与 *fcm_num_buffers* 配置参数一起使用来确定最大 FCM 缓冲池利用率。如果 buff_free_bottom 很低，那么应提高 *fcm_num_buffers* 以确保操作不会用完 FCM 缓冲区。如果 buff_free_bottom 很高，可降低 *fcm_num_buffers* 以节约系统资源。

byte_order – “事件数据的字节顺序”

数字数据的字节定序，具体地说是在“大尾数法”服务器（如 RS/6000®）还是“小尾数法”服务器（如基于 Intel 并且运行 Windows 2000 的 PC）上生成事件数据流。

表 198. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法 因为“大尾数法”服务器上的整数字节顺序与“小尾数法”服务器上的字节顺序方向相反，所以必须使用此信息以允许您解释数据流中的数字数据。

如果处理数据的应用程序识别它在一种类型的计算机硬件（如大尾数法计算机）上运行，而事件数据是在另一种类型的计算机硬件（如小尾数法计算机）上生成的，那么监视应用程序必须先使数字数据字段的字节反向，然后再解释它们。否则不需要进行字节定向。

此元素可设置为下列其中一种 API 常量:

- SQLM_BIG_ENDIAN

- SQLM_LITTLE_ENDIAN

cat_cache_inserts - "目录高速缓存插入数"

系统尝试将表描述符或权限信息插入到目录高速缓存中的次数。

元素标识

cat_cache_inserts

元素类型

计数器

表 199. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 200. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 通过与"目录高速缓存查询"一起使用，可借助以下公式来计算目录高速缓存命中率:

```
1 - (目录高速缓存插入数 / 目录高速缓存查询数)
```

有关使用此元素的更多信息，请参阅 cat_cache_lookups。

cat_cache_lookups - "目录高速缓存查询数"

为获取表描述符信息或权限信息而引用目录高速缓存的次数。

元素标识

cat_cache_lookups

元素类型

计数器

表 201. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 202. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

表 202. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 此元素包括对目录高速缓存的成功访问和不成功访问。每当出现下列情况，就会引用目录高速缓存:

- 在编译 SQL 语句期间处理表、视图或别名
- 访问数据库权限信息
- 在编译 SQL 语句期间处理例程

要计算目录高速缓存命中率，请使用以下公式:

```
(1 - (cat_cache_inserts / cat_cache_lookups))
```

来指示目录高速缓存避免目录访问的效果如何。如果比率很高（超过 0.8），那么表示高速缓存确实起到作用。如果比率偏低，那么表示应提高 *catalogcache_sz*。首次连接至数据库之后应该有较大的比率。

执行涉及表、视图或别名的数据定义语言（DDL）SQL 语句会从目录高速缓存中除去该对象的表描述符信息，从而导致这些信息在下一次引用时重新插入。此外，用于数据库权限和例程执行特权的 GRANT 和 REVOKE 语句会从目录高速缓存中除去主题权限信息。因此，大量使用 DDL 语句和 GRANT/REVOKE 语句也会提高该比率。

有关目录高速缓存大小配置参数的更多信息，请参阅《*管理指南*》。

cat_cache_overflows -“目录高速缓存溢出数”

目录高速缓存溢出其分配内存边界的次数。

表 203. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 204. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法

将此元素与 **cat_cache_size_top** 监视元素配合使用来确定是否需要增加目录高速缓存的大小以避免溢出。

目录高速缓存空间是通过除去表、视图或别名的表描述符信息或当前未被任何事务使用的权限信息来回收的。

如果 **cat_cache_overflows** 监视元素的值很大，那么相对工作负载而言目录高速缓存可能会太小。扩大目录高速缓存可以改进性能。如果工作负载包括的事务将编译大量 SQL 语句，而这些语句又引用单个工作单元中的多个表、视图、别名、用户定义的函数或存储过程，那么在单个事务中编译较少的 SQL 语句可以改进目录高速缓存的性能。或者，如果工作负载包括绑定包含许多 SQL 语句的程序包，而这些语句又引用多个表、视图、别名、用户定义的函数或存储过程，那么可以尝试分割程序包以使这些程序包包括较少的 SQL 语句从而改进性能。

cat_cache_size_top - "目录高速缓存高水位标记"监视元素

目录高速缓存达到最大逻辑大小。

表 205. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 206. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

此元素指示在激活数据库之后，对其运行的工作负载在逻辑上所需的目录高速缓存的最大字节数。

目录高速缓存通过逻辑大小进行管理，此大小不包括内存管理开销。数据库快照中的 **pool_watermark** 元素提供了目录高速缓存所使用的内存的物理高水位标记值。执行目录高速缓存监视和调整工作时，应该使用逻辑大小而不是物理大小。

如果目录高速缓存溢出，那么表示此元素包含溢出期间目录高速缓存达到的最大大小。检查 **cat_cache_overflows** 监视元素以确定是否发生此类情况。

可通过以下公式来确定工作负载需要的目录高速缓存最小大小:

```
最大目录高速缓存大小 / 4096
```

通过将结果四舍五入为整数，指示为避免溢出目录高速缓存最少需要的页数（每页 4K 字节）。

catalog_node - "目录节点号"

存储数据库目录表的节点的编号。

元素标识

catalog_node

元素类型

信息

表 207. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 208. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 目录节点是存储所有系统目录表的节点。对系统目录表的所有访问都必须通过此节点进行。

catalog_node_name -“目录节点网络名”

目录节点的网络名。

元素标识

catalog_node_name

元素类型

信息

表 209. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 210. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 使用此元素来确定数据库的位置。

ch_free -“当前可用的通道数”

此元素指示当前可用的节点间通信信道数。

表 211. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法 将当前可用的通信信道数与 *fcm_num_channels* 配置参数一起使用来确定当前连接条目利用率。可使用此信息来调整 *fcm_num_channels*。

ch_free_bottom -“最低可用通道数”

处理期间达到的最低可用节点间通信信道数。

表 212. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法 将此元素与 *fcm_num_channels* 配置参数一起使用来确定最大连接条目利用率。

client_acctng -"客户机记帐字符串"监视元素

如果在此连接中发出了 sqleseti API，那么此项是为了用于记录和诊断而传递至目标数据库的数据。 此连接、工作单元或活动的 CLIENT_ACCTNG 专用寄存器的当前值。

此监视元素与 **tpmon_acc_str** 监视元素同义。**client_acctng** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_acc_str** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 213. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	始终收集
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 214. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-

用法

此元素用于问题确定和记帐目的。

client_applname -"客户机应用程序名称"监视元素

如果在此连接中发出 sqleseti API，那么此项标识执行事务时出现的服务器事务程序问题。 此连接、工作单元或活动的 CLIENT_APPLNAME 专用寄存器的当前值。

此监视元素与 **tpmon_client_app** 监视元素同义。**client_applname** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_client_app** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 215. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_CONNECTION 表函数 - 获取连接度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集

表 216. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-

用法

此元素用于问题确定和记帐目的。

client_db_alias - “应用程序使用的数据库别名”

由要连接至数据库的应用程序提供的数据库别名。

元素标识

client_db_alias

元素类型

信息

表 217. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本

表 218. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

用法 此元素可用来标识应用程序正在访问的实际数据库。此名称与 *db_name* 之间的映射可通过在客户机节点和数据库管理器服务器节点上使用数据库目录来实现。

这是在发出数据库连接请求的数据库管理器中定义的别名。

此元素还可用于帮助您确定认证类型，原因是不同数据库别名可能具有不同的认证类型。

client_idle_wait_time - "客户机空闲等待时间"监视元素

此监视元素用于记录等待客户机发送下一个请求时耗用的时间。此值以毫秒计。

表 219. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 220. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定处理请求时（而不是等待来自客户机的请求时）耗用的时间。如果客户机空闲时间过长，那么表明客户机端（而不是服务器端）存在需要解决的性能问题。

client_pid – “客户机进程标识”

建立与数据库的连接的客户机应用程序的进程标识。

表 221. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 222. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
连接	event_connheader	-

用法

可使用此元素使 CPU 和 I/O 时间之类的监视器信息与客户机应用程序相关。

如果是 DRDA AS 连接，那么此元素将设置为 0。

client_platform – “客户机操作平台”

运行客户机应用程序的操作系统。

表 223. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 224. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
连接	event_connheader	-

用法

此元素可用于远程应用程序的问题确定。可在头文件 sqlmon.h 中找到此字段的值。

client_prdid – “客户机产品和版本标识”监视元素

正在客户机上运行的产品和版本。

表 225. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本

表 225. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

表 226. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
连接	event_connheader	-

用法

可使用此元素来标识IBM® 数据服务器客户机的产品和代码版本。其格式为 PPPVVRRM，其中:

- PPP 标识产品，对于 DB2 产品为“SQL”。
- VV 标识两位版本号（版本只有一位时则高位为 0）
- RR 标识两位发行版号（发行版只有一位时高位为 0）
- M 标识 1 个字符的修改级别（0-9 或 A-Z）。

client_protocol – “客户机通信协议”

客户机应用程序用于与服务器通信的通信协议。

表 227. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 228. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
连接	event_connheader	-

用法

此元素可用于远程应用程序的问题确定。此字段的值是:

SQLM_PROT_UNKNOWN
客户机使用未知协议进行通信。仅当将来客户机与较早级别的服务器连接时，才返回此值。

SQLM_PROT_LOCAL
客户机与服务器在同一节点上运行，未使用任何通信协议。

SQLM_PROT_TCPIP
TCP/IP

client_userid – "客户机用户标识"监视元素

如果使用 sqleseti API，那么此项是由事务管理器生成的并且提供给服务器的客户机用户标识。 此连接、工作单元或活动的 CLIENT_USERID 专用寄存器的当前值。

此监视元素与 **tpmon_client_userid** 监视元素同义。**client_userid** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_client_userid** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 229. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 – 获取连接度量值	始终收集
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 230. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-

用法

通过在应用程序服务器或事务处理监视器环境中使用此元素，可以标识为其执行事务的最终用户。

client_wrkstnname – "客户机工作站名称"监视元素

如果在此连接中发出了 sqleseti API，那么此项标识客户机的系统或工作站（如 CICS EITERMID）。此连接、工作单元或活动的 CLIENT_WRKSTNNAME 专用寄存器的当前值。

此监视元素与 **tpmon_client_wkstn** 监视元素同义。**client_wrkstnname** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_client_wkstn** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 231. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	始终收集
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 232. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-

用法

使用此元素并借助节点标识、终端标识或类似标识来标识用户的机器。

codepage_id - “应用程序使用的代码页标识”

代码页标识。

表 233. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 234. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-
连接	event_connheader	-

用法 对于快照监视器数据，这是启动被监视应用程序的分区的代码页。此标识可用于远程应用程序的问题确定。可以使用此信息来确保应用程序代码页与数据库代码页（或者，对于 DRDA 主机数据库则为主机 CCSID）之间的数据转换是受支持的。有关受支持代码页的信息，请参阅*《管理指南》*。

对于事件监视器数据，这是对其收集事件数据的数据库的代码页。可使用此元素来确定事件监视器应用程序是否在与数据库使用的代码页不同的代码页中运

行。事件监视器写下的数据使用数据库代码页。如果事件监视器应用程序使用另一代码页，那么可能需要执行一些字符转换来使数据可读。

comm_private_mem - "已落实的专用内存"

目前数据库管理器的实例在获取快照时已落实的专用内存量。返回的 comm_private_mem 值仅在 Windows 操作系统上有用。

表 235. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

commit_sql_stmts - "尝试的落实语句数"

尝试的 SQL COMMIT 语句总数。

表 236. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器复位。

表 237. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 如果监视期间此计数器数字很少变化，那么指示应用程序执行的落实操作较少，这可能导致登录和数据并行性出现问题。

还可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将仅包括发生以下情况之后出现的工作单元:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

此计算可在数据库级别或应用程序级别完成。

comp_env_desc - "编译环境"监视元素

此元素存储关于编译 SQL 语句时使用的编译环境的信息。

表 238. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 239. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-
活动	event_activitystmt	-

用法 此监视元素将编译环境描述存储在二进制大对象中。要以可读格式查看此信息，请使用 COMPILATION_ENV 表函数。

可提供此元素作为 COMPILATION_ENV 表函数的输入，或者作为 SET COMPILATION ENVIRONMENT SQL 语句的输入。

completion_status - "完成状态"监视元素

工作单元的状态。

表 240. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-

用法

使用此元素来确定工作单元是否因为死锁或异常终止而结束。 sqllib/misc/DB2EvmonUOW.xsd 文件列示了可能的值:

- UNKNOWN
- COMMIT
- ROLLBACK
- GLOBAL_COMMIT
- GLOBAL_ROLLBACK
- XA_END
- XA_PREPARE

con_elapsed_time - "最新连接耗用时间"

连接最新与此主机数据库断开连接的 DCS 应用程序所耗用的时间。

元素标识

con_elapsed_time

元素类型

时间

表 241. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

用法 将此元素用作应用程序保持与主机数据库的连接的时间长度的指示符。

con_local_dbases - "带有当前连接的本地数据库"

连接了应用程序的本地数据库的数目。

表 242. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 此值指示在数据库级别收集数据时您可以期望的数据库信息记录数。

应用程序可在本地或远程运行，并且可能执行也可能不执行数据库管理器中的工作单元。

con_response_time - "连接的最新响应时间"

对于连接至此数据库的最新 DCS 应用程序，此项为连接处理开始与实际建立连接之间所耗用的时间。

元素标识

con_response_time

元素类型

时间

表 243. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

用法 将此元素用作当前应用程序连接至特定主机数据库所花时间的指示符。

concurrent_act_top - "并行活动顶部"监视元素

自最后一次复位以后服务子类中并发活动的高水位标记（在任何嵌套级别）。

表 244. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此元素来了解在收集的时间间隔内，服务子类的分区上达到的最高活动（包括嵌套活动）并行数。

concurrent_connection_top －“并行连接顶部”监视元素

自最后一次复位后此服务类中的并行协调程序连接的高水位标记。在同一超类的每个子类中，此字段的值都相同。

表 245. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

通过显示当前高水位标记所在，此元素可帮助确定在何处对连接并行性设置阈值。此元素还可帮助验证该阈值是否正确配置且正常工作。

concurrent_wlo_act_top －“并行 WLO 活动顶部”监视元素

自最后一次复位后此工作负载的任何项的并行活动高水位标记（在任何嵌套级别）。

表 246. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-

用法

使用此元素来了解在收集的时间间隔内，此工作负载的任何顶的分区上达到的最高并行活动数。

concurrent_wlo_top －“并行工作负载项顶部”监视元素

自最后一次复位后工作负载并行项的高水位标记。

表 247. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-
统计信息	event_scstats	-

用法

使用此元素来了解在收集的时间间隔内，工作负载的分区上达到的最高工作负载项并行数。

concurrentdbcoordactivities_db_threshold_id – “并发数据库协调程序活动数据库阈值标识”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值的标识。

表 248. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值（如果有的话）。

concurrentdbcoordactivities_db_threshold_queued – “已由并发数据库协调程序活动数据库阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES 数据库阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 249. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值是否已对此活动进行排队。

concurrentdbcoordactivities_db_threshold_value – “并发数据库协调程序活动数据库阈值”监视元素

此监视元素返回应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值的上限。

表 250. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值（如果有的话）。

concurrentdbcoordactivities_db_threshold_violated -“违反并发数据库协调程序活动数据库阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES 数据库阈值。“No”表明此活动尚未违反该阈值。

表 251. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值。

concurrentdbcoordactivities_subclass_threshold_id -“并发数据库协调程序活动服务子类阈值标识”监视元素

此监视元素返回应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值的标识。

表 252. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值（如果有的话）。

concurrentdbcoordactivities_subclass_ threshold_queued -“已由并发数据库协调程序活动服务子类阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES 服务子类阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 253. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值是否已对此活动进行排队。

concurrentdbcoordactivities_subclass_ threshold_value - “并发数据库协调程序活动服务子类阈值”监视元素

此监视元素返回应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值的上限。

表 254. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值（如果有的话）。

concurrentdbcoordactivities_subclass_ threshold_violated - “违反并发数据库协调程序活动服务子类阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES 服务子类阈值。“No”表明此活动尚未违反该阈值。

表 255. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值。

concurrentdbcoordactivities_superclass_ threshold_id -“并发数据库协调程序活动服务超类阈值标识”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES_SUPERCLASS 阈值的标识。

表 256. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值（如果有的话）。

concurrentdbcoordactivities_superclass_ threshold_queued -“已由并发数据库协调程序活动服务超类阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES 服务超类阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 257. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值是否已对此活动进行排队。

concurrentdbcoordactivities_superclass_ threshold_value -“并发数据库协调程序活动服务超类阈值”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值的上限。

表 258. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值（如果有的话）。

concurrentdbcoordactivities_superclass_ threshold_violated -“违反并发数据库协调程序活动服务超类阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES 服务超类阈值。“No”表明此活动尚未违反该阈值。

表 259. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值。

concurrentdbcoordactivities_work_action_set_ threshold_id -“并发数据库协调程序活动工作操作集阈值标识”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作操作集阈值的标识。

表 260. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作操作集阈值（如果有的话）。

concurrentdbcoordactivities_work_action_set_ threshold_queued -“已由并发数据库协调程序活动工作操作集阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 261. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值是否已对此活动进行排队。

concurrentdbcoordactivities_work_action_set_ threshold_value - “并发数据库协调程序活动工作操作集阈值”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值的上限。

表 262. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK 阈值（如果有的话）。

concurrentdbcoordactivities_work_action_set_ threshold_violated - “违反并发数据库协调程序活动工作操作集阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值。“No”表明此活动尚未违反该阈值。

表 263. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值。

conn_complete_time - “连接请求完成时间戳记”

对连接请求授权的日期和时间。

元素标识

conn_complete_time

元素类型

时间戳记

表 264. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

用法 使用此元素来确定对数据库的连接请求授权的时间。

conn_time - "数据库连接时间"监视元素

在数据库级别第一次连接至数据库的日期和时间，或者发出激活数据库命令的日期和时间。

表 265. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
数据库	event_dbheader	-
连接	event_connheader	-

用法

通过将此元素与 **disconn_time** 监视元素配合使用，可以计算出现以下情况之后耗用的时间:

- 数据库处于活动状态（用于数据库级别的信息）。
- 连接处于活动状态（用于连接级别的信息）。

connection_status - "连接状态"

此元素指示发出 GET SNAPSHOT 命令的节点与 *db2nodes.cfg* 文件中列示的其他节点间的通信连接状态。

元素标识

connection_status

元素类型

信息

表 266. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

用法 连接值包括:

SQLM_FCM_CONNECT_INACTIVE
当前无连接

SQLM_FCM_CONNECT_ACTIVE
连接处于活动状态

SQLM_FCM_CONNECT_CONGESTED
连接已阻塞

两个节点可以处于活动状态，但除非节点之间存在通信，否则两个节点之间的通信连接仍然不活动。

connections_top - "最大并行连接数"

自数据库激活后同时与数据库进行的连接的最大数目。

元素标识

connections_top

元素类型

水位标记

表 267. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 268. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 可使用此元素来评估 *maxappls* 配置参数的设置。

如果此元素的值与 *maxappls* 参数相同，那么可能拒绝了某些数据库连接请求，原因是 *maxappls* 限制了允许的数据库连接数。

可使用以下公式来计算获取快照时的当前连接数:

```
rem_cons_in + local_cons
```

consistency_token - "程序包一致性标记"监视元素

对于给定程序包名称和创建者，可以有（从 DB2 版本 8 开始）多个版本。程序包一致性标记帮助标识包含 SQL 当前执行的程序包的版本。

表 269. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 270. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
语句	event_stmt	-

用法

可使用此元素来帮助标识程序包和正在执行的 SQL 语句。

container_accessible - "容器可访问"监视元素

此元素指示容器是否可访问。值为 1 表示"是"，值为 0 表示"否"。

表 271. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 272. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法 通过将此元素与 **container_id**、**container_name**、**container_type**、**container_total_pages**、**container_usable_pages** 和 **container_stripe_set** 元素配合使用，可以描述容器。

container_id - "容器标识"监视元素

在表空间中唯一定义容器的整数。

表 273. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取容器度量值	DATA OBJECT METRICS BASE

表 274. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法 此元素可与元素 container_name、container_type、container_total_pages、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

container_name - "容器名称"监视元素

容器的名称。

表 275. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取容器度量值	DATA OBJECT METRICS BASE

表 276. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法 此元素可与元素 container_id、container_type、container_total_pages、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

container_stripe_set - "容器分割集"监视元素

容器所属的分割集。

表 277. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 278. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法

通过将此监视元素与 **container_id**、**container_name**、**container_type**、**container_total_pages**、**container_usable_pages** 和 **container_accessible** 元素配合使用，可以描述容器。此元素仅适用于 DMS 表空间。

container_total_pages - "容器中的总页数"监视元素

容器占用的总页数。

表 279. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 280. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法 此元素可与元素 container_id、container_name、container_type、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

container_type - "容器类型"监视元素

容器的类型。

表 281. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取容器度量值	DATA OBJECT METRICS BASE

表 282. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法

此元素返回容器的类型，它可能是目录路径（仅适用于 SMS）、文件（适用于 DMS）或原始设备（适用于 DMS）。通过将此监视元素与 **container_id**、**container_name**、**container_type**、**container_total_pages**、**container_usable_pages** 和 **container_accessible** 元素配合使用，可以描述容器。

此监视元素的有效值由 sqlutil.h 文件定义。

container_usable_pages - "容器中的可用页数"监视元素

容器中的可用页的总数。

表 283. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 284. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法 此元素可与元素 container_id、container_name、container_type、container_total_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。对于 SMS 表空间，此值与 container_total_pages 相同。

coord_act_aborted_total - "异常终止的协调程序活动总数"监视元素

自最后一次复位后在任何嵌套级别完成时出错的协调程序活动总数。对于服务类，此值在活动完成时更新。对于工作负载，此值在其工作单元结束时由每个工作负载项更新。

对于服务类而言，如果在活动异常终止前通过 REMAP ACTIVITY 操作将其重新映射到另一个子类，那么此活动将仅计入在其中异常终止此活动的子类的总数。

表 285. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-

用法

使用此元素来了解系统上的活动是否成功完成。活动可能因取消、错误或反应性阈值而异常终止。

coord_act_completed_total - "完成的协调程序活动总数"监视元素

自最后一次复位后在任何嵌套级别成功完成的协调程序活动总数。对于服务类，此值在活动完成时更新。对于工作负载，此值在其工作单元结束时由每个工作负载项更新。

对于服务类而言，如果在活动完成前通过 REMAP ACTIVITY 操作将其重新映射到另一个子类，那么此活动将仅计入在其中完成此活动的子类的总数。

表 286. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-
统计信息	event_scstats	-

用法

此元素可用来确定系统中活动的吞吐量或用来帮助计算多分区间的平均活动生存期。

coord_act_est_cost_avg - "平均协调程序活动估计成本"监视元素

自最后一次复位以来，在嵌套级别 0 处的协调程序 DML 活动估计成本的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA EXTENDED 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。单位为毫秒。

对于服务类而言，活动的估计成本只计入活动从其中进入系统的服务子类。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动重新映射到的服务子类的 coord_act_est_cost_avg 平均值不受影响。

表 287. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定自上次统计信息复位以来嵌套级别 0 处的协调程序 DML 活动（与已完成或中止的此服务子类、工作负载或工作类相关联）的估计成本的算术平均值。

另外，还可以使用此平均值来确定用于活动估计成本直方图的直方图模板是否合适。根据活动估计成本直方图计算平均活动估计成本。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动估计成本直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_exec_time_avg - “平均协调程序活动执行时间”监视元素

自最后一次复位以来，在嵌套级别 0 处的协调程序活动执行时间的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动所映射到但未完成的服务子类的 coord_act_exec_time_avg 平均值不受影响。

表 288. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定与已完成或中止的服务子类、工作负载或工作类相关联的协调程序活动执行时间的算术平均值。

另外，还可以使用此平均值来确定用于活动执行时间直方图的直方图模板是否合适。根据活动执行时间直方图来计算平均活动执行时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动执行时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_interarrival_time_avg - “平均协调程序活动到达时间”监视元素

自最后一次复位以来，在嵌套级别 0 处的协调程序活动到达间隔时间的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA EXTENDED 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，将对活动进入系统时所借助于的服务子类计算到达之间时间平均值。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动重新映射到的服务子类的 coord_act_interarrival_time_avg 不受影响。

表 289. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定在与此服务子类、工作负载或工作类相关联的嵌套级别 0 处协调程序活动到达间隔时间的算术平均值。

到达间隔时间可用于确定到达速率，即到达间隔时间的倒数。另外，还可以使用此平均值来确定用于活动到达间隔时间直方图的直方图模板是否合适。根据活动到达间隔时间直方图来计算平均活动到达间隔时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动到达间隔时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_lifetime_avg – “平均协调程序活动生存期”监视元素

自从上次复位以来，在嵌套级别 0 与此服务子类、工作负载或工作类相关联的协调程序活动的生存期算术平均值。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，只有完成活动的最终服务类的 coord_act_lifetime_avg 平均值受影响。

表 290. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定与已完成或中止的服务子类、工作负载或工作类相关联的协调程序活动生存期的算术平均值。

另外，还可以使用此统计信息来确定用于活动生存期直方图的直方图模板是否合适。根据活动生存期直方图计算平均活动生存期。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动生存期直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_lifetime_top - "协调程序活动生存期顶部"监视元素

在所有嵌套级别计量的协调程序活动生存期的高水位标记。单位为毫秒。对于服务类，当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。

在同时使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动的情况下，要有效地将此统计信息与服务类配合使用，必须将任何给定服务子类的 coord_act_lifetime_top 高水位标记与相同重新映射阈值所影响的其他子类的该高水位标记进行聚集。这是因为，活动在被重新映射阈值重新映射到另一服务子类后将完成，该活动被重新映射前在其他服务子类中的耗用时间只计入在其中完成该活动的服务类。

表 291. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wcstats	-
统计信息	event_scstats	-
统计信息	event_wlstats	-

用法

此元素可用来帮助确定活动生存期的域值是否有效，还可以帮助确定如何配置这些阈值。

coord_member - "协调程序成员"监视元素

给定工作单元或工作负载的协调成员。

表 292. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

coord_act_queue_time_avg - "平均协调程序活动队列时间"监视元素

自最后一次复位以来，在嵌套级别 0 处的协调程序 DML 活动估计成本的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，队列时间只计入在其中完成或中止该活动的服务子类。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动所映射到但未完成的服务子类的 coord_act_queue_time_avg 平均值不受影响。

元素标识

coord_act_queue_time_avg

元素类型

信息

-->

表 293. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定与已完成或中止的服务子类、工作负载或工作类相关联的协调程序活动队列时间的算术平均值。

另外，还可以使用此统计信息来确定用于活动队列时间直方图的直方图模板是否合适。根据活动队列时间直方图来计算平均活动队列时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动队列时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_rejected_total - “被拒绝的协调程序活动总数”监视元素

自最后一次复位后，在任何嵌套级别被拒绝而未被允许执行的协调程序活动总数。此计数器在活动被预测性阈值或阻止执行的工作操作阻止执行时更新。对于服务类，此值在活动完成时更新。对于工作负载，此值在其工作单元结束时由每个工作负载项更新。

表 294. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-

用法

此元素可用来帮助确定阻止执行的预测性阈值或工作操作是否有效，以及它们的限制是否太严格。

coord_agent_pid - “协调代理程序标识”监视元素

应用程序的协调代理程序的引擎可分派单元（EDU）标识。除在 Linux 操作系统上以外，EDU 标识与线程标识映射。在 Linux 操作系统上，EDU 标识是 DB2 生成的唯一标识。

表 295. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

表 296. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

用法

可以使用此元素来将数据库系统监视器信息链接至其他诊断信息源，如系统跟踪。

coord_agents_top -“最大协调代理程序数”

同时工作的最大协调代理程序数。

表 297. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
数据库	dbase	基本

用法

如果协调代理程序的峰值数目显示此节点的工作负载过高，可通过更改 **max_coordagents** 配置参数来降低此上限。

coord_node -“协调节点”

在多节点系统中，这是应用程序连接至实例的节点的节点号。

表 298. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 299. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 每个连接的应用程序都有一个协调程序节点为其提供服务。

coord_partition_num -“协调程序分区号”监视元素

工作单元或活动的协调程序分区。在多分区系统中，应用程序将从协调程序分区中连接到数据库。

表 300. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
活动	event_activity	-
阈值违例	event_thresholdviolations	-

用法

对于那些在除协调程序分区以外的分区中有记录的活动或工作单元，此元素用来标识它们的协调程序分区。

corr_token - "DRDA 关联标记"

DRDA AS 关联标记。

表 301. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本

表 302. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

用法 DRDA 关联标记用于关联应用程序服务器与应用程序请求器之间的处理。它是出现错误时转储至日志的标识，可使用该标识来标识存在错误的对话。在某些情况下，它将成为对话的 LUWID。

如果通信未使用 DRDA，那么此元素返回 *appl_id*（请参阅 appl_id）。

如果要使用数据库系统监视器 API，那么注意 API 常量 SQLM_APPLID_SZ 用于定义此元素的长度。

cost_estimate_top - "成本估计顶部"监视元素

服务子类或工作类中所有嵌套级别的 DML 活动估计成本的高水位标记。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。

对于服务类而言，DML 活动的估计成本只计入活动从其中进入系统的服务子类。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动重新映射到的服务子类的 cost_estimate_top 不受影响。

表 303. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-

表 303. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-

用法

使用此元素来确定在收集时间间隔内，分区中服务类、工作负载或工作类达到的最高 DML 活动估计成本。

count - "事件监视器溢出数"

发生的连续溢出数。

元素标识
计数

元素类型
计数器

表 304. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

用法 可使用此元素来了解丢失的监视器数据量。

事件监视器对一组连续溢出发送一个溢出记录。

cputime_threshold_id - "CPU 时间阈值标识"监视元素

应用于此活动的 CPUTIME 阈值的标识。

表 305. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CPUTIME 阈值（如果有的话）。

cputime_threshold_value - "CPU 时间阈值"监视元素

应用于此活动的 CPUTIME 阈值的上限。

表 306. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CPUTIME 阈值（如果有的话）。

cputime_threshold_violated －“违反 CPU 时间阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CPUTIME 阈值。“No”表明此活动尚未违反该阈值。

表 307. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 CPUTIME 阈值。

cputimeinsc_threshold_id －“服务类中 CPU 时间阈值标识”监视元素

应用于此活动的 CPUTIMEINSC 阈值的标识。

表 308. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CPUTIMEINSC 阈值（如果有的话）。

cputimeinsc_threshold_value －“服务类中 CPU 时间阈值”监视元素

应用于此活动的 CPUTIMEINSC 阈值的上限。

表 309. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 CPUTIMEINSC 阈值（如果有的话）。

cputimeinsc_threshold_violated – “违反服务类中 CPU 时间阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CPUTIMEINSC 阈值。“No”表明此活动尚未违反该阈值。

表 310. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 CPUTIMEINSC 阈值。

create_nickname – “创建昵称数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位以后（取较晚者），联合服务器代表任何应用程序对驻留在此数据源上的对象创建昵称的总次数。

表 311. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法

使用此元素来确定此联合服务器实例或应用程序对此数据源执行的 CREATE NICKNAME 活动的级别。CREATE NICKNAME 处理将导致对数据源目录运行多个查询；因此，如果此元素的值很高，那么应确定原因并在可能的情况下限制此活动的执行。

create_nickname_time – “创建昵称响应时间”

此元素包含此数据源处理 CREATE NICKNAME 语句所花的总时间（以毫秒计），这些 CREATE NICKNAME 语句来自在此联合服务器实例上运行的所有应用程序或单个应用程序。响应时间自联合服务器实例启动或数据库监视计数器最后一次复位（取较晚者）起计。 the latest. 响应时间是以联合服务器开始从数据源接收信息以处理 CREATE NICKNAME 语句的时间与检索数据源中的所有必需数据所花时间之差量度的。

表 312. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

用法 使用此元素来确定用于为此数据源创建昵称所花的实际时间。

creator －“应用程序创建者”

预编译应用程序的用户的授权标识。

表 313. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 314. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
语句	event_stmt	-
活动	event_activitystmt	-

用法 将此元素与目录中的程序包段信息的 CREATOR 列一起使用来帮助标识正在处理的 SQL 语句。

如果设置了 CURRENT PACKAGE PATH 专用寄存器，那么 *creator* 值会在 SQL 语句有效期限内反映不同的值。如果快照或事件监视器记录是在解析 PACKAGE PATH 之前获取的，那么 *creator* 值将反映客户机请求中体现的值。如果快照或事件监视器记录是在解析 PACKAGE PATH 之后获取的，那么 *creator* 值将反映已解析程序包的创建者。已解析程序包将是这样一个程序包，其 *creator* 值在 CURRENT PACKAGE PATH SPECIAL REGISTER 中最早出现，并且其程序包名称和唯一标识与客户机请求的程序包名称和唯一标识相匹配。

current_active_log －“当前活动日志文件编号”

DB2 数据库系统当前写入的活动日志文件的编号。

表 315. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 316. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *first_active_log* 和 *last_active_log* 元素一起使用以确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

current_archive_log – “当前归档日志文件编号”

DB2 数据库系统当前归档的日志文件的文件号。如果 DB2 数据库系统未归档日志文件，那么此元素的值为 SQLM_LOGFILE_NUM_UNKNOWN。

表 317. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 318. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 使用此元素来确定归档日志文件时是否存在问题。这类问题包括:

- 归档介质速度太慢
- 归档介质不可用

current_extent – “当前正在移动的扩展数据块”监视元素

表空间重新平衡过程当前正在移动的扩展数据块的数字标识。

表 319. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_EXTENT_MOVEMENT_STATUS – 获取扩展数据块移动进度状态度量值	始终收集

cursor_name – “游标名称”

对应此 SQL 语句的游标的名称。

元素标识

cursor_name

元素类型

信息

表 320. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 321. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-
语句	event_stmt	-

用法 可使用此元素来标识正在处理的 SQL 语句。将在 SQL SELECT 语句的 OPEN、FETCH、CLOSE 和 PREPARE 上使用此名称。如果未使用游标，那么此字段将为空白。

data_object_pages – “数据对象页数”

表消耗的磁盘页数。此大小仅表示基本表大小。索引对象、LOB 数据和长数据消耗的空间分别由 *index_object_pages*、*lob_object_pages* 和 *long_object_pages* 报告。

表 322. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 323. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法 此元素提供了一种机制，可用来查看特定表消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内表的增长率。

data_partition_id – “数据分区标识”监视元素

与返回的信息相对应的数据分区的标识。

表 324. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 – 获取表度量值	始终收集
MON_GET_INDEX 表函数 – 获取索引度量值	始终收集

表 325. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
锁定	lock	锁定
锁定	lock_wait	锁定

表 326. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
死锁	lock	-

用法

此元素仅适用于分区表和分区索引。否则，此监视元素的值是 NULL。

当返回锁定级别信息时，值 -1 代表一个控制对整个表进行访问的锁定。

datasource_name – “数据源名称”

此元素包含其远程访问信息由联合服务器显示的数据源的名称。此元素对应于 SYSCAT.SERVERS 中的 ’SERVER’ 列。

元素标识

datasource_name

元素类型

信息

表 327. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

用法 使用此元素来标识已收集并正在返回其访问信息的数据源。

db2_status – “DB2 实例的状态”

数据库管理器的实例的当前状态。

表 328. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 可使用此元素来确定数据库管理器实例的状态。

此元素的值包括:

API 常量	值	描述
SQLM_DB2_ACTIVE	0	数据库管理器实例是活动的。
SQLM_DB2_QUIESCE_PEND	1	实例和实例中的数据库处于停顿 – 暂挂状态。不允许新建与任何实例数据库的连接，也不能启动新的工作单元。根据停顿请求，将允许活动工作单元完成或立即回滚。
SQLM_DB2_QUIESCED	2	实例和实例中的数据库已停顿。不允许新建与任何实例数据库的连接，也不能启动新的工作单元。

db2start_time – “启动数据库管理器时间戳记”

使用 db2start 命令启动数据库管理器的日期和时间。

元素标识

db2start_time

元素类型

时间戳记

表 329. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 此元素可与 *time_stamp* 监视元素配合使用，以计算自数据库管理器启动直到获取快照所耗用的时间。

db_conn_time - "数据库激活时间戳记"监视元素

在数据库级别第一次连接至数据库的日期和时间，或者发出激活数据库命令的日期和时间。

表 330. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	时间戳记
表空间	tablespace_list	缓冲池，时间戳记
表	table_list	时间戳记

表 331. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-

用法

将此元素与 **disconn_time** 监视元素配合使用以计算总连接时间。

db_heap_top - "分配的最大数据库堆"

此元素将保留以获取 DB2 版本兼容性。它现在量度内存使用情况，但并非由数据库堆专用。

注: 从 DB2 版本 9.5 开始，不推荐使用 **db_heap_top** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 332. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 333. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

db_location - "数据库位置"

与应用程序相关的数据库的位置。

表 334. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 确定有关获取快照的应用程序的数据库服务器的相对位置。值包括:

- SQLM_LOCAL
- SQLM_REMOTE

db_name - "数据库名称"

对其收集信息或应用程序连接至的数据库的真实名称。这是创建时给定的数据库名称。

表 335. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl_id_info	基本
应用程序	appl_remote	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁定	db_lock_list	基本
动态 SQL	dynsql_list	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

表 336. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	-

用法 可使用此元素来标识应用数据的特定数据库。

对于未使用 DB2 Connect 连接至主机或 System i® 数据库服务器的应用程序，可将此元素与 **db_path** 监视元素配合使用来唯一标识该数据库，并协助使监视器提供的信息的不同级别相关。

db_path - "数据库路径"

数据库在被监视系统上的存储位置的完整路径。

表 337. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_id_info	基本
表空间	tablespace_list	缓冲池

表 337. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池
表	table_list	表
锁定	db_lock_list	基本
动态 SQL	dynsql_list	基本

表 338. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	-

用法 此元素可与 *db_name* 监视元素配合使用以标识应用数据的特定数据库。

db_status - "数据库状态"

数据库的当前状态。

表 339. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 可使用此元素来确定数据库的状态。

此字段的值是:

API 常量	值	描述
SQLM_DB_ACTIVE	0	数据库是活动的。
SQLM_DB_QUIESCE_PEND	1	数据库处于停顿 - 暂挂状态。不允许新建与数据库的连接，也不能启动新的工作单元。根据停顿请求，将允许活动工作单元完成或立即回滚。
SQLM_DB_QUIESCED	2	数据库已停顿。不允许新建与数据库的连接，也不能启动新的工作单元。
SQLM_DB_ROLLFWD	3	数据库正在进行前滚。

db_storage_path - "自动存储器路径"监视元素

此元素显示数据库用于放置自动存储器表空间的位置的完整路径。一个数据库可以有 0 个或多个相关联的存储器路径。

表 340. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	基本

用法

将此元素与 **num_db_storage_paths** 监视元素配合使用来标识与此数据库相关联的存储器路径。

db_storage_path_state - "存储器路径状态"监视元素

自动存储器路径状态指示存储器路径是否正被数据库使用。

表 341. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	基本

用法

使用此监视元素来确定存储器路径是否正被数据库使用。可能的值如下所示:

NOT_IN_USE
在指定的数据库分区中，没有任何表空间正在使用此存储器路径。

IN_USE
在指定的数据库分区中，有一些表空间正在使用此存储器路径。

DROP_PENDING
此存储器路径已被删除，但某些表空间仍在使用此路径。以物理方式从数据库中删除存储器路径之前，所有表空间都必须停止使用这些路径。要停止使用已删除的存储器路径，请删除该表空间，或者使用 ALTER TABLESPACE 语句的 REBALANCE 子句对该表空间进行重新平衡。

db_storage_path_with_dpe - "包含数据库分区表达式的存储器路径"监视元素

包含尚未进行求值的数据库分区表达式的自动存储器路径。

表 342. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	基本

用法

当存储器路径包含数据库分区表达式时，请使用此监视元素来确定 CREATE DATABASE 命令或 ALTER DATABASE 语句对数据库指定的存储器路径。

如果存储器路径未包含数据库分区表达式，那么此监视元素将返回空值。

db_work_action_set_id - "数据库工作操作集标识"监视元素

如果此活动已划分到数据库作用域的某个工作类，那么此监视元素显示与该工作类所属的工作类集相关联的工作操作集的标识。否则，此监视元素将显示值 0。

表 343. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE（在 XML 文档 DETAILS 中报告）	始终收集

表 344. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

可将此元素与 **db_work_class_id** 元素配合使用，以唯一地标识活动的数据库工作类（如果有）。

db_work_class_id – “数据库工作类标识”监视元素

如果此活动已划分到数据库作用域的某个工作类，那么此监视元素将显示该工作类的标识。否则，此监视元素将显示值 0。

表 345. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 346. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

可将此元素与 **db_work_action_set_id** 元素配合使用，以唯一地标识活动的数据库工作类（如果有）。

dcs_appl_status – “DCS 应用程序状态”

DB2 Connect 网关上的 DCS 应用程序的状态。

表 347. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。值包括:

- SQLM_DCS_CONNECTPEND_OUTBOUND

  应用程序已启动从 DB2 Connect 网关至主机数据库的数据库连接，但请求尚未完成。
- SQLM_DCS_UOWWAIT_OUTBOUND

DB2 Connect 网关正在等待主机数据库应答应用程序的请求。

- SQLM_DCS_UOWWAIT_INBOUND

  已建立从 DB2 Connect 网关至主机数据库的连接并且网关正在等待来自应用程序的 SQL 请求。或者 DB2 Connect 网关正在代表应用程序中的工作单元等待。这通常意味着正在执行应用程序的代码。

dcs_db_name - "DCS 数据库名称"

DCS 目录中编目的 DCS 数据库的名称。

表 348. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

ddl_sql_stmts - "数据定义语言（DDL）SQL 语句数"

此元素指示执行的 SQL 数据定义语言（DDL）语句数。

表 349. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 350. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。DDL 语句的运行成本很高，这是因为它们对系统目录表有影响。因此，如果此元素的值很高，那么应确定原因并在可能的情况下限制此活动的执行。

还可使用此元素并借助以下公式来确定 DDL 活动的百分比:

```
ddl_sql_stmts / 语句总数
```

此信息对于分析应用程序活动和吞吐量非常有用。DDL 语句还会影响:

- 目录高速缓存（通过使该处存储的表描述符信息和权限信息无效并导致因为从系统目录检索信息而产生其他系统开销）
- 程序包高速缓存（通过使该处存储的段无效并导致因为段重新编译而产生其他系统开销）。

DDL 语句的示例包括 CREATE TABLE、CREATE VIEW、ALTER TABLE 和 DROP INDEX。

deadlock_id - "死锁事件标识"

死锁的标识。

元素标识

deadlock_id

元素类型

信息

表 351. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_stmt_history	-
带有详细信息的死锁的历史记录值	event_data_value	-
带有详细信息的死锁的历史记录值	event_detailed_dlconn	-
带有详细信息的死锁的历史记录值	event_stmt_history	-

用法 在监视应用程序中使用此元素，以将死锁连接和语句历史记录事件记录与死锁事件记录相关联。

deadlock_node - "发生死锁的分区号"

发生死锁的分区号。

元素标识

deadlock_node

元素类型

信息

表 352. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

用法 此元素仅与分区数据库有关。在监视应用程序中使用此元素，以将死锁连接事件记录与死锁事件记录相关联。

deadlocks - “检测到的死锁数”监视元素

发生的死锁总数。

表 353. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 354. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	锁定

可将快照监视的计数器复位。

表 355. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 355. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-

用法

此元素可指示应用程序遇到争用问题。这些问题可能是由下列情况导致的:

- 数据库发生锁定升级
- 在系统生成的行锁定已足够的情况下应用程序显式锁定了表
- 绑定时应用程序使用了不适当的隔离级别
- 目录表已被锁定以供可重复读
- 应用程序正以不同的顺序获取相同的锁定，从而导致死锁。

可通过确定发生死锁的应用程序（或应用程序进程）来解决问题。然后，您可以修改该应用程序，以使它能够更好地并行运行。但某些应用程序可能无法并行运行。

可以使用连接时间戳记监视元素（**last_reset**、**db_conn_time** 和 **appl_con_time**）来确定死锁的严重性。例如，5 分钟内发生 10 个死锁比 5 小时内发生 10 个死锁要严重得多。

以上列示的相关元素的描述还可提供其他调整建议。

degree_parallelism - “并行度”

绑定查询时请求的并行度。

元素标识

degree_parallelism

元素类型

信息

表 356. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

用法 与 agents_top 配合使用来确定查询是否达到最大级别的并行性。

del_keys_cleaned - “清除伪删除键数目”监视元素

已清除的伪删除键的数目。

表 357. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

delete_sql_stmts - “删除数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位以后（取较晚者），联合服务器代表任何应用程序对此数据源发出 DELETE 语句的总次数。

表 358. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法 使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
        (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
        (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
        DELETE 语句数)
```

delete_time - “删除响应时间”

此元素包含此数据源响应 DELETE 所花的总时间（以毫秒计），这些 DELETE 来自联合服务器启动后或数据库监视计数器上一次复位后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。

响应时间是以联合服务器将 DELETE 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 DELETE 的时间之差量度的。

表 359. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

用法 使用此元素来确定等待处理对此数据源执行 DELETE 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

destination_service_class_id - “目标服务类标识”监视元素

生成此元素所属的阈值违例记录时此活动重新映射到的服务子类的标识。对于除 REMAP ACTIVITY 以外的任何阈值操作，此元素的值为零。

元素标识

destination_service_class_id

元素类型

信息

表 360. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来跟踪活动在它重新映射到的各个服务类之间的路径。此元素还可用于计算映射到给定服务子类的活动数的聚集值。

diaglog_write_wait_time - "诊断日志文件写等待时间"监视元素

等待写 db2diag 日志文件时耗用的时间。此值以毫秒计。

表 361. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 362. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过使用此元素，可以确定写 db2diag 日志文件时的耗用时间。在分区数据库环境中，如果此时间过长，并且正在将共享存储器用于 diagpath，那么可能表明 db2diag 日志文件被争用。较大的值也可能表明日志记录量过大，例如，已将 **diaglevel** 设置为记录所有参考消息。

diaglog_writes_total - "写诊断日志文件总次数"监视元素

代理程序写 db2diag 日志文件的次数。

表 363. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 364. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过将此元素与 diaglog_write_wait_time 配合使用，可以了解写 db2diag 日志文件时的平均耗用时间。

direct_read_reqs - "直接读请求数"监视元素

对一个或多个扇区的数据执行直接读取的请求数。

表 365. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 365. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 366. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 367. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

表 367. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

用法

使用以下公式来计算直接读操作读取的平均扇区数:

```
direct_reads / direct_read_reqs
```

direct_read_time - "直接读时间"监视元素

执行直接读操作时耗用的时间。此值以毫秒计。

表 368. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 368. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 369. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 370. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

用法

使用以下公式来计算每扇区平均直接读取时间:

```
direct_read_time / direct_reads
```

如果平均时间很长，那么指示可能存在 I/O 冲突。

direct_reads - “直接读数据库数目”监视元素

未使用缓冲池的读操作的数目。

表 371. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 371. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 372. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 373. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 373. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

用法

使用以下公式来计算直接读操作读取的平均扇区数:

```
direct_reads / direct_read_reqs
```

使用系统监视器跟踪 I/O 时，此元素可帮助您区分设备上的数据库 I/O 与非数据库 I/O。

直接读操作是以单元为单位执行的，最小单元为 512 字节扇区。在下列情况下将使用它们:

- 读取 LONG VARCHAR 列
- 读取 LOB（大对象）列
- 执行备份

direct_write_reqs - “直接写请求数”监视元素

对一个或多个扇区的数据执行直接写入的请求数。

表 374. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 374. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 375. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 376. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

用法

使用以下公式来计算直接写入操作写入的平均扇区数:

```
direct_writes / direct_write_reqs
```

direct_write_time – “直接写时间”监视元素

执行直接写操作时耗用的时间。此值以毫秒计。

表 377. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 – 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 – 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 – 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 – 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 – 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 – 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 – 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 378. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 379. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

用法

使用以下公式来计算每扇区平均直接写入时间:

```
direct_write_time / direct_writes
```

如果平均时间很长，那么指示可能存在 I/O 冲突。

direct_writes - “直接写数据库数目”监视元素

未使用缓冲池的写操作的数目。

表 380. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 380. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 381. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 382. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
表空间	event_tablespace	-

用法

使用以下公式来计算直接写入操作写入的平均扇区数:

```
direct_writes / direct_write_reqs
```

使用系统监视器跟踪 I/O 时，此元素可帮助您区分设备上的数据库 I/O 与非数据库 I/O。

直接写入操作是以单元为单位执行的，最小单元为 512 字节扇区。在下列情况下将使用它们:

- 写入 LONG VARCHAR 列
- 写入 LOB（大对象）列
- 执行恢复
- 执行装入
- 如果启用了 MPFA（这是缺省情况），那么为 SMS 表空间分配新的扩展数据块

disconn_time -“数据库释放时间戳记”

应用程序与数据库在数据库级别断开连接的日期和时间，这是应用程序最后一次断开连接。

元素标识

disconn_time

元素类型

时间戳记

表 383. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 使用此元素来计算自出现以下情况以来耗用的时间:

- 数据库处于活动状态（用于数据库级别的信息）
- 连接处于活动状态（用于连接级别的信息）。

disconnects -“断开连接次数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位以后（取较晚者），联合服务器代表任何应用程序与此数据源断开连接的总次数。

表 384. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本

可将快照监视的计数器复位。

用法

使用此元素来确定联合服务器代表任何应用程序与此数据源断开连接的总次数。此元素与 CONNECT 计数一起使用可提供一种机制，您可以通过这种机制来确定此联合服务器实例相信且当前已连接至数据源的应用程序数。

dl_conns -“死锁中涉及的连接数”监视元素

死锁中涉及的连接数。

表 385. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	event_deadlock	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

在监视应用程序中使用此元素以标识事件监视器数据流中将会出现的死锁连接事件记录数。

dynamic_sql_stmts - "尝试的动态 SQL 语句数"

尝试的动态 SQL 语句数。

表 386. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 387. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
    dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

eff_stmt_text - "有效语句文本"监视元素

如果 SQL 语句作为语句集中器的结果而被修改，那么此元素包含该语句的有效文本。

表 388. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 389. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	-

用法

如果已启用语句集中器，并且语句文本已作为语句集中器的结果而被修改，那么此监视元素包含有效的语句文本。否则，此监视元素包含长度为 0 字节的文本字符串。

effective_isolation - "有效隔离级别"监视元素

此活动的有效隔离级别。

表 390. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集

表 391. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

用法

通过使用此元素，可以确定执行活动期间使用的隔离级别。

effective_lock_timeout - "有效锁定超时"监视元素

此活动的有效锁定超时值。

表 392. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

effective_query_degree - "有效查询并行度"监视元素

此活动的有效查询并行度。

表 393. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 394. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

elapsed_exec_time - "语句执行耗用时间"

在 DCS 语句级别，这是在主机数据库服务器上处理 SQL 请求所耗用的时间。此值由此服务器报告。和 host_response_time 元素相比，此元素不包括 DB2 Connect 与主机数据库服务器之间的网络耗用时间。在其他级别，此值表示对特定数据库或应用程序执行的所有语句的主机执行时间的总和，或者是使用给定数目的数据传输的那些语句的主机执行时间的总和。

表 395. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句，时间戳记
应用程序	appl	语句，时间戳记
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

用法 将此元素与其他耗用时间监视元素一起使用以评估数据库服务器的 SQL 请求处理情况和帮助隔离性能问题。

从 host_response_time 元素减去此元素，以计算 DB2 Connect 与主机数据库服务器之间的网络耗用时间。

注: 对于 dcs_dbase、dcs_appl、dcs_stmt 和 stmt_transmissions 级别，*elapsed_exec_time* 元素仅适用于 z/OS® 数据库。如果 DB2 Connect 网关连接至 Windows、Linux、AIX 或其他 UNIX 数据库，那么 *elapsed_exec_time* 报告为零。

empty_pages_deleted - "删除的空页数"监视元素

已进行伪删除的伪空页中所有的键。此监视元素报告已删除的伪空页数。

表 396. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

empty_pages_reused - "复用的空页数"监视元素

已进行伪删除的伪空页中所有的键。此监视元素报告已复用的伪空页数。

表 397. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

entry_time - "进入时间"监视元素

此活动进入系统时的时间。

表 398. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

estimatedsqlcost_threshold_id - "估计 SQL 成本阈值标识"监视元素

应用于此活动的 ESTIMATEDSQLCOST 阈值的标识。

表 399. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 ESTIMATEDSQLCOST 阈值（如果有的话）。

estimatedsqlcost_threshold_value - "估计 SQL 成本阈值"监视元素

应用于此活动的 ESTIMATEDSQLCOST 阈值的上限。

表 400. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 ESTIMATEDSQLCOST 阈值（如果有的话）。

estimatedsqlcost_threshold_violated – “违反估计 SQL 成本阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 ESTIMATEDSQLCOST 阈值。“No”表明此活动尚未违反该阈值。

表 401. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 ESTIMATEDSQLCOST 阈值。

event_monitor_name – “事件监视器名称”

创建事件数据流的事件监视器的名称。

元素标识

event_monitor_name

元素类型

信息

表 402. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法 此元素允许您将要分析的数据与系统目录表中的特定事件监视器相关联。这是可在 SYSCAT.EVENTMONITORS 目录表的 NAME 列中找到的相同名称，该名称是在 CREATE EVENT MONITOR 和 SET EVENT MONITOR 语句上指定的。

event_time – “事件时间”

发生事件的日期和时间。

表 403. 事件监视信息

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	-
表	event_table	-

用法 可使用此元素来帮助相关事件按时间排序。

evmon_activates – “事件监视器激活数”

激活事件监视器的次数。

元素标识

evmon_activates

元素类型

计数器

表 404. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表	event_table	-
表空间	event_tablespace	-
缓冲池	event_bufferpool	-
死锁	event_deadlock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

用法 使用此元素以使上述事件类型返回的信息相关。此元素仅适用于写至表事件监视器。未对写至文件或管道的事件监视器保留此监视元素。

只有某些类型的写至表事件监视器使用 evmon_activates 监视元素（使用此元素的事件监视器类型列示在先前的表“事件监视信息”中）。在激活时，这些事件监视器将更新 SYSCAT.EVENTMONITORS 目录表的 evmon_activates 列。将记录此更改，所以 DATABASE CONFIGURATION 将显示:

```
数据库是一致的                          = 否
```

如果事件监视器是使用 AUTOSTART 选项创建的，并且第一个用户连接至数据库后立即断开连接从而使得数据库被释放，那么会生成日志文件。

executable_id - “可执行文件标识”监视元素

在数据服务器上生成的加密二进制标记，用于唯一地标识已执行的 SQL 语句节。对于非 SQL 活动，将返回长度为 0 的字符串值。

表 405. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 406. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	-

用法

使用此监视元素作为不同监视接口的输入，以获取关于该节的数据。MON_GET_PKG_CACHE_STMT 表函数（用于获取程序包高速缓存中的 SQL 语句活动度量值）接受可执行文件标识作为输入。

evmon_flushes - "事件监视器清空数"

发出 FLUSH EVENT MONITOR SQL 语句的次数。

元素标识

evmon_flushes

元素类型

信息

表 407. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表	event_table	-
表空间	event_tablespace	-
缓冲池	event_bufferpool	-

用法 应用程序连接至数据库之后，数据库管理器每次成功处理 FLUSH EVENT MONITOR SQL 请求时，此标识就会递增。此元素有助于唯一标识数据库、表、表空间和缓冲池数据。

execution_id - "用户登录标识"

用户在登录至操作系统时指定的标识。此标识与 auth_id 不同，auth_id 是用户在连接至数据库时指定的。

表 408. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 409. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

用法 可使用此元素来确定个人的操作系统用户标识，这些人正在运行正在监视的应用程序。

failed_sql_stmts - "失败的语句操作"

尝试但失败了的 SQL 语句数。

表 410. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器复位。

表 411. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
    dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

此计数包括接收到负值 SQLCODE 的所有 SQL 语句数。

此元素还可帮助您确定性能低下的原因，这是因为失败的语句意味着数据库管理器浪费了时间并因而导致数据库的吞吐量很低。

fcm_message_recv_volume - "接收 FCM 消息量"监视元素

为 FCM 通信层所分发的内部请求（例如 RPC）接收的数据量。此值以字节计。

表 412. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 412. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 413. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定在通过 FCM 子系统发送的数据量中，用于请求或应答消息流量（而不是实际的表数据）的数据量。

fcm_message_recv_wait_time - “接收 FCM 消息等待时间”监视元素

代理程序在等待 FCM 应答消息时耗用的时间（该消息包含先前发送的 FCM 请求消息的结果）。此时间既反映在另一端处理请求消息所需的时间，也反映使用 FCM 在分区之间发送响应所需的时间。此值以毫秒计。

表 414. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 415. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

此元素可用于确定在多分区实例的给定分区中，等待在其他分区中处理请求时耗用的时间。

fcm_message_recvs_total - "接收 FCM 消息总数"监视元素

作为 FCM 应答消息的组成部分接收的缓冲区的总数（该消息包含先前发送的 FCM 请求消息的结果）。

表 416. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

此元素可用于确定所接收的每条 FCM 消息的平均数据量以及等待接收单一 FCM 消息时的平均耗用时间。

fcm_message_send_volume - "发送 FCM 消息量"监视元素

通过内部 FCM 请求发送的数据量。此值以字节计。

表 417. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定在通过 FCM 子系统发送的数据量中，用于发送请求和应答消息流量（而不是实际的表数据）的数据量。

fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素

发送 FCM 消息时被阻塞的时间。此值以毫秒计。此监视元素反映在数据库系统中分发内部请求期间将 FCM 缓冲区从 FCM 通道中清仓时被阻塞的时间。

表 418. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定代理程序等待通过 FCM 子系统发送 FCM 请求消息时耗用的时间。根据 FCM 守护程序的繁忙程度不同，代理程序在尝试发送消息时可能需要等待。

fcm_message_sends_total - "发送 FCM 消息总数"监视元素

使用 FCM 通信机制作为内部请求的组成部分分发的缓冲区的总数。

表 419. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 420. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定为每条 FCM 请求消息发送的平均数据量以及处理每条 FCM 消息时的平均等待时间。

fcm_recv_volume - "FCM 接收量"监视元素

通过 FCM 通信层接收的数据总量。此值以字节计。

表 421. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 422. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

指示在此分区中使用 FCM 接收的数据总量，其中包括消息流量和表队列数据量。

fcm_recv_wait_time - “FCM 接收等待时间”监视元素

等待通过 FCM 接收数据时的耗用时间总计。此值以毫秒计。

表 423. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 424. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定在此数据库分区中等待通过 FCM 接收数据时的耗用时间总计。这既包括请求消息应答数据也包括表队列数据。

fcm_recvs_total - “FCM 接收总计”监视元素

使用 FCM 通信机制为内部请求接收的缓冲区的总数。fcm_recvs_total 监视元素值是 fcm_message_recvs_total 监视元素值与 fcm_tq_recvs_total 监视元素值之和。

表 425. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 426. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过将此元素与 **fcm_recv_wait_time** 监视元素配合使用，可以确定每个 FCM 接收操作的平均等待时间以及 FCM 接收操作所返回的平均数据量。

fcm_send_volume -“FCM 发送量”监视元素

通过 FCM 通信层分发的数据总量。此值以字节计。

表 427. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

用法

使用此监视元素来确定使用 FCM 发送的数据总量，其中包括消息流量和表队列数据量。

fcm_send_wait_time - “FCM 发送等待时间”监视元素

执行 FCM 发送操作时被阻塞的时间。这包括等待内部请求的缓冲区被清仓时耗用的时间以及通过表队列发送数据时等待窗口计数确认时耗用的时间。此值以毫秒计。

表 428. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 428. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 429. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定等待通过 FCM 发送数据时的耗用时间总计。这既包括发送请求消息时耗用的时间，也包括发送表队列数据时耗用的时间。

fcm_sends_total - “FCM 发送总计”监视元素

使用内部 FCM 通信层发送的缓冲区的总数。

表 430. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 430. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 431. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定每个 FCM 接收操作的平均等待时间以及 FCM 接收操作所返回的平均数据量。

fcm_tq_recv_volume - “FCM 表队列接收量”监视元素

FCM 通信层通过表队列接收的数据量。此值以字节计。

表 432. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 432. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 433. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定通过表队列接收的数据总量。

fcm_tq_recv_wait_time - “FCM 表队列接收等待时间”监视元素

等待从表队列中接收下一个缓冲区时耗用的时间。此值以毫秒计。

表 434. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 434. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 435. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定代理程序等待通过表队列接收数据时耗用的时间。

fcm_tq_recvs_total - “FCM 表队列接收总量”监视元素

使用内部 FCM 通信机制从表队列接收的缓冲区的总数。

表 436. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 437. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 437. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过将此元素与 **fcm_tq_recv_volume** 和 **fcm_tq_recv_wait_time** 配合使用，可以确定接收每个表队列缓冲区时的平均等待时间和接收数据量。

fcm_tq_send_volume - "FCM 表队列发送量"监视元素

FCM 通信层通过表队列发送的数据量。此值以字节计。

表 438. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 439. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定使用 FCM 通过表队列缓冲区发送的数据总量。

fcm_tq_send_wait_time - "FCM 表队列发送等待时间"监视元素

等待通过表队列发送下一个缓冲区时耗用的时间。此元素反映等待来自表队列接收端的窗口计数确认时耗用的时间。此值以毫秒计。

表 440. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 441. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此监视元素来确定等待使用 FCM 通过表队列发送数据缓冲区时耗用的时间。

fcm_tq_sends_total - "FCM 表队列发送总次数"监视元素

使用内部 FCM 通信机制发送的缓冲区（包含表队列数据）的总数。

表 442. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 443. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过将此元素与 **fcm_tq_send_volume** 和 **fcm_tq_send_wait_time** 监视元素配合使用，可以确定使用表队列发送的缓冲区的平均数据量和平均等待时间。

fetch_count - "成功的访存数"

成功的物理访存数或尝试的物理访存数，取决于快照监视级别。

- 对于 stmt 和 dynsql 快照监视级别和语句事件类型：对特定游标执行的成功访存数。
- 对于 dcs_stmt 快照监视级别：在语句执行期间尝试的物理访存数（不管应用程序访存的行数是多少）。在这种情况下，**fetch_count** 表示处理语句时服务器需要将应答数据发送回网关的次数。

表 444. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句
动态 SQL	dynsql	语句

对于动态 SQL 快照监视，可以复位此计数器。

表 445. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

用法

可使用此元素来了解数据库管理器内的当前活动级别。

由于性能原因，语句事件监视器不会对每个 FETCH 语句生成语句事件记录。仅当 FETCH 返回非零 SQLCODE 时，才生成记录事件。

files_closed - “关闭数据库文件数”监视元素

关闭的数据库文件的总数。

表 446. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 447. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 448. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法

数据库管理器打开文件以便读入缓冲池和写出缓冲池。任何时间应用程序打开的最大数据库文件数都由 **maxfilop** 配置参数控制。如果达到最大文件数，那么在打开新文件之前必须先关闭某个文件。注意，实际的打开文件数可能不等于关闭文件数。

您可以使用此元素来帮助确定最佳的 **maxfilop** 配置参数值。

first_active_log - “第一个活动日志文件编号”

第一个活动日志文件的文件号。

元素标识

first_active_log

元素类型

信息

表 449. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 450. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *last_active_log* 和 *current_active_log* 元素一起使用来确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

first_overflow_time - "第一次事件溢出时间"

此溢出记录的第一次溢出的日期和时间。

表 451. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

用法 将此元素与 *last_over_flow time* 配合使用来计算生成溢出记录所耗用的时间。

fs_caching - "文件系统高速缓存"监视元素

指示特定表空间是否使用文件系统高速缓存。如果 **fs_caching** 为 0，那么表明已启用文件系统高速缓存。如果 **fs_caching** 为 1，那么表明已将文件系统高速缓存禁用。

表 452. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 453. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

表 454. 事件监视信息

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	-

fs_id - "唯一文件系统标识号"监视元素

此元素指示操作系统为存储器路径或容器所指向的文件系统提供的唯一标识号。

表 455. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 456. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法

通过将此元素与下列元素配合使用，可以收集数据库的空间利用率数据:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_used_size**
- **fs_total_size**
- **fs_type**

fs_total_size - "文件系统总大小"监视元素

此元素指示存储器路径或容器所指向的文件系统的容量。

表 457. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 458. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法

可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_used_size**
- **fs_id**
- **fs_type**

fs_type - “文件系统类型”

此元素显示存储路径指向的文件系统的类型。此文件系统类型是由操作系统提供的。

元素标识

fs_type

元素类型

信息

表 459. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法 可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- db_storage_path
- sto_path_free_sz
- fs_used_size
- fs_total_size
- fs_id

fs_used_size - “文件系统中的已用空间量”监视元素

此元素指示存储器路径或容器所指向的文件系统中的已用空间量。

表 460. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 461. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法

可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_total_size**
- **fs_id**
- **fs_type**

gw_comm_error_time - “通信错误时间”

DCS 应用程序尝试连接至主机数据库时或处理 SQL 语句时发生最新通信错误（SQL30081）的日期和时间。

表 462. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

用法

将此元素与通信错误及管理通知日志中记录的通信错误一起使用来确定问题。

gw_comm_errors - "通信错误"

DCS 应用程序尝试连接至主机数据库时或处理 SQL 语句时发生通信错误（SQL30081）的次数。

表 463. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本

可将快照监视的计数器复位。

用法 通过监视一段时间内发生通信错误的次数，可评估 DB2 Connect 网关与特定主机数据库的连接是否存在问题。可设置您希望的正常错误阈值，这样每当错误数超过此阈值时都会进行通信错误调查。

将此元素与管理通知日志中记录的通信错误一起使用来确定问题。

gw_con_time - "DB2 Connect 网关首次启动的连接"

从 DB2 Connect 网关首次启动与主机数据库的连接的日期和时间。

表 464. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记
DCS 应用程序	dcs_appl	时间戳记

用法 此元素用于 DCS 应用程序的问题确定。

gw_connections_top - "与主机数据库的最大并行连接数"

自首次数据库连接后由 DB2 Connect 网关处理的与主机数据库的并行连接的最大数目。

元素标识

gw_connections_top

元素类型

水位标记

表 465. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

gw_cons_wait_client -“等待客户机发送请求的连接数”

由 DB2 Connect 网关处理并且等待客户机发送请求的与主机数据库的当前连接的数目。

元素标识

gw_cons_wait_client

元素类型

标尺

表 466. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

用法 此值可能经常更改。在延长时间段内应按一定时间间隔对其进行采样以了解真实的网关使用情况。

gw_cons_wait_host -“等待主机应答的连接数”

由 DB2 Connect 网关处理并且等待主机应答的与主机数据库的当前连接的数目。

元素标识

gw_cons_wait_host

元素类型

标尺

表 467. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

用法 此值可能经常更改。在延长时间段内应按一定时间间隔对其进行采样以了解真实的网关使用情况。

gw_cur_cons -“DB2 Connect 的当前连接数”

由 DB2 Connect 网关处理的与主机数据库的当前连接的数目。

表 468. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

gw_db_alias - “网关上的数据库别名”

DB2 Connect 网关上用来连接至主机数据库的别名。

元素标识

gw_db_alias

元素类型

信息

表 469. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

gw_exec_time - “DB2 Connect 网关处理所耗用的时间”

DB2 Connect 网关处理应用程序请求（自建立连接后）或处理单个语句所花的时间（以秒和微秒计）。

表 470. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

可将快照监视的计数器复位。

用法 使用此元素来确定整体处理时间的哪一部分用于 DB2 Connect 网关处理。

gw_total_cons - “对 DB2 Connect 尝试连接的总数”

自最后一次 `db2start` 命令或最后一次复位后尝试从 DB2 Connect 网关进行连接的总次数。

表 471. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

可将快照监视的计数器复位。

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

hadr_connect_status - “HADR 连接状态”监视元素

数据库的高可用性灾难恢复（HADR）连接状态。

表 472. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 连接状态。

此元素的数据类型是整型。

如果数据库具有 HADR 主角色或备用角色，那么此元素的值是下列其中一个常量:

SQLM_HADR_CONN_CONNECTED
数据库连接至其伙伴节点。

SQLM_HADR_CONN_DISCONNECTED
数据库未连接至其伙伴节点。

SQLM_HADR_CONN_CONGESTED
数据库连接至其伙伴节点，但连接拥塞。当主数据库与备用数据库之间的 TCP/IP 套接字连接仍然活动但一端不能将信息发送至另一端时连接拥塞。例如，接收端未从套接字连接接收，导致 TCP/IP 发送空间变满。网络连接拥塞的原因包括下列几项:

- 网络被太多资源共享，或者网络相对于主 HADR 节点的事务量而言不够快。
- 备用 HADR 节点所在的服务器处理能力不足，无法以必要的速率检索通信子系统中的信息。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_connect_time - "HADR 连接时间"监视元素

显示下列之一: 高可用性灾难恢复（HADR）连接时间、HADR 拥塞时间或 HADR 断开连接时间。

表 473. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定当前 HADR 连接状态开始的时间。

如果数据库为 HADR 主数据库或备用数据库，那么此元素的含义取决于 **hadr_connect_status** 元素的值:

- 如果 **hadr_connect_status** 元素的值为 SQLM_HADR_CONN_CONNECTED，那么此元素显示连接时间。
- 如果 **hadr_connect_status** 元素的值为 SQLM_HADR_CONN_CONGESTED，那么此元素显示拥塞开始的时间。

- 如果 **hadr_connect_status** 元素的值为 SQLM_HADR_CONN_DISCONNECTED，那么此元素显示断开连接时间。

如果自 HADR 引擎可拆离单元（EDU）启动后没有任何连接，那么会将连接状态报告为“已断开连接”并且 HADR EDU 启动时间将用于断开连接时间。因为 HADR 连接和断开连接事件相对少见，所以即使 DFT_MON_TIMESTAMP 开关为 OFF，也会收集并报告该时间。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_heartbeat - “HADR 脉动信号”监视元素

在高可用性灾难恢复（HADR）连接上已丢失的脉动信号数。如果数据库具有 HADR 主角色或备用角色，那么此元素指示 HADR 连接的运行状况。

表 474. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

对于快照监视来说，无法复位此计数器。

用法说明:

使用此元素来确定 HADR 连接的运行状况。

脉动信号是以固定时间间隔从其他 HADR 数据库发送的消息。如果此元素的值为零，那么表明未丢失脉动信号，并且连接的运行状况正常。此值越大，连接的运行状况就越差。

在断开连接方式下，因为丢失的脉动信号不适用，所以它始终显示为 0。

HADR 数据库期望伙伴数据库在特定时间间隔内至少发送一条脉动信号消息，该时间间隔是 HADR_TIMEOUT 数据库配置参数中定义的时间间隔的 1/4 或者 30 秒（以较小者为准）。例如，如果 HADR_TIMEOUT 值是 80（秒），那么 HADR 数据库期望伙伴数据库至少每 20 秒发送一条脉动信号消息。

此元素的数据类型是整型。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_local_host - “HADR 本地主机”监视元素

本地高可用性灾难恢复（HADR）主机名。该值显示为主机名字符串或 IP 地址字符串，如“1.2.3.4”。

表 475. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 本地主机名称。HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

注: 使用的任何名称都必须解析为一个 IP 地址。尝试启动 HADR 时，解析为多个地址的名称将导致错误。

hadr_local_service - "HADR 本地服务"监视元素

本地 HADR TCP 服务。此值将显示为服务名称字符串或端口号字符串。

表 476. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 本地服务名称。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_log_gap - "HADR 日志间隔"

此元素显示主日志序号（LSN）与备用日志 LSN 之间正在运行的平均间隔。该间隔是以字节数度量的。

表 477. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库日志与备用 HADR 数据库日志之间的间隔。

当日志文件被截断时，下一个日志文件中的 LSN 将会开始，就好像上一个文件未被截断一样。此 LSN 洞口未包含任何日志数据。这样的洞口可能导致日志间隔不反映主 HADR 数据库日志与备用 HADR 数据库日志之间的实际日志差别。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_peer_window – “HADR 对等窗口”监视元素

HADR_PEER_WINDOW 数据库配置参数的值。

表 478. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定 HADR_PEER_WINDOW 数据库配置参数的值。

hadr_peer_window_end – “HADR 对等时间结束”监视元素

只要主数据库处于活动状态，在高可用性灾难恢复（HADR）主数据库承诺将时间点停留在对等或断开对等状态之前的时间点。

表 479. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定在主数据库承诺将时间点停留在对等或断开对等状态之前的时间点。

主数据库报告的值可能与备用数据库报告的值不同。发生此情况的原因是：主数据库在发送脉动信号消息时对值进行了更新，而新值仅在备用数据库接收并处理消息之后才显示在备用数据库上。

如果数据库脱离对等或断开对等状态，那么此监视元素的值不会被复位。将保留并返回最后知道的值。如果数据库从未达到对等状态，那么将返回零值。

对等时间结束时间由主数据库设置，然后发送至备用数据库。因此，对等时间结束值将基于主数据库时钟。当您将对等时间结束时间与主数据库停机时间进行比较时，如果两个时钟未同步良好，那么您可能需要添加偏移量以将时间戳记转换为主数据库时钟。

hadr_primary_log_file – “HADR 主日志文件”监视元素

当前日志文件在主 HADR 数据库上的名称。

表 480. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库上的当前日志文件。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_primary_log_lsn －“HADR 主日志 LSN”监视元素

主 HADR 数据库的当前日志位置。日志序号（LSN）是数据库的日志流中的字节位移。

表 481. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库上的当前日志位置。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_primary_log_page －“HADR 主日志页”监视元素

当前日志文件中的页号，指示当前日志在主 HADR 数据库上的位置。页号与日志文件相关。例如，页零是文件的开头。

表 482. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库上的当前日志页。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_remote_host －“HADR 远程主机”监视元素

远程高可用性灾难恢复（HADR）主机名。该值显示为主机名字符串或 IP 地址字符串，如“1.2.3.4”。

表 483. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 远程主机名称。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

注: 使用的任何名称都必须解析为一个 IP 地址。尝试启动 HADR 时，解析为多个地址的名称将导致错误。

hadr_remote_instance - "HADR 远程实例"监视元素

远程 HADR 实例名。

表 484. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 远程实例名。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_remote_service - "HADR 远程服务"监视元素

远程 HADR TCP 服务。此值将显示为服务名称字符串或端口号字符串。

表 485. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 远程服务名称。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_role - "HADR 角色"

数据库的高可用性灾难恢复（HADR）角色。

表 486. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 角色。

此元素的数据类型是整型。

此元素的值是下列其中一个常量:

SQLM_HADR_ROLE_STANDARD
数据库不是 HADR 数据库。

SQLM_HADR_ROLE_PRIMARY
数据库是主 HADR 数据库。

SQLM_HADR_ROLE_STANDBY
数据库是备用 HADR 数据库。

hadr_standby_log_file - "HADR 备用日志文件"监视元素

当前日志文件在备用 HADR 数据库上的名称。

表 487. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定备用 HADR 数据库上的当前日志文件。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_standby_log_lsn - "HADR 备用日志 LSN"监视元素

备用 HADR 数据库的当前日志位置。日志序号（LSN）是数据库的日志流中的字节位移。

表 488. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定备用 HADR 数据库上的当前日志位置。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_standby_log_page - "HADR 备用日志页"监视元素

当前日志文件中的页号，指示当前日志在备用 HADR 数据库上的位置。页号与日志文件相关。例如，页零是文件的开头。

表 489. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定备用 HADR 数据库上的当前日志页。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_state - "HADR 状态"监视元素

数据库的高可用性灾难恢复（HADR）状态。

表 490. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 状态。

此元素的数据类型是整型。如果数据库具有 HADR 主角色或备用角色，那么此元素的值是下列其中一个常量:

SQLM_HADR_STATE_DISCONNECTED
数据库未连接至它的伙伴数据库。

SQLM_HADR_STATE_LOC_CATCHUP
数据库正在进行本地同步复制。

SQLM_HADR_STATE_REM_CATCH_PEND
数据库正在等待连接至它的伙伴数据库以执行远程同步复制。

SQLM_HADR_STATE_REM_CATCHUP
数据库正在进行远程同步复制。

SQLM_HADR_STATE_PEER
在主数据库与备用数据库之间已建立连接，并且它们处于对等状态。

SQLM_HADR_STATE_DISCONN_PEER
主数据库与备用数据库处于断开对等状态。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_syncmode - "HADR 同步方式"监视元素

数据库的高可用性灾难恢复（HADR）同步方式。

表 491. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 同步方式。

此元素的数据类型是整型。

HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

如果数据库具有 HADR 主角色或备用角色，那么此元素的值是下列其中一个常量:

SQLM_HADR_SYNCMODE_SYNC
同步方式。

SQLM_HADR_SYNCMODE_NEARSYNC
接近同步方式。

SQLM_HADR_SYNCMODE_ASYNC
异步方式。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_timeout - "HADR 超时"监视元素

没有与其伙伴进行任何通信的秒数，HADR 数据库服务器在此时间之后将认为它们之间的连接出现故障。

表 492. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 超时值。 HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hash_join_overflows - "散列连接溢出数"

散列连接数据超过可用排序堆空间的次数。

元素标识
hash_join_overflows

元素类型

计数器

表 493. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 494. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 在数据库级别，如果 hash_join_small_overflows 的值大于此 hash_join_overflows 的 10%，那么应该考虑增加排序堆大小。应用程序级别的值可用于评估各个应用程序的散列连接性能。

hash_join_small_overflows - "散列连接小溢出数"

散列连接数据超过可用排序堆空间的部分小于 10% 的次数。

元素标识

hash_join_small_overflows

元素类型

计数器

表 495. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 496. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 如果此值和 hash_join_overflows 很高，那么应考虑增加排序堆阈值。如果此值超过 hash_join_overflows 的 10%，那么应考虑增加排序堆大小。

histogram_type - "直方图类型"监视元素

直方图的类型，字符串格式。

有六种直方图类型。

CoordActQueueTime

在协调程序分区中测量的非嵌套活动排队（例如，在阈值队列中）的耗用时间直方图。

CoordActExecTime

非嵌套活动在协调程序分区执行时所花的时间的直方图。执行时间不包括初始化或排队所花的时间。对于游标来说，执行时间只包括打开、访存和关闭请求所花的时间。在服务子类之间重新映射活动时，只有在该活动所在的服务子类执行完成的情况下，才会更新执行时间直方图。

CoordActLifetime

从非嵌套活动被数据库管理器标识到该活动执行完成的耗用时间直方图（在协调程序分区中测量）。在服务子类之间重新映射活动时，只有在该活动所在的服务子类执行完成的情况下，才会更新生存期直方图。

CoordActInterArrivalTime

非嵌套协调程序活动的到达之间的时间间隔的直方图。将对活动进入系统时所借助于的服务子类计算到达之间时间平均值。在服务子类之间重新映射活动时，活动重新映射到的服务子类的到达之间时间直方图不受影响。

CoordActEstCost

非嵌套 DML 活动估计成本直方图。活动的估计成本只计入该活动从中进入系统的服务子类。

ReqExecTime

请求执行次数的直方图，它包括对协调程序分区的请求和对协调程序与非协调程序分区的子请求（如 RPC 请求或 SMP 子代理程序请求）。所包括的请求可能与活动相关联，也可能不会与活动相关联：例如，PREPARE 和 OPEN 请求均包括在此直方图中，但当 OPEN 请求始终与游标活动相关联时，PREPARE 请求不是任何活动的一部分。重新映射所涉及的服务子类的执行时间直方图对该服务子类中的不完整请求所耗用的执行时间进行计数。

表 497. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

使用此元素来标识直方图的类型。可以有几个直方图属于同一统计信息记录，但每种类型只能有一个。

host_ccsid - “主机编码字符集标识”

此项是主机数据库的编码字符集标识（CCSID）。

元素标识

host_ccsid

元素类型

信息

表 498. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

host_db_name -"主机数据库名称"

对其收集信息或应用程序连接至的主机数据库的真实名称。这是创建数据库时给定的名称。

表 499. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

host_prdid -"主机产品/版本标识"

正在服务器上运行的产品和版本。

表 500. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 用于标识 DRDA 主机数据库产品及其代码版本号。其格式为 PPPVVRRM，其中:

- PPP 标识主机 DRDA 产品
  - ARI 用于 DB2 服务器 VSE 和 VM 版
  - DSN 用于 DB2 z/OS 版
  - QSQ 用于 DB2 i 版
  - SQL 用于其他 DB2 产品。
- VV 标识两位版本号（版本只有一位时则高位为 0）
- RR 标识两位发行版号（发行版只有一位时高位为 0）
- M 标识 1 个字符的修改级别（0-9 或 A-Z）

host_response_time -"主机响应时间"

在 DCS 语句级别，此项是语句从 DB2 Connect 网关发送至主机以进行处理的时间与从主机接收到结果的时间之间所耗用的时间。在 DCS 数据库和 DCS 应用程序级别，此项是对特定应用程序或数据库执行的所有语句所耗用的时间。在数据传输级别，此项是使用这么多数据传输的所有语句的主机响应时间的总和。

表 501. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

用法 将此元素与发送的出站字节数和接收的出站字节配合使用来计算出站响应时间（传输速率）：

```
(发送的出站字节数 + 接收的出站字节数) / 主机响应时间
```

idle_agents - "空闲代理程序数"

代理程序池中当前未分配给应用程序而"空闲"的代理程序数。

表 502. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 可使用此元素来帮助设置 *num_poolagents* 配置参数。如果具有可用来处理代理程序请求的空闲代理程序，就可以改进性能。

iid - "索引标识"监视元素

索引的标识。

表 503. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

inbound_bytes_received - "接收的入站字节数"

DB2 Connect 网关从客户机接收的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。

表 504. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句

对于应用程序级别的快照监视，可复位此计数器。不能在其他级别复位此计数器。

用法 使用此元素来度量从客户机至 DB2 Connect 网关的吞吐量。

inbound_bytes_sent - "发送的入站字节数"

DB2 Connect 网关发送至客户机的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。

表 505. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句

对于应用程序级别的快照监视，可复位此计数器。不能在其他级别复位此计数器。

用法 使用此元素来度量从 DB2 Connect 网关至客户机的吞吐量。

inbound_comm_address - "入站通信地址"

此项是客户机的通信地址。例如，它可能是 SNA 网络标识和 LU 伙伴名称，或者是 TCP/IP 的 IP 地址和端口号。

表 506. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

include_col_updates - "更新包括列次数"监视元素

更新包括列次数。

表 507. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

index_object_pages - "索引对象页数"

对表定义的所有索引消耗的磁盘页数。

表 508. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 509. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法 此元素提供了一种机制，可用来查看对特定表定义的索引消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内索引的增长率。不会对分区表返回此元素。

index_only_scans - "纯索引扫描次数"监视元素

纯索引扫描次数。如果仅通过访问索引即可获得扫描结果，那么将执行纯索引扫描。

表 510. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

index_scans - "索引扫描次数"监视元素

索引扫描次数。

表 511. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

index_tbsp_id - "索引表空间标识"监视元素

一个表空间的标识，此表空间用于存放对此表创建的索引。

表 512. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

用法

此元素的值与视图 SYSCAT.TABLESPACES 的列 TBSPACEID 中的值相匹配。

input_db_alias - "输入数据库别名"

调用快照函数时提供的数据库别名。

表 513. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_id_info	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁定	db_lock_list	基本

用法 此元素可用来标识监视器数据适用的特定数据库。除非请求与特定数据库相关的监视器信息，否则它将包含空白。

此字段的值可能不同于 *client_db_alias* 监视元素的值，原因是数据库可能具有许多不同别名。不同应用程序和用户可使用不同别名来连接至同一数据库。

insert_sql_stmts - "插入数"

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位以后（取较晚者），联合服务器代表任何应用程序对此数据源发出 INSERT 语句的总次数。

表 514. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法

使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
         (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
         (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
         DELETE 语句数)
```

insert_time - "插入响应时间"

此元素包含此数据源响应 INSERT 所花的总时间（以毫秒计），这些 INSERT 来自联合服务器启动后或数据库监视计数器上一次复位后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。

响应时间是以联合服务器将 INSERT 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 INSERT 的时间之差量度的。

表 515. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

用法

使用此元素来确定等待处理对此数据源执行 INSERT 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

insert_timestamp - "语句插入时间戳记"监视元素

语句插入到高速缓存中的时间。

表 516. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 517. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

用法

此元素指定语句插入到高速缓存中的时间。它可用来估计高速缓存中语句的生存期。

int_auto_rebinds - "内部自动重新绑定次数"

尝试的自动重新绑定（或重新编译）数。

表 518. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 519. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 自动重新绑定是程序包无效时系统执行的内部绑定。当数据库管理器需要从程序包执行 SQL 语句时，将首次执行重新绑定。例如，当您执行以下操作时程序包将变得无效:

- 删除计划所依赖的对象，如表、视图或索引
- 添加或删除外键
- 撤销计划所依赖的对象特权。

可使用此元素来确定应用程序或数据库级别的数据库活动的级别。因为 int_auto_rebinds 对性能有严重影响，所以应尽可能少使用它们。

还可使用此元素并借助以下公式来确定重新绑定活动的百分比:

```
int_auto_rebinds / 语句总数
```

此信息对于分析应用程序活动和吞吐量非常有用。

int_commits - "内部落实数"

数据库管理器在内部启动的落实总数。

表 520. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 521. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 在执行下列任一操作期间可能发生内部落实:

- 重组
- 导入
- 绑定或预编译
- 应用程序在未执行显式 SQL COMMIT 语句的情况下结束（在 UNIX 上）。

此值（不包括显式 SQL COMMIT 语句）表示自出现下列情况后发生的内部落实数:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将仅包括发生以下情况之后出现的工作单元:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

此计算可在应用程序或数据库级别完成。

int_deadlock_rollbacks - “死锁导致的内部回滚数”

数据库管理器因为死锁而启动的强制回滚总数。为解决死锁，将对数据库管理器选择的应用程序中的当前工作单元执行回滚。

元素标识

int_deadlock_rollbacks

元素类型

计数器

表 522. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 522. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

可将快照监视的计数器复位。

表 523. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 此元素显示已中断并且可用作并行性问题的指示符的死锁数。这很重要，原因是 int_deadlock_rollbacks 会降低数据库的吞吐量。

此值包括在 int_rollbacks 给定的值中。

int_node_splits - "中间节点分割次数"监视元素

在插入操作期间分割中间索引节点的次数。

表 524. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

int_rollbacks - "内部回滚数"

数据库管理器在内部启动的回滚总数。

元素标识

int_rollbacks

元素类型

计数器

表 525. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 526. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 当下列任何一项**不能**成功完成时，将发生内部回滚:

- 重组
- 导入

- 绑定或预编译
- 应用程序因为出现死锁或锁定超时而结束
- 应用程序在未执行显式落实或回滚语句的情况下结束（在 Windows 上）。

此值表示自发生下列情况之后出现的内部回滚数:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

虽然此值不包括显式 SQL ROLLBACK 语句，但包括 int_deadlock_rollbacks 中的计数。

可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将包括发生以下情况之后出现的工作单元:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次复位。

此计算可在应用程序或数据库级别完成。

int_rows_deleted －“删除的内部行数”

此项是因为内部活动而从数据库中删除的行数。

表 527. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 528. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

用法 此元素可帮助了解您不熟悉的数据库管理器的内部活动。如果此活动很多，那么您可能想要评估表设计以确定对数据库定义的引用约束或触发器是否必要。

内部删除活动可能是由下列原因引起的:

- 强制实施 ON CASCADE DELETE 引用约束的级联删除
- 被触发的触发器。

int_rows_inserted – “插入的内部行数”

因为触发器导致的内部活动而插入到数据库中的行数。

表 529. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 530. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

用法 此元素可帮助您了解数据库管理器的内部活动。如果此活动很多，那么您可能想要评估设计以确定是否可以更改它以降低此活动。

int_rows_updated – “更新的内部行数”

此项是因为内部活动而从数据库中更新的行数。

表 531. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 532. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-

用法 此元素可帮助了解您不熟悉的数据库管理器的内部活动。如果此活动很多，那么您可能想要评估表设计以确定对数据库定义的引用约束是否必要。

内部更新活动可能是由下列原因引起的:

- 强制使用 ON DELETE SET NULL 规则定义的引用约束的 *set null* 行更新
- 被触发的触发器。

invocation_id - “调用标识”监视元素

此标识用于将此活动的一次特定调用与同一嵌套级别的其他调用区分开。

表 533. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

ipc_recv_volume - “进程间通信接收量”监视元素

数据服务器通过 IPC 从客户机接收的数据量。此值以字节计。

表 534. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 535. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_recv_wait_time - "进程间通信接收等待时间"监视元素

代理程序使用 IPC 通信协议接收传入客户机请求时耗用的时间。此值以毫秒计。

表 536. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 537. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_recvs_total - "进程间通信接收总次数"监视元素

数据库服务器使用 IPC 从客户机应用程序接收数据的次数。

表 538. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 538. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 539. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_send_volume - “进程间通信发送量”监视元素

数据服务器通过 IPC 协议发送到客户机的数据量。此值以字节计。

表 540. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 540. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 541. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_send_wait_time - “进程间通信发送等待时间”监视元素

通过 IPC 向客户机发送数据时被阻塞的时间。此值以毫秒计。

表 542. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 543. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_sends_total - "进程间通信发送总次数"监视元素

数据库服务器使用 IPC 向客户机应用程序发送数据的次数。

表 544. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 545. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

is_system_appl - "是系统应用程序"监视元素

指示应用程序是否是系统应用程序。

表 546. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

用法

is_system_appl 监视元素指示应用程序是否是内部系统应用程序。可能的值包括

0 用户应用程序

1 系统应用程序

key_updates - "更新键次数"监视元素

更新键次数。

表 547. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

last_active_log - "最后一个活动日志文件编号"

最后一个活动日志文件的文件号。

元素标识

last_active_log

元素类型

信息

表 548. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 549. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *first_active_log* 和 *current_active_log* 元素一起使用来确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

last_backup - "上次备份时间戳记"

完成上次数据库备份的日期和时间。

元素标识

last_backup

元素类型

时间戳记

表 550. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	时间戳记

用法 可使用此元素来帮助您标识最近没有备份的数据库，或者标识最新的数据库备份文件。如果数据库从未进行备份，那么此时间戳记将初始化为零。

last_extent - "移动的最后一个扩展数据块"监视元素

表空间重新平衡程序过程所移动的最后一个扩展数据块的数字标识。

表 551. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	始终收集

last_overflow_time - "最后一次事件溢出时间"

此溢出记录的最后一次溢出的日期和时间。

表 552. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

用法 将此元素与 *first_overflow_time* 配合使用来计算生成溢出记录所耗用的时间。

last_reference_time - "上次引用时间"监视元素

此活动上次被某个请求访问时的时间。

表 553. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

last_reset - "最后复位时间戳记"

指示监视器计数器对发出 GET SNAPSHOT 的应用程序复位的日期和时间。

元素标识

last_reset

元素类型

时间戳记

表 554. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	时间戳记
数据库	dbase	时间戳记
应用程序	appl	时间戳记
表空间	tablespace_list	缓冲池，时间戳记
表	table_list	时间戳记
DCS 数据库	dcs_dbase	时间戳记
DCS 应用程序	dcs_appl	时间戳记

用法 可使用此元素来帮助定义数据库系统监视器返回的信息的作用域。

如果计数器从未复位，那么此元素将为零。

仅当复位所有活动数据库时，数据库管理器计数器才会复位。

last_wlm_reset －“最后一次复位时间”监视元素

此元素为本地时间戳记形式，显示创建此类型的最后一个统计信息事件记录的时间。

表 555. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-
统计信息	event_wcstats	-
统计信息	event_qstats	-

用法

使用 **wlm_last_reset** 和 **statistics_timestamp** 监视元素来确定收集事件监视器统计信息记录中的统计信息的时间段。收集时间间隔从 **wlm_last_reset** 时间开始，并且在 **statistics_timestamp** 结束。

lob_object_pages －“LOB 对象页数”

LOB 数据消耗的磁盘页数。

表 556. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 557. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法 此元素提供了一种机制，可用来查看特定表中的 LOB 数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内 LOB 数据的增长率。

local_cons -“本地连接数”

当前连接至正在监视的数据库管理器实例中的数据库的本地应用程序数。

表 558. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此数目可帮助您确定数据库管理器中进行的并行处理的级别。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。

此数目仅包括从与数据库管理器相同的实例启动的应用程序。这些应用程序将进行连接，但它们可能执行也可能不执行数据库中的工作单元。

与 rem_cons_in 监视元素一起使用时，此元素可帮助您调整 **max_connections** 配置参数的设置。

local_cons_in_exec -“数据库管理器中正在执行的本地连接数”

当前连接至正在监视的数据库管理器实例中的数据库并且正在处理工作单元的本地应用程序数。

表 559. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此数目可帮助您确定数据库管理器中进行的并行处理的级别。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目仅包括从与数据库管理器相同的实例启动的应用程序。

与 **rem_cons_in_exec** 监视元素使用使用时，此元素可帮助您调整 **max_coordagents** 配置参数的设置。

以下建议仅适用于非集中器配置。如果启用了集中器，DB2 会将大量客户机连接多路复用到较小的协调代理程序池。在这种情况下，通常可以使 **rem_cons_in_exec** 与 **local_cons_in_exec** 之和接近 **max_coordagents** 值。

- 如果 **max_coordagents** 设置为 AUTOMATIC，那么不要作任何调整。
- 如果 **max_coordagents** 未设置为 AUTOMATIC，并且 **rem_cons_in_exec** 与 **local_cons_in_exec** 之和接近 **max_coordagents**，那么请增加 **max_coordagents** 的值。

local_start_time – “本地开始时间”监视元素

此活动开始在分区中处理工作时的时间。这是本地时间。如果此活动已进入系统，但由于正在队列中排队而尚未开始执行，那么此字段是空字符串。

表 560. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

lock_attributes – “锁定属性”监视元素

锁定属性。

表 561. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	基本

表 562. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

以下是可能的锁定属性设置。每个锁定属性设置都以 sqlmon.h 中定义的位标志值为基础。

API 常量	描述
SQLM_LOCKATTR_WAIT_FOR_AVAIL	等待可用性。
SQLM_LOCKATTR_ESCALATED	由升级获取。
SQLM_LOCKATTR_RR_IN_BLOCK	块中的 RR 锁定。
SQLM_LOCKATTR_INSERT	插入锁定。
SQLM_LOCKATTR_DELETE_IN_BLOCK	块中的已删除行。
SQLM_LOCKATTR_RR	由 RR 扫描锁定。
SQLM_LOCKATTR_UPDATE_DELETE	更新/删除行锁定。
SQLM_LOCKATTR_ALLOW_NEW	允许新锁定请求。
SQLM_LOCKATTR_NEW_REQUEST	新锁定请求者。

API 常量	描述
SQLM_LOCKATTR_INDOUBT	由不确定事务挂起的锁定。
SQLM_LOCKATTR_LOW_PRIORITY	由低优先级应用程序挂起的锁定。

lock_count - "锁定计数"监视元素

正在挂起的锁定上的锁定数目。

表 563. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本

表 564. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此值的范围在 1 到 255 之间。获得新锁定时，此值递增；释放锁定时，此值递减。

lock_count 监视元素的值为 255 时，指示正在挂起事务持续时间锁定。此时，获得或释放锁定时，**lock_count** 不再递增或递减。在下列任一可能方式下，**lock_count** 监视元素的值设置为 255:

1. **lock_count** 监视元素值因为获取新锁定而递增 255 次。
2. 显式获得事务持续时间锁定。例如使用 LOCK TABLE 语句或 INSERT 语句。

lock_current_mode - "转换前的原始锁定方式"

在锁定转换操作期间，完成转换前挂起的锁定的类型。

表 565. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	基本

表 566. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

以下方案描述锁定转换示例。在更新或删除操作期间，可以等待针对目标行的 X 锁定。如果事务要挂起针对行的 S 或 V 锁定，那么需要转换。此时将对 **lock_current_mode** 元素指定值 S 或 V，而锁定等待转换为 X 锁定。

lock_escalation - "锁定升级"监视元素

指示锁定请求是否作为锁定升级的组成部分。

表 567. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	锁定
锁定	lock_wait	锁定

表 568. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

使用此元素来更好地理解死锁的原因。如果遇到涉及执行锁定升级的应用程序的死锁，那么您可能想要增加锁定内存量或更改任一应用程序可请求的锁定百分比。

lock_escals - "锁定升级次数"监视元素

锁定已从若干行锁定升级至表锁定的次数。

表 569. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 569. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 570. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 571. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
事务	event_xact	-

用法 当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量，或者所有应用程序消耗的锁定列表空间达到总锁定列表空间时，锁定将会升级。可用锁定列表空间量由 **maxlocks** 和 **locklist** 配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，它将使用锁定列表中为其他应用程序分配的空间。当整个锁定列表已满时，将发生错误。

此数据项包含包括互斥锁定升级在内的所有锁定升级的计数。

以下几种原因可能会导致产生过量锁定升级:

- 锁定列表大小（**locklist**）对于并行应用程序数目而言可能太小
- 可供每个应用程序使用的锁定列表百分比（**maxlocks**）可能太小
- 一个或多个应用程序使用的锁定数可能过量。

要解决这些问题，可以:

- 增加 **locklist** 配置参数值。
- 增加 **maxlocks** 配置参数值。
- 确定执行大量锁定的应用程序（参见 **locks_held_top** 监视元素），或者使用以下公式来确定过多占用锁定列表的应用程序:

  ```
  (((locks held * 36) / (locklist * 4096)) * 100)
  ```

  然后，将结果值与 **maxlocks** 进行比较。这些应用程序还可能因为在锁定列表中使用过量资源而导致其他应用程序中发生锁定升级。这些应用程序可能需要使用表锁定来代替行锁定，尽管表锁定可能会导致 **lock_waits** 和 **lock_wait_time** 监视元素的值增大。

lock_hold_count - “锁定挂起计数”监视元素

挂起锁定的次数。使用 WITH HOLD 子句和一些 DB2 实用程序注册的游标放置在锁定上的挂起数。落实事务时，不会释放带有挂起的锁定。

表 572. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本

表 573. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

lock_list_in_use - “使用中的锁定列表内存总量”

使用中的锁定列表内存总量（以字节计）。

元素标识

lock_list_in_use

元素类型

水位标记

表 574. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 此元素可以与 *locklist* 配置参数配合使用以计算锁定列表利用率。如果锁定列表利用率很高，那么您可能要考虑增大该参数。

注: 在计算利用率时，注意下面这一点十分重要: *locklist* 配置参数是以 4K 字节页为单位分配的，而此监视元素提供的结果以字节计。

lock_mode - "锁定方式"监视元素

正在挂起的锁定的类型。

表 575. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	lock	锁定
锁定	lock_wait	锁定

表 576. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此方式可帮助您确定资源争用的源头。

根据要检查的监视器信息的类型，此元素指示下列其中一种情况:

- 另一应用程序针对此应用程序等待锁定的对象挂起的锁定类型（对于应用程序监视级别和死锁监视级别）。
- 此应用程序针对对象挂起的锁定类型（对于对象锁定级别）。

此字段的值包括:

方式	锁定类型	API 常量
	没有锁定	SQLM_LNON
IS	意向共享锁定	SQLM_LOIS
IX	意向互斥锁定	SQLM_LOIX
S	共享锁定	SQLM_LOOS

方式	锁定类型	API 常量
SIX	与意向互斥锁定共享	SQLM_LSIX
X	互斥锁定	SQLM_LOOX
IN	无任何意向	SQLM_LOIN
Z	超级互斥锁定	SQLM_LOOZ
U	更新锁定	SQLM_LOOU
NS	扫描共享锁定	SQLM_LONS
NW	下一键弱互斥锁定	SQLM_LONW

lock_mode_requested - "请求的锁定方式"监视元素

应用程序请求的锁定方式。

表 577. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock_wait	锁定

表 578. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

应用程序请求的锁定方式。此值可帮助您确定资源争用的源头。

lock_name - "锁定名称"监视元素

内部二进制锁定名称。此元素将充当锁定的唯一标识。

表 579. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	lock_wait

表 580. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

lock_node - "锁定节点"

锁定涉及的节点。

表 581. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句
死锁	event_dlconn	语句
带有详细信息的死锁	event_detailed_dlconn	语句

用法 它可以用于故障诊断。

lock_object_name - "锁定对象名称"

此元素仅供参考。它是应用程序对其挂起锁定的对象的名称（对于对象锁定级别信息），或者应用程序等待对其获取锁定的对象的名称（对于应用程序级别和死锁级别信息）。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 582. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock	基本

表 583. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

用法 对于表级别锁定，该对象名是 SMS 和 DMS 表空间的文件标识（FID）。对于行级别锁定，该对象名是行标识（RID）。对于表空间锁定，该对象名留为空白。对于缓冲池锁定，该对象名是缓冲池的名称。

要确定挂起锁定的表，请使用 *table_name* 和 *table_schema* 而不是文件标识，原因是文件标识可能不是唯一的。

要确定挂起锁定的表空间，请使用 *tablespace_name*。

lock_object_type - "等待的锁定对象类型"监视元素

应用程序对其挂起锁定的对象的类型（对于对象锁定级别信息），或者应用程序等待对其获取锁定的对象的类型（对于应用程序级别和死锁级别信息）。

表 584. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock	基本
锁定	lock_wait	锁定

表 585. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此元素可帮助您确定资源争用的源头。

对象类型标识是在 sqlmon.h 中定义的。对象可能是下列其中一种类型:

- 表空间（sqlmon.h 中的 SQLM_TABLESPACE_LOCK）
- 表
- 缓冲池
- 块
- 记录（或行）
- 数据分区（sqlmon.h 中的 SQLM_TABLE_PART_LOCK）
- 内部（数据库管理器内部挂起的另一类型的锁定）
- 自动调整大小
- 自动存储器。

lock_release_flags - "锁定释放标志"监视元素

锁定释放标志。

表 586. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	基本

表 587. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

以下是可能的释放标志设置。每个释放标志都以 sqlmon.h 中定义的位标志值为基础。

API 常量	描述
SQLM_LOCKRELFLAGS_SQLCOMPILER	SQL 编译器进行的锁定。
SQLM_LOCKRELFLAGS_UNTRACKED	非唯一的未记录锁定。

注: 所有非指定位将用于应用程序游标。

lock_status – “锁定状态”监视元素

指示锁定的内部状态。

表 588. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本

表 589. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此元素可帮助说明应用程序等待获取对某个对象的锁定时所发生的情况。虽然可能已经具有对所需对象的锁定，但应用程序必须等待以获取对同一对象的另一类型的锁定。

锁定可以是下列其中一种状态:

已授予状态
应用程序的锁定处于 **lock_mode** 监视元素所指定状态的锁定。

转换状态

应用程序正在尝试将拥有的锁定更改为另一类型；例如，从共享锁定更改为互斥锁定。

注： API 用户应参考包含数据库系统监视器常量定义的 sqlmon.h 头文件。

lock_timeout_val －“锁定超时值”监视元素

指示应用程序发出 SET CURRENT LOCK TIMEOUT 语句时的超时值（以秒计）。如果未执行语句，那么将显示数据库级别锁定超时。

表 590. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	agent	基本

表 591. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

用法

可使用 SET CURRENT LOCK TIMEOUT 语句来指定应用程序代理程序等待表或索引锁定的最长持续时间。

如果应用程序等待锁定的时间过长，那么可以检查 **lock_timeout_val** 监视元素值以了解该值在应用程序中是否设置得过高。如果符合应用程序逻辑，那么可以修改应用程序以降低锁定超时值，从而允许应用程序超时。可使用 SET CURRENT LOCK TIMEOUT 语句来完成此修改。

如果应用程序经常超时，那么请检查锁定超时值是否设置得过低，并在适当时增大该值。

lock_timeouts －“锁定超时次数”监视元素

要锁定对象的请求由于发生超时而未获得锁定的次数。

表 592. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 － 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 － 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 － 获取服务子类度量值	REQUEST METRICS BASE

表 592. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 593. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 594. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-

用法

此元素可帮助您调整 **locktimeout** 数据库配置参数的设置。如果锁定超时次数与正常操作水平相比显得过多，那么可能表明应用程序挂起锁定的时间过长。在这种情况下，此元素可能指示应该分析某些其他锁定和死锁监视元素，以确定是否存在应用程序问题。

如果 **locktimeout** 数据库配置参数设置得过大，那么锁定超时情况将会过少。在这种情况下，应用程序在获取锁定前的等待时间将会过长。

lock_wait_start_time －“锁定等待开始时间戳记”

此应用程序开始等待获取锁定的日期和时间，该锁定针对当前被另一应用程序锁定的对象。

元素标识

lock_wait_start_time

元素类型

时间戳记

表 595. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定，时间戳记
锁定	lock_wait	锁定，时间戳记

表 596. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	时间戳记
带有详细信息的死锁	event_detailed_dlconn	时间戳记

用法 此元素可帮助您确定资源争用的严重性。

lock_wait_time －“等待锁定时间”监视元素

等待锁定的耗用时间总计。此值以毫秒计。

表 597. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 － 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 － 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 － 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 － 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 597. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 598. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	锁定
应用程序	appl	锁定
锁定	appl_lock_list	appl_lock_list

可将快照监视的计数器复位。

表 599. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
事务	event_xact	-

用法 在数据库级别，此项表示所有应用程序在此数据库中等待锁定的总次数。

在应用程序连接和事务级别，此项表示此连接或事务等待授予锁定的耗用时间总计。

此元素的值不包括当前仍处于锁定等待状态的代理程序的锁定等待时间。它仅包括已完成锁定等待的代理程序的锁定等待时间。

此元素可与 **lock_waits** 监视元素一起使用以计算锁定的平均等待时间。此计算可在数据库级别或应用程序连接级别执行。

使用提供耗用时间的监视元素时，应考虑:

- 耗用时间受系统负载影响，所以运行的进程越多，此耗用时间值越高。

- 要在数据库级别计算此元素，数据库系统监视器会将应用程序级别时间求和。这会导致对数据库级别的耗用时间双倍计数，原因是多个应用程序进程可能同时运行。

  为提供有意义的数据，可按上述计算锁定的平均等待时间。

lock_wait_time_top －“锁定等待时间顶部”监视元素

工作负载中任何请求的锁定等待时间的高水位标记。单位为毫秒。系统始终对工作负载收集 lock_wait_time_top 高水位标记。仅当启用了请求度量值时，请求才会影响此高水位标记。

表 600. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-

用法

使用此元素来确定在收集时间间隔内，分区中某个工作负载的任何请求的最大锁定等待时间。

lock_waits －“等待锁定次数”监视元素

应用程序或连接等待锁定的总次数。

表 601. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 － 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 － 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 － 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 － 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 － 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 － 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 601. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 602. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 603. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-

用法 在数据库级别，此项表示应用程序在数据库中等待锁定的总次数。

在应用程序连接级别，此项表示此连接请求锁定但必须等待（因为另一连接已经挂起针对该数据的锁定）的总次数。

此元素可与 **lock_wait_time** 配合使用以在数据库级别计算锁定的平均等待时间。此计算可在数据库级别或应用程序连接级别完成。

如果平均锁定等待时间很长，那么应查找挂起许多锁定或具有锁定升级的应用程序，并把重点放在调整应用程序以改进并行性上（如果适当）。如果平均锁定等待时间很长是升级导致的，那么 **locklist** 和/或 **maxlocks** 配置参数的值可能太低了。

locks_held - “挂起的锁定数”

当前挂起的锁定数目。

元素标识

locks_held

元素类型

标尺

表 604. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
锁定	db_lock_list	基本
锁定	appl_lock_list	基本

表 605. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-

用法 如果监视器信息为数据库级别，那么此项表示数据库中的所有应用程序当前挂起的总锁定数。

如果监视器信息为应用程序级别，那么此项表示应用程序的所有代理程序当前挂起的总锁定数。

locks_held_top - "挂起的最大锁定数"

此事务期间挂起的最大锁定数。

元素标识

locks_held_top

元素类型

计数器

表 606. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

用法 可使用此元素来确定应用程序是否达到 *maxlocks* 配置参数定义的最大可用锁定数。此参数指示发生锁定升级之前每个应用程序可以使用的锁定列表百分比。锁定升级可能导致连接至数据库的应用程序之间的并行度下降。

因为 *maxlocks* 参数被指定为百分比并且此元素为计数器，所以可以借助以下公式进行计算以将此元素提供的计数与应用程序可挂起的总锁定数进行比较:

```
(locklist * 4096 / 36 ) * (maxlocks / 100)
```

如果您有大量锁定，那么可能需要在应用程序中执行更多落实操作以便可以释放一些锁定。

locks_in_list - "报告的锁定数"

事件监视器要报告的特定应用程序挂起的锁定数。

表 607. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	-

locks_waiting - "等待锁定的当前代理程序数"

指示等待锁定的代理程序数。

表 608. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
锁定	db_lock_list	基本

用法 与 **appls_cur_cons** 一起使用时，此元素指示等待锁定的应用程序百分比。如果此数字很高，那么表示应用程序可能具有并行性问题，您应该标识长时间挂起锁定或互斥锁定的应用程序。

log_buffer_wait_time - "日志缓冲区等待时间"监视元素

代理程序等待日志缓冲区空间时耗用的时间。此值以毫秒计。

表 609. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 610. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

log_disk_wait_time - "日志磁盘等待时间"监视元素

代理程序等待日志记录被清仓到磁盘时耗用的时间。此值以毫秒计。

表 611. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 612. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 612. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

log_disk_waits_total - "日志磁盘等待总次数"监视元素

代理程序被迫等待日志数据被写入磁盘的次数。

表 613. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 614. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 614. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

log_held_by_dirty_pages -“脏页占用的日志空间量”

对应数据库中的最旧脏页与活动日志顶部之间的差的日志量（以字节计）。

元素标识

log_held_by_dirty_pages

元素类型

水位标记

表 615. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 616. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 获取快照时，将根据该快照的时间上的条件计算此值。

使用此元素来评估对缓冲池中的旧页进行页清除的效果。

缓冲池中的旧页清除由 *softmax* 数据库配置参数控制。如果页清除有效果，那么 *log_held_by_dirty_pages* 应小于或大致等于:

```
(softmax / 100) * logfilsiz * 4096
```

如果不是这样，那么增加页清除程序的数目（*num_iocleaners*）配置参数。

如果符合条件并且期望脏页占用的日志少一些，那么降低 *softmax* 配置参数。

log_read_time -“日志读取时间”

记录器从磁盘读取日志数据的耗用时间总计。

表 617. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 618. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *log_reads*、*num_log_read_io* 和 *num_log_data_found_in_buffer* 元素一起使用来确定以下情况是否属实:

- 当前磁盘对于记录操作而言是够用的。
- 日志缓冲区大小够用。

log_reads - "读取的日志页数"

记录器从磁盘读取的日志页数。

元素标识

log_reads

元素类型

计数器

表 619. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 620. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 可将此元素与操作系统监视器配合使用来确定设备上可归因于数据库活动的 I/O 量。

log_to_redo_for_recovery - "要为恢复重做的日志量"

必须为崩溃恢复重做的日志量(以字节计)。

元素标识

log_to_redo_for_recovery

元素类型

水位标记

表 621. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 622. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 获取快照时,将根据该快照的时间上的条件计算此值。如果值较大,那么指示系统崩溃后的恢复时间较长。如果值看起来过大,那么检查 *log_held_by_dirty_pages* 监视元素以了解是否需要调整页清除。还应检查是否存在任何长时间运行并且需要终止的事务。

log_write_time - "日志写入时间"

记录器将日志数据写至磁盘的耗用时间总计。

表 623. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 624. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *log_writes* 和 *num_log_write_io* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

log_writes - "写入的日志页数"

记录器写至磁盘的日志页数。

元素标识

log_writes

元素类型

计数器

表 625. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 626. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 可将此元素与操作系统监视器配合使用来确定设备上可归因于数据库活动的 I/O 量。

注: 当日志页写至磁盘时，最后一页可能未满。在这种情况下，部分日志页保留在日志缓冲区中，而其他日志记录将写至页。因此记录器可能会多次将日志页写至磁盘。不应使用此元素来量度 DB2 生成的页数。

long_object_pages - "长对象页数"

表中的长数据消耗的磁盘页数。

表 627. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 628. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法 此元素提供了一种机制，可用来查看特定表中的长数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内长数据的增长率。

long_tbsp_id - "长表空间标识"监视元素

一个表空间的标识，此表空间用于存放此表的长数据（LONG 或 LOB 类型的列）。

表 629. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

用法

此元素的值与视图 SYSCAT.TABLESPACES 的列 TBSPACEID 中的值相匹配。

max_agent_overflows - "最大代理程序溢出次数"

达到"最大代理程序数"（**maxagents**）配置参数后接收到创建新代理程序的请求的次数。

注: 从 DB2 版本 9.5 开始，不推荐使用 **max_agent_overflows** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 630. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

如果达到 **maxagents** 配置参数后仍然接收到代理程序创建请求，那么可能指示此节点的工作负载太高。

max_data_received_1024 - "接收的出站字节数在 513 到 1024 字节之间的语句数"

此元素表示接收的出站字节数在 513 到 1024 字节之间（包括 513 字节和 1024 字节）的语句或链的数目。

元素标识

max_data_received_1024

元素类型

计数器

表 631. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_128 - "接收的出站字节数在 1 到 128 字节之间的语句数"

此元素表示接收的出站字节数在 1 到 128 字节之间（包括 1 字节和 128 字节）的语句或链的数目。

元素标识

max_data_received_128

元素类型

计数器

表 632. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_16384 - "接收的出站字节数在 8193 到 16384 字节之间的语句数"

此元素表示接收的出站字节数在 8193 到 16384 字节之间（包括 8193 字节和 16384 字节）的语句或链的数目。

元素标识

max_data_received_16384

元素类型

计数器

表 633. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_2048 - "接收的出站字节数在 1025 到 2048 字节之间的语句数"

此元素表示接收的出站字节数在 1025 到 2048 字节之间（包括 1025 字节和 2048 字节）的语句或链的数目。

元素标识

max_data_received_2048

元素类型

计数器

表 634. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_256 - "接收的出站字节数在 129 到 256 字节之间的语句数"

此元素表示接收的出站字节数在 129 到 256 字节之间（包括 129 字节和 256 字节）的语句或链的数目。

元素标识

max_data_received_256

元素类型

计数器

表 635. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_31999 - "接收的出站字节数在 16385 到 31999 字节之间的语句数"监视元素

此元素表示接收的出站字节数在 16385 到 31999 字节之间（包括 16385 字节和 31999 字节）的语句或链的数目。

表 636. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_4096 - "接收的出站字节数在 2049 到 4096 字节之间的语句数"

此元素表示接收的出站字节数在 2049 到 4096 字节之间（包括 2049 字节和 4096 字节）的语句或链的数目。

元素标识

max_data_received_4096

元素类型

计数器

表 637. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_512 - "接收的出站字节数在 257 到 512 字节之间的语句数"

此元素表示接收的出站字节数在 257 到 512 字节之间（包括 257 字节和 512 字节）的语句或链的数目。

元素标识

max_data_received_512

元素类型

计数器

表 638. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_64000 - "接收的出站字节数在 32000 到 64000 字节之间的语句数"监视元素

此元素表示接收的出站字节数在 32000 到 64000 字节之间（包括 32000 字节和 64000 字节）的语句或链的数目。

表 639. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_8192 - "接收的出站字节数在 4097 到 8192 字节之间的语句"

此元素表示接收的出站字节数在 4097 到 8192 字节之间（包括 4097 字节和 8192 字节）的语句或链的数目。

元素标识

max_data_received_8192

元素类型

计数器

表 640. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_gt64000 - "接收的出站字节数高于 64000 的语句数"

此元素表示接收的出站字节数高于 64000 的语句或链的数目。

元素标识

max_data_received_gt64000

元素类型

计数器

表 641. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_1024 - "发送的出站字节数在 513 到 1024 字节之间的语句数"

此元素表示发送的出站字节数在 513 到 1024 字节之间（包括 513 字节和 1024 字节）的语句或链的数目。

元素标识

max_data_sent_1024

元素类型

计数器

表 642. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_128 - "发送的出站字节数在 1 到 128 字节之间的语句数"

此元素表示发送的出站字节数在 1 到 128 字节之间（包括 1 字节和 128 字节）的语句或链的数目。

元素标识

max_data_sent_128

元素类型

计数器

表 643. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_16384 - "发送的出站字节数在 8193 到 16384 字节之间的语句数"

此元素表示发送的出站字节数在 8193 到 16384 字节之间（包括 8193 字节和 16384 字节）的语句或链的数目。

元素标识

max_data_sent_16384

元素类型

计数器

表 644. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_2048 - "发送的出站字节数在 1025 到 2048 字节之间的语句数"

此元素表示发送的出站字节数在 1025 到 2048 字节之间（包括 1025 字节和 2048 字节）的语句或链的数目。

元素标识

max_data_sent_2048

元素类型

计数器

表 645. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句

表 645. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_256 - "发送的出站字节数在 129 到 256 字节之间的语句数"

此元素表示发送的出站字节数在 129 到 256 字节之间（包括 129 字节和 256 字节）的语句或链的数目。

元素标识

max_data_sent_256

元素类型

计数器

表 646. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_31999 - "发送的出站字节数在 16385 到 31999 字节之间的语句数"

此元素表示发送的出站字节数在 16385 到 31999 字节之间（包括 16385 字节和 31999 字节）的语句或链的数目。

元素标识

max_data_sent_31999

元素类型

计数器

表 647. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_4096 - "发送的出站字节数在 2049 到 4096 字节之间的语句数"

此元素表示发送的出站字节数在 2049 到 4096 字节之间（包括 2049 字节和 4096 字节）的语句或链的数目。

元素标识

max_data_sent_4096

元素类型

计数器

表 648. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_512 - "发送的出站字节数在 257 到 512 字节之间的语句数"

此元素表示发送的出站字节数在 257 到 512 字节之间（包括 257 字节和 512 字节）的语句或链的数目。

元素标识

max_data_sent_512

元素类型

计数器

表 649. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_64000 - "发送的出站字节数在 32000 到 64000 字节之间的语句数"

此元素表示发送的出站字节数在 32000 到 64000 字节之间（包括 32000 字节和 64000 字节）的语句或链的数目。

元素标识

max_data_sent_64000

元素类型

计数器

表 650. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_8192 - "发送的出站字节数在 4097 到 8192 字节之间的语句数"

此元素表示发送的出站字节数在 4097 到 8192 字节之间（包括 4097 字节和 8192 字节）的语句或链的数目。

元素标识

max_data_sent_8192

元素类型

计数器

表 651. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_gt64000 - "发送的出站字节数高于 64000 的语句数"

此元素表示发送的出站字节数高于 64000 的语句或链的数目。

元素标识

max_data_sent_gt64000

元素类型

计数器

表 652. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_100_ms -“网络时间在 16 到 100 毫秒之间的语句数”

此元素表示网络时间大于 16 毫秒但小于或等于 100 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

表 653. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_16_ms -“网络时间在 4 到 16 毫秒之间的语句数”

此元素表示网络时间大于 4 毫秒但小于或等于 16 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

元素标识

max_network_time_16_ms

元素类型

计数器

表 654. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_1_ms - “网络时间最多为 1 毫秒的语句数”

此元素表示网络时间小于或等于 1 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

表 655. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_4_ms - “网络时间在 1 到 4 毫秒之间的语句数”

此元素表示网络时间大于 1 毫秒但小于或等于 4 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

元素标识

max_network_time_4_ms

元素类型

计数器

表 656. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_500_ms - “网络时间在 100 到 500 毫秒之间的语句数”

此元素表示网络时间大于 100 毫秒但小于或等于 500 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

表 657. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_gt500_ms - "网络时间大于 500 毫秒的语句数"

此元素表示网络时间大于 500 毫秒的语句或链的数目。(网络时间是主机响应时间与语句或链所耗用的执行时间之差。)

表 658. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

member - "数据库成员"监视元素

数据库成员的数字标识，此结果记录的数据接收自该成员。

表 659. 表函数监视信息

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	始终收集
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	始终收集
MON_GET_CONNECTION 表函数 - 获取连接度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	始终收集
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	始终收集
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	始终收集
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	始终收集

表 659. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	始终收集
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

表 660. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

DB2 成员是在单一主机上运行 DB2 服务器软件的数据库管理器实例。DB2 成员接受并处理来自与其相连接的应用程序的数据库请求。

message - "控制表消息"

MESSAGE_TIME 列中的时间戳记特征。此元素仅供写至表事件监视器在 CONTROL 表中使用。

表 661. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	-

用法

以下是可能的值:

DROPPED RECORDS: *n*
因为无法为其分配 MONHEAP 而被删除的活动记录的数目。

FIRST_CONNECT
数据库激活后第一次连接至数据库的时间。

EVMON_START
EVMONNAME 列中列示的事件监视器启动的时间。

OVERFLOWS: *n*
说明因为缓冲区溢出而删除了 *n* 个记录。

LAST DROPPED RECORD

上次删除活动记录的时间。

message_time - "时间戳记控制表消息"

对应于 MESSAGE 列中描述的事件的时间戳记。此元素仅供写至表事件监视器在 CONTROL 表中使用。

表 662. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	-

nesting_level - "嵌套级别"监视元素

此元素代表此活动的嵌套级别。嵌套级别是此活动在其最顶部父活动中的嵌套深度。

表 663. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

network_time_bottom - "语句的最短网络时间"

此元素表示对此 DCS 数据库或在此 DCS 应用程序中执行语句的最短网络时间，或者表示执行使用这么多数据传输的语句的最短网络时间。（网络时间是主机响应时间与语句所耗用的执行时间之差。）

元素标识

network_time_bottom

元素类型

水位标记

表 664. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

network_time_top - "语句的最长网络时间"

此元素表示对此 DCS 数据库或在此 DCS 应用程序中执行语句的最长网络时间，或者表示执行使用这么多数据传输的语句的最长网络时间。（网络时间是主机响应时间与语句所耗用的执行时间之差。）

元素标识

network_time_top

元素类型

水位标记

表 665. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

可将快照监视的计数器复位。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。注意，时间戳记开关设置为 OFF 时不收集此元素。

nleaf - "叶子页数"监视元素

大致的叶子页数。

表 666. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

nlevels - "索引层数"监视元素

索引层数。这是近似值。

表 667. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

node_number - "节点号"

在 *db2nodes.cfg* 文件中指定给节点的编号。

表 668. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本
数据库管理器	memory_pool	基本
数据库管理器	fcm	基本

表 668. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本
数据库管理器	utility_info	基本
数据库	detail_log	基本
缓冲池	bufferpool_nodeinfo	缓冲池
表空间	rollforward	基本
锁定	lock	基本
锁定	lock_wait	基本
数据库	db_sto_path_info	缓冲池

表 669. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-
死锁	lock	-
溢出记录	event_overflow	-
数据库	event_dbmemuse	-
连接	event_connmemuse	-

用法 此值标识当前节点号，在监视多个节点时可使用节点号。

nonboundary_leaf_node_splits - "非边界叶节点分割次数"监视元素

插入操作期间分割非边界叶节点的次数。

表 670. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

num_agents - "正在处理语句的代理程序数"

当前执行语句或子节的并行代理程序数。

元素标识

num_agents

元素类型

标尺

表 671. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
应用程序	subsection	语句

用法 指示查询并行度的指示符。此项对于通过获取连续快照来跟踪查询执行进度非常有用。

num_assoc_agents - "关联代理程序数"

在应用程序级别，此项是与应用程序相关联的子代理程序数。在数据库级别，此项是用于所有应用程序的子代理程序数。

元素标识

num_assoc_agents

元素类型

标尺

表 672. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_info	基本

用法 可使用此元素来帮助您评估代理程序配置参数的设置。

num_compilations - "语句编译次数"

特定 SQL 语句的不同编译的次数。

表 673. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

用法 对于某些对不同模式发出的 SQL 语句（例如"select t1 from foo"）来说，尽管它们引用不同的访问方案，但却表现为在 DB2 高速缓存中的同一个语句中。将此值与 num_executions 配合使用，以确定不佳编译环境是否会导致动态 SQL 快照统计信息的结果出现偏差。

num_db_storage_paths - "自动存储器路径数"

此元素显示与此数据库相关联的自动存储器路径的数目。

表 674. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 可将此元素与 db_storage_path 监视元素配合使用来标识与此数据库相关联的存储器路径。

num_executions - "语句执行次数"监视元素

已执行 SQL 语句的次数。

表 675. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 676. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

可将快照监视的计数器复位。

用法

可使用此元素来标识系统中执行得最频繁的 SQL 语句。

在程序包高速缓存级别，使用此选项来计算对每个语句报告的活动度量值的平均值。例如，通过使用以下公式，可以计算在程序包高速缓存级别报告的语句执行的平均 CPU 使用量:

```
total_cpu_time / num_exec_with_metrics
```

在计算平均值时，请使用 **num_exec_with_metrics** 监视元素来代替 **num_executions** 监视元素，这是因为 **num_executions** 监视元素对语句的所有执行进行计数，而不考虑该语句的执行是否与报告的活动度量值相关。

num_exec_with_metrics - “在收集度量值情况下的执行次数”监视元素

在收集度量值情况下执行此 SQL 语句节的次数。此元素可用于计算程序包高速缓存中各个语句的监视元素的每次执行值。

表 677. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

num_extents_left - “尚未处理的扩展数据块数”监视元素

在此表重新平衡过程中尚未移动的扩展数据块数。

表 678. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	始终收集

num_extents_moved - “移动的扩展数据块数”监视元素

在此扩展数据块移动操作期间迄今为止移动的扩展数据块数。

表 679. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	始终收集

num_gw_conn_switches - "连接交换次数"

代理程序池中的代理程序准备好进行连接但被重新分配以与另一 DRDA 数据库配合使用的次数。

表 680. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

对于大多数用户而言，**num_poolagents** 配置参数的缺省设置能确保最优性能。此配置参数的缺省设置自动管理代理程序分组并避免重新分配代理程序。

要减小此监视元素的值，请调整 **num_poolagents** 配置参数的值。

num_indoubt_trans - "不确定事务数"

数据库中的不确定事务的数目。

表 681. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 不确定事务具有未落实事务的日志空间，这可能导致日志变满。日志变满时，就不能完成更多事务。此问题的解决办法涉及以启发方式解决不确定事务的手动过程。此监视元素提供当前未完成并且必须以启发方式解决的不确定事务的数目。

num_log_buffer_full - "日志缓冲区变满次数"监视元素

在将日志记录复制至日志缓冲区时，代理程序必须等待日志数据写入磁盘的次数。每个代理程序每一次发生这种情况时，此值都会递增。例如，如果两个代理程序尝试在缓冲区变满时复制日志数据，那么此值将加上 2。

表 682. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 682. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 683. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 684. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定是否需要增大 **logbufsz** 数据库配置参数。

num_log_data_found_in_buffer - “在缓冲区中找到日志数据的次数”

代理程序从缓冲区读取日志数据的次数。从缓冲区读取日志数据比从磁盘读取数据要好一些，这是因为从磁盘读取时速度较慢。

表 685. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 686. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *num_log_read_io* 元素一起使用来确定是否需要增加 LOGBUFSZ 数据库配置参数的大小。

num_log_part_page_io - "部分日志页写入数"

记录器为了将部分日志数据写至磁盘而发出的 I/O 请求数。

表 687. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 688. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *log_writes*、*log_write_time* 和 *num_log_write_io* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

num_log_read_io - "日志读取数"

记录器为了从磁盘读取日志数据而发出的 I/O 请求数。

表 689. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 690. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *log_reads* 和 *log_read_time* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

num_log_write_io - "日志写入次数"

记录器为了将日志数据写至磁盘而发出的 I/O 请求数。

表 691. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 692. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 将此元素与 *log_writes* 和 *log_write_time* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

num_nodes_in_db2_instance - "分区中的节点数"

获取快照的实例上的节点数。

表 693. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

表 694. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法 使用此元素来确定实例的节点数。对于非分区系统数据库，此值将为 1。

num_remaps - "重新映射次数"监视元素

此活动被重新映射的次数。如果 num_remaps 大于零，那么此活动记录的 service_class_id 是此活动上次重新映射到的服务类的标识。

表 695. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 696. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

使用此信息来验证此活动是否已被映射期望的次数。

num_threshold_violations - "阈值违例次数"监视元素

自最后一次激活此数据库后，在其中发生的阈值违例次数。

表 697. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 698. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

此元素可用来帮助确定阈值对此特定应用程序是否有效，或者阈值违例是否过多。

num_transmissions －“传输次数”

DB2 Connect 网关与用于处理此 DCS 语句的主机之间的数据传输次数。(一次数据传输包括一次发送或一次接收)。

注:

这是旧的监视元素，DB2 UDB 版本 8.1.2 或更高版本中已不再使用此元素。如果在使用 DB2 UDB 版本 8.1.2 或更高版本，那么参阅 **num_transmissions_group** 监视元素。

元素标识

num_transmissions

元素类型

计数器

-->

表 699. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 语句	dcs_stmt	语句

用法 使用此元素来更好地了解特定语句执行时间很长的原因。例如，返回的结果集很大的查询可能需要许多次数据传输才能完成。

num_transmissions_group －“传输组数目”

DB2 Connect 网关与用于处理此 DCS 语句的主机之间的数据传输范围。(一次数据传输包括一次发送或一次接收)。

表 700. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 语句	dcs_stmt	语句

用法 使用此元素来更好地了解特定语句执行时间很长的原因。例如，返回的结果集很大的查询可能需要许多次数据传输才能完成。

下面描述了表示传输范围的常量，它们是在 sqlmon.h 中定义的。

API 常量	描述
SQLM_DCS_TRANS_GROUP_2	2 次传输
SQLM_DCS_TRANS_GROUP_3TO7	3 到 7 次传输
SQLM_DCS_TRANS_GROUP_8TO15	8 到 15 次传输
SQLM_DCS_TRANS_GROUP_16TO64	16 到 64 次传输
SQLM_DCS_TRANS_GROUP_GT64	大于 64 次传输

number_in_bin - “条形中的数目”监视元素

此元素为直方图条形内的活动数或请求数。

表 701. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

使用此元素来表示直方图中条形的高度。

olap_func_overflows - “OLAP 函数溢出次数”监视元素

OLAP 函数数据超过可用排序堆空间的次数。

表 702. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 703. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法

在数据库级别，将此元素与 total_olap_funcs 配合使用以计算溢出至磁盘的 OLAP 函数的百分比。如果此百分比很高，并且使用 OLAP 函数的应用程序的性能需要提高，那么您应该考虑增加排序堆大小。

在应用程序级别，使用此元素来评估各个应用程序的 OLAP 函数性能。

open_cursors - "打开的游标数"

当前对应用程序打开的游标数。

表 704. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句

用法 使用此元素来评估分配的内存量。DB2 客户机、 DB2 Connect 或目标数据库上的数据库代理分配的内存量与当前打开的游标数有关。了解此信息可帮助您进行容量规划。例如，每个执行分块的打开游标的缓冲区大小为 RQRIOBLK。如果启用了 *deferred_prepare*，那么将分配两个缓冲区。

此元素不包括之前的关闭操作关闭的游标。之前的关闭操作是在主机数据库将最后一个记录返回至客户机时发生的。游标在主机和网关上关闭，但在客户机上仍然是打开的。可使用 DB2 调用级接口来设置之前的关闭游标。

open_loc_curs - "打开的本地游标数"

当前对此应用程序打开的本地游标数，包括 *open_loc_curs_blk* 计数的那些游标数。

元素标识

open_loc_curs

元素类型

标尺

表 705. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_loc_curs_blk* 一起使用来计算作为分块游标的本地游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能。

有关远程应用程序使用的游标，请参阅 *open_rem_curs*。

open_loc_curs_blk - "打开的本地分块游标数"

当前对此应用程序打开的本地分块游标数。

元素标识

open_loc_curs_blk

元素类型

标尺

表 706. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_loc_curs* 一起使用来计算作为分块游标的本地游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能:

- 检查记录分块的预编译选项以了解模糊游标的处理方式
- 重新定义游标以允许进行分块（例如，如果可能对游标指定 FOR FETCH ONLY）。

rej_curs_blk 和 acc_curs_blk 提供了附加信息，这些信息可帮助您调整配置参数以改进应用程序中的行分块。

有关远程应用程序使用的分块游标，请参阅 *open_rem_curs_blk*。

open_rem_curs －“打开的远程游标数”

当前对此应用程序打开的远程游标数，包括 *open_rem_curs_blk* 计数的那些游标数。

元素标识

open_rem_curs

元素类型

标尺

表 707. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_rem_curs_blk* 配合使用以计算作为分块游标的远程游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能。有关更多信息，请参阅 *open_rem_curs_blk*。

有关应用程序用于连接至本地数据库的打开的游标数，请参阅 *open_loc_curs*。

open_rem_curs_blk －“打开的远程分块游标数”

当前对此应用程序打开的远程分块游标数。

元素标识

open_rem_curs_blk

元素类型

标尺

表 708. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_rem_curs* 配合使用以计算作为分块游标的远程游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能:

- 检查记录分块的预编译选项以了解模糊游标的处理方式
- 重新定义游标以允许进行分块（例如，如果可能对游标指定 FOR FETCH ONLY）。

rej_curs_blk 和 *acc_curs_blk* 提供了附加信息，这些信息可帮助您调整配置参数以改进应用程序中的行分块。

有关应用程序用来连接至本地数据库的打开的分块游标的数目，请参阅 *open_loc_curs_blk*。

outbound_appl_id – “出站应用程序标识”

此标识是在应用程序连接至 DRDA 主机数据库时生成的。它用来将 DB2 Connect 网关连接至主机，而 **appl_id** 监视元素用来将客户机连接至 DB2 Connect 网关。

注: 不再支持 NetBIOS。也不再支持 SNA，包括其 API APPC、APPN 和 CPI-C。如果您使用这些协议，那么必须对使用诸如 TCP/IP 之类受支持协议的节点和数据库进行重新编目。应忽略对这些协议的引用。

元素标识

outbound_appl_id

元素类型

信息

-->

表 709. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法

可以将此元素与 **appl_id** 一起使用，以使应用程序信息的客户机部分与服务器部分相关联。

此标识在网络上是唯一的。

当网关集中器处于打开状态，或者 DCS 应用程序不在逻辑工作单元中时，此元素将是空白的。

格式 `Network.LU Name.Application instance`

示例 `CAIBMTOR.OSFDBM0.930131194520`

详细信息

此应用程序标识是实际 SNA LUWID（逻辑工作单元标识）的可显示格式，该 SNA LUWID 将在分配 APPC 对话时在网络上流动。APPC 生成的应用程序标识是通过并置网络名、LU 名和 LUWID 实例号构成的，这些应用程序标识将为客户机/服务器应用程序创建唯一标注。网络名和 LU 名的最大长度都是 8 个字符。应用程序实例对应于 12 位十进制字符的 LUWID 实例号。

outbound_bytes_received – “接收的出站字节数”

DB2 Connect 网关从主机接收的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。对于数据传输级别: 处理使用此数目的数据传输的所有语句期间 DB2 Connect 网关从主机接收的字节数。

表 710. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

表 710. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 语句	dcs_stmt	语句
数据传输	stmt_transmissions	语句

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

用法

使用此元素来度量从主机数据库至 DB2 Connect 网关的吞吐量。

outbound_bytes_received_bottom －“接收的最小出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链从主机接收的最少字节数。

表 711. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“接收的出站字节数”一起使用，将后者作为说明主机数据库至 DB2 Connect 网关的吞吐量的另一参数。

outbound_bytes_received_top －“接收的最大出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链从主机接收的最多字节数。

表 712. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“接收的出站字节数”一起使用，将后者作为说明主机数据库至 DB2 Connect 网关的吞吐量的另一参数。

outbound_bytes_sent －“发送的出站字节数”

DB2 Connect 网关发送至主机的字节数，通信协议开销（如 TCP/IP 或 SNA 头）除外。对于数据传输级别: 处理使用此数目的数据传输的所有语句期间 DB2 Connect 网关发送至主机的字节数。

表 713. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句
数据传输	stmt_transmissions	语句

对于语句级别的快照监视，不能复位此计数器。可在其他级别复位此计数器。

用法 使用此元素来度量从 DB2 Connect 网关至主机数据库的吞吐量。

outbound_bytes_sent_bottom －“发送的最小出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链发送至主机的最少字节数。

表 714. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“发送的出站字节数”一起使用，将后者作为说明 DB2 Connect 网关至主机数据库的吞吐量的另一参数。

outbound_bytes_sent_top －“发送的最大出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链发送至主机的最多字节数。

表 715. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“发送的出站字节数”一起使用，将后者作为说明 DB2 Connect 网关至主机数据库的吞吐量的另一参数。

outbound_comm_address －“出站通信地址”

此项是目标数据库的通信地址。例如，它可能是 SNA 网络标识和 LU 伙伴名称，或者是 TCP/IP 的 IP 地址和端口号。

元素标识

outbound_comm_address

元素类型

信息

表 716. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

outbound_comm_protocol －“出站通信协议”

DB2 Connect 网关与主机之间使用的通信协议。

表 717. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法

此元素用于 DCS 应用程序的问题确定。有效值为:

- SQLM_PROT_TCPIP

outbound_sequence_no -"出站序号"

当网关集中器处于打开状态，或者 DCS 应用程序不在逻辑工作单元中时，此元素将是空白的。

表 718. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

overflow_accesses -"访问溢出记录次数"监视元素

对此表的溢出行的访问（读和写）次数。

表 719. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

表 720. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

可将快照监视的计数器复位。

表 721. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法

溢出行表明已发生数据分段。如果此数目较高，那么您可以使用 REORG 实用程序来重组该表（这将消除分段情况），从而提高表的性能。

如果某行已更新并且无法再装入到一开始写入的数据页中，那么该行将会溢出。这种情况通常是因为更新 VARCHAR 或 ALTER TABLE 语句造成的。

overflow_creates - "创建溢出行数"监视元素

对此表创建的溢出行的数目。

表 722. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

用法

package_name - "程序包名"监视元素

当前正在执行的 SQL 语句所在程序包的名称。

表 723. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集

表 724. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 725. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁[1]	event_detailed_dlconn	-
语句	event_stmt	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可使用此元素来帮助标识正在执行的应用程序和 SQL 语句。

package_schema - "程序包模式"监视元素

如果此活动是 SQL 语句，那么此元素代表其程序包的模式名。

表 726. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集

表 727. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

package_version_id - "程序包版本"监视元素

对于给定程序包名称和创建者，可以有（从 DB2 版本 8 开始）多个版本。程序包版本指定当前正在执行的 SQL 语句所在程序包的版本标识。程序包版本由嵌入式 SQL 程序在预编译（PREP）时使用 VERSION 关键字确定。如果在预编译时未指定，那么程序包版本的值为"（空字符串）。

表 728. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集

表 729. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 730. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
语句	event_stmt	-
活动	event_activitystmt	-

用法

使用此元素来帮助确定程序包以及当前正在执行的 SQL 语句。

page_allocations - "分配页数"监视元素

已分配给索引的页数。

表 731. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

page_reorgs - "页重组"

对表执行的页重组数。

表 732. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

可将快照监视的计数器复位。

表 733. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法

尽管页具有足够空间，但可能会因为下列情况而导致页形成碎片:

- 插入新行时
- 更新现有行并且更新导致记录大小增加时

页形成碎片时可能需要重组。重组会将所有碎片空间移至连续区域，可在该区域中写入新记录。这种页重组可能需要几千个指令。它还会生成操作的日志记录。

如果页重组过多，那么可能导致插入性能无法达到最优。可使用 REORG TABLE 实用程序来重组表并消除碎片。还可对 ALTER TABLE 语句使用 APPEND 参数来指示将所有插入追加在表的结尾以避免页重组。

如果行更新导致行长度增加，那么页可能有足够的空间来容纳新行，但可能需要页重组来对该空间进行碎片整理。如果页没有足够的空间来容纳增大的新行，那么将创建导致在读取期间产生 *overflow_accesses* 的溢出记录。可通过使用固定长度的列而不是可变长度的列来避免出现这两种情况。

pages_from_block_ios - "块 I/O 读取总页数"监视元素

块 I/O 读取到缓冲池的块区域中的总页数。

表 734. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 735. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

用法

如果已启用基于块的缓冲池，那么此元素将报告块 I/O 读取的总页数。否则，此元素将返回 0。

要计算每个基于块的 I/O 按顺序预取的平均页数，请将 **pages_from_block_ios** 监视元素的值除以 **block_ios** 监视元素的值。如果此值比您在 CREATE BUFFERPOOL 或 ALTER BUFFERPOOL 语句中为基于块的缓冲池定义的 BLOCKSIZE 选项小很多，那么表明未充分利用基于块的 I/O。出现这种情况的一个可能原因是，正在按顺序预取的表空间的扩展数据块大小与基于块的缓冲池的块大小不匹配。

pages_from_vectored_ios - "向量 I/O 读取总页数"监视元素

向量 I/O 读取到缓冲池的页区域中的总页数。

表 736. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 737. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

pages_merged - "合并页数"监视元素

已合并的索引页数。

表 738. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

pages_read - "读取页数"监视元素

从常规表空间和大型表空间的物理表空间容器读取的页数（包括数据页、索引页和 XML 页）。

表 739. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

用法

pages_written -“写入页数”监视元素

以物理方式写入表空间容器的页数（包括数据页、索引页和 XML 页）。

表 740. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

用法

parent_activity_id -“父活动标识”监视元素

此活动的父活动在父活动的工作单元中的唯一标识。如果没有父活动，那么此监视元素的值为 0。

表 741. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 742. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

将此元素与 **parent_uow_id** 元素和 **appl_id** 元素配合使用，以唯一地标识此活动记录中描述的活动的父活动。

parent_uow_id -“父工作单元标识”监视元素

应用程序句柄内的唯一工作单元标识。此活动的父活动所来源于的工作单元的标识。如果没有父活动，那么该值为 0。

表 743. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 744. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

将此元素与 **parent_activity_id** 元素和 **appl_id** 元素配合使用，以唯一地标识此活动记录中描述的活动的父活动。

partial_record - "部分记录"监视元素

指示事件监视器记录只是部分记录。

表 745. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表	event_table	-
表空间	event_tablespace	-
缓冲池	event_bufferpool	-
连接	event_conn	-
语句	event_stmt	-
语句	event_subsection	-
事务	event_xact	-
活动	event_activity	-

用法

在数据库释放之前，大多数事件监视器不会输出结果。可使用 FLUSH EVENT MONITOR <monitorName> 语句来强制将监视器值输出至事件监视器输出写程序。这允许您在不需要停止并重新启动事件监视器的情况下强制它将记录输出至写程序。此元素指示事件监视器记录是不是清空操作的结果并且因此成为部分记录。

清空事件监视器不会导致值复位。这意味着触发事件监视器时仍会生成完整的事件监视器记录。

在 event_activity 逻辑数据分组时，**partial_record** 监视元素可能的值包括：

0 活动记录通常在活动结束时生成。

1 活动记录在调用 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 存储过程后生成。

2 由于没有足够的存储器来创建记录，所以缺少此活动的信息。event_activity、event_activitystmt 或 event_activityvals 记录可能缺少信息。

participant_no -“死锁参与者”

唯一标识死锁参与者的序号。

元素标识

participant_no

元素类型

信息

表 746. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

用法 在监视应用程序中使用此元素，以将死锁连接事件记录与死锁事件记录相关联。

participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”

对某个对象挂起锁定的应用程序的参与者编号，此应用程序正在等待对该对象获取锁定。

元素标识

participant_no_holding_lk

元素类型

信息

表 747. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

用法 此元素可帮助您确定存在资源争用的应用程序。

partition_number -“分区号”

此元素仅供分区数据库环境中的写至表事件监视器在目标 SQL 表中使用。此值指示插入事件监视器数据的分区的编号。

表 748. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	-

passthru_time - "传递时间"

此元素包含此数据源响应 PASSTHRU 语句所花的总时间（以毫秒计），这些 PASSTHRU 语句来自联合服务器启动后或数据库监视计数器上一次复位后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。响应时间是以联合服务器将 PASSTHRU 语句提交给数据源的时间与数据源响应以指示已处理 PASSTHRU 语句的时间之差量度的。

表 749. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

用法

使用此元素来确定在此数据源上以通过方式处理语句所花的实际时间。

passthrus - "传递数"

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位以后（取较晚者），联合服务器代表任何应用程序直接传递到此数据源的 SQL 语句总数。

表 750. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法 使用此元素来确定可由联合服务器在本地处理的 SQL 语句的百分比以及需要以通过方式处理的 SQL 语句的百分比。如果此值很高，那么应确定原因和调查方法以便更好地利用本机支持。

piped_sorts_accepted - "接受的管道排序数"

接受的管道排序数。

表 751. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器复位。

用法 系统上每个活动排序分配内存可能导致排序占用过多可用系统内存。

如果接受的管道排序数低于请求的管道排序数，可通过调整下列配置参数中的一个或全部来改进排序性能:

- sortheap

- sheapthres

如果拒绝管道排序，那么可以考虑降低排序堆或提高排序堆阈值。您应该了解其中每个选项的可能含义。如果提高排序堆阈值，那么可能会保留更多内存以便分配给排序使用。这可能导致内存至磁盘的页面调度。如果降低排序堆，那么可能需要进行其他的合并阶段，这可能会导致排序速度下降。

piped_sorts_requested －“请求的管道排序数”

请求的管道排序数。

表 752. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器复位。

用法 系统上每个活动排序分配内存可能导致排序占用过多可用系统内存。

排序列表堆（*sortheap*）和排序堆阈值（*sheapthres*）配置参数帮助控制用于排序操作的内存量。这些参数还将用于确定排序是否以管道方式进行。

因为管道排序可以降低磁盘 I/O，允许更多管道排序可以改进排序操作的性能并且可能改进整个系统的性能。如果对排序分配排序堆时将超过排序堆阈值，那么不接受管道排序。如果遇到拒绝管道排序的情况，请参阅 *piped_sorts_accepted* 以获取更多信息。

SQL EXPLAIN 输出将显示优化器是否请求管道排序。

pkg_cache_inserts －“程序包高速缓存插入数”

请求段不可用并且必须装入到程序包高速缓存中的总次数。此计数包括系统执行的所有隐式编译。

元素标识

pkg_cache_inserts

元素类型

计数器

表 753. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 754. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 通过与“程序包高速缓存查询数”一起使用，可借助以下公式来计算程序包高速缓存命中率:

```
1 -（程序包高速缓存插入数/程序包高速缓存查询数）
```

有关使用此元素的更多信息，请参阅 pkg_cache_lookups。

pkg_cache_lookups -“程序包高速缓存查询数”

应用程序在程序包高速缓存中查找某个段或程序包的次数。在数据库级别，这指示自数据库启动或监视器数据复位后的整体引用数。此计数器包括该段已装入到高速缓存中的情况和该段必须装入到高速缓存中的情况。在代理程序要与不同应用程序相关联的集中器环境中，如果新的代理程序在本地存储器中没有必需的段或程序包可用，那么可能需要附加程序包高速缓存查询。

表 755. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 756. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法

要计算程序包高速缓存不命中率，请使用以下公式:

```
1 -（程序包高速缓存插入数/程序包高速缓存查询数）
```

程序包高速缓存不命中率显示程序包高速缓存的使用是否有效率。如果比率很低（低于 0.2），那么表示高速缓存确实起到作用。如果不命中率偏高，那么可能表示应该增加程序包高速缓存。

您需要使用程序包高速缓存大小进行试验，以找出 *pckcachesz* 配置参数的最优数字。例如，如果降低高速缓存的大小时 *pkg_cache_inserts* 元素中的值没有增加，那么可以使用更小的程序包高速缓存大小。降低程序包高速缓存大小将释放系统资源以供其他任务使用。如果增加程序包高速缓存大小使得 pkg_cache_inserts 数降低，那么可以通过这样做来改进整体系统性能。在满工作负载条件下能够最好地完成此实验。

可将此元素与 *ddl_sql_stmts* 配合使用以确定执行 DDL 语句是否会影响程序包高速缓存的性能。执行 DDL 语句时，动态 SQL 语句的各段可能会变得无效。下次使用时系统会对无效段进行隐式编译。执行 DDL 语句可能会使许多段变得无效，并且编译这些段时产生的其他开销会严重影响性能。在此情况下，程序包高速缓存命中率会影响无效

段的隐式重新编译。但它不会影响将新段插入到高速缓存中，所以增加程序包高速缓存大小不会改进整体性能。您可能会发现在满负载环境中工作之前独自为应用程序调整高速缓存会更加清晰明了。

在决定采取什么操作之前，需要确定 DDL 语句在程序包高速缓存命中率的值中所起的作用。如果 DDL 语句很少出现，那么可通过增加高速缓存大小来改进性能。如果 DDL 语句频繁出现，那么可能需要通过限制 DDL 语句的使用（将 DDL 语句的可能使用时间限制为特定时间段）来改进性能。

static_sql_stmts 和 *dynamic_sql_stmts* 计数可用来帮助提供有关要高速缓存的段的数量和类型的信息。

有关程序包高速缓存大小（*pckcachesz*）配置参数的更多信息，请参阅《*管理指南*》。

注: 您可能想要在数据库级别使用此信息来计算每个应用程序的平均程序包高速缓存命中率。您应该在应用程序级别查看此信息以找出给定应用程序的精确程序包高速缓存命中率。为了满足很少执行的应用程序的高速缓存要求而增加程序包高速缓存大小是不值得的。

pkg_cache_num_overflows －“程序包高速缓存溢出数”

程序包高速缓存溢出其分配内存边界的次数。

表 757. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器复位。

表 758. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

将此元素与 **pkg_cache_size_top** 监视元素配合使用来确定是否需要增加程序包高速缓存的大小以避免溢出。

pkg_cache_size_top －“程序包高速缓存高水位标记”

程序包高速缓存达到的最大大小。

注: 从 DB2 版本 9.5 开始，不推荐使用 **pkg_cache_size_top** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 759. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 760. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

如果程序包高速缓存溢出，那么表示此元素包含溢出期间程序包高速缓存达到的最大大小。

检查 **pkg_cache_num_overflows** 监视元素以确定是否发生此类情况。

可通过以下公式来确定工作负载需要的程序包高速缓存最小大小:

```
最大程序包高速缓存大小 / 4096
```

通过将结果四舍五入为整数，指示为避免溢出程序包高速缓存最少需要的页数（每页 4K 字节）。

pool_async_data_read_reqs - "缓冲池异步读请求数"监视元素

预取程序对操作系统发出的异步读请求的数目。这些请求通常是涉及多页的大型块 I/O。

表 761. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 762. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 763. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法

要计算每个读请求的平均数据页数，请使用下列公式:

```
pool_async_data_reads / pool_async_data_read_reqs
```

此平均值可以帮助您确定预取程序所使用的平均读 I/O 大小。此数据还可以帮助您了解所测量的工作负载的大型块 I/O 要求。

预取程序读 I/O 最大大小是所涉及表空间的 CREATE TABLESPACE 语句中 EXTENTSIZE 选项指定的值，但在某些情况下可能更小:

- 扩展数据块的某些页已经在缓冲池中
- 超出操作系统能力
- EXTENTSIZE 选项值非常大，执行大型 I/O 将导致整体性能下降

pool_async_data_reads - "缓冲池异步数据读次数"监视元素

指示异步引擎可派遣单元（EDU）从所有类型的表空间的物理表空间容器中读取的数据页数。

表 764. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 765. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 766. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法

可将此元素与 **pool_data_p_reads** 配合使用，以计算同步执行的物理读取数（即数据库管理器代理程序执行的物理数据页读取数）。请使用以下公式:

```
 1-((pool_data_p_reads+pool_index_p_reads)-(pool_async_data_reads+pool_async_index_reads))
/(pool_data_l_reads+pool_index_l_reads)
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

异步读取是由数据库管理器预取程序执行的。

pool_async_data_writes - "缓冲池异步数据写次数"监视元素

异步页清除程序或预取程序将缓冲池数据页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

表 767. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 768. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 769. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法 通过将此元素与 **buffer_pool_data_writes** 监视元素配合使用，可以计算以同步方式执行的物理写请求（即，由数据库管理器代理程序执行的物理数据页写操作）的数目。请使用以下公式:

```
pool_data_writes - pool_async_data_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 **num_iocleaners** 配置参数时，此比率会非常有用。

pool_async_index_read_reqs - "缓冲池异步索引读请求数"监视元素

针对索引页的异步读请求的数目。

表 770. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 771. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

表 771. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 772. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法 要计算每个异步请求读取的索引页数，请使用以下公式:

```
pool_async_index_reads / pool_async_index_read_reqs
```

此平均数可帮助您确定每次与预取程序交互时对索引页进行的异步 I/O 量。

pool_async_index_reads - "缓冲池异步索引读取数"监视元素

指示异步引擎可派遣单元（EDU）从所有类型的表空间的物理表空间容器中读取的索引页数。

表 773. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 774. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 775. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法

通过将此元素与 **pool_index_p_reads** 监视元素配合使用，可以计算以同步方式执行的物理读操作（即，由数据库管理器代理程序执行的物理索引页读操作）的数目。请使用以下公式:

```
1 - ((pool_data_p_reads + pool_index_p_reads) - (pool_async_data_reads + pool_async_index_reads))
/ (pool_data_l_reads + pool_index_l_reads)
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

异步读取是由数据库管理器预取程序执行的。

pool_async_index_writes - "缓冲池异步索引写次数"监视元素

异步页清除程序或预取程序将缓冲池索引页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

表 776. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 777. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 778. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法

通过将此元素与 **pool_index_writes** 监视元素配合使用，可以计算以同步方式执行的物理索引写请求（即，由数据库管理器代理程序执行的物理索引页写操作）的数目。请使用以下公式:

```
pool_index_writes - pool_async_index_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 **num_iocleaners** 配置参数时，此比率会非常有用。

pool_async_read_time – “缓冲池异步读取时间”

指示从表空间容器（物理）中由异步引擎可派遣单元（EDU）对所有类型的表空间读取数据和索引页所花费的总时间。此值以毫秒计。

元素标识

pool_async_read_time

元素类型

计数器

表 779. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 780. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法 可使用此元素并借助以下公式来计算同步读取所耗用的时间:

```
pool_read_time - pool_async_read_time
```

还可使用此元素并借助以下公式来计算平均异步读取时间:

```
pool_async_read_time / pool_async_data_reads
```

这些计算可用来了解要执行的 I/O 处理。

pool_async_write_time – “缓冲池异步写入时间”

数据库管理器页清除程序将数据或索引页从缓冲池写至磁盘的耗用时间总计。

元素标识

pool_async_write_time

元素类型

计数器

表 781. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 782. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法 要计算同步写入页所耗用的时间，请使用以下公式:

```
pool_write_time - pool_async_write_time
```

还可使用此元素并借助以下公式来计算平均异步读取时间:

```
pool_async_write_time
  / (pool_async_data_writes
      + pool_async_index_writes)
```

这些计算可用来了解要执行的 I/O 处理。

pool_async_xda_read_reqs - "缓冲池异步 XDA 读请求数"监视元素

针对 XML 存储器对象（XDA）数据的异步读取请求数。

表 783. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 784. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 785. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法 要计算每个异步请求读取的平均 XML 存储器对象数据页数，请使用以下公式:

```
pool_async_xda_reads / pool_async_xda_read_reqs
```

此平均数可帮助您确定每次与预取程序交互时完成的异步 I/O 量。

pool_async_xda_reads - "缓冲池异步 XDA 数据读取数"监视元素

指示异步引擎可派遣单元（EDU）从所有类型的表空间的物理表空间容器中读取的 XML 存储器对象（XDA）数据页数。

表 786. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 787. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 788. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法

通过使用 **pool_async_xda_reads** 和 **pool_xda_p_reads** 监视元素，可以计算以同步方式对 XML 存储器对象数据页执行的物理读操作（即，数据库管理器代理程序对 XML 数据执行的物理数据页读操作）的数目。请使用以下公式:

```
pool_xda_p_reads - pool_async_xda_reads
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

异步读取是由数据库管理器预取程序执行的。

pool_async_xda_writes - “缓冲池异步 XDA 数据写次数”监视元素

异步页清除程序或预取程序将 XML 存储器对象（XDA）的缓冲池数据页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

表 789. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 790. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 791. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法 通过将此元素与 **pool_xda_writes** 监视元素配合使用，可以计算以同步方式对 XML 存储器对象数据页执行的物理写请求（即，数据库管理器代理程序对 XML 数据执行的物理数据页写操作）的数目。请使用以下公式:

```
pool_xda_writes - pool_async_xda_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 **num_iocleaners** 配置参数时，此比率会非常有用。

pool_config_size –“内存池的已配置大小”

DB2 数据库系统中的内存池的内部配置大小。

表 792. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 793. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

用法 要跟踪系统内存使用情况，请将此值与 *pool_cur_size*、*pool_id* 和 *pool_watermark* 一起使用。

要查看内存池是否接近已满状态，将 *pool_config_size* 与 *pool_cur_size* 进行比较。例如，假定实用程序堆非常小。可以按一定时间间隔获取快照并查看快照输出的实用程序堆部分，以诊断这一特定问题。如果需要，可能会允许 *pool_cur_size* 超过 *pool_config_size* 以避免内存不足故障。如果这种情况不常发生，那么可能不需要进一步的操作。但是，如果 *pool_cur_size* 一直接近或大于 *pool_config_size*，那么可能要考虑增加实用程序堆的大小。

pool_cur_size - "内存池的当前大小"

内存池的当前大小。

表 794. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 795. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

用法 要跟踪系统内存使用情况，请将此值与 *pool_config_size*、*pool_id* 和 *pool_watermark* 一起使用。

要查看内存池是否接近已满状态，将 *pool_config_size* 与 *pool_cur_size* 进行比较。例如，假定实用程序堆非常小。可以按一定时间间隔获取快照并查看快照输出的实用程序堆部分，以诊断这一特定问题。如果 *pool_cur_size* 的值一直接近 *pool_config_size*，那么您可能要考虑增加实用程序堆的大小。

pool_data_l_reads - "缓冲池数据逻辑读取数"监视元素

从常规表空间和大型表空间的逻辑缓冲池中请求获取的数据页的数目。

表 796. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 796. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 797. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 798. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

此计数包括数据处于下列情况时对数据的访问:

- 当数据库管理器需要处理页时，数据已经在缓冲池中。
- 应读取到缓冲池中，数据库管理器才能处理页。

通过使用 **pool_data_l_reads** 和 **pool_data_p_reads** 监视元素并借助以下公式，可以计算缓冲池的整体数据页命中率:

```
1 - ((pool_data_p_reads - pool_async_data_reads) / pool_data_l_reads)
```

增加缓冲池大小一般会改进命中率，但您会达到一个最优状态，而无法继续改进。从理论上说，如果能够分配大到足以存储整个数据库的缓冲池，那么系统启动并运行后你可以得到 100% 的命中率。但在许多情况下这是不现实的。命中率的高低实际上取决于数据的大小以及访问数据的方式。如果数据库很大并且数据访问比较平均，那么命中率将会很低。对于非常大的表，您几乎无能为力。

对于较小并且访问较为频繁的表和索引，要提高其命中率，请将其分配到不同的缓冲池。

pool_data_p_reads - “缓冲池数据物理读取数”监视元素

指示从常规表空间和大型表空间的物理表空间容器中读取的数据页数。

表 799. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 799. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 800. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 801. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

通过将此元素与 **pool_data_l_reads** 和 **pool_async_data_reads** 监视元素配合使用，可以计算以同步方式执行的物理读操作（即，数据库管理器代理程序执行的物理数据页读操作）的数目。请使用以下公式:

```
1 - ((pool_data_p_reads + pool_index_p_reads) - (pool_async_data_reads + pool_async_index_reads))
 / (pool_data_l_reads + pool_index_l_reads)
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此信息非常有用。

pool_data_writes - “缓冲池数据写次数”监视元素

以物理方式将缓冲池数据页写入磁盘的次数。

表 802. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 803. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 804. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 804. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

用法

如果缓冲池数据页被写入磁盘的次数在 **pool_data_p_reads** 监视元素值中占较高的百分比，那么可通过增加可供数据库使用的缓冲池页数来提高性能。

缓冲池数据页将因为下列原因写至磁盘:

- 释放缓冲池中的某一页以便可以读取另一页
- 将缓冲池清仓

系统不会总是通过写入页为新页腾出空间。如果该页未被更新，那么只需将其替换。此元素不考虑此替换。

在需要缓冲池空间之前，数据页可由异步页清除程序代理程序写入，这由 **pool_async_data_writes** 监视元素报告。这些异步页写入与同步页写入一起包括在此元素的值中。

计算此百分比时，忽略一开始填充缓冲池所需的物理读取的数目。要确定写入的页数:

1. 运行应用程序以装入缓冲区。
2. 记录此元素的值。
3. 再次运行应用程序。
4. 从此元素的新值中减去步骤 2 中记录的值。

为了避免在应用程序的各次运行之间取消分配缓冲池，应该执行下列其中一项操作:

- 使用 ACTIVATE DATABASE 命令来激活数据库。
- 将一个空闲的应用程序连接至数据库。

如果所有应用程序都要更新该数据库，那么提高缓冲池大小对性能可能没有多大影响，原因是大多数缓冲池页包含已更新数据，而这些数据必须写入磁盘。但是，如果已更新页在写出之前可供其他工作单元使用，那么缓冲池可保存读写操作，这将对性能有所改进。

pool_drty_pg_steal_clns - “触发缓冲池牺牲页清除程序次数”监视元素

因为数据库的牺牲缓冲区替换期间需要同步写入而调用页清除程序的次数。

表 805. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 806. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器复位。

表 807. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

通过使用以下公式，可以计算通过此元素表示的所有清除程序调用的百分比:

```
pool_drty_pg_steal_clns
  / (pool_drty_pg_steal_clns
   +  pool_drty_pg_thrsh_clns
   +  pool_lsn_gap_clns)
```

如果此比率很低，那么可能指示您定义的页清除程序过多。如果 **chngpgs_thresh** 配置参数设置得太小，那么可能会写出将来会成为脏页的页。主动清除使得缓冲池失去了尽可能延迟写入这一用途。

如果此比率很高，那么可能表明未定义足够的页清除程序。页清除程序不足将导致发生故障后的恢复时间延长。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- **pool_drty_pg_steal_clns** 监视元素将插入到监视器流中。
- **pool_drty_pg_steal_clns** 监视元素计算因为数据库的牺牲缓冲区替换期间需要同步写入而调用页清除程序的次数。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- **pool_drty_pg_steal_clns** 监视元素将 0 插入到监视器流中。
- 牺牲缓冲区替换期间需要同步写入时不会显式触发页清除程序。要确定为数据库或特定缓冲池配置的页清除程序数是否正确，请参考 **pool_no_victim_buffer** 监视元素。

注: 虽然会将脏页写入磁盘，但除非需要空间来读入新页，否则不会立即从缓冲池中除去这些页。

pool_drty_pg_thrsh_clns - “触发缓冲池阈值清除程序次数”监视元素

因为缓冲池达到数据库的脏页阈值条件而调用页清除程序的次数。

表 808. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 809. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器复位。

表 810. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 该阈值由 **chngpgs_thresh** 配置参数设置。它是应用于缓冲池大小的百分比。如果池中的脏页数超过此值，将触发清除程序。

如果 **chngpgs_thresh** 配置参数值设置得太小，那么可能会过早将页写出，从而导致需要重新读入那些页。如果此值设置得过大，那么可能会累积过多的页，从而要求用户以同步方式将这些页写出。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- **pool_drty_pg_thrsh_clns** 监视元素将插入到监视器流中。
- **pool_drty_pg_thrsh_clns** 监视元素计算因为缓冲池达到数据库的脏页阈值条件而调用页清除程序的次数。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- **pool_drty_pg_thrsh_clns** 监视元素将 0 插入到监视器流中。
- 页清除程序总是处于活动状态，以试图确保有足够的可用缓冲区以供牺牲，而不是等待条件值触发。

pool_id - “内存池标识”

内存池的类型。

表 811. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 812. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

用法

要跟踪系统内存使用情况，请将此值与 **pool_max_size**、**pool_cur_size** 和 **pool_watermark** 配合使用。

使用 **pool_id** 来标识系统监视器输出中讨论的内存池。各种内存池标识都可在 sqlmon.h 中找到。在正常操作情况下，可以使用下列每个内存池的其中一个或多个。

API 常量	描述
SQLM_HEAP_APPLICATION	应用程序堆
SQLM_HEAP_DATABASE	数据库堆
SQLM_HEAP_LOCK_MGR	锁管理器堆
SQLM_HEAP_UTILITY	备份/恢复/实用程序堆
SQLM_HEAP_STATISTICS	统计信息堆
SQLM_HEAP_PACKAGE_CACHE	程序包高速缓存堆
SQLM_HEAP_CAT_CACHE	目录高速缓存堆
SQLM_HEAP_MONITOR	数据库监视器堆
SQLM_HEAP_STATEMENT	语句堆
SQLM_HEAP_FCMBP	FCMBP 堆
SQLM_HEAP_IMPORT_POOL	导入池
SQLM_HEAP_OTHER	其他内存
SQLM_HEAP_BP	缓冲池堆
SQLM_HEAP_APPL_SHARED	应用程序共享堆
SQLM_HEAP_SHARED_SORT	排序共享堆

pool_index_l_reads - “缓冲池索引逻辑读取数”监视元素

指示从常规表空间和大型表空间的逻辑缓冲池中请求获取的索引页数。

表 813. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 813. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 814. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 815. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

此计数包括索引页处于下列情况时对索引页的访问:

- 当数据库管理器需要处理页时索引页已经在缓冲池中
- 应读取到缓冲池中，数据库管理器才能处理页。

通过与 **pool_index_p_reads** 配合使用并借助以下公式，可以计算缓冲池的索引页命中率:

```
1 - ((pool_index_p_reads - pool_async_index_reads) / pool_index_l_reads)
```

通过使用 **pool_data_l_reads** 和 **pool_data_p_reads** 监视元素并借助以下公式，可以计算缓冲池的整体数据页命中率:

```
1 - ((pool_data_p_reads - pool_async_data_reads) / pool_data_l_reads)
```

如果命中率很低，那么提高缓冲池页数可以改进性能。

pool_index_p_reads - "缓冲池索引物理读取数"监视元素

指示从常规表空间和大型表空间的物理表空间容器中读取的索引页数。

表 816. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 816. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 817. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 818. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

通过与 **pool_index_l_reads** 监视元素配合使用并借助以下公式，可以计算缓冲池的索引页命中率:

```
1 - ((pool_index_p_reads - pool_async_index_reads) / pool_index_l_reads)
```

pool_index_writes - “缓冲池索引写次数”监视元素

指示缓冲池索引页物理写至磁盘的次数。

表 819. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 820. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 821. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 821. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

用法

与数据页一样，缓冲池索引页将因为下列原因写至磁盘:

- 释放缓冲池中的某一页以便可以读取另一页
- 将缓冲池清仓

系统不会总是通过写入页为新页腾出空间。如果该页未被更新，那么只需将其替换。此元素不考虑此替换。

在需要缓冲池空间之前，索引页可由异步页清除程序代理程序写入。这些异步索引页写次数与同步索引页写次数一起包括在此元素的值中（参见 **pool_async_index_writes** 监视元素）。

如果缓冲池索引页被写入磁盘的次数在 **pool_index_p_reads** 监视元素值中占较高的百分比，那么可通过增加可供数据库使用的缓冲池页数来提高性能。

计算此百分比时，忽略一开始填充缓冲池所需的物理读取的数目。要确定写入的页数:

1. 运行应用程序以装入缓冲区。
2. 记录此元素的值。
3. 再次运行应用程序。
4. 从此元素的新值中减去步骤 2 中记录的值。

为了避免在应用程序的各次运行之间取消分配缓冲池，应该执行下列其中一项操作:

- 使用 ACTIVATE DATABASE 命令来激活数据库。
- 将一个空闲的应用程序连接至数据库。

如果所有应用程序都要更新该数据库，那么提高缓冲池大小对性能可能没有多大影响，原因是大多数页包含已更新数据，而这些数据必须写入磁盘。

pool_lsn_gap_clns - “触发缓冲池日志空间清除程序次数”监视元素

因为使用的记录空间已达到数据库的预定义条件而调用页清除程序的次数。

表 822. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 823. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器复位。

表 824. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

此元素可用来帮助评估您是否具有足够的记录空间以及您是否需要更多或更大的日志文件。

页清除条件将由 **softmax** 配置参数的设置确定。如果缓冲池中最旧的页包含一个更新，此更新由比条件值定义的当前日志位置还要旧的日志记录描述，那么将触发页清除程序。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- **pool_lsn_gap_clns** 监视元素将插入到监视器流中。
- 如果缓冲池中最旧的页包含一个更新，此更新由比条件值定义的当前日志位置还要旧的日志记录描述，那么将触发页清除程序。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- **pool_lsn_gap_clns** 监视元素会将 0 插入到监视器流中。
- 页清除程序主动写入页而不是等待条件值触发。

pool_no_victim_buffer - “缓冲池无牺牲缓冲区次数”监视元素

代理程序没有预先选择的可用牺牲缓冲区的次数。

表 825. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 826. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池

表 826. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

可将快照监视的计数器复位。

表 827. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-

用法 此元素可用来帮助评估使用前摄页清除时您是否具有足够的页清除程序用于给定缓冲池。

DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时，pool_no_victim_buffer 元素计算代理程序找不到预先选择的牺牲缓冲区可供立即使用，并且强制搜索缓冲池以查找适合的牺牲缓冲区的次数。

如果 pool_no_victim_buffer 元素的值相对于缓冲池中的逻辑读取数过高，那么 DB2 数据库系统难以确保有足够的良好牺牲缓冲区可供使用。增加页清除程序数目将增加 DB2 提供预选牺牲缓冲区的能力。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时，pool_no_victim_buffer 元素没有预测值，并且可以安全地忽略。在此配置中，DB2 数据库系统不会尝试确保代理程序预选可供使用的牺牲缓冲区，所以对缓冲池的大多数访问需要代理程序搜索缓冲池以查找牺牲缓冲区。

pool_read_time - “缓冲池物理读时间总计”监视元素

指示从所有类型的表空间的物理表空间容器读取数据和索引页时的耗费时间总计。此值以毫秒计。

表 828. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 828. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 829. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 830. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

用法

通过将此元素与 **pool_data_p_reads** 和 **pool_index_p_reads** 监视元素配合使用，可以计算平均读页时间。此平均值非常重要，它表示存在 I/O 等待状态，而 I/O 等待状态又表示应该将数据移至另一设备。

在数据库和表空间级别，此元素包括 **pool_async_read_time** 监视元素的值。

pool_secondary_id - "内存池辅助标记"

一个附加标识，用于帮助确定对其返回监视器数据的内存池。

元素标识

pool_secondary_id

元素类型

信息

表 831. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 832. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

用法 与 pool_id 一起使用来确定对其返回监视器数据的内存池。pool_secondary_id 的数据仅在必要时才会出现。例如，当指示的 pool_id 是用于确定与监视器数据相关的缓冲池的缓冲池堆时，它就会出现。

创建新数据库后，它将具有缺省缓冲池 IBMDEFAULTBP，其大小将由平台确定。此缓冲池的辅助标识为"1"。除了此缓冲池及您创建的所有缓冲池之外，在缺省情况下还会创建一组系统缓冲池，每个缓冲池对应不同页大小。这些缓冲池的标识会出现在 pool_secondary_id 的快照中:

- 系统 32K 缓冲池
- 系统 16K 缓冲池
- 系统 8K 缓冲池
- 系统 4K 缓冲池

pool_temp_data_l_reads - "缓冲池临时数据逻辑读取数"监视元素

指示向临时表空间的逻辑缓冲池请求的数据页数。

表 833. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 833. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 834. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 835. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 835. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

与 **pool_temp_data_p_reads** 元素配合使用，通过以下公式计算临时表空间中缓冲池的数据页命中率:

```
1 - (pool_temp_data_p_reads / pool_temp_data_l_reads)
```

缓冲池整体命中率的计算公式如下:

```
     1 - ((pool_data_p_reads + pool_xda_p_reads +
 pool_index_p_reads + pool_temp_data_p_reads
 + pool_temp_xda_p_reads + pool_temp_index_p_reads )
           / (pool_data_l_reads + pool_xda_l_reads + pool_index_l_reads +
           pool_temp_data_l_reads + pool_temp_xda_l_reads
      + pool_temp_index_l_reads )) * 100%
```

此计算公式考虑缓冲池高速缓存的所有页（索引和数据）。

pool_temp_data_p_reads - “缓冲池临时数据物理读取数”监视元素

指示从临时表空间的物理表空间容器中读取的数据页数。

表 836. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 836. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 837. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 838. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

通过将此元素与 **pool_temp_data_l_reads** 元素配合使用并借助以下公式，可以计算临时表空间中缓冲池的数据页命中率:

```
1 - (pool_temp_data_p_reads / pool_temp_data_l_reads)
```

pool_temp_index_l_reads - "缓冲池临时索引逻辑读取数"监视元素

指示向临时表空间的逻辑缓冲池请求的索引页数。

表 839. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 840. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 841. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

通过将此元素与 **pool_temp_index_p_reads** 元素配合使用并借助以下公式，可以计算临时表空间中缓冲池的索引页命中率:

```
1 - (pool_temp_index_p_reads / pool_temp_index_l_reads)
```

pool_temp_index_p_reads - “缓冲池临时索引物理读取数”监视元素

指示从临时表空间的物理表空间容器中读取的索引页数。

表 842. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 842. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 843. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 844. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 844. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

通过将此元素与 **pool_temp_index_l_reads** 元素配合使用并借助以下公式，可以计算临时表空间中缓冲池的索引页命中率:

```
1 - (pool_temp_index_p_reads / pool_temp_index_l_reads)
```

pool_temp_xda_l_reads - "缓冲池临时 XDA 数据逻辑读取数"监视元素

指示向临时表空间的逻辑缓冲池请求的 XML 存储器对象（XDA）数据页数。

表 845. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 845. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 846. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 847. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

通过将 **pool_temp_xda_l_reads** 监视元素与 **pool_temp_xda_p_reads**、**pool_temp_data_l_reads** 和 **pool_temp_data_p_reads** 监视元素配合使用并借助以下公式，可以计算临时表空间中缓冲池的数据页命中率:

```
1 - ((pool_temp_data_p_reads + pool_temp_xda_p_reads)
      / (pool_temp_data_l_reads + pool_temp_xda_l_reads))
```

pool_temp_xda_p_reads – “缓冲池临时 XDA 数据物理读取数”监视元素

指示从临时表空间的物理表空间容器中读取的 XML 存储器对象（XDA）数据页数。

表 848. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 – 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 – 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 – 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 – 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 – 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 – 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 – 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 – 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 849. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 850. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

通过将 **pool_temp_xda_p_reads** 监视元素与 **pool_temp_xda_l_reads**、**pool_temp_data_l_reads** 和 **pool_temp_data_p_reads** 监视元素配合使用并借助以下公式，可以计算临时表空间中缓冲池的数据页命中率:

```
1 - ((pool_temp_data_p_reads + pool_temp_xda_p_reads) /
(pool_temp_data_l_reads + pool_temp_xda_l_reads))
```

pool_watermark - “内存池水位标记”

自创建内存池后内存池的最大大小。

元素标识

pool_watermark

元素类型

信息

表 851. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 852. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

用法 在一直运行的系统上，可将 *pool_watermark* 和 *pool_config_size* 元素一起使用来预测潜在的内存问题。

例如，按一定时间间隔获取快照（如每天），并检查 *pool_watermark* 和 *pool_config_size* 值。如果发现 *pool_watermark* 的值开始逐步接近 *pool_config_size*（预示将来可能出现内存相关问题），那么可能指示应增加内存池的大小。

pool_write_time －“缓冲池物理写时间总计”监视元素

提供以物理方式将缓冲池中的数据或索引页写至磁盘时耗用的总时间。耗用时间以毫秒计。

表 853. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 － 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 － 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 － 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 － 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 － 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 － 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 － 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 － 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 － 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 854. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 855. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

用法

通过将此元素与 **buffer_pool_data_writes** 和 **pool_index_writes** 监视元素配合使用，可以计算平均写页时间。此平均值非常重要，它表示存在 I/O 等待状态，而 I/O 等待状态又表示应该将数据移至另一设备。

在数据库和表空间级别，此元素包括 **pool_async_write_time** 监视元素的值。

pool_xda_l_reads - “缓冲池 XDA 数据逻辑读取数”监视元素

指示向常规表空间和大型表空间的逻辑缓冲池请求的 XML 存储器对象（XDA）数据页数。

表 856. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 856. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 857. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 858. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

此计数包括数据处于下列情况时对数据的访问:

- 当数据库管理器需要处理页时索引页已经在缓冲池中
- 应读取到缓冲池中，数据库管理器才能处理页。

通过使用 **pool_xda_l_reads**、**pool_xda_p_reads**、**pool_data_l_reads** 和 **pool_data_p_reads** 监视元素并借助以下公式，可以计算缓冲池的数据页命中率:

```
1 - ((pool_data_p_reads + pool_xda_p_reads)
   / (pool_data_l_reads + pool_xda_l_reads))
```

缓冲池整体命中率的计算公式如下:

```
    1 - ((pool_data_p_reads + pool_xda_p_reads +
pool_index_p_reads + pool_temp_data_p_reads
+ pool_temp_xda_p_reads + pool_temp_index_p_reads )
          / (pool_data_l_reads + pool_xda_l_reads + pool_index_l_reads +
          pool_temp_data_l_reads + pool_temp_xda_l_reads
     + pool_temp_index_l_reads )) * 100%
```

此计算公式考虑缓冲池高速缓存的所有页（索引和数据）。

增加缓冲池大小一般会改进命中率，但您会达到一个最优状态，而无法继续改进。从理论上说，如果能够分配大到足以存储整个数据库的缓冲池，那么系统启动并运行后你可以得到 100% 的命中率。但在许多情况下这是不现实的。命中率的高低取决于数据的大小以及访问数据的方式。如果数据库很大并且数据访问比较平均，那么命中率将会很低。对于非常大的表，您几乎无能为力。在此类情况下，应该将重点放在较小并且访问较为频繁的表以及索引上。

pool_xda_p_reads - “缓冲池 XDA 数据物理读取数”监视元素

指示从常规表空间和大型表空间的物理表空间容器中读取的 XML 存储器对象（XDA）数据页数。

表 859. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 859. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 860. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器复位。

表 861. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	缓冲池，语句

用法

通过使用 **pool_async_xda_reads** 和 **pool_xda_p_reads** 监视元素，可以计算以同步方式对 XML 存储器对象数据页执行的物理读操作（即，数据库管理器代理程序对 XML 数据执行的物理数据页读操作）的数目。请使用以下公式:

```
pool_xda_p_reads - pool_async_xda_reads
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

通过使用 **pool_xda_l_reads**、**pool_xda_p_reads**、**pool_data_l_reads** 和 **pool_data_p_reads** 监视元素并借助以下公式，可以计算缓冲池的数据页命中率:

```
1 - ((pool_data_p_reads + pool_xda_p_reads)
   / (pool_data_l_reads + pool_xda_l_reads))
```

pool_xda_writes – “缓冲池 XDA 数据写次数”监视元素

指示以物理方式将 XML 存储器对象（XDA）的缓冲池数据页写入磁盘的次数。

表 862. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 – 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_BUFFERPOOL 表函数 – 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 – 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 – 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 – 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 862. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 863. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 864. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

用法

此监视元素帮助您评估通过增加数据库的可用缓冲池页数能否改进性能。对于包含 XML 数据的数据库，您既应该考虑有关 XML 数据的缓冲池页写入数与缓冲池页读取数比率（使用 **pool_xda_writes** 和 **pool_xda_p_reads** 监视元素），也应该考虑有关关系数据类型的缓冲池页写入数与缓冲池页读取数比率（使用 **pool_data_writes** 和 **pool_data_p_reads** 监视元素）。

通过使用 **pool_xda_l_reads**、**pool_xda_p_reads**、**pool_data_l_reads** 和 **pool_data_p_reads** 监视元素并借助以下公式，可以计算缓冲池的数据页命中率:

```
1 - ((pool_data_p_reads + pool_xda_p_reads)
   / (pool_data_l_reads + pool_xda_l_reads))
```

post_shrthreshold_hash_joins - "阈值后散列连接数"

排序内存限量算法已限制的散列连接总数。内存受限散列连接是指获得的内存量少于排序内存管理器所请求内存量的散列连接。

表 865. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

可将快照监视的计数器复位。

表 866. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

当从共享排序堆中分配的内存量接近数据库配置参数 *sheapthres_shr* 设置的限制时，就会对散列连接进行内存限制。这种内存限制将显著减少在配置不当的系统中因超过 *sheapthres_shr* 限制导致的内存溢出次数。此元素报告的数据仅反映正在使用从共享排序堆分配的内存的散列连接。

post_shrthreshold_sorts - "共享阈值后排序数"监视元素

排序内存限量算法已限制的排序总数。内存受限排序是指获得的内存量少于排序内存管理器所请求内存量的排序。当为排序分配的内存量接近数据库配置参数 *sheapthres_shr* 设置的限制时，就会对排序进行内存限制。这种内存限制将显著减少在配置不当的系统中因超过 *sheapthres_shr* 限制导致的内存溢出次数。此元素报告的数据仅反映正在使用从共享排序堆分配的内存的排序。

表 867. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 867. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 868. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	排序

可将快照监视的计数器复位。

表 869. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-

post_threshold_hash_joins - “散列连接阈值”

散列连接堆请求因为并行使用共享或专用排序堆空间而受限的总次数。

元素标识

post_threshold_hash_joins

元素类型

计数器

表 870. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器复位。

用法 如果此值很大（超过 hash_join_overflows 的 5%），那么应增加排序堆阈值。

post_threshold_olap_funcs - "OLAP 函数阈值"监视元素

超过排序堆阈值后请求排序堆的 OLAP 函数数。

表 871. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器复位。

用法

排序、散列连接和 OLAP 函数是使用排序堆的操作的例子。在正常条件下，数据库管理器将使用 sortheap 配置参数指定的值来分配排序堆。如果分配给排序堆的内存量超过排序堆阈值（sheapthres 配置参数），那么数据库管理器将使用小于 sortheap 配置参数指定值的值来分配后续排序堆。

在达到排序堆阈值后启动的 OLAP 函数可能无法接收最优内存量来执行。

为提高排序、散列连接、OLAP 函数的性能以及系统整体性能，请修改排序堆阈值和排序堆大小配置参数。

如果此元素的值过高，那么请增加排序堆阈值（sheapthres）。

post_threshold_sorts - "超出阈值后的排序次数"监视元素

超过排序堆阈值后请求堆的排序数。

表 872. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 872. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 873. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	排序

可将快照监视的计数器复位。

表 874. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

在正常条件下，数据库管理器将使用 **sortheap** 配置参数指定的值来分配排序堆。如果分配给排序堆的内存量超过排序堆阈值（**sheapthres** 配置参数），那么数据库管理器将使用小于**sortheap**配置参数指定值的值来分配排序堆。

系统上每个活动排序分配内存可能导致排序占用过多系统可用内存。在达到排序堆阈值后启动的排序可能无法接收最优内存量来执行，但整个系统将因此而获益。通过修改排序堆阈值和排序堆大小配置参数，排序操作性能和整体系统性能可以得到改进。如果此元素的值过高，那么可以:

- 提高排序堆阈值（**sheapthres**）或者
- 通过 SQL 查询更改将应用程序调整为使用数量较少范围较小的排序。

prefetch_wait_time -“等待预取的时间”监视元素

应用程序等待 I/O 服务器（预取程序）完成将页装入到缓冲池中的操作所花的时间。 此值以毫秒计。

表 875. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
应用程序	appl	缓冲池

表 876. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可通过更改 I/O 服务器数目和 I/O 服务器大小来使用此元素进行试验。

prep_time - "编译时间"监视元素

活动为 SQL 语句时编译 SQL 语句所需的时间（以毫秒计）。

表 877. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 878. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

此元素可用来标识当活动为 SQL 活动时编译 SQL 语句所花的活动总生存期的时间。

prep_time_best - "语句最短编译时间"监视元素

编译特定 SQL 语句所需的最短时间（以毫秒计）。

表 879. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

用法

将此值与 **prep_time_worst** 配合使用来标识编译成本高昂的 SQL 语句。

prep_time_worst - "语句最长编译时间"监视元素

编译特定 SQL 语句所需的最长时间（以毫秒计）。

表 880. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

用法

将此值与 **prep_time_best** 配合使用来标识编译成本高昂的 SQL 语句。

prev_uow_stop_time - "上一个工作单元完成时间戳记"

这是工作单元的完成时间。

元素标识

prev_uow_stop_time

元素类型

时间戳记

表 881. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

用法 可将此元素与 *uow_stop_time* 配合使用来计算 COMMIT/ROLLBACK 点之间的耗用时间总计，还可将其与 *uow_start_time* 配合使用来计算在工作单元之间的应用程序上耗用的时间。以下其中一个时间:

- 对于当前在工作单元中的应用程序，这是最新工作单元的完成时间。
- 对于当前不在工作单元中的应用程序（该应用程序已完成某个工作单元，但尚未启动新的工作单元），这是刚刚完成的工作单元之前完成的最后一个工作单元的停止时间。刚刚完成的工作单元的时间用 uow_stop_time 指示。
- 对于第一个工作单元中的应用程序，这是数据库连接请求的完成时间。

priv_workspace_num_overflows - "专用工作空间溢出数"

专用工作空间溢出其分配内存边界的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 882. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 883. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 将此元素与 priv_workspace_size_top 配合使用来确定专用工作空间是否需要增加大小以避免溢出。专用工作空间的溢出可能导致性能下降，以及从代理程序专用内存分配的其他堆出现内存不足错误。

在数据库级别，报告的元素将来自报告为具有相同最大专用工作空间大小的元素的专用工作空间。在应用程序级别，此项是为当前应用程序提供服务的每个代理程序的工作空间的溢出数。

priv_workspace_section_inserts －“专用工作空间节插入数”

应用程序在专用工作空间中插入 SQL 节的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 884. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 885. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可执行节的工作副本存储在专用工作空间中。

此计数器指示副本何时不可用并且必须插入。在数据库级别，此项是针对数据库的所有专用工作空间中的每个应用程序进行的所有插入的累积总数。在应用程序级别，此项是针对此应用程序的专用工作空间中的所有节的所有插入的累积总数。

在代理程序要与不同应用程序相关联的集中器环境中，如果新的代理程序在专用工作空间中没有必需的节可用，那么可能需要附加专用工作空间插入。

priv_workspace_section_lookups －“专用工作空间节查询数”

应用程序在其代理程序的专用工作空间中对 SQL 节进行查询的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 886. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 887. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 每个应用程序都可以访问为其工作的代理程序的专用工作空间。

此计数器指示为找到应用程序的特定节而访问专用工作空间的次数。在数据库级别，此项是针对数据库的所有专用工作空间中的每个应用程序进行的所有查询的累积总数。在应用程序级别，此项是针对此应用程序的专用工作空间中的所有节的所有查询的累积总数。

可将此元素与“专用工作空间节插入数”一起使用来调整专用工作空间的大小。专用工作空间的大小由 applheapsz 配置参数控制。

priv_workspace_size_top - “最大专用工作空间大小”

专用工作空间达到的最大大小。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 888. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 889. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 每个代理程序都有专用工作空间，它所服务的应用程序对该工作空间具有访问权。此元素指示专用工作空间中提供服务的任何代理程序所需的最多字节数。在数据库级别，此项是连接至当前数据库的所有代理程序在所有专用工作空间中必需的最多字节数。 在应用程序级别，此项是为当前应用程序提供服务的所有代理程序的专用工作空间中的最大大小。

专用工作空间溢出时，将会临时从代理程序专用内存的其他实体借出内存。这可能导致这些实体出现内存不足错误，也可能导致性能下降。可通过增加 APPLHEAPSZ 来降低溢出的机率。

product_name - "产品名称"

正在运行的 DB2 实例的版本的详细信息。

表 890. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

progress_completed_units - "完成的进度工作单元数"

当前阶段完成的工作单元数。

表 891. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

此元素的值通常会在实用程序操作时递增。此元素总是小于或等于 *progress_total_units*（如果同时定义了这两个元素）。

注:

1. 可能并非所有实用程序都包含此元素。
2. 此元素在 *progress_work_metric* 监视元素中以单元表示。

用法 使用此元素来确定某个阶段完成的工作量。此元素本身可用来监视正在运行的实用程序的活动。实用程序执行时，此元素应不断递增。如果 *progress_completed_units* 在很长一段时间内未能递增，那么实用程序可能已停止。

如果定义了 *progress_total_units*，那么此元素可用来计算完成工作的百分比:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

progress_description - "进度描述"

描述工作阶段。

表 892. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

Load 实用程序的示例值包括:

- DELETE
- LOAD
- REDO

用法 使用此元素来获取对某个阶段的一般描述。

progress_list_attr - "当前进度列表属性"

此元素描述如何解释进度元素列表。

表 893. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress_list	基本

用法

此元素的值是下列其中一个常量:

- SQLM_ELM_PROGRESS_LIST_ATTR_SERIAL - 列表中的元素将解释为一组串行阶段，这意味着第一次更新元素 *n*+1 的已完成工作之前，已完成工作必须等于元素 *n* 的总工作。此属性用于描述由一组串行阶段组成的任务的进度，其中串行阶段意味着必须彻底完成一个阶段才能开始下一个阶段。
- SQLM_ELM_PROGRESS_LIST_ATTR_CONCURRENT - 进度列表中的任何元素可以随时更新。

使用此元素来确定更新进度列表的各个元素的方式。

progress_list_cur_seq_num - "当前进度列表序号"

如果实用程序包含多个顺序阶段，那么此元素显示当前阶段的编号。

表 894. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress_list	基本

用法 使用此元素来确定多阶段实用程序的当前阶段。请参阅『progress_seq_num - "进度序号"』。

progress_seq_num - "进度序号"

阶段号。

注: 仅对由多个执行阶段组成的实用程序显示阶段号。

表 895. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

用法 使用此元素来确定多阶段实用程序内的阶段顺序。该实用程序将按进度序号递增顺序来顺序执行各个阶段。可通过将 *progress_seq_num* 与 *progress_list_current_seq_num* 的值相匹配来找到多阶段实用程序的当前阶段。

progress_start_time - "进度开始时间"

表示阶段开始的时间戳记。

表 896. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

用法 使用此元素来确定阶段开始的时间。如果该阶段尚未开始，那么省略此元素。

progress_total_units - "进度工作单元总数"

为了完成某个阶段而执行的工作总量。

表 897. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

某些实用程序可能无法确定总工作量，所以它们将持续更新此元素。其他实用程序可能无法估计总工作量，所以可能完全省略此元素。

此元素在 *progress_work_metric* 监视元素中以单元表示。

用法 使用此元素来确定某个阶段的总工作量。将此元素与 *progress_completed_units* 配合使用来计算某个阶段完成的工作百分比:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

progress_work_metric - "进度工作度量"

解释 *progress_total_units* 和 *progress_completed_units* 元素的度量。

表 898. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

示例值包括:

- SQLM_WORK_METRIC_BYTES
- SQLM_WORK_METRIC_EXTENTS

注:

1. 可能并非所有实用程序都包含此元素。
2. 此元素的值可在 sqlmon.h 中找到。

用法 使用此元素的值来确定用作报告度量的 *progress_total_units* 和 *progress_completed_units*。

pseudo_deletes - "伪删除数"监视元素

已进行伪删除的伪空页中所有的键。此监视元素报告已删除的伪空页数。

表 899. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

pseudo_empty_pages - "伪空页数"监视元素

已进行伪删除的伪空页中所有的键。此监视元素报告已被标识为伪空页的页数。

表 900. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

用法

注: 此监视元素并不报告伪空页的当前数目。

qp_query_id - "Query Patroller 查询标识"监视元素

如果此活动是查询，那么此元素包含 Query Patroller 对此活动指定的查询标识。查询标识为 0 表明 Query Patroller 未对此活动指定查询标识。

要点: 由于 qp_query_id monitor 监视元素与 Query Patroller 功能相关联，因此建议您不要使用此监视元素。由于 DB2 版本 9.5 引入了新的工作负载管理功能部件，因此版本 9.7 建议您不要使用 Query Patroller 及其相关组件，它们在将来的发行版中可能会被除去。

表 901. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

query_card_estimate - "行查询数估计"

查询将返回的行数估计。

表 902. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句
活动	event_activity	-

用法 SQL 编译器进行的此估计可与运行时实际结果进行比较。

在监视 DB2 Connect时，此元素还会返回有关下列 SQL 语句的信息。

- INSERT、UPDATE 和 DELETE

指示受影响的行数。

- PREPARE

  估计将返回的行数。仅当 DRDA 服务器为 DB2 数据库 Linux 版、UNIX 版和 Windows 版、DB2 VM 版和 VSE 版或 DB2 OS/400® 版时才收集此信息。

- FETCH

  设置为访存行数。仅当 DRDA 服务器为 DB2 OS/400 版时，才会收集此信息。

如果不对 DRDA 服务器收集信息，那么该元素设置为零。

query_cost_estimate －“查询估计成本”监视元素

由 SQL 编译器确定的查询估计成本。此值以 timeron 计。

表 903. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 － 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 904. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句
活动	event_activity	-

用法 此项允许将实际运行时估计与编译时估计相关联。

在监视 DB2 Connect时，此元素还会返回有关下列 SQL 语句的信息。

- PREPARE

  表示预编译 SQL 语句的相对成本。

- FETCH

  包含已检索行的长度。仅当 DRDA 服务器为 DB2 OS/400 版时，才会收集此信息。

如果不对 DRDA 服务器收集信息，那么该元素设置为零。

注: 如果 DRDA 服务器为 DB2 OS/390® 版和 z/OS 版，那么此估计可能高于 2**32 - 1（可通过不带符号的长整型变量表示的最大整数）。在此情况下，监视器对此元素返回的值将为 2**32 - 1。

queue_assignments_total －“队列分配总次数”监视元素

自从上次复位后任何连接或活动被分配到此阈值队列的次数。

表 905. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

此元素可用于确定，任何连接或活动在由统计信息收集时间间隔确定的给定时间段内在此特定队列中进行排队的次数。这样可帮助确定队列阈值的有效性。

queue_size_top －“队列大小顶部”监视元素

自最后一次复位后达到的最大队列大小。

表 906. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

使用此元素来测量队列阈值的有效性以及检测队列何时变得过长。

queue_time_total －“总队列时间”监视元素

自最后一次复位以后，此队列中的所有连接或活动在队列中所花的总时间。单位为毫秒。

表 907. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

使用此元素来测量队列阈值的有效性以及检测队列何时变得过长。

quiescer_agent_id －“停顿者代理程序标识”

具有停顿状态的代理程序的代理程序标识。

元素标识

quiescer_agent_id

元素类型

信息

表 908. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 将此元素与 quiescer_auth_id 配合使用，以确定停顿表空间的人员。

quiescer_auth_id – “停顿者用户授权标识”

具有停顿状态的用户的授权标识。

元素标识

quiescer_auth_id

元素类型

信息

表 909. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 使用此元素来确定停顿表空间的人员。

quiescer_obj_id – “停顿者对象标识”

导致表空间停顿的对象的对象标识。

元素标识

quiescer_obj_id

元素类型

信息

表 910. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 将此元素与 quiescer_ts_id 和 quiescer_auth_id 配合使用以确定停顿表空间的人员。此元素的值与视图 SYSCAT.TABLES 的列 TABLEID 中的值相匹配。

quiescer_state – “停顿者状态”

要完成的停顿类型（如“SHARE”、“INTENT TO UPDATE”或“EXCLUSIVE”）。

元素标识

quiescer_state

元素类型

信息

表 911. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 此元素的值与 sqlutil.h 中的常量 SQLB_QUIESCED_SHARE、SQLB_QUIESCED_UPDATE 或 SQLB_QUIESCED_EXCLUSIVE 的值相匹配。

quiescer_ts_id - "停顿者表空间标识"

导致表空间停顿的对象的表空间标识。

元素标识

quiescer_ts_id

元素类型

信息

表 912. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 将此元素与 quiescer_obj_id 和 quiescer_auth_id 配合使用，以确定停顿表空间的人员。此元素的值与视图 SYSCAT.TABLES 的列 TBSPACEID 中的值相匹配。

range_adjustment - "范围调整"

此值表示容器数组中范围实际开始的偏移。

表 913. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_container_id - "范围容器"

在范围内唯一定义容器的整数。

表 914. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_end_stripe - "结束分割区"

此值表示范围中的最后一个分割区的编号。

表 915. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_max_extent - "范围中的最大扩展数据块"

此值表示范围映射的最大扩展数据块编号。

表 916. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_max_page_number - "范围中的最大页"

此值表示范围映射的最大页号。

表 917. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_num_containers - "范围中的容器数"

此值表示当前范围中的容器数目。

表 918. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_number - "范围编号"

此值表示表空间映射内的范围的编号。

表 919. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_offset - "范围偏移"

从范围所属的分割集开头的分割区 0 开始的偏移。

表 920. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_start_stripe - "起始分割区"

此值表示范围中的第一个分割区的编号。

表 921. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_stripe_set_number - "分割集编号"

此值表示范围所在的分割集。

表 922. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

reclaimable_space_enabled - "已启用可回收空间指示器"监视元素

如果已对表空间启用可回收存储器，那么此监视元素将返回值 1。否则，它返回值 0。

表 923. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

rej_curs_blk - "拒绝的块游标请求数"

在服务器上拒绝请求 I/O 块并且请求转换为非分块 I/O 的次数。

元素标识

rej_curs_blk

元素类型

计数器

表 924. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 925. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-

用法 如果有许多游标分块数据，那么通信堆可能会变满。此堆变满时，不会返回错误。而是不会再对分块游标分配 I/O 块。如果游标无法对数据进行分块，那么性能会受到影响。

如果大量游标无法执行数据分块，那么可通过执行下列操作来改进性能:

- 增加 *query_heap* 数据库管理器配置参数的大小。

rem_cons_in - "与数据库管理器的远程连接数"

从远程客户机启动的与正在监视的数据库管理器实例的当前连接数。

表 926. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

显示此实例中从远程客户机至数据库的连接的数目。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目不包括从与数据库管理器相同的实例启动的应用程序。

与 local_cons 监视元素一起使用时，这些元素可帮助您调整 **max_coordagents** 和 **max_connections** 配置参数的设置。

rem_cons_in_exec - "数据库管理器中正在执行的远程连接数"

当前连接至数据库，并且正在处理要监视的数据库管理器实例中的工作单元的远程应用程序数。

表 927. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此数目可帮助您确定数据库管理器上进行的并行处理的级别。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目不包括从与数据库管理器相同的实例启动的应用程序。

与 local_cons_in_exec 监视元素一起使用时，此元素可帮助您调整 **max_coordagents** 配置参数的设置。

如果 **max_coordagents** 设置为 AUTOMATIC，那么您不需要作任何调整。如果不是设置为 AUTOMATIC，并且 rem_cons_in_exec 与 local_cons_in_exec 的和接近 **max_coordagents**，那么应该增加 **max_coordagents** 的值。

remote_lock_time – “远程锁定时间”

此元素包含此数据源对远程锁定所花的总时间（以毫秒计），这些远程锁定来自联合服务器启动后或数据库监视计数器上一次复位后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。响应时间是以联合服务器将远程锁定提交给数据源的时间与联合服务器在数据源上释放远程锁定的时间之差量度的。

表 928. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

用法

使用此元素来确定此数据源对远程锁定所花的实际时间。

remote_locks – “远程锁定”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位以后（取较晚者），联合服务器代表任何应用程序在此数据源上调用的远程锁定总数。

表 929. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法 使用此元素来确定在数据源上进行的远程锁定的数目。

reorg_completion – “重组完成标志”

表重组成功指示器，这包括从多维集群（MDC）表中回收扩展数据块。对于分区表来说，此元素还指示数据分区的完成状态。

表 930. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 如果表或数据分区重组操作成功，此元素的值将为 0。如果表或数据分区重组操作不成功，那么此元素的值将为 -1。成功和失败值将在 sqlmon.h 中作如下定义:

- 成功: SQLM_REORG_SUCCESS
- 失败: SQLM_REORG_FAIL

如果表重组不成功，那么请参阅历史记录文件以获取任何诊断信息，包括警告和错误。可使用 LIST HISTORY 命令来访问此数据。对于分区表，将对每个数

据分区指示完成状态。如果索引重建在分区表上失败，那么将在所有数据分区上更新失败状态。有关进一步的诊断信息，请参阅管理通知日志。

reorg_current_counter - “重组进度”

指示重组完成量的进度单元。此值表示的进度与 reorg_max_counter 的值有关，后者表示要完成的表重组的总量。

表 931. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法

可使用以下公式来确定已完成的表重组的百分比:

```
表重组进度 = reorg_current_counter / reorg_max_counter * 100
```

reorg_end - “表重组结束时间”

表重组（包括为了从多维集群（MDC）表中回收扩展数据块而进行的重组）的结束时间。对于分区表来说，此元素还将指示每个数据分区重组的结束时间。

表 932. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_index_id - “用于重组表的索引”

用于重组表的索引。

表 933. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_long_tbspc_id - “用来重组长对象的表空间”监视元素

将用来重组任何长对象（LONG VARCHAR 或 LOB 数据）的表空间。对于分区表来说，这是将用来重组每个分区的 LONG VARCHAR 和 LOB 的表空间。

表 934. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_max_counter - "重组总量"

此值指示重组期间要完成的工作总量，其中包括用于从多维集群（MDC）表中回收扩展数据块的重组操作。此值可与 reorg_current_counter 配合使用以确定重组进度，reorg_current_counter 表示完成的工作量。

表 935. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_max_phase - "最大重组阶段"

重组处理期间将发生的最大重组阶段数。此项仅适用于经典（脱机）重组。值的范围为 2 到 4（[SORT], BUILD, REPLACE,[INDEX_RECREATE]）。此值还可能指示执行重组时为了从多维集群（MDC）表中回收扩展数据块而完成的工作总量。执行这样的重组时，此值是 3（SCAN、DRAIN 和 RELEASE）。

表 936. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_phase - "表重组阶段"监视元素

指示表的重组阶段。对于分区表来说，此元素还将指示每个数据分区的重组阶段。此元素仅适用于脱机表重组。

表 937. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法

对于分区表来说，重组是逐个数据分区进行的。对于传统表重组而言，可能的阶段如下所示（这些阶段与它们在 sqlmon.h 中的相应定义一起列示）：

- 排序：SQLM_REORG_SORT
- 构建：SQLM_REORG_BUILD
- 替换：SQLM_REORG_REPLACE
- 索引重新创建：SQLM_REORG_INDEX_RECREATE
- 字典构建：SQLM_REORG_DICT_SAMPLE

对于分区表而言，在数据分区的"替换"阶段完成后，可以直接进入分区索引（如果有的话）的"索引重建"阶段。仅当每个数据分区上的所有先前阶段成功完成后，**reorg_phase** 元素才会指示"索引重新创建"阶段。

在 XDA 对象压缩期间，XML 数据重组阶段涉及识别表的 XML 存储器对象。XML 字典构建阶段涉及尝试为 XML 存储器对象创建压缩字典。对于 XDA 对象压缩而言，可能的两个阶段如下所示:

- XML 重组: SQLM_REORG_XML_DATA
- XML 字典构建: SQLM_REORG_XML_DICT_SAMPLE

对于分区表，在执行扩展数据块回收操作时，可能的阶段如下所示:

- 扫描: SQLM_REORG_SCAN
- 漏出: SQLM_REORG_DRAIN
- 释放: SQLM_REORG_RELEASE

reorg_phase_start - "重组阶段开始时间"

表重组或回收重组阶段的开始时间。对于分区表来说，此元素还将指示每个数据分区的重组阶段的开始时间。对于非分区索引而言，在索引重建阶段，所有数据分区的数据组将同时进行更新。

表 938. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_rows_compressed - "压缩行数"

重组期间在表中压缩的行数。

表 939. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 重组期间在表中压缩的行数的连续计数。某些记录永远不会被压缩（如果记录长度小于最小记录长度）。

重要的是要注意，这个行数未反映数据压缩效率，它只显示了符合压缩条件的记录的个数。

reorg_rows_rejected_for_compression - "拒绝压缩行数"

重组期间由于记录长度小于或等于最小记录长度而未压缩的行数。

表 940. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 如果记录长度小于或等于最小记录长度，就不会压缩该记录。已拒绝的行数反映了这些未符合此压缩要求的记录的连续计数。

reorg_start - "表重组开始时间"

表重组（包括为了从多维集群（MDC）表中回收扩展数据块而进行的重组）的开始时间。对于分区表来说，此元素还将指示每个数据分区重组的开始时间。

表 941. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_status - "表重组状态"

现场（联机）表重组或数据分区级别重组的状态。此项不适用于传统（脱机）表重组。

表 942. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 归位表或数据分区重组可能处于下列其中一种状态（状态与它们在 sqlmon.h 中的相应定义列示在一起）：

- 启动/继续: SQLM_REORG_STARTED
- 暂停: SQLM_REORG_PAUSED
- 停止: SQLM_REORG_STOPPED
- 完成: SQLM_REORG_COMPLETED
- 截断: SQLM_REORG_TRUNCATE

用于回收扩展数据块的归位表或数据分区重组可能处于下列其中一种状态:

- 启动: SQLM_REORG_STARTED
- 停止: SQLM_REORG_STOPPED
- 完成: SQLM_REORG_COMPLETED

reorg_tbspc_id - "用来重组表或数据分区的表空间"

用来重组表的表空间。对于分区表来说，这将指示用来重组每个数据分区的表空间。

表 943. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_type - "表重组属性"

表重组属性设置。

表 944. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 可能的属性设置如下所示。每个属性设置都基于 db2ApiDf.h 中定义的位标志值。

- 允许写访问: DB2REORG_ALLOW_WRITE

- 允许读访问：DB2REORG_ALLOW_READ
- 不允许访问：DB2REORG_ALLOW_NONE
- 通过索引扫描重新集群：DB2REORG_INDEXSCAN
- 重组长型字段 LOB 数据：DB2REORG_LONGLOB
- 不截断表：DB2REORG_NOTRUNCATE_ONLINE
- 替换压缩字典：DB2REORG_RESET_DICTIONARY
- 保留压缩字典：DB2REORG_KEEP_DICTIONARY
- 回收扩展数据块：DB2REORG_RECLAIM_EXTS

除了上述属性设置以外，在 GET SNAPSHOT FOR TABLES 命令的 CLP 输出中还列示了下列属性。这些属性设置基于其他属性设置值或表重组监视元素值。

- 重新集群：如果 reorg_index_id 监视元素值不为零，那么表重组操作具有此属性。
- 重新声明：如果 reorg_index_id 监视元素值为零，那么表重组操作具有此属性。
- 原位表重组：如果 reorg_status 监视元素值不为空，那么表示正在使用原位（联机）重组方法。
- 表重组：如果 reorg_phase 监视元素值不为空，那么表示正在使用传统（脱机）重组方法。
- 通过表扫描重新集群：如果未设置 DB2REORG_INDEXSCAN 标志，那么表重组操作具有此属性。
- 仅重组数据：如果未设置 DB2REORG_LONGLOB 标志，那么表重组操作具有此属性。

reorg_xml_regions_compressed - “已压缩的 XML 区域数”监视元素

在表重组过程中压缩的 XML 区域的数目。

表 945. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_xml_regions_rejected_for_compression - “拒绝压缩的 XML 区域数”监视元素

在表重组过程中未压缩的 XML 区域的数目。

表 946. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

request_exec_time_avg - “平均请求执行时间”监视元素

自最后一次复位以后与此服务子类相关联的请求的执行时间算术平均值。如果内部跟踪的平均值已溢出，那么将返回值 -2。当服务子类的 COLLECT AGGREGATE REQUEST DATA 设置为 NONE 时，此监视元素返回 -1。单位为毫秒。

使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，request_exec_time_avg 平均值将对重新映射所涉及的每个子类中的不完整请求进行计数。

表 947. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此统计信息以快速了解在数据库分区上处理此服务子类中的每个请求所花的平均时间量。

另外，还可以使用此平均值来确定用于请求执行时间直方图的直方图模板是否合适。根据请求执行时间直方图来计算平均请求执行时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改请求执行时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

rf_log_num - “正在前滚的日志”

要处理的日志。

元素标识

rf_log_num

元素类型

信息

表 948. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法 如果正在进行前滚，那么此元素标识前滚涉及的日志。

rf_status - “日志阶段”

恢复的状态。

元素标识

rf_status

元素类型

信息

表 949. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法 此元素指示恢复的进度。它指示恢复是处于撤销（回滚）阶段还是处于重做（前滚）阶段。

rf_timestamp - "前滚时间戳记"

上次落实的事务的时间戳记。

元素标识

rf_timestamp

元素类型

时间戳记

表 950. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	时间戳记

用法 如果正在进行前滚，那么这是前滚恢复操作所处理的上次落实事务的时间戳记。这是前滚操作的进度指示符。

rf_type - "前滚类型"

正在进行的前滚的类型。

元素标识

rf_type

元素类型

信息

表 951. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法 指示是在数据库级别还是表空间级别进行恢复的指示符。

rollback_sql_stmts - "尝试的回滚语句数"

尝试的 SQL ROLLBACK 语句总数。

元素标识

rollback_sql_stmts

元素类型

计数器

表 952. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本

表 952. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器复位。

表 953. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 回滚可能是应用程序请求、死锁或错误情况导致的。此元素**仅**对从应用程序发出的回滚语句计数。

在应用程序级别，此元素可帮助您确定应用程序的数据库活动的级别以及与其他应用程序的冲突程度。在数据库级别，它可以帮助您确定数据库中的活动量以及数据库上的应用程序间的冲突程度。

注：应尝试将回滚次数降至最低，原因是回滚活动越高，数据库的吞吐量越低。

还可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

rolled_back_agent_id - "回滚的代理程序"

发生死锁时回滚的代理程序。

元素标识

rolled_back_agent_id

元素类型

信息

表 954. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

rolled_back_appl_id - "回滚的应用程序"

发生死锁时回滚的应用程序标识。

元素标识

rolled_back_appl_id

元素类型

信息

表 955. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

用法　系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

rolled_back_participant_no －“回滚的应用程序参与者”监视元素

用于标识已回滚的应用程序的参与者编号。

表 956. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	event_deadlock	-

1　建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

rolled_back_sequence_no －“回滚的序号”

发生死锁时回滚的应用程序的序号。

元素标识

rolled_back_sequence_no

元素类型

信息

表 957. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	-

用法　系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

root_node_splits －“根节点分割次数”监视元素

在插入操作期间分割索引根节点的次数。

表 958. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

routine_id - "例程标识"监视元素

此监视元素是唯一的例程标识。如果此活动不是任何例程的组成部分，那么此元素将返回零。

表 959. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

rows_deleted - "删除行数"监视元素

这是所尝试的行删除操作的数目。

表 960. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

表 961. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 962. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 您可以使用此元素来确定数据库的当前活动级别。

此计数不包括 **int_rows_deleted** 监视元素中记录的尝试次数。

rows_fetched – “访存的行数”监视元素

从表中读取的行数。

此监视元素是 **rows_read** 监视元素的别名。

注: 此监视元素仅报告为共记录了此信息的数据库分区的值。在 DPF 系统上，这些值可能不能反映整个活动的正确总计。

元素标识

rows_fetched

元素类型

计数器

-->

表 963. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	语句

用法

有关详细信息，请参阅 **rows_read** 监视元素。

rows_inserted – “插入行数”监视元素

尝试插入的行数。

表 964. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 – 获取表度量值	始终收集

表 965. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 966. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 您可以使用此元素来确定数据库的当前活动级别。

在联合系统中，每个 INSERT 语句可插入多行，原因是联合服务器会在适当时将 INSERT FROM SUBSELECT 推送至数据源。

此计数不包括 **int_rows_inserted** 监视元素中记录的尝试次数。

rows_modified - "修改的行数"监视元素

插入、更新或删除的行数。

此监视元素是 **rows_written** 监视元素的别名。

表 967. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 968. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

表 968. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activity	语句

用法

有关详细信息，请参阅 **rows_written** 监视元素。

rows_read - “读取行数”监视元素

从表中读取的行数。

表 969. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 970. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
表	表	表
应用程序	appl	基本

表 970. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	stmt	基本
应用程序	subsection	语句
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 971. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
连接	event_conn	-
表	event_table	-
语句	event_stmt	-
事务	event_xact	-

用法

此元素帮助标识使用频率很高并且您可能想要为其创建附加索引的表。为了避免维护非必需的索引，请使用 SQL EXPLAIN 语句来确定程序包是否使用索引。

此计数不是返回至调用应用程序的行数。而是必须读取以返回结果集的行数。例如，以下语句向应用程序返回一行，但读取了许多行以确定平均薪水:

```
SELECT AVG(SALARY) FROM USERID.EMPLOYEE
```

此计数包括 **overflow_accesses** 监视元素中的值。另外，此计数不包括任何索引访问。即，如果访问方案仅使用索引访问方法，并且不会访问该表以查看实际的行，那么 **rows_read** 监视元素的值不会递增。

rows_returned - "返回的行数"监视元素

已选择并返回到应用程序的行数。对于部分活动记录，此元素具有 0 值（例如，当活动仍在执行或当完整活动记录因内存限制而无法写入事件监视器时，如果收集活动，那么就会出现此情况）。

此监视元素是 **fetch_count** 监视元素的别名。

表 972. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 973. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
活动	event_activity	-

用法

此元素可用来帮助确定返回到应用程序的行数的阈值，或者用来验证该阈值配置是否正确且正常工作。

rows_returned_top - "最高实际返回行数"监视元素

服务类或工作类中所有嵌套级别的 DML 活动实际返回行数的高水位标记。对于服务类，当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更新完成该活动的服务子类的 rows_returned_top 高水位标记。该活动所映射到但未在其中完成该活动的服务子类的高水位标记不受影响。

表 974. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此元素来确定在收集时间间隔内，分区中服务类、工作负载或工作类达到的最高 DML 活动实际返回行数。

rows_selected - "选择的行数"

此项是已选择并且返回至应用程序的行数。

表 975. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 976. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 您可以使用此元素来确定数据库的当前活动级别。

此元素不包括对 COUNT(*) 或连接这样的操作读取的行计数。

对于联合系统，可计算将数据源中的一行返回至联合服务器的平均时间:

```
平均时间 = 返回的行数 / 聚集查询响应时间
```

可使用这些结果来修改 SYSCAT.SERVERS 中的 CPU 速度或通信速度参数。修改这些参数会影响优化器是否将请求发送至数据源。

注: 如果被监视的网关为 DB2 数据库版本 7.2 或更低版本，那么将在 dcs_dbase 和 dcs_appl 快照监视器逻辑数据组中收集此元素。

rows_updated - "更新行数"监视元素

这是尝试更新的行数。

表 977. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

表 978. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 979. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 您可以使用此元素来确定数据库的当前活动级别。

此值不包括 **int_rows_updated** 监视元素中记录的更新计数。但是，将对每个更新计算多个更新语句更新的行数。

rows_written - "写入的行数"

此项是表中更改（插入、删除或更新）的行数。

表 980. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	基本
应用程序	stmt	基本
应用程序	subsection	语句
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

表 981. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
表	event_table	-
语句	event_stmt	-
事务	event_xact	-

用法 如果表级别信息的值很高，那么指示该表的使用频率太高，您可能使用“运行统计”（RUNSTATS）实用程序来保持用于此表的程序包的效率。

对于应用程序连接和语句，此元素包括临时表中插入、更新和删除的行数。

在应用程序、事务和语句级别，此元素对于分析相对活动级别和标识调整候选对象会非常有用。

rqsts_completed_total - “完成请求总数”监视元素

已执行的请求的总数，其中包括应用程序请求和内部请求。对于服务子类而言，将仅在此请求的完成位置更新此监视元素。如果此请求曾在不同服务子类之间移动，那么将计数两次。

表 982. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 983. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 983. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

sc_work_action_set_id - "服务类工作操作集标识"监视元素

如果此活动已划分到服务类作用域的某个工作类，那么此监视元素显示与该工作类所属的工作类集相关联的工作操作集的标识。否则，此监视元素将显示值 0。

表 984. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 985. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

可将此元素与 **sc_work_class_id** 元素配合使用，以唯一地标识活动的服务类工作类（如果有）。

sc_work_class_id - "服务类工作类标识"监视元素

如果此活动已划分到服务类作用域的某个工作类，那么此监视元素将显示分配给此活动的工作类的标识。否则，此监视元素将显示值 0。

表 986. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 987. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

可将此元素与 **sc_work_action_set_id** 元素配合使用，以唯一地标识活动的服务类工作类（如果有）。

sec_log_used_top - “使用的最大辅助日志空间”

使用的最大辅助日志空间（以字节计）。

表 988. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 989. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 可将此元素与 *sec_logs_allocated* 和 *tot_log_used_top* 配合使用以显示当前对辅助日志的依赖性。如果此值很高，那么可能需要更大的日志文件或者更多的主日志文件，又或是在应用程序中更频繁地使用 COMMIT 语句。

因此，可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond
- logretain

如果数据库没有任何辅助日志文件，那么该值将为零。如果未定义辅助日志文件，那么将出现此情况。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

sec_logs_allocated - “当前分配的辅助日志数”

当前用于数据库的辅助日志文件总数。

表 990. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 可将此元素与 *sec_log_used_top* 和 *tot_log_used_top* 一起使用来显示当前对辅助日志的依赖性。如果此值一直很高，那么可能需要更大的日志文件或者更多的主日志文件，又或是在应用程序中更频繁地使用 COMMIT 语句。

因此，可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond
- logretain

section_env - “节环境”监视元素

提供活动的节详细信息的句柄。

表 991. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	-

用法

此元素用来与将来的 IBM 工具配合使用来对此记录中描述的活动提取节信息

section_number - "节号"监视元素

当前正在处理的或最近处理的静态 SQL 语句在程序包中的内部节号。

表 992. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集

表 993. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 994. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁[1]	event_detailed_dlconn	-
语句	event_stmt	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

对于静态 SQL 语句，可以将此元素与 **creator**、**package_version_id** 和 **package_name** 监视元素配合使用，以便使用以下样本查询来查询 SYSCAT.STATEMENTS 系统目录表并获取静态 SQL 语句文本:

```
SELECT SEQNO, SUBSTR(TEXT,1,120)
       FROM SYSCAT.STATEMENTS
       WHERE PKGNAME   = 'package_name' AND
             PKGSCHEMA = 'creator'      AND
             VERSION = 'package_version_id' AND
             SECTNO    = section_number
       ORDER BY SEQNO
```

注: 在获取静态语句文本时将产生警告，原因是针对系统目录表的此查询可能导致锁定争用。尽可能在没有其他针对该数据库的活动时才使用此查询。

section_type - "节类型指示器"监视元素

指示 SQL 语句节是动态的还是静态的。

表 995. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

用法

此监视元素的可能值如下所示:

- D: 动态
- S: 静态

select_sql_stmts - "执行的 Select SQL 语句数"

执行的 SQL SELECT 语句数。

表 996. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
表空间	tablespace	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

表 997. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。

还可使用以下公式来确定 SELECT 语句数与语句总数的比率:

```
    select_sql_stmts
/ ( static_sql_stmts
    + dynamic_sql_stmts )
```

此信息对于分析应用程序活动和吞吐量非常有用。

select_time – “查询响应时间”

此元素包含此数据源响应查询所花的总时间（以毫秒计），这些查询来自联合服务器启动后或数据库监视计数器上一次复位后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。

注: 因为存在查询分块，并非联合服务器检索行的所有尝试都能进行通信处理，所以获取下一行的请求可能要通过返回行块来得到处理。因此，聚集查询响应时间并不总是指示数据源上的处理，但它通常指示数据源或客户机上的处理。

表 998. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

用法

使用此元素来确定等待此数据源中的数据所花的实际时间。它在容量规划和容量调整 SYSCAT.SERVERS 中的 CPU 速度和通信速率时很有用。修改这些参数会影响优化器是否将请求发送至数据源。

响应时间是以联合服务器请求数据源中的行的时间与该行可供联合服务器使用的时间之差量度的。

sequence_no – “序号”监视元素

每当工作单元结束（即 COMMIT 或 ROLLBACK 终止工作单元）时，此标识就会递增。**appl_id** 与 **sequence_no** 一起唯一地标识一个事务。

表 999. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本

表 1000. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
连接	event_connheader	-
语句	event_stmt	-
事务	event_xact	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_detailed_dlconn	-

表 1000. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史纪录	event_stmt_history	-
带有详细信息的死锁的历史记录值	event_detailed_dlconn	-
带有详细信息的死锁的历史记录值	event_stmt_history	-

sequence_no_holding_lk -“挂起锁定的序号”

对某个对象挂起锁定的应用程序的序号，此应用程序正在等待对该对象获取锁定。

元素标识

sequence_no_holding_lk

元素类型

信息

表 1001. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
锁定	appl_lock_list	基本

表 1002. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-

用法 此标识与 appl_id 配合使用以唯一标识对某个对象挂起锁定的事务，此应用程序正在等待对该对象获取锁定。

server_db2_type -“受监视的（服务器）节点上的数据库管理器类型”

标识要监视的数据库管理器的类型。

表 1003. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法 它包含数据库管理器的下列配置类型的其中一个:

API 符号常量
命令行处理器输出

sqlf_nt_server
带本地客户机和远程客户机的数据库服务器

sqlf_nt_stand_req
带本地客户机的数据库服务器

API 符号常量是在包含文件 *sqlutil.h* 中定义的。

server_instance_name －“服务器实例名称”

对其获取快照的数据库管理器实例的名称。

元素标识

server_instance_name

元素类型

信息

表 1004. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

表 1005. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法 如果同一系统上存在多个数据库管理器实例，那么此数据项将用于唯一标识对其发出快照调用的实例。如果要将监视器输出保存在文件或数据库中以便以后进行分析，并且需要区分来自不同数据库管理器实例的数据，那么此信息将会非常有用。

server_platform －“服务器操作系统”

运行数据库服务器的操作系统。

元素标识

server_platform

元素类型

信息

表 1006. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 1007. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 此元素可用于远程应用程序的问题确定。可在头文件 *sqlmon.h* 中找到此字段的值。

server_prdid －“服务器产品/版本标识”

正在服务器上运行的产品和版本。

表 1008. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

表 1009. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法 其格式为 PPPVVRRM，其中:

PPP 为 SQL

VV 标识两位版本号（版本只有一位时高位为 0）

RR 标识两位发行版号（发行版只有一位时高位为 0）

M 标识 1 个字符的修改级别（0-9 或 A-Z）

server_version －“服务器版本”

返回该信息的服务器的版本。

表 1010. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法

此字段标识收集数据库系统监视器信息的数据库服务器的级别。这允许应用程序根据返回数据的服务器级别来解释数据。有效值为:

SQLM_DBMON_VERSION1
数据由 DB2 版本 1 返回

SQLM_DBMON_VERSION2
数据由 DB2 版本 2 返回

SQLM_DBMON_VERSION5
数据由 DB2 通用数据库版本 5 返回

SQLM_DBMON_VERSION5_2
数据由 DB2 通用数据库版本 5.2 返回

SQLM_DBMON_VERSION6
数据由 DB2 通用数据库版本 6 返回

SQLM_DBMON_VERSION7
数据由 DB2 通用数据库版本 7 返回

SQLM_DBMON_VERSION8
数据由 DB2 通用数据库版本 8 返回

SQLM_DBMON_VERSION9
数据由 DB2 数据库 Linux? 版、UNIX? 版和 Windows? 版版本 9 返回

SQLM_DBMON_VERSION9_5
数据由 DB2 数据库 Linux 版、UNIX 版和 Windows 版版本 9.5 返回

service_class_id - "服务类标识"监视元素

服务子类的唯一标识。对于工作负载而言，此标识代表此工作负载所映射到的服务子类。对于工作单元而言，此标识代表发出此工作单元的连接的相关联工作负载的服务子类标识。

表 1011. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	始终收集
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	始终收集
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 1012. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
锁定	-	-
工作单元	-	-
统计信息	event_histogrambin	-
统计信息	event_scstats	-

用法

此元素的值与视图 SYSCAT.SERVICECLASSES 的列 SERVICECLASSID 中的某个值匹配。使用此元素来查找服务子类名或者对来自不同来源的服务子类的信息进行链接。例如，将服务类统计信息与直方图条形记录相连接。

service_level - "服务级别"

这是 DB2 实例的当前校正服务级别。

表 1013. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

service_subclass_name – “服务子类名”监视元素

服务子类的名称。

表 1014. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集

表 1015. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
锁定	-	-
工作单元	-	-
活动	event_activity	-
统计信息	event_scstats	-
统计信息	event_qstats	-

用法

将此元素与其他活动元素配合使用来分析活动的行为，或者与其他统计信息元素配合使用来分析服务类或阈值队列。

service_superclass_name – “服务超类名”监视元素

服务超类的名称。

表 1016. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	始终收集
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	始终收集

表 1016. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集

表 1017. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	-
活动	event_activity	-
统计信息	event_scstats	-
统计信息	event_qstats	-

用法

将此元素与其他活动元素配合使用来分析活动的行为，或者与其他统计信息元素配合使用来分析服务类或阈值队列。

session_auth_id - “会话授权标识”监视元素

此应用程序将使用的会话的当前授权标识。为监视工作负载管理活动，此监视元素描述将活动插入到系统中时使用的会话授权标识。

表 1018. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
锁定	appl_lock_list	基本

表 1019. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
活动	event_activity	-
阈值违例	event_activity	-

用法

可使用此元素来确定要用于预编译 SQL 语句和/或执行 SQL 语句的授权标识。此监视元素不报告在执行存储过程内设置的任何会话授权标识值。

shr_workspace_num_overflows - “共享工作空间溢出数”

共享工作空间溢出其分配内存边界的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1020. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1021. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 将此元素与 shr_workspace_size_top 配合使用来确定是否需要增加共享工作空间的大小以避免溢出。共享工作空间的溢出可能导致性能下降，以及从应用程序共享内存分配的其他堆出现内存不足错误。

在数据库级别，报告的元素将来自报告具有最大共享工作空间大小的元素的共享工作空间。在应用程序级别，此项是当前应用程序使用的工作空间的溢出数。

shr_workspace_section_inserts - "共享工作空间节插入数"

应用程序将 SQL 节插入到共享工作空间中的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1022. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1023. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可执行段的工作副本存储在共享工作空间中。此计数器指示副本何时不可用并且必须插入。

在数据库级别，此项是针对数据库的所有共享工作空间中的每个应用程序进行的所有插入的累积总数。在应用程序级别，此项是针对此应用程序的共享工作空间中的所有段的所有插入的累积总数。

shr_workspace_section_lookups - "共享工作空间节查询数"

应用程序在共享工作空间中对 SQL 节进行查询的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1024. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1025. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 每个应用程序都可以访问共享工作空间，该工作空间中保留了可执行段的工作副本。

此计数器指示为找到应用程序的特定段而访问共享工作空间的次数。在数据库级别，此项是针对数据库的所有共享工作空间中的每个应用程序进行的所有查询的累积总数。在应用程序级别，此项是针对此应用程序的共享工作空间中的所有段的所有查询的累积总数。

可将此元素与"共享工作空间节插入数"一起使用来调整共享工作空间的大小。共享工作空间的大小由 app_ctl_heap_sz 配置参数控制。

shr_workspace_size_top - "最大共享工作空间大小"

共享工作空间达到的最大大小。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1026. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 1027. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 此元素指示对激活后的数据库运行工作负载所需的共享工作空间最多字节数。

在数据库级别，此项是所有共享工作空间达到的最大大小。 在应用程序级别，此项是当前应用程序使用的共享工作空间的最大大小。

如果共享工作空间溢出，那么表示此元素包含溢出期间共享工作空间达到的最大大小。检查共享工作空间溢出数以确定是否存在此情况。

工作空间溢出时，将会临时从应用程序共享内存的其他实体借出内存。这可能导致这些实体出现内存不足错误，也可能导致性能下降。可通过增加 APP_CTL_HEAP_SZ 来降低溢出的机率。

smallest_log_avail_node －“带有最少可用日志空间的节点”

此元素指示带有最少可用日志空间量（以字节计）的节点并且仅对全局快照返回。

元素标识

smallest_log_avail_node

元素类型

信息

表 1028. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 将此元素与 appl_id_oldest_xact 一起使用以确保数据库有足够的日志空间可用。在全局快照中，appl_id_oldest_xact、total_log_used 和 total_log_available 对应于此节点上的值。

sort_heap_allocated －“分配的总排序堆”

排序堆空间的总分配页数，用于获取快照时处于所选级别的所有排序。

表 1029. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
数据库	dbase	基本

用法 为每个排序分配的内存量可以是可用排序堆大小的一部分或全部。排序堆大小是按 *sortheap* 数据库配置参数中定义的可用于每个排序的内存量。

单个应用程序可使多个排序同时处于活动状态。例如，在某些情况下带有子查询的 SELECT 语句可能导致并行排序。

可在两个级别收集信息:

- 在数据库管理器级别，它表示为数据库管理器的所有活动数据库中的所有排序分配的排序堆空间的总和
- 在数据库级别，它表示为数据库中的所有排序分配的排序堆空间的总和。

常规内存估计不包括排序堆空间。如果出现过多排序，除了需要用于运行数据库管理器的基本内存之外，还应额外添加用于排序堆的内存。通常排序堆越大，排序越有效。适当使用索引可以降低必需的排序量。

可使用在数据库管理器级别返回的信息来帮助您调整 *sheapthres* 配置参数。如果元素值大于或等于 *sheapthres*，那么表示排序未达到 *sortheap* 参数定义的排序堆已满的状态。

sort_heap_top - "排序专用堆高水位标记"

数据库管理器范围的专用排序内存高水位标记（以 4 KB 页计）。

表 1030. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 此元素可用来确定是否已将 SHEAPTHRES 配置参数设置为最佳的值。例如，如果此水位标记接近或超过 SHEAPTHRES，那么可能应该增大 SHEAPTHRES。这是因为，超过 SHEAPTHRES 后，专用排序操作可使用的内存就会减少，这会对系统性能产生不利影响。

sort_overflows - "排序溢出数"监视元素

用完排序堆并且可能需要磁盘空间以供临时存储器使用的排序总数。

表 1031. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1032. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	基本

可将快照监视的计数器复位。

表 1033. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	语句，排序

用法

在数据库或应用程序级别，将此元素与 **total_sorts** 一起使用来计算必须溢出至磁盘的排序的百分比。如果此百分比很高，那么您可能想要通过增加 **sortheap** 的值来调整数据库配置。

在语句级别，使用此元素来标识需要大量排序的语句。如果另外进行调整以降低所需排序量，这些语句将从中受益。

当排序溢出时，因为排序需要合并阶段并且可能需要更多 I/O（如果数据需要写至磁盘），所以可能会导致其他开销。

此元素提供有关一个语句、一个应用程序或访问一个数据库的所有应用程序的信息。

sort_shrheap_allocated – “对当前分配的共享堆进行排序”

数据库中分配的共享排序内存总量。

表 1034. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 此元素可用来评估共享排序内存的阈值。如果此值经常大幅高于或低于当前共享排序内存阈值，那么可能应该调整阈值。

注: 如果 SHEAPTHRES_SHR 数据库配置参数为 0，那么“共享排序内存阈值”由 SHEAPTHRES 数据库管理器配置参数确定。否则将由 SHEAPTHRES_SHR 的值确定。

sort_shrheap_top - “排序共享堆高水位标记”

数据库范围的共享排序内存高水位标记（以 4 KB 页计）。

表 1035. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 此元素可用来评估是否已将 SHEAPTHRES（或 SHEAPTHRES_SHR）设置为最佳的值。例如，如果这个高水位标记持续地远小于共享排序内存阈值，那么有可能需要减小这个阈值，从而释放内存以供其他数据库功能使用。相反，如果这个高水位标记开始接近共享排序内存阈值，那么可能表示需要增大这个阈值。由于共享排序内存阈值是硬限制，所以这一点很重要。排序内存总量达到这个阈值后，就无法启动更多的共享排序操作。

此元素和专用排序内存高水位标记还可以帮助用户确定是否需要相互独立地设置共享排序阈值和专用排序阈值。通常，如果 SHEAPTHRES_SHR 数据库配置选项值为 0，那么共享排序内存阈值由 SHEAPTHRES 数据库管理器配置选项值确定。但是，如果专用排序内存高水位标记与共享排序内存高水位标记相差很大，那么可能表明用户需要重设 SHEAPTHRES，并将 SHEAPTHRES_SHR 设置为基于共享排序内存高水位标记的更合适的值。

source_service_class_id - “源服务类标识”监视元素

生成此元素所属的阈值违例记录时从中重新映射此活动的服务子类的标识。对于除 REMAP ACTIVITY 以外的任何阈值操作，此元素的值为零。

元素标识

source_service_class_id

元素类型

信息

表 1036. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来跟踪活动在它重新映射到的各个服务类之间的路径。此元素还可用于计算从给定服务子类向外映射的活动数的聚集值。

sp_rows_selected - “存储过程返回的行数”

此元素包含自启动联合服务器实例或最后一次复位数据库监视计数器以后，因对此应用程序执行存储过程操作而导致从数据源发送至联合服务器的行数。

表 1037. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法 此元素有若干用途。可使用它并借助以下公式来计算对于每个存储过程而言从数据源发送至联合服务器的平均行数:

```
    每个存储过程的行数
  = 返回的行数
  / 调用的存储过程数
```

还可以计算对于此应用程序而言行从数据源返回至联合服务器的平均时间:

```
平均时间 = 聚集存储过程响应时间 / 返回的行数
```

sql_chains – "尝试的 SQL 链数"

表示语句处理期间在 DB2 Connect 网关与主机之间采用 *n* 个数据传输的 SQL 语句数。范围 *n* 是由 *num_transmissions_group* 元素指定的。

表 1038. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	基本

可将快照监视的计数器复位。

例如，如果链接已打开，并且 PREP 和 OPEN 语句链接在一起，该链一共占用了两个传输，那么 *sql_chains* 报告为"1"，而 *sql_stmts* 报告为"2"。

如果链接已关闭，那么 *sql_chains* 计数等于 *sql_stmts* 计数。

用法 使用此元素来获取有关处理期间使用 2、3、4（等等）个数据传输的语句数的统计信息。（要处理语句，至少需要 2 个数据传输：发送和接收。）这些统计信息能够让您更好地了解数据库或应用程序级别的数据库或应用程序活动和网络流量。

注: *sql_stmts* 监视元素表示尝试将 SQL 语句发送至服务器的次数。在传输级别，同一游标内的所有语句在计数时都被当作单个 SQL 语句。

sql_req_id – "SQL 语句的请求标识"

SQL 语句中某个操作的请求标识。

表 1039. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

用法 第一个应用程序连接至数据库之后，数据库管理器每次成功处理 SQL 操作时，此标识就会递增。它的值在数据库中是唯一的，可以唯一地标识语句操作。

sql_reqs_since_commit - "上次落实后的 SQL 请求数"

自上次落实后提交的 SQL 请求数。

表 1040. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可使用此元素来监视事务的进度。

sql_stmts - "尝试的 SQL 语句数"

对于数据传输快照，此元素表示语句处理期间在 DB2 Connect 网关与主机之间采用 *n* 个数据传输的 SQL 语句数。范围 *n* 是由 *num_transmissions_group* 元素指定的。

表 1041. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
数据传输	stmt_transmissions	基本

可将快照监视的计数器复位。

对于 DCS DATABASE 快照，此语句计数为自激活数据库后处理的语句数。

对于 DCS APPLICATION 快照，此语句计数为自此应用程序建立与数据库的连接后处理的语句数。

用法 使用此元素来度量数据库或应用程序级别的数据库活动。要计算给定时间段的 SQL 语句吞吐量，可按两次快照之间所耗用的时间来分割此元素。

对于数据传输级别: 使用此元素来获取有关处理期间使用 2、3、4（等等）个数据传输的语句数的统计信息。（要处理语句，至少需要 2 个数据传输: 发送和接收。）这些统计信息能够让您更好地了解数据库或应用程序级别的数据库或应用程序活动和网络流量。

注:

1. *sql_stmts* 监视元素表示尝试将 SQL 语句发送至服务器的次数:
   - 在应用程序级别和数据库级别，游标内的每个 SQL 语句是分开计数的。
   - 在传输级别，同一游标内的所有语句在计数时都被当作单个 SQL 语句。

sqlca - "SQL 通信区（SQLCA）"

在语句完成时返回至应用程序的 SQLCA 数据结构。

表 1042. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-
活动	event_activity	-

用法

SQLCA 数据结构可用来确定语句是否成功完成。有关 SQLCA 内容的信息，请参阅 *SQL Reference, Volume 1* 中的『SQLCA（SQL 通信区）』或 *Administrative API Reference* 中的『SQLCA 数据结构』。

sqlrowsread_threshold_id - "读取 SQL 行数阈值标识"监视元素

应用于此活动的 SQLROWSREAD 阈值的标识。

表 1043. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLROWSREAD 阈值（如果有的话）。

sqlrowsread_threshold_value - "读取 SQL 行数阈值"监视元素

应用于此活动的 SQLROWSREAD 阈值的上限。

表 1044. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLROWSREAD 阈值（如果有的话）。

sqlrowsread_threshold_violated - "违反读取 SQL 行数阈值"监视元素

此监视元素返回"Yes"表明此活动已违反 SQLROWSREAD 阈值。"No"表明此活动尚未违反该阈值。

表 1045. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLROWSREAD 阈值。

sqlrowsreadinsc_threshold_id – “服务类中读取 SQL 行数阈值标识”监视元素

应用于此活动的 SQLROWSREADINSC 阈值的标识。

表 1046. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLROWSREADINSC 阈值（如果有的话）。

sqlrowsreadinsc_threshold_value – “服务类中读取 SQL 行数阈值”监视元素

应用于此活动的 SQLROWSREADINSC 阈值的上限。

表 1047. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLROWSREADINSC 阈值（如果有的话）。

sqlrowsreadinsc_threshold_violated – “违反服务类中读取 SQL 行数阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 SQLROWSREADINSC 阈值。“No”表明此活动尚未违反该阈值。

表 1048. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLROWSREADINSC 阈值。

sqlrowsreturned_threshold_id - "返回所读取 SQL 行数阈值标识"监视元素

应用于此活动的 SQLROWSRETURNED 阈值的标识。

表 1049. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLROWSRETURNED 阈值（如果有的话）。

sqlrowsreturned_threshold_value - "返回所读取 SQL 行数阈值"监视元素

应用于此活动的 SQLROWSRETURNED 阈值的上限。

表 1050. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLROWSRETURNED 阈值（如果有的话）。

sqlrowsreturned_threshold_violated - "违反返回所读取 SQL 行数阈值"监视元素

此监视元素返回"Yes"表明此活动已违反 SQLROWSRETURNED 阈值。"No"表明此活动尚未违反该阈值。

表 1051. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLROWSRETURNED 阈值。

sqltempspace_threshold_id - "SQL 临时空间阈值标识"监视元素

应用于此活动的 SQLTEMPSPACE 阈值的标识。

表 1052. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLTEMPSPACE 阈值（如果有的话）。

sqltempspace_threshold_value - "SQL 临时空间阈值"监视元素

应用于此活动的 SQLTEMPSPACE 阈值的上限。

表 1053. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定应用于此活动的 SQLTEMPSPACE 阈值（如果有的话）。

sqltempspace_threshold_violated - "违反 SQL 临时空间阈值"监视元素

此监视元素返回"Yes"表明此活动已违反 SQLTEMPSPACE 阈值。"No"表明此活动尚未违反该阈值。

表 1054. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLTEMPSPACE 阈值。

ss_exec_time - "子节执行耗用时间"

执行子节所花的时间（以秒计）。

表 1055. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1056. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 允许您跟踪子节进度。

ss_node_number - "子节节点号"

执行子节的节点。

表 1057. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1058. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 用于使每个子节与执行该子节的数据库分区相关联。

ss_number - "子节号"

标识与返回的信息相关联的子节。

表 1059. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1060. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 此编号与访问方案中可使用 db2expln 获取的子节号相关联。

ss_status - "子节状态"

正在执行的子节的当前状态。

表 1061. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 当前状态值可能是:

- 正在执行（sqlmon.h 中的 SQLM_SSEXEC）
- 等待锁定
- 在表队列上等待接收数据
- 在表队列上等待发送数据

ss_sys_cpu_time - "子节使用的系统 CPU 时间"

当前执行的语句子节使用的总系统 CPU 时间（以秒和微秒计）。

元素标识

ss_sys_cpu_time

元素类型

时间

表 1062. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	时间戳记

表 1063. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

ss_usr_cpu_time - "子节使用的用户 CPU 时间"

当前执行的语句子节使用的总用户 CPU 时间（以秒和微秒计）。

元素标识

ss_usr_cpu_time

元素类型

时间

表 1064. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	时间戳记

表 1065. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

start_time - "事件启动时间"

启动工作单元、启动语句或检测死锁的日期和时间。此元素在 event_start API 结构中指示事件监视器的启动时间。

表 1066. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_start	时间戳记
事务	event_xact	时间戳记
语句	event_stmt	时间戳记
死锁	event_deadlock	时间戳记
死锁	event_dlconn	时间戳记
带有详细信息的死锁	event_detailed_dlconn	时间戳记

用法 可使用此元素来将死锁连接记录与死锁事件记录相关联，并与 *stop_time* 一起使用来计算执行语句或事务所耗用的时间。

注: 时间戳记开关设置为 OFF 时，此元素显示为"0"。

static_sql_stmts - "尝试的静态 SQL 语句数"

尝试的静态 SQL 语句数。

表 1067. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1068. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
    dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

statistics_timestamp - "统计信息时间戳记"监视元素

生成此统计信息记录的时间。

表 1069. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-
统计信息	event_wcstats	-
统计信息	event_qstats	-
统计信息	event_histogrambin	-

用法

使用此元素来确定生成此统计信息记录的时间。

将此元素与 **last_wlm_reset** 元素配合使用以确定生成此统计信息记录中的统计信息的时间间隔。

此监视元素还可用来将对同一收集时间间隔生成的所有统计信息记录分组。

stats_cache_size - "统计信息高速缓存大小"监视元素

统计信息高速缓存的当前大小，它用在目录分区，可高速缓存实时统计信息收集生成的统计信息。

注: 由于统计信息高速缓存驻留在目录分区中，所以只有在目录分区捕获的快照才报告统计信息高速缓存大小。而在其他分区捕获的快照将报告零值。当捕获全局快照时，所有数据库分区报告的值将汇总合计。

表 1070. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

表 1071. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

使用此元素来确定当前统计信息高速缓存的大小。此值经常更改。为了评估系统使用情况，请长期在特定时间间隔捕获快照。使用此元素来调整 **catalogcache_sz** 配置参数的值。

stats_fabricate_time - "产生统计信息的活动所花的总时间"监视元素

实时统计信息收集在产生统计信息所花的总时间（以毫秒计）。产生统计信息是在查询编译期间生成统计信息所需的统计信息收集活动。如果在数据库级别收集监视元素，那么此监视元素表示在数据库上运行的所有应用程序的实时统计信息收集活动所花的总时间。如果是在语句级别收集，那么它表示语句的最新实时统计信息收集所花的时间。所有数据库分区报告的时间将汇总合计。

表 1072. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句
动态 SQL	dynsql	语句

对于快照监视来说，此元素可以复位。

表 1073. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
语句	event_stmt	-

用法

将此元素与 **stats_fabrications** 配合使用，以评估数据库级别的实时统计信息收集对性能的影响。对于动态 SQL 快照监视器，可以将此元素与 **total_exec_time** 和 **num_executions** 配合使用以评估产生统计信息的影响。对于语句事件监视器，可以将此元素与 **stmt_start** 和 **stmt_stop** 配合使用，以进一步评估实时统计信息收集的影响。

stats_fabrications －“产生统计信息的次数”监视元素

实时统计信息在查询编译期间为所有数据库应用程序执行的产生统计信息的总次数。与通过扫描表中或索引中存储的数据来获取统计信息不同，统计信息是根据索引和数据管理器维护的元数据来产生的。所有数据库分区报告的值将汇总合计。

表 1074. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句

可将快照监视的计数器复位。

表 1075. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

使用此元素来确定数据库中产生统计信息的频率。此值经常更改。为更好地了解系统使用情况，请长期在特定时间间隔捕获快照。与 **stats_fabricate_time** 配合使用时，此元素可帮助您评估产生统计信息的影响。

status_change_time －“应用程序状态更改时间”

应用程序进入当前状态的日期和时间。

元素标识

status_change_time

元素类型

时间戳记

表 1076. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	工作单元，时间戳记
锁定	appl_lock_list	工作单元，时间戳记
DCS 应用程序	dcs_appl_info	工作单元，时间戳记

用法 此元素允许您确定应用程序进入当前状态后所经历的时间。如果应用程序已处于同一状态很长一段时间，那么可能指示存在问题。

stmt_elapsed_time - "最新语句耗用时间"

最新完成的语句耗用的执行时间。

表 1077. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

用法 将此元素用作完成语句所花时间的指示符。

stmt_first_use_time - "语句第一次使用时间"

此元素显示第一次处理语句条目的时间。对于游标操作，**stmt_first_use_time** 将显示打开游标的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。

表 1078. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	时间戳记
带有详细信息的死锁的历史纪录	event_stmt_history	时间戳记
活动	event_activitystmt	时间戳记

用法 将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_history_id - "语句历史记录标识"

此数字元素显示相对于其他语句历史记录元素而言，该语句在 sequence_no 元素指示的工作单元中的运行位置。在工作单元中最早运行的语句的值最低。如果同一语句在同一工作单元中运行两次，那么该语句在两个不同位置出现时将显示两个不同的 stmt_history_id 值。

表 1079. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史记录值	event_data_value	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

用法 可使用此信息来了解导致死锁的 SQL 语句的顺序。

inact_stmthist_sz – “语句历史记录列表大小”

当带有历史记录的详细信息死锁事件监视器运行时，此元素将报告数据库监视器堆（MON_HEAP_SZ）中用于记录语句历史记录列表项的字节数。

表 1080. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	-
数据库	db	-

用法 可在调整数据库监视器堆时使用此元素。

stmt_invocation_id – “语句调用标识”监视元素

此元素显示运行 SQL 语句的例程调用的标识。该值指示在当前嵌套级别进行的例程调用次数，这些调用是该级别在应用程序中活动时进行的。

表 1081. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录[1]	event_stmt_history	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **stmt_nest_level** 监视元素配合使用，可以唯一地标识特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_isolation - "语句隔离"

此元素显示运行语句时对该语句生效的隔离值。

表 1082. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-
活动	event_activitystmt	-

可能的隔离级别值包括:

- SQLM_ISOLATION_LEVEL_NONE 0（未指定隔离级别）
- SQLM_ISOLATION_LEVEL_UR 1（未落实的读）
- SQLM_ISOLATION_LEVEL_CS 2（游标稳定性）
- SQLM_ISOLATION_LEVEL_RS 3（读稳定性）
- SQLM_ISOLATION_LEVEL_RR 4（可重复读）

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因和特定 SQL 语句的执行行为。

stmt_last_use_time - "语句上一次使用时间"监视元素

此元素显示上一次处理语句条目的时间。对于游标操作，stmt_last_use_time 显示操作可能为打开、访存或关闭的游标的上一次操作的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。

表 1083. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	时间戳记
带有详细信息的死锁的历史纪录	event_stmt_history	时间戳记
活动	event_activitystmt	-

用法 将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_lock_timeout - "语句锁定超时"监视元素

此元素显示运行语句时对该语句生效的锁定超时值。

表 1084. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

表 1084. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录[1]	event_stmt_history	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因和特定 SQL 语句的执行行为。

stmt_nest_level - "语句嵌套级别"监视元素

此元素显示运行语句时生效的嵌套或递归的级别；每个嵌套级别对应一个存储过程或用户定义的函数（UDF）的嵌套或递归调用。

表 1085. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录[1]	event_stmt_history	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **stmt_invocation_id** 监视元素配合使用，可以唯一地标识特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_node_number - "语句节点"

执行语句的节点。

表 1086. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

用法 用于使每个语句与执行该语句的节点相关联。

stmt_operation/operation －“语句操作”监视元素

当前处理或最新处理的语句操作（如果当前未运行任何操作）。

表 1087. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 1088. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁[1]	event_detailed_dlconn	-
语句	event_stmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可使用此元素来确定正在执行或最近完成的操作。

它可以是下列其中一项:

对于 SQL 操作:

- SELECT
- PREPARE
- EXECUTE
- EXECUTE IMMEDIATE
- OPEN
- FETCH
- CLOSE
- DESCRIBE
- STATIC COMMIT
- STATIC ROLLBACK
- FREE LOCATOR
- PREP_COMMIT
- CALL
- PREP_OPEN
- PREP_EXEC
- COMPILE
- DROP PACKAGE

对于非 SQL 操作:

- RUN STATISTICS
- REORG
- REBIND
- REDISTRIBUTE
- GET TABLE AUTHORIZATION
- GET ADMINISTRATIVE AUTHORIZATION

注: API 用户应参考包含数据库系统监视器常量定义的 sqlmon.h 头文件。

stmt_pkgcache_id - "语句程序包高速缓存标识"

此元素显示动态 SQL 语句的内部程序包高速缓存标识。

表 1089. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集

表 1090. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

表 1091. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录[1]	event_stmt_history	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

在多分区环境中，对于高速缓存的语句，每个分区都有一个唯一的语句标识。给定语句在各分区中的标识可能不同。

在全局动态 SQL 快照中，只返回第一个语句标识。

stmt_query_id - "语句查询标识"监视元素

此元素显示对用作游标的任何 SQL 语句指定的内部查询标识。

表 1092. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录[1]	event_stmt_history	-
活动	event_activitystmt	-

用法

通过将此元素与 **stmt_nest_level** 监视元素配合使用，可以唯一地标识特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_sorts - "语句排序数"

为处理 stmt_operation 而对一组数据排序的总次数。

表 1093. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句
应用程序	stmt	语句
动态 SQL	dynsql	语句

用法 可使用此元素来帮助标识对索引的需求，原因是索引可以降低数据排序的需求。通过使用上表中的相关元素，您可以标识此元素为其提供排序信息的 SQL 语句，然后通过查看要排序的列（如 ORDER BY 和 GROUP BY 子句和连接列中使用的列）分析此语句以确定索引候选项。有关检查索引是否用于优化排序性能的信息，请参阅《*管理指南*》中的**说明**。

此计数包括数据库管理器为执行语句而在内部生成的临时表排序。排序数与 SQL 语句的第一个 FETCH 操作相关联。当语句的操作是第一个 FETCH 时，将返回此信息。您应注意到对于分块游标而言，打开游标时将执行若干次访存。在这种情况下，很难使用快照监视器来获取排序数，原因是 DB2 在内部发出第一个 FETCH 时需要获取快照。

如果要确定使用分块游标时执行的排序数，那么更可靠的方法是使用对语句声明的事件监视器。CLOSE 游标的语句事件中的 total_sorts 计数器包含执行定义了游标的语句时执行的排序总数。

stmt_source_id - "语句源标识"

此元素显示对运行的 SQL 语句的源指定的内部标识。

表 1094. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录[1]	event_stmt_history	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **appl_id** 配合使用，可以唯一地标识运行特定 SQL 语句的请求的来源。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_start - "语句操作开始时间戳记"

stmt_operation 开始执行的日期和时间。

元素标识

stmt_start

元素类型

时间戳记

表 1095. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

用法 可将此元素与 stmt_stop 一起使用来计算执行语句操作时耗用的时间。

stmt_stop - "语句操作停止时间戳记"

stmt_operation 停止执行的日期和时间。

元素标识

stmt_stop

元素类型

时间戳记

表 1096. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

用法 可将此元素与 stmt_start 一起使用来计算执行语句操作时耗用的时间。

stmt_sys_cpu_time - "语句使用的系统 CPU 时间"

当前执行的语句使用的总系统 CPU 时间（以秒和微秒计）。

元素标识

stmt_sys_cpu_time

元素类型

时间

表 1097. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句，时间戳记
应用程序	stmt	语句，时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

stmt_text - "SQL 语句文本"监视元素

QL 语句的文本。

表 1098. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	始终收集

表 1099. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
动态 SQL	dynsql	基本
DCS 语句	dcs_stmt	语句

表 1100. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

表 1100. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁[1]	event_detailed_dlconn	-
带有详细信息的死锁的历史记录[1]	event_stmt_history	-
语句	event_stmt	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

对于应用程序快照，此语句文本帮助您标识获取快照时执行的应用程序或最新处理的应用程序（如果获取快照时正好没有在处理的语句）。

此元素返回的信息是从 SQL 语句高速缓存中获取的，并且在高速缓存溢出时可能不可用。如果要捕获语句的 SQL 文本，那么唯一保险的方法是对语句使用事件监视器。

对于动态 SQL 语句，此元素标识与程序包相关联的 SQL 文本。

对于语句事件监视器而言，将仅对动态语句返回此元素。如果语句事件监视器记录在语句事件监视器的 BUFFERSIZE 选项所指定大小的缓冲区中放不下，那么 **stmt_text** 监视器的值可能会被截断以使该记录能够放得下。

对于 EVENT_STMT_HISTORY 事件监视器而言，将仅对动态语句返回此元素。对于其余事件监视器而言，仅当 **stmt_text** 包含在 SQL 语句高速缓存中时，才会对动态和静态语句返回此元素。

有关如何查询系统目录表以获取因为性能注意事项而未提供的静态 SQL 语句文本的信息，请参阅 **section_number** 监视元素。

stmt_type - “语句类型”监视元素

所处理语句的类型。

表 1101. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 1102. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁[1]	event_detailed_dlconn	-
语句	event_stmt	-
活动	event_activitystmt	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可使用此元素来确定正在执行的语句的类型。它可以是下列其中一项:

- 静态 SQL 语句
- 动态 SQL 语句
- SQL 语句之外的操作；如绑定或预编译操作。

对于快照监视器，此元素描述当前正在处理或最新处理的语句。

注: API 用户应参考包含数据库系统监视器常量定义的 sqlmon.h 头文件。

stmt_usr_cpu_time - "语句使用的用户 CPU 时间"

当前执行的语句使用的总*用户* CPU 时间（以秒和微秒计）。

元素标识

stmt_usr_cpu_time

元素类型

时间

表 1103. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句，时间戳记
应用程序	stmt	语句，时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

stmt_value_data - "值数据"

此元素包含 SQL 语句的数据值的字符串表示。LOB、LONG 和结构化类型参数显示为空字符串。日期、时间和时间戳记字段记录为 ISO 格式。

表 1104. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	stmt_value_data	-

表 1104. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activityvals	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_index - "值索引"

此元素表示 SQL 语句中使用的输入参数标记或主变量的位置。

表 1105. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	stmt_value_data	-
活动	event_activityvals	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_isnull - "包含空值"监视元素

此元素显示与 SQL 语句相关联的数据值是否为空值。

表 1106. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	stmt_value_isnull	-
活动	event_activityvals	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_isreopt - "用于语句重新优化的变量"监视元素

此元素显示是否在语句重新优化期间使用了提供的值。如果语句重新优化（例如，因为设置 REOPT 绑定选项），或者该值在此重新优化期间用作 SQL 编译器的输入，那么返回值"True"。

表 1107. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_data_value	-
活动	event_activityvals	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可以将此元素与提供的编译环境一起使用，以允许就 SQL 编译器对 SQL 语句的处理进行完整的分析。

stmt_value_type - "值类型"监视元素

此元素包含与 SQL 语句相关联的数据值类型的字符串表示。

表 1108. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	stmt_value_type	-
活动	event_activityvals	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

sto_path_free_sz - "自动存储器路径可用空间量"

此元素显示存储器路径指向的文件系统上的可用空间量。如果多个存储器路径指向同一个文件系统，那么不会在它们之间划分可用大小。

元素标识

fs_free_size

元素类型

信息

表 1109. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法 可以将此元素与下列元素配合使用以收集每个节点的数据库空间利用率数据:

- db_storage_path
- fs_used_size
- fs_total_size
- fs_id
- fs_type

stop_time - "事件停止时间"

语句停止执行的日期和时间。

表 1110. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	时间戳记

用法 可将此元素与 *start_time* 配合使用来计算执行语句所耗用的时间。

对于 FETCH 语句事件，此项是上一次成功的访存操作的时间。

注: 时间戳记开关设置为 OFF 时，此元素显示为"0"。

stored_proc_time - "存储过程时间"

此元素包含此数据源响应存储过程语句所花的总时间（以毫秒计），这些存储过程语句来自联合服务器启动后或数据库监视计数器上一次复位后在此联合服务器实例上运行的所有应用程序或单个应用程序。

表 1111. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

响应时间是以联合服务器将存储过程提交给数据源的时间与数据源响应以指示已处理存储过程的时间之差量度的。

用法 使用此元素来确定在此数据源上处理存储过程所花的实际时间。

stored_procs - "存储过程"

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位后，联合服务器代表任何应用程序在此数据源上调用的存储过程总数。

表 1112. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法 使用此元素来确定联合数据库或应用程序在本地对联合数据库调用的存储过程数。

sync_runstats －"同步 RUNSTATS 活动总数"监视元素

实时统计信息收集对数据库中的所有应用程序触发的同步 RUNSTATS 总数。此值同时包括成功和不成功的同步 RUNSTATS 命令。所有数据库分区报告的值将汇总合计。

表 1113. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句

可将快照监视的计数器复位。

表 1114. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法

使用此监视元素来确定实时统计信息收集已在数据库中触发的同步 RUNSTATS 活动数。此值经常更改。为更好地了解系统使用情况，请长期在特定时间间隔捕获快照。与 **sync_runstats_time** 配合使用时，此元素可帮助您评估实时统计信息收集触发的同步 RUNSTATS 活动对性能的影响。

sync_runstats_time －"同步 RUNSTATS 活动所花的总时间"监视元素

实时统计信息收集触发的同步 RUNSTATS 活动所花的总时间（以毫秒计）。在查询编译期间发生同步 RUNSTATS 活动。在数据库级别，此监视元素表示由实时统计信息收集触发的在数据库上运行的所有应用程序的同步 RUNSTATS 活动所花的总时间。在语句级别，它表示由实时统计信息收集触发的特定语句的最新同步 RUNSTATS 活动所花的时间。所有数据库分区报告的值将汇总合计。

表 1115. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句
动态 SQL	dynsql	语句

对于快照监视来说，此元素可以复位。

表 1116. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
语句	event_stmt	-

用法

将此元素与 **sync_runstats** 配合使用，以评估在数据库级别由实时统计信息收集触发的同步 RUNSTATS 活动对性能的影响。

对于动态 SQL 快照监视器，将此元素与 **total_exec_time** 和 **num_executions** 配合使用以评估同步 RUNSTATS 对查询性能的影响。

对于语句事件监视器，将此元素与 **stmt_start** 和 **stmt_stop** 配合使用以进一步评估实时统计信息收集的影响。

system_cpu_time - "系统 CPU 时间"

数据库管理器代理进程、工作单元或语句使用的总系统 CPU 时间（以秒和微秒计）。

当监视开关或时间戳记开关未打开时，将不收集此元素。在这种情况下，监视元素改为显示 -1。

表 1117. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
事务	event_xact	-
语句	event_stmt	-
活动	event_activity	-

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为其他调整而受益的应用程序。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

tab_file_id - "表文件标识"监视元素

表的文件标识（FID）。

表 1118. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE_METRICS 表函数 - 获取表度量值	始终收集

用法

tab_type – “表类型”监视元素

此接口返回基于 sqlmon.h 中的定义的文本标识，即 USER_TABLE、DROPPED_TABLE、TEMP_TABLE、CATALOG_TABLE 或 REORG_TABLE。

表 1119. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE_METRICS 表函数 – 获取表度量值	始终收集

用法

table_file_id – “表文件标识”监视元素

此项是表的文件标识（FID）。

表 1120. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 – 获取表度量值	始终收集

表 1121. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
表	表	基本
锁定	appl_lock_list	锁定
锁定	lock	锁定

表 1122. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	-

用法 此元素仅供参考。返回它是为了获取与数据库系统监视器的先前版本的兼容性，并且它可能未唯一地标识该表。请使用 **table_name** 和 **table_schema** 监视元素来标识表。

table_name – “表名”监视元素

表的名称。

表 1123. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 – 获取表度量值	始终收集
MON_GET_INDEX 表函数 – 获取索引度量值	始终收集

表 1124. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock	锁定
锁定	lock_wait	锁定

表 1125. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
表	event_table	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **table_schema** 配合使用，可以确定资源争用的根源。

在应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表，原因是它目前被另一应用程序锁定。对于快照监视而言，仅当“锁定”监视器组信息设置为 ON 并且 **lock_object_type** 表明应用程序正在等待获取表锁定时，此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其挂起这些锁定的表。

对于表级别的快照和事件监视，此项是对其收集信息的表。对于临时表，**table_name** 的格式为 『TEMP (*n*, *m*)』，其中:

- *n* 是表空间标识
- *m* 是 **table_file_id** 元素

table_scans - “表扫描次数”监视元素

对此表执行扫描的次数。

表 1126. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

用法

table_schema - “表模式名”监视元素

表的模式。

表 1127. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集
MON_GET_INDEX 表函数 - 获取索引度量值	始终收集

表 1128. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock	锁定
锁定	lock_wait	锁定

表 1129. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
表	event_table	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **table_name** 配合使用，可以确定资源争用的根源。

对于应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表的模式，原因是它目前被另一应用程序锁定。仅当 **lock_object_type** 指示应用程序正在等待获取表锁定时，才设置此元素。对于应用程序级别和应用程序锁定级别的快照监视而言，仅当“锁定”监视器组信息设置为 ON 时，此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其挂起这些锁定的表。

对于表级别的快照和事件监视，此元素标识对其收集信息的表的模式。对于临时表，**table_schema** 的格式为 『*<agent_id><auth_id>*』，其中:

- *agent_id* 是创建临时表的应用程序的应用程序句柄
- *auth_id* 是应用程序用来连接至数据库的授权标识

table_type - "表类型"监视元素

与返回的信息相对应的表的类型。

表 1130. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	始终收集

表 1131. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 1132. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法

使用此元素来帮助确定与返回的信息相对应的表。如果该表是用户表或系统目录表，那么可使用 **table_name** 和 **table_schema** 来标识表。

表的类型可以是下列其中一项。可能的值是基于 sqlmon.h 文件中的定义的文本字符串。

USER_TABLE
用户表。

TEMP_TABLE
临时表。将返回有关临时表的信息，即使这些表在使用后未保留在数据库中。您会发现有关此类型的表的信息仍然有用。

CATALOG_TABLE
系统目录表。

tablespace_auto_resize_enabled - "允许自动调整表空间大小"监视元素

此元素描述是否允许自动调整表空间的大小。值为 1 表示"是"，值为 0 表示"否"。

表 1133. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1134. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

此元素仅适用于 DMS 表空间和非临时自动存储器表空间。如果此元素设置为 1，那么能够自动调整大小。有关表空间的增长率和最大大小的信息，请参阅下列监视元素。

- **tablespace_max_size**
- **tablespace_increase_size**
- **tablespace_increase_size_percent**

tablespace_content_type －“表空间内容类型”监视元素

表空间中的内容类型。

表 1135. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1136. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

表空间中的内容类型（在 sqlmon.h 中定义）可以是下列其中一项:

- 所有类型的永久数据。
  - 常规表空间: SQLM_TABLESPACE_CONTENT_ANY
  - 大型表空间: SQLM_TABLESPACE_CONTENT_LARGE
- 系统临时数据: SQLM_TABLESPACE_CONTENT_SYSTEMP
- 用户临时数据: SQLM_TABLESPACE_CONTENT_USRTEMP

tablespace_cur_pool_id －“当前使用的缓冲池”监视元素

表空间当前使用的缓冲池的标识。

表 1137. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1138. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法 每个缓冲池由唯一的整数标识。此元素的值与视图 SYSCAT.BUFFERPOOLS 的列 BUFFERPOOLID 中的值相匹配。

tablespace_current_size -“当前表空间大小”

此元素显示表空间的当前大小（以字节计）。

表 1139. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于 DMS 和自动存储器表空间，此元素表示所有表空间容器的总大小（以字节计）。此值等于表空间的总页数（tablespace_total_pages）乘以表空间的页大小（tablespace_page_size）的积。此元素不适用于 SMS 表空间或临时自动存储器表空间。

在为自动存储器表空间创建表空间时，当前大小可能与初始大小不匹配。当前大小的值应在页大小乘以扩展数据块大小乘以创建时初始大小的存储器路径数的积的范围内 （通常大于该积，但有时小于该积）。它将总是小于或等于 tablespace_max_size（如果设置了）。这是因为容器只能以完整的扩展数据块大小增长，并且必须以集合的方式增长。

tablespace_extent_size -“表空间扩展数据块大小”监视元素

表空间使用的扩展数据块大小。

表 1140. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1141. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

tablespace_free_pages -“表空间中的空闲页数”监视元素

表空间中当前空闲的总页数。

表 1142. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1143. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

此元素仅适用于 DMS 表空间。

tablespace_id - "表空间标识"监视元素

唯一表示当前数据库使用的表空间的整数。

表 1144. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	始终收集
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	始终收集

表 1145. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
表	表	基本

表 1146. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法

此元素的值与视图 SYSCAT.TABLESPACES 的列 TBSPACEID 中的值相匹配。

tablespace_increase_size - "增加字节大小"

此元素显示表空间已满并且需要更多空间时自动调整大小的表空间将增加的大小（以字节计）。

表 1147. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项表示将添加表空间的空间量，当该表空间变满并且请求更多空间，同时又未达到最大表空间大小时，该表空间可自动调整大小。如果此元素的值为 -1（或者在快照输出中为 『AUTOMATIC』），那么 DB2 将自动确定需要添加空间时的值。此元素仅适用于能够自动调整大小的表空间。

tablespace_increase_size_percent - "增加大小（以百分比计）"监视元素

此元素显示表空间已满并且需要更多空间时自动调整大小的表空间将增加的量。实际字节数是根据表空间当时的大小来调整大小时确定的。

表 1148. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项表示将添加表空间的空间量，当该表空间变满并且请求更多空间，同时又未达到最大表空间大小时，该表空间可自动调整大小。增长率以调整表空间大小时当前表空间大小（tablespace_current_size）所占的百分比为基础。此元素仅适用于能够自动调整大小的表空间。

tablespace_initial_size －“初始表空间大小”

自动存储器表空间的初始大小（以字节计）。

表 1149. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于非临时自动存储器表空间，此监视元素表示创建表空间时的初始大小（以字节计）。

tablespace_last_resize_failed －“上一次调整大小尝试失败”

此元素描述上一次尝试自动增加表空间大小是否失败。值为 1 意味着肯定；0 意味着否定。

表 1150. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于自动存储器表空间，此元素可能显示任何数据库存储器路径上都没有留下空间。对于非自动存储器表空间，失败意味着因为文件系统已满而未能扩展其中一个容器。失败的另一个原因是已达到表空间的最大大小。此元素仅适用于能够自动调整大小的表空间。

tablespace_last_resize_time －“上次成功调整大小的时间”

此元素显示用来表示上次成功增加表空间的大小的时间戳记。

表 1151. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于可自动调整大小的表空间，此元素表示上次表空间变满并且请求更多空间，同时又未达到最大表空间大小时，自动对该表空间添加空间的时间。此元素仅适用于能够自动调整大小的表空间。

tablespace_max_size - "最大表空间大小"

此元素显示表空间可自动调整大小或增长至的最大大小（以字节计）。

表 1152. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项表示可自动调整大小的表空间可自动增长至的最大大小（以字节计）。如果此值等于 tablespace_current_size 元素，那么没有空间来容纳表空间的增长。如果此元素的值为 -1，那么最大大小被认为是『无限』并且表空间可自动调整大小直到文件系统已满或达到表空间的体系结构大小限制。（此限制将在*SQL Reference*中的 SQL 限制附录中描述）。此元素仅适用于能够自动调整大小的表空间。

tablespace_min_recovery_time - "前滚的最短恢复时间"

显示可前滚表空间的最早时间点的时间戳记。

元素标识

tablespace_min_recovery_time

元素类型

信息

表 1153. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 只有不为零时才显示。

tablespace_name - "表空间名称"监视元素

表空间的名称。

表 1154. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	始终收集

表 1155. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
锁定	appl_lock_list	基本
锁定	lock	锁定

表 1155. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
锁定	lock_wait	锁定

表 1156. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁[1]	lock	-
死锁[1]	event_dlconn	-
带有详细信息的死锁[1]	event_detailed_dlconn	-
表空间	tablespace_list	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此元素可帮助您确定资源争用的源头。

它相当于数据库目录表 SYSCAT.TABLESPACES 中的 TBSPACE 列。在应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表空间的名称。另一个应用程序当前挂起针对此表空间的锁定。

在锁定级别，此项是应用程序当前对其挂起锁定的表空间的名称。

在表空间级别（缓冲池监视器组设置为“ON”时），此项是与所返回信息相对应的表空间的名称。

对于针对分区表挂起的表锁定，将不会返回此元素。

tablespace_next_pool_id - “下次启动时使用的缓冲池”监视元素

表空间在下一次数据库启动时使用的缓冲池的标识。

表 1157. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1158. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法 每个缓冲池由唯一的整数标识。此元素的值与视图 SYSCAT.BUFFERPOOLS 的列 BUFFERPOOLID 中的值相匹配。

tablespace_num_containers - "表空间中的容器数目"

表空间中的容器总数。

表 1159. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

tablespace_num_quiescers - "停顿者数目"

停顿表空间的用户数目（可以在范围 0 到 5 之间）。

表 1160. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此值表示停顿表空间的代理程序数（以 "SHARE"、"UPDATE"或"EXCLUSIVE"方式）。对于每个停顿者，将在 tablespace_quiescer 逻辑数据组中返回以下信息:

- 停顿者的用户授权标识
- 停顿者的代理程序标识
- 因为停顿而导致此表空间停顿的对象的表空间标识
- 因为停顿而导致此表空间停顿的对象的对象标识
- 停顿状态

tablespace_num_ranges - "表空间映射中的范围数"

表空间映射中的范围（条目）数。此项的范围在 1 到 100 之间（通常小于 12）。仅 DMS 表空间存在表空间映射。

表 1161. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

tablespace_page_size - "表空间页大小"监视元素

表空间使用的页大小（以字节计）。

表 1162. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1163. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

tablespace_page_top －“表空间高水位标记”监视元素

表空间中包含高水位标记的页。

表 1164. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1165. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

对于 DMS，此元素表示上一次分配的表空间扩展数据块之后第一个可用扩展数据块的页号。注意，这并非实际意义上的“高水位标记”，而是“当前水位标记”，这是因为该值可能下降。对于 SMS，此项不适用。

tablespace_paths_dropped －“表空间正在使用已删除的路径”监视元素

指示表空间正在使用已删除的存储器路径。

表 1166. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1167. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

对于正在使用自动存储器的表空间，请使用此监视元素来确定是否有任何表空间容器驻留在已被删除的存储器路径中。以物理方式从数据库中删除存储器路径之前，所有表空间都必须停止使用这些路径。要停止使用已删除的存储器路径，请删除该表空间，或者使用 ALTER TABLESPACE 语句的 REBALANCE 子句对该表空间进行重新平衡。

tablespace_pending_free_pages －“表空间中的暂挂可用页数”监视元素

如果已落实或回滚所有暂挂事务，并且已经为对象请求了新的空间，那么在表空间中将变得可用的页的数目。

表 1168. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1169. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

此元素仅适用于 DMS 表空间。

tablespace_prefetch_size - "表空间预取大小"监视元素

预取程序一次从磁盘获取的最大页数。

表 1170. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1171. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
表空间	tablespace_nodeinfo	基本

用法

- 如果启用了自动预取大小，那么此元素在*表空间*逻辑数据分组中将显示值"-1"，而在 *tablespace_nodeinfo* 逻辑数据分组中显示实际值。
- 如果未启用自动预取大小，那么此元素将在*表空间*逻辑数据分组中显示实际值，并且该元素不会出现在 *tablespace_nodeinfo* 逻辑数据分组中。

tablespace_rebalancer_extents_processed - "重新平衡程序已经处理的扩展数据块数"

自重新平衡程序启动或重新启动后（选择最近的时间）重新平衡程序已经移动的扩展数据块数。

元素标识

tablespace_rebalancer_extents_processed

元素类型

信息

表 1172. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 和 rebalance_mode 来检查重新平衡是否完成。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_extents_remaining –“重新平衡程序要处理的扩展数据块总数”

要移动的扩展数据块数目。此值是在重新平衡程序启动时间或重新启动时间计算的（选择最近的时间）。

元素标识

tablespace_rebalancer_extents_remaining

元素类型

信息

表 1173. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 来检查重新平衡是否完成。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_last_extent_moved –“重新平衡程序移动的最后一个扩展数据块”

重新平衡程序移动的最后一个扩展数据块。

元素标识

tablespace_rebalancer_last_extent_moved

元素类型

信息

表 1174. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 和 rebalance_mode 来检查重新平衡是否完成。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_mode –“重新平衡程序方式”监视元素

指示当前重新平衡过程是正在从表空间中除去空间还是正在对表空间添加空间。

表 1175. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1176. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

在添加新容器或者增大现有容器的大小时，将执行*正向重新平衡*。在正向重新平衡操作中，数据移动以表空间中的第一个扩展数据块开始并以高水位标记扩展数据块结束。

在除去容器或者减小其大小并且需要从正在释放的空间中移出数据时，将执行*反向重新平衡*。在反向重新平衡操作中，数据移动以高水位标记扩展数据块开始按反向顺序处理整个表空间，并以表空间中的第一个扩展数据块结束。

*双程重新平衡*是指先执行正向重新平衡，然后再执行反向重新平衡。如果在重新平衡操作期间既添加容器也删除容器，那么将执行双程重新平衡。

对于 DMS 非自动存储器表空间而言，此监视元素指示正在对该表空间执行的重新平衡的类型。对于 DMS 非自动表空间而言，只能执行单程正向重新平衡或单程反向重新平衡。

对于自动存储器表空间而言，此监视元素指示当前重新平衡过程正在对该表空间执行的操作。通常，启动重新平衡操作时，只需要执行单程正向重新平衡或单程反向重新平衡。但是，在某些情况下，有必要对自动存储器表空间执行双程重新平衡。

可能的 **tablespace_rebalancer_mode** 值由 sqlmon.h 文件定义，如下所示:

SQLM_TABLESPACE_NO_REBAL
未执行重新平衡。

SQLM_TABLESPACE_FWD_REBAL
正在执行正向重新平衡。

SQLM_TABLESPACE_REV_REBAL
正在执行反向重新平衡。

SQLM_TABLESPACE_FWD_REBAL_OF_2PASS
正在执行双程重新平衡操作的正向重新平衡阶段。

SQLM_TABLESPACE_REV_REBAL_OF_2PASS
正在执行双程重新平衡操作的反向重新平衡阶段。

tablespace_rebalancer_priority - “当前重新平衡程序优先级”

在数据库中运行的重新平衡程序的优先级。

表 1177. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此元素仅适用于 DMS 表空间。

tablespace_rebalancer_restart_time -“重新平衡程序重新启动时间”

表示重新平衡程序在暂停或停止后重新启动的时间戳记。

元素标识

tablespace_rebalancer_restart_time

元素类型

信息

表 1178. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项可用作重新平衡程序的完成级别的指示符。它将说明重新平衡程序重新启动的时间，并且允许派生重新平衡程序的速度和直到完成所耗用的时间。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_start_time -“重新平衡程序启动时间”

表示重新平衡程序最初启动时间的时间戳记。

元素标识

tablespace_rebalancer_start_time

元素类型

信息

表 1179. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项用于指示重新平衡程序最初启动的时间。它可以用来派生度量，以测量运行重新平衡程序的速度和完成重新平衡的估计时间。此元素仅适用于 DMS 表空间。

tablespace_state -“表空间状态”监视元素

此元素描述表空间的当前状态。

表 1180. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1181. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

此元素包含指示当前表空间状态的十六进制值。表空间的外部可视状态由特定状态值的十六进制和组成。例如，如果状态为“停顿: EXCLUSIVE”和“装入暂挂”，那么值为 0x0004 + 0x0008，即 0x000c。使用 db2tbst 命令来获取与给定十六进制值相关联的表空间状态。

表 1182. sqlutil.h 中列示的位定义

十六进制值	十进制值	State
0x0	0	正常（请参阅 sqlutil.h 中的定义 SQLB_NORMAL）
0x1	1	停顿: SHARE
0x2	2	停顿: UPDATE
0x4	4	停顿: EXCLUSIVE
0x8	8	装入暂挂
0x10	16	删除暂挂
0x20	32	备份暂挂
0x40	64	正在前滚
0x80	128	前滚暂挂
0x100	256	复原暂挂
0x100	256	恢复暂挂（未使用）
0x200	512	禁用暂挂
0x400	1024	正在重组
0x800	2048	正在备份
0x1000	4096	必须定义存储器
0x2000	8192	正在复原
0x4000	16384	脱机并且不可访问
0x8000	32768	删除暂挂
0x80000	524288	正在移动
0x2000000	33554432	可以定义存储器
0x4000000	67108864	存储器定义处于“最终”状态
0x8000000	134217728	在前滚之前已更改存储器定义
0x10000000	268435456	DMS 重新平衡程序处于活动状态
0x20000000	536870912	正在进行 TBS 删除
0x40000000	1073741824	正在进行 TBS 创建

tablespace_state_change_object_id – “状态更改对象标识”

导致表空间状态设置为“装入暂挂”或“删除暂挂”的对象。

元素标识

tablespace_state_change_object_id

元素类型

信息

表 1183. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 仅当表空间状态为“装入暂挂”或“删除暂挂”时，此元素才有意义。如果此元素的值非零，那么此元素的值与视图 SYSCAT.TABLES 的列 TABLEID 中的值相匹配。

tablespace_state_change_ts_id - “状态更改表空间标识”

如果表空间状态为“装入暂挂”或“删除暂挂”，那么此项显示导致设置表空间状态的对象的表空间标识。

元素标识

tablespace_state_change_ts_id

元素类型

信息

表 1184. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 仅当表空间状态为“装入暂挂”或“删除暂挂”时，此元素才有意义。如果此元素的值非零，那么此元素的值与视图 SYSCAT.TABLES 的列 TABLESPACEID 中的值相匹配。

tablespace_total_pages - “表空间中的总页数”监视元素

表空间中的总页数。

表 1185. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1186. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法

表空间占用的总操作系统空间。对于 DMS，此项是容器大小的总和（包括开销）。对于 SMS，此项是用于此表空间中存储的表的所有文件空间的总和（并且仅在缓冲池开关设置为 ON 时收集此项）。

tablespace_type －“表空间类型”监视元素

表空间的类型。

表 1187. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1188. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

此元素显示此表空间是数据库管理的表空间（DMS）还是系统管理的表空间（SMS）。

tablespace_type（它是在 sqlmon.h 中定义的）的值如下所示:

- 对于 DMS: SQLM_TABLESPACE_TYP_DMS
- 对于 SMS: SQLM_TABLESPACE_TYP_SMS

tablespace_usable_pages －“表空间中的可用页数”监视元素

表空间中的总页数减去开销页数。

表 1189. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1190. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法

此元素仅适用于 DMS 表空间。对于 SMS 表空间，此元素的值与 **tablespace_total_pages** 的值相同。

在表空间重新平衡期间，可用页数将包括新添加的容器的页数，但在重新平衡完成之前，这些新页可能不会反映在空闲页数中。如果未进行表空间重新平衡，那么已使用页数加上空闲页数再加上暂挂空闲页数，将等于可用页数。

tablespace_used_pages －“表空间中的已使用页数”监视元素

当前在表空间中使用的（非空闲）总页数。

表 1191. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1192. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法

这是用于 DMS 表空间的总页数。对于 SMS 表空间，它等于 **tablespace_total_pages** 监视元素的值。

tablespace_using_auto_storage －“已对表空间启用自动存储器”监视元素

此元素描述表空间是否被创建为自动存储器表空间。值为 1 表示“是”，值为 0 表示“否”。

表 1193. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 － 获取表空间度量值	DATA OBJECT METRICS BASE

表 1194. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

可使用此元素来确定是否使用自动存储器创建指定表空间（即使用 MANAGED BY AUTOMATIC STORAGE 子句创建），而不是使用显式提供的容器创建。表空间的某些容器可以在与数据库相关联的某些或所有存储器路径中。

tbsp_max_page_top －“最大表空间页号高水位标记”监视元素

上次激活数据库之后 DMS 表空间的最大分配页号。

表 1195. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

每当 **tablespace_page_top** 监视元素的值递增时，此值都将更改。

tcpip_recv_volume - "TCP/IP 接收量"监视元素

数据服务器通过 TCP/IP 从客户机接收的数据量。此值以字节计。

表 1196. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1197. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_recv_wait_time – “TCP/IP 接收等待时间”监视元素

通过 TCP/IP 等待传入客户机请求时耗用的时间（不包括空闲时间）。此值以毫秒计。

表 1198. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 – 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 – 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 – 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1199. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_recvs_total – “TCP/IP 接收总次数”监视元素

数据库服务器通过 TCP/IP 从客户机应用程序接收数据的次数。

表 1200. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 – 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1200. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1201. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_send_volume - “TCP/IP 发送量”监视元素

数据服务器发送到客户机的数据量。此值以字节计。

表 1202. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 1202. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1203. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_send_wait_time - “TCP/IP 发送等待时间”监视元素

通过 TCP/IP 向客户机发送数据时被阻塞的时间。此值以毫秒计。

表 1204. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1205. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_sends_total – “TCP/IP 发送总次数”监视元素

通过 TCP/IP 从数据库服务器向客户机应用程序发送数据的次数。

表 1206. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 – 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 – 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 – 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 – 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 – 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 – 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 – 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1207. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

temp_tablespace_top - "临时表空间顶部"监视元素

服务类或工作类中所有嵌套级别的 DML 活动临时表空间使用量的高水位标记（以 KB 计）。对于服务类，当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。 对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更改完成该活动的服务子类的 temp_tablespace_top 高水位标记。该活动所映射到但未在其中完成该活动的服务子类的高水位标记不受影响。

表 1208. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此元素来确定在收集时间间隔内，分区中服务类、工作负载或工作类达到的最高 DML 活动系统临时表空间使用量。

此元素仅由对其应用了临时表空间阈值的活动更新。如果未对活动应用临时表空间阈值，那么将返回 0。

territory_code - "数据库地域代码"

对其收集监视器数据的数据库的地域代码。此监视元素先前称为 country_code。

表 1209. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本

表 1210. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-
连接	event_connheader	-

用法 地域代码信息记录在数据库配置文件中。

对于 DRDA AS 连接，此元素将设置为 0。

threshold_action - "阈值操作"监视元素

此阈值违例记录适用的阈值操作。可能的值包括 Stop、Continue 和 Remap。

表 1211. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来确定违反阈值的活动在发生违例时被停止、被允许继续执行还是被重新映射到另一个服务子类。如果该活动被停止，那么提交活动的应用程序将接收到 SQL4712N 错误。如果该活动被重新映射到另一个服务子类，那么分区中为该活动工作的代理程序将移至该阈值的目标服务子类。

threshold_domain - "阈值域"监视元素

负责此队列的阈值的域。

可能的值包括

- 数据库
- 工作操作集
- 服务超类
- 服务子类
- 工作负载

表 1212. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

此元素可用来区分谓词相同但域不同的阈值的队列统计信息。

threshold_maxvalue - "阈值最大值"监视元素

对于非队列阈值，此元素表示被超过而导致此阈值违例的值。对于队列阈值，此监视元素表示导致队列的并行级别。导致队列阈值违例的并行级别是 **threshold_maxvalue** 和 **threshold_queuesize** 监视元素之和。

表 1213. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

对于活动阈值，此元素提供阈值违例时域值最大值的历史记录。如果阈值的最大值在违例后更改了，且旧值在 SYSCAT.THRESHOLDS 视图中不再可用，那么此元素将很有用。

threshold_name - “阈值名称”监视元素

负责此队列的阈值的唯一名称。

表 1214. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

使用此元素来唯一地标识此记录所表示的统计信息的队列阈值。

threshold_predicate - “阈值谓词”监视元素

标识已违反的阈值类型或对其收集统计信息的阈值类型。

表 1215. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-
统计信息	event_qstats	-

用法

将此监视元素与其他统计信息或阈值违例监视元素配合使用来分析阈值违例。

threshold_queuesize - “阈值队列大小”监视元素

队列阈值的队列大小。尝试超过此大小将导致阈值违例。对于非队列阈值，此值为 0。

表 1216. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来确定违反阈值时此阈值的队列中的活动或连接数。

thresholdid - “阈值标识”监视元素

标识阈值违例记录适用的阈值或为其收集队列统计信息的阈值。

表 1217. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-
统计信息	event_qstats	-

用法

将此监视元素与其他活动历史监视元素配合使用来分析阈值队列或分析违反阈值的活动。

time_completed - "完成时间"监视元素

此活动记录描述的活动完成执行的时间。此元素为本地时间戳记。

如果由于内存局限性而无法将完整活动记录写入表事件监视器，或者如果活动在执行时被捕获，那么此字段的值将为"0000-00-00-00.00.00.000000"。

元素标识

time_completed

元素类型

信息

-->

表 1218. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

time_created - "创建时间"监视元素

用户提交此活动记录描述的活动的时间。此元素为本地时间戳记。

表 1219. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

time_of_violation - "违例时间"监视元素

发此此阈值违例记录描述的阈值违例的时间。此元素为本地时间戳记。

表 1220. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

将此元素与其他阈值违例监视元素配合使用来分析阈值违例。

time_stamp - "快照时间"

收集数据库系统监视器信息的日期和时间。

表 1221. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法 如果要将结果保存在文件或数据库中以便将来进行分析，那么可使用此元素来帮助将相关数据按时间顺序排列。

time_started - "开始时间"监视元素

此活动记录描述的活动开始执行的时间。此元素为本地时间戳记。

表 1222. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

time_zone_disp - "时区偏移"

表示本地时区与格林威治标准时间（GMT）之间的偏移的秒数。

表 1223. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法 数据库系统监视器报告的所有时间为 GMT，而此偏移计算本地时间。

top - "直方图类别顶部"监视元素

直方图类别范围的包含顶端。此监视元素的值也是下一直方图类别的范围的不包含底端。

表 1224. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

将此元素与相应的 **bottom** 元素配合使用来确定直方图内的类别范围。

tot_log_used_top - "使用的最大总日志空间"

使用的最大总日志空间（以字节计）。

表 1225. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 1226. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 可使用此元素来帮助评估分配的主日志空间量。将此元素的值与分配的主日志空间量进行比较可帮助您评估配置参数设置。可借助以下公式计算主日志空间分配:

```
logprimary x logfilsiz x 4096（请参阅以下注释）
```

可将此元素与 *sec_log_used_top* 和 *sec_logs_allocated* 一起使用来显示当前对辅助日志的依赖性。

此值同时包括在主日志文件和辅助日志文件中使用的空间。

可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

total_act_time - "活动时间总计"监视元素

执行活动时的耗用时间总计。此值以毫秒计。

表 1227. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

total_act_wait_time - "活动等待时间总计"监视元素

处理活动期间，在 DB2 数据库服务器中进行等待时的耗用时间总计。此值以毫秒计。

表 1228. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	COLLECT ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

用法

通过将此元素与 **total_act_request_time** 元素配合使用，可以确定数据服务器处理此活动时耗用的时间所占的百分比。

```
(total_act_request_time - total_act_wait_time)/(total_act_request_time) =
 数据服务器主动处理活动时耗用的时间所占的百分比
```

total_app_rqst_time - “应用程序请求时间总计”监视元素

处理应用程序请求时的耗用时间总计；这是服务器上的协调代理程序执行应用程序请求时的耗用时间总计。

表 1229. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

用法

使用此监视元素来确定应用程序请求在 DB2 数据服务器中耗用的时间。此值可用于帮助确定数据服务器是否是所检测到的性能问题的根源。

例如，如果用户报告应用程序遇到问题，其返回时间长达 20 分钟，而您确定应用程序请求时间总计为 1 分钟，并且当前未通过该连接处理应用程序请求，那么性能问题可能并非由 DB2 数据服务器所致。

total_buffers_rcvd －“接收到的 FCM 缓冲区总数”

发出 GET SNAPSHOT 命令的节点从 *node_number*（请参阅 *db2nodes.cfg* 文件）标识的节点接收到的总 FCM 缓冲区数。

表 1230. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

用法 可使用此元素来量度当前节点与远程节点之间的通信量级别。如果从此节点接收到的总 FCM 缓冲区数很高，您可能想要重新分发数据库或移动表以降低节点间通信量。

total_buffers_sent －“发送的 FCM 缓冲区总数”

从发出 GET SNAPSHOT 命令的节点发送至 *node_number*（请参阅 *db2nodes.cfg* 文件）标识的节点的 FCM 缓冲区总数。

表 1231. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

用法 可使用此元素来量度当前节点与远程节点之间的通信量级别。如果发送至此节点的总 FCM 缓冲区数很高，您可能想要重新分发数据库或移动表以降低节点间通信量。

total_cons －“数据库激活以后的连接数”

指示第一次连接、激活或上一次复位（协调代理程序）后与数据库的连接的数目。

表 1232. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本

可将快照监视的计数器复位。

表 1233. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-

用法 可将此元素与 db_conn_time 和 db2start_time 监视元素配合使用以计算应用程序与数据库进行连接的频率。

如果连接频率很低，那么您可能想要在连接任何其他应用程序之前使用 ACTIVATE DATABASE 命令显式激活数据库，这是因为第一个与数据库的连接存在其他开销（例如，初始缓冲池分配）。这将导致后续连接以较高的频率进行处理。

注：复位此元素时，其值将设置为当前连接的应用程序数而不是设置为零。

total_cpu_time －“CPU 时间总计”监视元素

在 DB2 中耗用的 CPU 时间总计。此值代表用户 CPU 时间与系统 CPU 时间的总计。此值以微秒计。

表 1234. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 － 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 － 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 － 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 － 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 － 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 － 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 － 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 － 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 － 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1235. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1235. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

total_exec_time - "执行语句所耗用的时间"

在 SQL 高速缓存中执行特定语句所花的总时间（以秒和微秒计）。

表 1236. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

用法 将此元素与 num_executions 配合使用来确定语句的平均耗用时间并标识调整 SQL 时受益最多的 SQL 语句。在评估此元素的内容时，必须考虑 num_compilation。

total_move_time - "扩展数据块移动时间总计"监视元素

表空间重新平衡过程期间移动的所有扩展数据块的移动时间总计（以毫秒计）。

表 1237. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	始终收集

total_hash_joins - "散列连接总数"

已执行的散列连接的总数。

元素标识

total_hash_joins

元素类型

计数器

表 1238. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1239. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 在数据库或应用程序级别，将此值与 hash_join_overflows 和 hash_join_small_overflows 配合使用，以确定适度增大排序堆是否能够使相当百分比的散列连接受益。

total_hash_loops －“总散列循环数”

散列连接的单个分区大于可用排序堆空间的总次数。

表 1240. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1241. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 此元素的值指示散列连接的执行没有效率。它可能指示排序堆大小太小或者排序堆阈值太小。将此值与其他散列连接变量一起使用来调整排序堆大小（*sortheap*）和排序堆阈值（*sheapthres*）配置参数。

total_log_available －“可用的总日志量”

数据库中未被未落实事务使用的活动日志空间量（以字节计）。

表 1242. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法

将此元素与 total_log_used 一起使用来确定是否需要调整下列配置参数以避免用完日志空间:

- logfilsiz
- logprimary
- logsecond

如果 total_log_available 降低至 0，那么将返回 SQL0964N。您可能需要增加以上配置参数，或通过 COMMIT、ROLLBACK 或 FORCE APPLICATION 结束最旧的事务。

如果 logsecond 设置为 -1，那么此元素将包含 SQLM_LOGSPACE_INFINITE。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

total_log_used －“使用的总日志空间”

当前在数据库中使用的总活动日志空间量（以字节计）。

表 1243. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 将此元素与 total_log_available 一起使用来确定是否需要调整下列配置参数以避免用完日志空间:

- logfilsiz
- logprimary
- logsecond

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

total_olap_funcs －“OLAP 函数总数”监视元素

执行的 OLAP 函数总数

表 1244. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1245. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法

在数据库或应用程序级别，将此值与 olap_func_overflows 配合使用，以确定适度增大排序堆是否能够使相当百分比的 OLAP 函数受益。

total_rqst_mapped_in －“映入请求总数”监视元素

通过重新映射阈值或工作操作集映射到此服务子类中的请求的总数。

表 1246. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1247. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

total_rqst_mapped_out - "映出请求总数"监视元素

通过重新映射阈值或工作操作集从此服务子类中映射出的请求的总数。

表 1248. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1249. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

total_rqst_time - "请求时间总计"监视元素

处理请求时的耗用时间总计。此值以毫秒计。

表 1250. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1250. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

用法

total_sec_cons - "辅助连接数"

子代理程序与节点上的数据库建立的连接的数目。

元素标识

total_sec_cons

元素类型

计数器

表 1251. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 可将此元素与 total_cons、db_conn_time 和 db2start_time 监视元素一起使用来计算应用程序与数据库建立连接的频率。

total_section_sorts - "节排序总次数"监视元素

执行节期间执行的排序的总次数（执行节是指执行已编译的查询方案，此方案由客户机应用程序发出的 SQL 语句生成）。

表 1252. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1252. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

用法

通过将此元素与 total_section_sort_time 配合使用，可以计算执行节期间执行排序时耗用的平均时间。

在活动和程序包高速缓存级别，使用此元素来标识执行大量排序的语句。如果另外进行调整以降低排序数目，这些语句将从中受益。还可使用 EXPLAIN 语句来标识语句执行的排序数。

total_section_sort_proc_time - “节排序处理时间总计”监视元素

在执行节期间执行排序时耗用的非等待时间总计（执行节是指执行已编译的查询方案，此方案由客户机应用程序发出的 SQL 语句生成）。

表 1253. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 1253. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

用法

在系统级别，通过将此元素与 total_section_sorts 配合使用，可以计算执行节期间的平均排序处理时间（不包括等待时间），此时间可以指示就性能而言排序是否存在问题。

在活动级别，使用此元素来确定耗用大量时间进行排序的语句。如果另外进行调整以降低排序时间，这些语句将从中受益。

total_section_sort_time - “节排序时间总计”监视元素

在执行节期间执行排序时的耗用时间总计（执行节是指执行已编译的查询方案，此方案由客户机应用程序发出的 SQL 语句生成）。

表 1254. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1254. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

用法

在系统级别，通过将此元素与 total_section_sorts 配合使用，可以计算执行节期间的平均排序时间，此时间可以指示就语句性能而言排序是否存在问题。

注：total_section_sort_time 元素既包含等待时间也包含处理时间。如果 (total_section_sort_time - total_section_sort_proc_time) 较大，那么表明排序操作进行等待时耗用了大量时间。例如，如果排序频繁地溢出到磁盘，那么 total_section_sort_time 将由于 I/O 等待而增大。此时间将不会包括在 total_section_sort_proc_time 中，而只是计作主动处理排序时耗用的时间。在这种情况下，您可以考虑调整排序内存以提高性能。

在活动级别，使用此元素来确定耗用大量时间进行排序的语句。如果另外进行调整以降低排序时间，这些语句将从中受益。

total_sort_time - “排序时间总计”监视元素

已执行的所有排序的耗用时间总计。此值以毫秒计。

表 1255. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	排序
应用程序	appl	排序
应用程序	stmt	排序
动态 SQL	dynsql	排序

可将快照监视的计数器复位。

表 1256. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	语句，排序

用法

在数据库或应用程序级别，将此元素与 **total_sorts** 配合使用来计算平均排序时间，它可以指示就性能而言排序是否存在问题。

在语句级别，使用此元素来标识花费大量时间进行排序的语句。如果另外进行调整以降低排序时间，这些语句将从中受益。

此计数还包括相关操作期间创建的临时表的排序时间。它提供有关一个语句、一个应用程序或访问一个数据库的所有应用程序的信息。

使用提供耗用时间的监视元素时，应考虑:

1. 耗用时间受系统负载影响，所以运行的进程越多，此耗用时间值越高。
2. 要在数据库级别计算此监视元素，数据库系统监视器会将应用程序级别时间求和。这会导致对数据库级别的耗用时间双倍计数，原因是多个应用程序进程可能同时运行。

   要在数据库级别提供有意义的数据，应将数据标准化以将其降至较低级别。例如:

   ```
   total_sort_time / total_sorts
   ```

   提供有关每个排序的平均耗用时间的信息。

total_sorts - "排序总数"监视元素

已执行的排序总数。

表 1257. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1257. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1258. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1259. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	语句，排序

用法

在数据库或应用程序级别，将此值与 **sort_overflows** 配合使用来计算需要更多堆空间的排序的百分比。还可将其与 **total_sort_time** 配合使用来计算平均排序时间。

如果排序溢出数相对排序总数较小，那么除非此缓冲区大小实际上增加了，否则提高排序堆大小对性能没什么影响。

在语句级别，使用此元素来标识执行大量排序的语句。如果另外进行调整以降低排序数目，这些语句将从中受益。还可使用 SQL EXPLAIN 语句来标识语句执行的排序数。

total_sys_cpu_time - “语句的系统 CPU 时间总计”监视元素

SQL 语句的总系统 CPU 时间。

表 1260. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

用法 将此元素与执行语句所耗用的时间和语句的总用户 CPU 配合使用以找出执行成本最高的语句。

total_sorts - "排序总数"监视元素

已执行的排序总数。

表 1261. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1262. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1263. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
数据库	event_db	-
连接	event_conn	-
语句	event_stmt	-
活动	event_activity	语句，排序

用法

在数据库或应用程序级别，将此值与 **sort_overflows** 配合使用来计算需要更多堆空间的排序的百分比。还可将其与 **total_sort_time** 配合使用来计算平均排序时间。

如果排序溢出数相对排序总数较小，那么除非此缓冲区大小实际上增加了，否则提高排序堆大小对性能没什么影响。

在语句级别，使用此元素来标识执行大量排序的语句。如果另外进行调整以降低排序数目，这些语句将从中受益。还可使用 SQL EXPLAIN 语句来标识语句执行的排序数。

total_usr_cpu_time - “语句的用户 CPU 时间总计”监视元素

SQL 语句的总用户 CPU 时间。

表 1264. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器复位。

用法

将此元素与执行语句所耗用的时间配合使用以找出运行时间最长的语句。

total_wait_time - “等待时间总计”监视元素

在 DB2 数据库服务器中进行等待时的耗用时间总计。此值以毫秒计。

表 1265. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1265. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

用法

要了解数据库服务器主动处理请求时耗用的时间所占的百分比，请使用以下比率:

```
(total_rqst_time - total_wait_time) / total_rqst_time
```

client_idle_wait_time 监视元素的值未包括在 **total_wait_time** 监视元素的值中。**total_wait_time** 元素仅代表数据库服务器处理请求期间进行等待时耗用的时间。

tpmon_acc_str - "TP 监视器客户机记帐字符串"监视元素

如果在此连接中发出了 sqleseti API，那么此项是为了用于记录和诊断而传递至目标数据库的数据。 此连接、工作单元或活动的 CLIENT_ACCTNG 专用寄存器的当前值。

此监视元素与 **client_acctng** 监视元素同义。**client_acctng** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_acc_str** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1266. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1267. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-

表 1267. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

用法

此元素用于问题确定和记帐目的。

tpmon_client_app – "TP 监视器客户机应用程序名称"监视元素

如果在此连接中发出 sqleseti API，那么此项标识执行事务时出现的服务器事务程序问题。 此连接、工作单元或活动的 CLIENT_APPLNAME 专用寄存器的当前值。

此监视元素与 **client_applname** 监视元素同义。**client_applname** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_client_app** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1268. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1269. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-
事务	event_xact	-

用法

此元素用于问题确定和记帐目的。

tpmon_client_userid – "TP 监视器客户机用户标识"监视元素

如果使用 sqleseti API，那么此项是由事务管理器生成的并且提供给服务器的客户机用户标识。 此连接、工作单元或活动的 CLIENT_USERID 专用寄存器的当前值。

此监视元素与 **client_userid** 监视元素同义。**client_userid** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_client_userid** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1270. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1271. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-
事务	event_xact	-

用法

通过在应用程序服务器或事务处理监视器环境中使用此元素，可以标识为其执行事务的最终用户。

tpmon_client_wkstn －“TP 监视器客户机工作站名称”监视元素

如果在此连接中发出了 sqleseti API，那么此项标识客户机的系统或工作站（如 CICS EITERMID）。此连接、工作单元或活动的 CLIENT_WRKSTNNAME 专用寄存器的当前值。

此监视元素与 **client_wrkstnname** 监视元素同义。**client_wrkstnname** 监视元素用于监视写入 DB2 版本 9.7 所引入的非格式化表的表函数和事件监视器。**tpmon_client_wkstn** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1272. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1273. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-
事务	event_xact	-

用法

使用此元素并借助节点标识、终端标识或类似标识来标识用户的机器。

tq_cur_send_spills －“当前溢出的表队列缓冲区数”监视元素

当前驻留在临时表中的表队列缓冲区数。

表 1274. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 写至表队列的代理程序可能正将行发送至若干阅读器。当向其发送行的代理程序不接受行而另一代理程序需要行以进行处理时，写代理程序会将缓冲区溢出至临时表。溢出至临时表允许写程序和其他阅读器同时继续进行处理。

当读取代理程序准备接受更多行时，已溢出的行将发送至该代理程序。

如果此数字很高，并且查询失败（其 sqlcode 为 -968），同时 db2diad.log 中的消息指示 TEMP 表空间中的临时空间已用完，那么可能是表队列溢出造成的。这可能指示另一节点存在问题（如锁定）。应通过在所有分区上获取此查询的快照来进行调查。

还可能出现下列情况：因为数据的分区方式而使得许多缓冲区需要溢出以供该查询使用。在这类情况下，您需要为临时表空间添加更多磁盘。

tq_id_waiting_on - "在表队列的节点上等待"

正在等待的代理程序。

元素标识

tq_id_waiting_on

元素类型

信息

表 1275. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 它可以用于故障诊断。

tq_max_send_spills - "最大表队列缓冲区溢出数"

溢出至临时表的最大表队列缓冲区数。

元素标识

tq_max_send_spills

元素类型

水位标记

表 1276. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1277. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 指示已写至临时表的最大表队列缓冲区数。

tq_node_waited_for - "在表队列上等待节点"

如果子节状态 ss_status 为*等待接收*或*等待发送*并且 tq_wait_for_any 为 FALSE，那么此项是此代理程序正在等待的节点的编号。

元素标识

tq_node_waited_for

元素类型

信息

表 1278. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 它可以用于故障诊断。您可能想要在子节正在等待的节点上获取应用程序快照。例如，应用程序在该节点上可能处于锁定等待状态。

tq_rows_read – “从表队列读取的行数”

从表队列读取的总行数。

元素标识

tq_rows_read

元素类型

计数器

表 1279. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1280. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 如果监视未指示此数字正在增加，那么表示处理未在进行。

如果此数字在节点间有很大差别，那么表示一些节点可能使用过度而另一些节点使用得不多。

如果此数字很大，那么表示节点间传递的数据很多，建议进行优化以改进访问方案。

tq_rows_written – “写至表队列的行数”

写至表队列的总行数。

元素标识

tq_rows_written

元素类型

计数器

表 1281. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1282. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 如果监视未指示此数字正在增加，那么表示处理未在进行。

如果此数字在节点间有很大差别，那么表示一些节点可能使用过度而另一些节点使用得不多。

如果此数字很大，那么表示节点间传递的数据很多，建议进行优化以改进访问方案。

tq_tot_send_spills - "溢出表队列缓冲区总数"监视元素

溢出至临时表的表队列缓冲区的总数。

表 1283. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1284. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1285. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
语句	event_subsection	-

用法

指示已写入临时表的表队列缓冲区的总数。有关更多信息，请参阅 **tq_cur_send_spills** 监视元素。

tq_wait_for_any －“在表队列上等待发送任何节点”

此标志用于指示子节已阻塞，原因是它正在等待从任何节点接收行。

元素标识

tq_wait_for_any

元素类型

信息

表 1286. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 如果 ss_status 指示*在表队列上等待接收数据*并且此标志为 TRUE，那么表示子节正等待从任何节点接收行。这通常指示 SQL 语句未处理至可将数据传递至等待代理程序的点。例如，写代理程序可能正在执行排序并且在排序完成之前不会写入行。从 db2expln 输出来确定与表队列相关联的子节号，代理程序正在等待接收来自该表队列的行。然后可通过在执行子节的每个节点上获取快照来检查该子节的状态。

ts_name －“正在前滚的表空间”监视元素

目前已前滚的表空间的名称。

元素标识

ts_name

元素类型

信息

表 1287. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法 如果正在进行前滚，那么此元素标识前滚涉及的表空间。

uid_sql_stmts －“执行的 Update/Insert/Delete SQL 语句数”

执行的 SQL UPDATE、INSERT 和 DELETE 语句数。

表 1288. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1289. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。

还可使用以下公式来确定 UPDATE、INSERT 和 DELETE 语句数与语句总数的比率:

```
    uid_sql_stmts
/ (static_sql_stmts + dynamic_sql_stmts )
```

此信息对于分析应用程序活动和吞吐量非常有用。

unread_prefetch_pages - "未读取的预取页数"监视元素

指示预取读入但从未使用的页数。

表 1290. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1291. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器复位。

表 1292. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表空间	event_tablespace	-
连接	event_conn	-

用法

如果此数目很高，那么预取程序会将不会使用到的页读入缓冲池，从而导致执行不必要的 I/O。

uow_comp_status - "工作单元完成状态"

工作单元的状态以及停止方式。

元素标识

uow_comp_status

元素类型

信息

表 1293. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元
DCS 应用程序	dcs_appl	基本

表 1294. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

用法 可使用此元素来确定工作单元是否因为死锁或异常终止而结束。它可能已经:

- 因为落实语句而落实
- 因为回滚语句而回滚
- 因为死锁而回滚
- 因为异常终止而回滚
- 在正常应用程序终止时落实。
- 因为对正在运行的工作单元执行 FLUSH EVENT MONITOR 命令而处于未知状态。

注: API 用户应参考包含数据库系统监视器常量的定义的头文件（*sqlmon.h*）。

uow_elapsed_time - "最新工作单元耗用时间"

最新完成的工作单元耗用的执行时间。

元素标识

uow_elapsed_time

元素类型

时间

表 1295. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

用法 将此元素用作完成工作单元所花时间的指示符。

uow_id - "工作单元标识"监视元素

工作单元标识。工作单元标识在应用程序句柄内是唯一的。

表 1296. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	始终收集
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	始终收集

表 1297. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-
活动	event_activity	-
活动	event_activitystmt	-
活动	event_activityvals	-
阈值违例	event_thresholdviolations	-

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

还可以将此元素与 **activity_id** 和 **appl_id** 监视元素配合使用来唯一地标识某项活动。

uow_lock_wait_time - "工作单元等待锁定的总时间"监视元素

此工作单元等待锁定时的耗用时间总计。此值以毫秒计。

元素标识

uow_lock_wait_time

元素类型

计数器

表 1298. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元

用法 此元素可帮助您确定资源争用问题的严重性。

uow_log_space_used - "使用的工作单元日志空间"

被监视应用程序的当前工作单元中使用的日志空间量（以字节计）。

元素标识

uow_log_space_used

元素类型

标尺

表 1299. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元

表 1300. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

用法 可使用此元素来了解工作单元级别的记录要求。

uow_start_time －“工作单元开始时间戳记”

工作单元第一次需要数据库资源的日期和时间。

表 1301. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

表 1302. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-

用法

此资源要求出现在该工作单元的第一个 SQL 语句执行中:

- 对于第一个工作单元，这是 **conn_complete_time** 之后的第一个数据库请求（SQL 语句执行）的时间。
- 对于后续工作单元，这是上一个 COMMIT 或 ROLLBACK 之后第一个数据库请求（SQL 语句执行）的时间。

注: *SQL Reference* 将工作单元的边界定义为 COMMIT 或 ROLLBACK 点。

数据库系统监视器排除 COMMIT/ROLLBACK 与工作单元定义中的下一个 SQL 语句之间所花的时间。此量度方法反映数据库管理器在处理数据库请求时所花的时间，并且将此时间与该工作单元的第一个 SQL 语句之间的应用程序逻辑所花的时间隔开。工作单元耗用时间包括在工作单元内的 SQL 语句之间运行应用程序逻辑所花的时间。

可将此元素与 **uow_stop_time** 监视元素配合使用来计算工作单元的耗用时间总计，并与 **prev_uow_stop_time** 监视元素配合使用来计算在工作单元之间的应用程序上耗用的时间。

可以使用 **uow_stop_time** 和 **prev_uow_stop_time** 监视元素来计算工作单元的 *SQL Reference* 定义的耗用时间。

uow_status - "工作单元状态"

工作单元的状态。

元素标识

uow_status

元素类型

信息

表 1303. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	-

用法 可使用此元素来确定工作单元的状态。API 用户应参考包含数据库系统监视器常量定义的 sqlmon.h 头文件。

uow_stop_time - "工作单元停止时间戳记"监视元素

最新工作单元的完成日期和时间，该工作单元将在落实或回滚数据库更改时完成。

表 1304. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

表 1305. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-

用法

通过将此元素与 **prev_uow_stop_time** 监视元素配合使用，可以计算 COMMIT/ROLLBACK 点之间的耗用时间总计，通过将此元素与 **uow_start_time** 监视元素配合使用，可以计算最后一个工作单元的耗用时间。

时间戳记内容的设置将为如下所示:

- 如果应用程序已完成一个工作单元并且尚未启动新的工作单元（由 **uow_start_time** 监视元素定义），那么此元素将报告有效的非零时间戳记。
- 如果应用程序当前在执行工作单元，那么此元素将报告零。
- 当应用程序第一次连接至数据库时，此元素将设置为 **conn_complete_time** 监视元素的值。

当新的工作单元启动时，此元素的内容将移至 **prev_uow_stop_time** 监视元素。

uow_total_time_top – “UOW 时间总计顶部”监视元素

保留以备将来使用。

表 1306. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-
统计信息	event_scstats	-

用法

保留以备将来使用。

update_sql_stmts – “更新数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次复位以后，联合服务器代表任何应用程序对此数据源发出 UPDATE 语句的总次数。

表 1307. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器复位。

用法 使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
        (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
        (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
        DELETE 语句数)
```

update_time – “更新响应时间”

此元素包含此数据源响应 UPDATE 所花的总时间（以毫秒计），这些 UPDATE 来自联合服务器启动后或数据库监视计数器上一次复位后在此联合服务器实例上运行的所有应用程序或单个应用程序。

表 1308. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器复位。

响应时间是以联合服务器将 UPDATE 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 UPDATE 的时间之差量度的。

用法 使用此元素来确定等待处理对此数据源执行 UPDATE 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

user_cpu_time - "用户 CPU 时间"

数据库管理器代理进程、工作单元或语句使用的总*用户* CPU 时间（以秒和微秒计）。

当监视开关或时间戳记开关未打开时，将不收集此元素，并改为写入 -1。

表 1309. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
事务	event_xact	-
语句	event_stmt	-
活动	event_activity	-

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

utility_dbname - "实用程序操作的数据库"

实用程序操作的数据库。

表 1310. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_description - "实用程序描述"

对实用程序执行的任务的简短描述。例如，重新平衡调用可能包含"Tablespace ID: 2"，表示此重新平衡程序正在处理标识为 2 的表空间。此字段的格式取决于实用程序的类，并且在发行版之间可能更改。

表 1311. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_id - "实用程序标识"

对应于实用程序调用的唯一标识。

表 1312. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	始终收集

表 1313. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_invoker_type - "实用程序调用者类型"

此元素描述实用程序是如何被调用的。

表 1314. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

用法 使用此元素来确定实用程序是如何被调用的。例如，可以使用此元素来确定实用程序是 DB2 自动调用的还是用户调用的。此元素的值（列示如下）是在 sqlmon.h 中定义的。

API 常量	实用程序
SQLM_UTILITY_INVOKER_USER	实用程序是用户调用的
SQLM_UTILITY_INVOKER_AUTO	实用程序是 DB2 自动调用的

utility_priority - "实用程序优先级"

实用程序优先级指定调速实用程序相对其调速层而言的相对重要性。优先级 0 指示实用程序以非调速方式执行。非零优先级的范围必须在 1 到 100 之间，100 表示最高优先级，1 表示最低优先级。

表 1315. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_start_time - "实用程序启动时间"

初次调用当前实用程序的日期和时间。

表 1316. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_state - "实用程序状态"

此元素描述实用程序的状态。

元素标识

utility_state

元素类型
信息

表 1317. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

用法 使用此元素来确定活动实用程序的状态。此字段的值（列示如下）是在 sqlmon.h 中定义的。

API 常量	描述
SQLM_UTILITY_STATE_EXECUTE	实用程序正在执行
SQLM_UTILITY_STATE_WAIT	实用程序正在等待事件发生，然后才会继续执行
SQLM_UTILITY_STATE_ERROR	实用程序遇到了错误

valid －“节有效性指示器”监视元素

指示动态 SQL 语句节是否有效。对于静态 SQL 语句，此监视元素的值始终为 Y。

表 1318. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 － 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

用法

此监视元素的有效值是 Y 和 N。有效节在下次被使用时将由系统以隐式方式准备。

utility_type －“实用程序类型”

实用程序的类。

表 1319. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

用法

此元素的值可以是在 sqlmon.h 中定义的名称以“SQLM_UTILITY_”开头的任何常量。

vectored_ios －“向量 I/O 请求数”监视元素

向量 I/O 请求的数目。更具体地说，就是 DB2 在缓冲池的页区域中执行顺序页预取的次数。

表 1320. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 1321. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

用法

使用此元素来确定执行向量 I/O 的频率。仅在顺序预取期间才监视向量 I/O 请求的数目。

version - "监视器数据版本"

产生事件监视器数据流的数据库管理器版本。

表 1322. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法

事件监视器使用的数据结构在数据库管理器发行版之间可能更改。因此，监视器应用程序应检查数据流版本以确定它们能否处理要接收的数据。

对于此发行版，此元素设置为 API 常量 SQLM_DBMON_VERSION9_5。

wlm_queue_assignments_total - "工作负载管理器队列分配总次数"监视元素

WLM 阈值对活动进行排队的次数。

表 1323. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1323. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

wlm_queue_time_total - “工作负载管理器队列时间总计”监视元素

等待 WLM 排队阈值时耗用的时间。此值以毫秒计。

表 1324. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1324. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

wlo_completed_total - "完成的工作负载项总数"监视元素

自最后一次复位后完成的工作负载项数。

表 1325. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-

用法

使用此元素来确定向该系统添加工作的给定工作负载项个数。

work_action_set_id - "工作操作集标识"监视元素

此统计信息记录适用的工作操作集的标识。

表 1326. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-
统计信息	event_wcstats	-

用法

将此元素与其他活动历史元素配合使用来分析活动的行为，或者与其他统计信息元素配合使用来分析工作类。

work_action_set_name - "工作操作集名称"监视元素

与作为此事件的一部分显示的统计信息相关联的工作操作集的名称。

表 1327. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-
统计信息	event_wcstats	-

用法

将此元素与 **work_class_name** 元素配合使用，以唯一地标识此记录中显示了其统计信息的工作类或唯一地标识属于此记录中显示了其统计信息的阈值队列的域的工作类。

work_class_id - "工作类标识"监视元素

此统计信息记录适用的工作类的标识。

表 1328. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wcstats	-
统计信息	event_histogrambin	-

用法

将此元素与其他统计信息元素配合使用来分析工作类。

work_class_name - "工作类名"监视元素

与作为此事件的一部分显示的统计信息相关联的工作类的名称。

表 1329. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-
统计信息	event_wcstats	-

用法

将此元素与 **work_action_set_name** 元素配合使用，以唯一地标识此记录中显示了其统计信息的工作类或唯一地标识属于此记录中显示了其统计信息的阈值队列的域的工作类。

workload_id - "工作负载标识"监视元素

一个用来唯一地标识工作负载的整数。

表 1330. 表函数监视信息

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	始终收集
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	始终收集

表 1331. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

表 1332. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-

表 1332. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
统计信息	event_wlstats	-
统计信息	event_histogrambin	-
活动	event_activity	-

用法

使用此标识来唯一地指定此活动、应用程序、直方图条形或工作负载统计信息记录所属的工作负载。

workload_name - “工作负载名称”监视元素

工作负载的名称。

表 1333. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	始终收集
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	始终收集
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	始终收集

表 1334. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
工作单元	-	-
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-
统计信息	event_wlstats	-

用法

在统计信息事件监视器和工作负载表函数中，工作负载名称用于标识正在为其收集并报告统计信息和度量值的工作负载。在工作单元事件监视器和工作单元表函数中，工作负载名称用于标识与工作单元相关联的工作负载。

请使用工作负载名称来标识适用于您感兴趣的特定工作负载的工作单元或信息集。

workload_occurrence_id - "工作负载项标识"监视元素

此活动所属的工作负载项的标识。

表 1335. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	始终收集

表 1336. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
活动	event_activity	-

用法

使用它来标识已提交活动的工作负载项。

workload_occurrence_state - "工作负载实例状态"监视元素

工作负载实例的状态。

表 1337. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	始终收集
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	始终收集

用法

可能的值包括:

DECOUPLED
未对此工作负载实例指定协调代理程序（集中器情况）。

DISCONNECTPEND
此工作负载实例正在与数据库断开连接。

FORCED
此工作负载实例已被强制终止并除去。

INTERRUPTED
此工作负载实例已被中断。

QUEUED
Query Patroller 或工作负载管理排队阈值已对此工作负载实例协调代理程序进行排队。在数据库分区功能部件（DPF）环境中，此状态可能表明协调代理程序已向目录分区发出 RPC 请求以获取阈值凭单并且尚未接收到响应。

TRANSIENT
尚未将此工作负载实例映射到服务超类。

UOWEXEC
此工作负载实例正在处理请求。

UOWWAIT
此工作负载实例正在等待来自客户机的请求。

x_lock_escals - "互斥锁定升级数"

锁定从若干行锁定升级至一个互斥表锁定的次数，或者针对行的互斥锁定导致表锁定成为互斥锁定的次数。

元素标识
x_lock_escals

元素类型
计数器

表 1338. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器复位。

表 1339. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-
事务	event_xact	-

用法 其他应用程序不能访问互斥锁定挂起的数据；因此需要跟踪互斥锁定，原因是它们会影响数据的并行性。

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量时，锁定将会升级。可用锁定列表空间量由 *locklist* 和 *maxlocks* 配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，它将使用锁定列表中为其他应用程序分配的空间。当整个锁定列表已满时，将发生错误。

有关导致出现过量互斥锁定升级的原因和解决办法，请参阅 *lock_escals*。

当共享锁定已足够时，应用程序可能会使用互斥锁定。尽管共享锁定可能不会降低锁定升级总数，但共享锁定升级可能比互斥锁定升级更好一些。

xda_object_pages - "XDA 对象页数"

XML 存储器对象（XDA）数据消耗的磁盘页数。

元素标识

xda_object_pages

元素类型

信息

表 1340. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 1341. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-

用法 此元素提供了一种机制，可用来查看特定表中的 XML 存储器对象（XDA）数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内 XML 存储器对象数据的增长率。

xid - "事务标识"

事务管理器在两阶段落实事务中生成的唯一事务标识（在所有数据库上）。

表 1342. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	工作单元

用法 此标识可用来将事务管理器生成的事务与对多个数据库执行的事务相关联。它还可用来帮助诊断事务管理器问题，方法是将涉及两阶段落实协议的数据库事务与事务管理器生成的事务关联在一起。

xquery_stmts - "尝试的 XQuery 语句数"

已为应用程序或数据库执行的 XQuery 语句。

元素标识

xquery_stmts

元素类型

计数器

表 1343. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 1343. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

可将快照监视的计数器复位。

表 1344. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
连接	event_conn	-

用法 可使用此元素来测量本地 XQuery 语言请求的活动。这不包括嵌入式 XQuery 语言请求，例如，xmlquery、xmltable 或 xmlexist。

第 3 部分 附录

附录 A. DB2 技术信息概述

可以通过下列工具和方法获取 DB2 技术信息:

- DB2 信息中心
  - 主题（任务、概念和参考主题）
  - DB2 工具的帮助
  - 样本程序
  - 教程
- DB2 书籍
  - PDF 文件（可下载）
  - PDF 文件（在 DB2 PDF DVD 中）
  - 印刷版书籍
- 命令行帮助
  - 命令帮助
  - 消息帮助

注: DB2 信息中心主题的更新频率比 PDF 书籍或硬拷贝书籍的更新频率高。要获取最新信息，请安装可用的文档更新，或者参阅 ibm.com 上的 DB2 信息中心。

可以在线访问 ibm.com 上的其他 DB2 技术信息，如技术说明、白皮书和 IBM Redbooks® 出版物。访问位于以下网址的 DB2 信息管理软件库站点: http://www.ibm.com/software/data/sw-library/。

文档反馈

我们非常重视您对 DB2 文档的反馈。如果您想就如何改善 DB2 文档提出建议，请将电子邮件发送至 db2docs@ca.ibm.com。 DB2 文档小组会阅读您的所有反馈，但不能直接答复您。请尽可能提供具体的示例，这样我们才能更好地了解您所关心的问题。如果您要提供有关具体主题或帮助文件的反馈，请加上标题和 URL。

请不要用以上电子邮件地址与 DB2 客户支持机构联系。如果您遇到文档不能解决的 DB2 技术问题，请与您当地的 IBM 服务中心联系以获得帮助。

硬拷贝或 PDF 格式的 DB2 技术库

下列各表描述 IBM 出版物中心（网址为 www.ibm.com/shop/publications/order）提供的 DB2 资料库。可从 www.ibm.com/support/docview.wss?rs=71&uid=swg2700947 下载 PDF 格式的 DB2 版本 9.7 手册的英文版本和翻译版本。

尽管这些表标识书籍有印刷版，但可能未在您所在国家或地区提供。

每次更新手册时，表单号都会递增。确保您正在阅读下面列示的手册的最新版本。

注: *DB2 信息中心*的更新频率比 PDF 或硬拷贝书籍的更新频率高。

表 1345. DB2 技术信息

书名	书号	是否提供印刷版	最近一次更新时间
Administrative API Reference	SC27-2435-00	是	2009 年 8 月
Administrative Routines and Views	SC27-2436-00	否	2009 年 8 月
Call Level Interface Guide and Reference, Volume 1	SC27-2437-00	是	2009 年 8 月
Call Level Interface Guide and Reference, Volume 2	SC27-2438-00	是	2009 年 8 月
Command Reference	SC27-2439-00	是	2009 年 8 月
《数据移动指南和参考》	S151-1186-00	是	2009 年 8 月
《数据恢复及高可用性指南与参考》	S151-1187-00	是	2009 年 8 月
《数据库管理概念和配置参考》	S151-1163-00	是	2009 年 8 月
《数据库监视指南和参考》	S151-1165-00	是	2009 年 8 月
《数据库安全性指南》	S151-1188-00	是	2009 年 8 月
DB2 Text Search Guide	SC27-2459-00	是	2009 年 8 月
《开发 ADO.NET 和 OLE DB 应用程序》	S151-1167-00	是	2009 年 8 月
《开发嵌入式 SQL 应用程序》	S151-1168-00	是	2009 年 8 月
Developing Java Applications	SC27-2446-00	是	2009 年 8 月
Developing Perl, PHP, Python, and Ruby on Rails Applications	SC27-2447-00	否	2009 年 8 月
Developing User-defined Routines (SQL and External)	S151-1169-00	是	2009 年 8 月
Getting Started with Database Application Development	G151-1170-00	是	2009 年 8 月
《Linux 和 Windows 上的 DB2 安装和管理入门》	G151-1172-00	是	2009 年 8 月
《全球化指南》	S151-1189-00	是	2009 年 8 月
《安装 DB2 服务器》	G151-1174-00	是	2009 年 8 月
《安装 IBM 数据服务器客户端》	G151-1175-00	否	2009 年 8 月
《消息参考第 1 卷》	S151-1182-00	否	2009 年 8 月

表 1345. DB2 技术信息 (续)

书名	书号	是否提供印刷版	最近一次更新时间
《消息参考第 2 卷》	S151-1183-00	否	2009 年 8 月
《Net Search Extender 管理和用户指南》	S151-1185-00	否	2009 年 8 月
《分区和集群指南》	S151-1190-00	是	2009 年 8 月
《pureXML 指南》	S151-1180-00	是	2009 年 8 月
Query Patroller Administration and User's Guide	SC27-2467-00	否	2009 年 8 月
《Spatial Extender 和地理数据管理功能部件用户指南和参考》	SC27-2468-00	否	2009 年 8 月
《SQL 过程语言: 应用程序启用和支持》	S151-1171-00	是	2009 年 8 月
SQL Reference, Volume 1	SC27-2456-00	是	2009 年 8 月
SQL Reference, Volume 2	SC27-2457-00	是	2009 年 8 月
《故障诊断和调整数据库性能》	S151-1164-00	是	2009 年 8 月
《升级到 DB2 版本 9.7》	S151-1173-00	是	2009 年 8 月
《Visual Explain 教程》	S151-1184-00	否	2009 年 8 月
《DB2 版本 9.7 新增内容》	S151-1179-00	是	2009 年 8 月
Workload Manager Guide and Reference	SC27-2464-00	是	2009 年 8 月
《XQuery 参考》	S151-1181-00	否	2009 年 8 月

表 1346. 特定于 DB2 Connect 的技术信息

书名	书号	是否提供印刷版	最近一次更新时间
《安装和配置 DB2 Connect 个人版》	S151-1177-00	是	2009 年 8 月
《安装和配置 DB2 Connect 服务器》	S151-1178-00	是	2009 年 8 月
《DB2 Connect 用户指南》	S151-1176-00	是	2009 年 8 月

表 1347. Information Integration 技术信息

书名	书号	是否提供印刷版	最近一次更新时间
Information Integration: Administration Guide for Federated Systems	SC19-1020-02	是	2009 年 8 月
Information Integration: ASNCLP Program Reference for Replication and Event Publishing	SC19-1018-04	是	2009 年 8 月

表 1347. Information Integration 技术信息 (续)

书名	书号	是否提供印刷版	最近一次更新时间
Information Integration: Configuration Guide for Federated Data Sources	SC19-1034-02	否	2009 年 8 月
Information Integration: SQL Replication Guide and Reference	SC19-1030-02	是	2009 年 8 月
Information Integration: Introduction to Replication and Event Publishing	GC19-1028-02	是	2009 年 8 月

订购印刷版的 DB2 书籍

如果您需要印刷版的 DB2 书籍，可以在许多（但不是所有）国家或地区在线购买。无论何时都可以从当地的 IBM 代表处订购印刷版的 DB2 书籍。请注意，*DB2 PDF 文档* DVD 上的某些软拷贝书籍没有印刷版。例如，*DB2 消息参考*的任何一卷都没有提供印刷版书籍。

只要支付一定费用，就可以从 IBM 获取 DB2 PDF 文档 DVD，该 DVD 包含许多 DB2 书籍的印刷版。 根据您下订单的位置，您可能能够从 IBM 出版物中心在线订购书籍。如果在线订购在您所在国家或地区不可用，您始终可以从当地的 IBM 代表处订购印刷版 DB2 书籍。注意，并非 DB2 PDF 文档 DVD 上的所有书籍都有印刷版。

注：最新最完整的 DB2 文档保留在 DB2 信息中心中，网址为：http://publib.boulder.ibm.com/infocenter/db2luw/v9r7。

要订购印刷版的 DB2 书籍：

- 要了解您是否可从所在国家或地区在线订购印刷版的 DB2 书籍，可查看 IBM 出版物中心站点，网址为：http://www.ibm.com/shop/publications/order。必须先选择国家、地区或语言才能访问出版物订购信息，然后再按照针对您所在位置的订购指示信息进行订购。
- 要从当地的 IBM 代表处订购印刷版的 DB2 书籍：
  1. 从下列其中一个 Web 站点找到当地代表处的联系信息：
     - IBM 全球联系人目录，网址为 www.ibm.com/planetwide。
     - IBM 出版物 Web 站点，网址为 http://www.ibm.com/shop/publications/order。必须先选择国家、地区或语言才能访问对应您的所在地的出版物主页。在此页面中访问“关于此站点”链接。
  2. 请在致电时说明您想订购 DB2 出版物。
  3. 请向您当地的代表提供想要订购的书籍的书名和书号。有关书名和书号的信息，请参阅第 709 页的『硬拷贝或 PDF 格式的 DB2 技术库』。

从命令行处理器显示 SQL 状态帮助

DB2 产品针对可能充当 SQL 语句结果的条件返回 SQLSTATE 值。SQLSTATE 帮助说明 SQL 状态和 SQL 状态类代码的含义。

要启动 SQL 状态帮助，请打开命令行处理器并输入:

```
? sqlstate or ? class code
```

其中，*sqlstate* 表示有效的 5 位 SQL 状态，*class code* 表示该 SQL 状态的前 2 位。例如，`? 08003` 显示 08003 SQL 状态的帮助，而 `? 08` 显示 08 类代码的帮助。

访问不同版本的 DB2 信息中心

对于 DB2 版本 9.7 主题，DB2 信息中心 URL 为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/

对于 DB2 版本 9.5 主题，DB2 信息中心 URL 为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/

对于 DB2 版本 9 主题，DB2 信息中心 URL 为 http://publib.boulder.ibm.com/infocenter/db2luw/v9/

对于 DB2 版本 8 主题，请访问版本 8 信息中心 URL: http://publib.boulder.ibm.com/infocenter/db2luw/v8/

在 DB2 信息中心中以您的首选语言显示主题

DB2 信息中心尝试以您在浏览器首选项中指定的语言显示主题。如果未提供主题的首选语言翻译版本，那么 DB2 信息中心将显示该主题的英文版。

- 要在 Internet Explorer 浏览器中以您的首选语言显示主题:
  1. 在 Internet Explorer 中，单击**工具** —> **Internet 选项** —> **语言...** 按钮。“语言首选项”窗口打开。
  2. 确保您的首选语言被指定为语言列表中的第一个条目。
     - 要将新语言添加至列表，请单击**添加...** 按钮。

       注: 添加语言并不能保证计算机具有以首选语言显示主题所需的字体。
     - 要将语言移至列表顶部，请选择该语言并单击**上移**按钮直到该语言成为语言列表中的第一个条目。
  3. 清除浏览器高速缓存然后刷新页面以便以首选语言显示 DB2 信息中心。
- 要在 Firefox 或 Mozilla 浏览器中以首选语言显示主题:
  1. 在**工具** —> **选项** —> **高级**对话框中的**语言**部分中选择按钮。“语言”面板将显示在“首选项”窗口中。
  2. 确保您的首选语言被指定为语言列表中的第一个条目。
     - 要将新语言添加至列表，请单击**添加...** 按钮以从“添加语言”窗口中选择一种语言。
     - 要将语言移至列表顶部，请选择该语言并单击**上移**按钮直到该语言成为语言列表中的第一个条目。

3. 清除浏览器高速缓存然后刷新页面以便以首选语言显示 DB2 信息中心。

在某些浏览器和操作系统组合上，可能还必须将操作系统的区域设置更改为您选择的语言环境和语言。

更新安装在您的计算机或内部网服务器上的 DB2 信息中心

本地安装的 DB2 信息中心必须定期进行更新。

开始前

必须已安装 DB2 版本 9.7 信息中心。有关详细信息，请参阅《安装 DB2 服务器》中的“使用 DB2 安装向导来安装 DB2 信息中心”主题。所有适用于安装信息中心的先决条件和限制同样适用于更新信息中心。

关于此任务

可自动或手动更新现有 DB2 信息中心:

- 自动更新 - 更新现有信息中心功能和语言。自动更新的一个优点是在更新期间，信息中心不可用的时间最短。另外，自动更新可设置为作为定期运行的其他批处理作业的一部分运行。
- 手动更新 - 应该在更新过程期间要添加功能或语言时使用。例如，如果本地信息中心最初安装的是英语和法语版，而现在还要安装德语版；那么手动更新将安装德语版，并更新现有信息中心的功能和语言。但是，手动更新要求您手动停止、更新和重新启动信息中心。在整个更新过程期间信息中心不可用。

过程

此主题详细说明了自动更新的过程。有关手动更新的指示信息，请参阅“手动更新安装在您的计算机或内部网服务器上的 DB2 信息中心”主题。

要自动更新安装在您的计算机或内部网服务器上的 DB2 信息中心:

1. 在 Linux 操作系统上,
   a. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 /opt/ibm/db2ic/V9.7 目录中。
   b. 从安装目录浏览至 doc/bin 目录。
   c. 运行 ic-update 脚本:
      ```
      ic-update
      ```
2. 在 Windows 操作系统上,
   a. 打开命令窗口。
   b. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 <Program Files>\IBM\DB2 Information Center\Version 9.7 目录中，其中 <Program Files> 表示 Program Files 目录的位置。
   c. 从安装目录浏览至 doc\bin 目录。
   d. 运行 ic-update.bat 文件:
      ```
      ic-update.bat
      ```

结果

DB2 信息中心自动重新启动。如果更新可用，那么信息中心会显示新的以及更新后的主题。如果信息中心更新不可用，那么会在日志中添加消息。日志文件位于 doc\eclipse\configuration 目录中。日志文件名称是随机生成的编号。例如，1239053440785.log。

手动更新安装在您的计算机或内部网服务器上的 DB2 信息中心

如果已经在本地安装了 DB2 信息中心，那么您可以从 IBM 获取文档更新并安装。

手动更新在本地安装的 DB2 信息中心要求您:

1. 停止计算机上的 DB2 信息中心，然后以独立方式重新启动信息中心。如果以独立方式运行信息中心，那么网络上的其他用户将无法访问信息中心，因而您可以应用更新。DB2 信息中心的工作站版本总是以独立方式运行。
2. 使用“更新”功能部件来查看可用的更新。如果有您必须安装的更新，那么请使用“更新”功能部件来获取并安装这些更新。

   注: 如果您的环境要求在一台未连接至因特网的机器上安装 DB2 信息中心更新，那么通过使用一台已连接至因特网并有已安装的 DB2 信息中心的机器将更新站点镜像至本地文件系统。如果网络中有许多用户将安装文档更新，那么可以通过在本地也为更新站点制作镜像并为更新站点创建代理来缩短每个人执行更新所需要的时间。如果提供了更新包，请使用“更新”功能部件来获取这些更新包。但是，只有在单机方式下才能使用“更新”功能部件。
3. 停止独立信息中心，然后在计算机上重新启动 DB2 信息中心。

注: 在 Windows 2008、Windows Vista 和更高版本上，稍后列示在此部分的命令必须作为管理员运行。要打开具有全面管理员特权的命令提示符或图形工具，请右键单击快捷方式，然后选择**以管理员身份运行**。

要更新安装在您的计算机或内部网服务器上的 DB2 信息中心:

1. 停止 DB2 信息中心。
   - 在 Windows 上，单击**开始 → 控制面板 → 管理工具 → 服务**。右键单击 **DB2 信息中心**服务，并选择**停止**。
   - 在 Linux 上，输入以下命令:

     ```
     /etc/init.d/db2icdv97 stop
     ```
2. 以独立方式启动信息中心。
   - 在 Windows 上:
     a. 打开命令窗口。
     b. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 <Program Files>\IBM\DB2 Information Center\Version 9.7 目录中，其中 <Program Files> 表示 Program Files 目录的位置。
     c. 从安装目录浏览至 doc\bin 目录。
     d. 运行 help_start.bat 文件:

        ```
        help_start.bat
        ```
   - 在 Linux 上:
     a. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 /opt/ibm/db2ic/V9.7 目录中。
     b. 从安装目录浏览至 doc/bin 目录。

c. 运行 help_start 脚本:

```
help_start
```

系统缺省 Web 浏览器将打开以显示独立信息中心。

3. 单击**更新**按钮 ( )。(必须在浏览器中启用 JavaScript™。) 在信息中心的右边面板上，单击**查找更新**。 将显示现有文档的更新列表。
4. 要启动安装进程，请检查您要安装的选项，然后单击**安装更新**。
5. 在安装进程完成后，请单击**完成**。
6. 要停止独立信息中心，请执行下列操作:
   - 在 Windows 上，浏览至安装目录的 doc\bin 目录并运行 help_end.bat 文件:

     ```
     help_end.bat
     ```

     注: help_end 批处理文件包含安全地停止使用 help_start 批处理文件启动的进程所需的命令。不要使用 Ctrl-C 或任何其他方法来停止 help_start.bat。
   - 在 Linux 上，浏览至安装目录的 doc/bin 目录并运行 help_end 脚本:

     ```
     help_end
     ```

     注: help_end 脚本包含安全地停止使用 help_start 脚本启动的进程所需的命令。不要使用任何其他方法来停止 help_start 脚本。
7. 重新启动 DB2 信息中心。
   - 在 Windows 上，单击**开始 → 控制面板 → 管理工具 → 服务**。右键单击 **DB2 信息中心**服务，并选择**启动**。
   - 在 Linux 上，输入以下命令:

     ```
     /etc/init.d/db2icdv97 start
     ```

更新后的 DB2 信息中心将显示新的以及更新后的主题。

DB2 教程

DB2 教程帮助您了解 DB2 产品的各个方面。这些课程提供了逐步指示信息。

开始之前

可从信息中心查看 XHTML 版的教程: http://publib.boulder.ibm.com/infocenter/db2help/ 。

某些课程使用了样本数据或代码。有关其特定任务的任何先决条件的描述，请参阅教程。

DB2 教程

要查看教程，请单击标题。

《pureXML 指南》中的“pureXML™”
设置 DB2 数据库以存储 XML 数据以及对本机 XML 数据存储执行基本操作。

《Visual Explain 教程》中的“Visual Explain”
使用 Visual Explain 来分析、优化和调整 SQL 语句以获取更好的性能。

DB2 故障诊断信息

提供了很多故障诊断和问题确定信息以帮助您使用 DB2 数据库产品。

DB2 文档

故障诊断信息可在《DB2 故障诊断指南》或 *DB2 信息中心*的“数据库基础”部分中找到。可在该处找到有关如何使用 DB2 诊断工具和实用程序来隔离和找出问题的信息、某些最常见问题的解决方案以及有关如何解决使用 DB2 数据库产品时可能遇到的问题的建议。

DB2 技术支持 Web 站点

如果您遇到了问题并且想要获取查找可能的原因和解决方案的帮助，请参阅 DB2 技术支持 Web 站点。该“技术支持”站点具有指向最新 DB2 出版物、技术说明、授权程序分析报告（APAR 或错误修订）、修订包和其他资源的链接。可搜索此知识库并查找问题的可能解决方案。

请访问 DB2 技术支持 Web 站点: http://www.ibm.com/software/data/db2/support/db2_9/。

条款和条件

如果符合以下条款和条件，那么授予您使用这些出版物的许可权。

个人使用: 只要保留所有的专有权声明，您就可以为个人、非商业使用复制这些出版物。未经 IBM 明确同意，您不可以分发、展示或制作这些出版物或其中任何部分的演绎作品。

商业使用: 只要保留所有的专有权声明，您就可以仅在企业内复制、分发和展示这些出版物。未经 IBM 明确同意，您不可以制作这些出版物的演绎作品，或者在您的企业外部复制、分发或展示这些出版物或其中的任何部分。

除非本许可权中明确授予，否则不得授予对这些出版物或其中包含的任何信息、数据、软件或其他知识产权的任何许可权、许可证或权利，无论是明示的还是暗含的。

当使用这些出版物损害了 IBM 的利益，或者根据 IBM 的规定，未正确遵守上述指导说明时，那么 IBM 保留自主决定撤销本文授予的许可权的权利。

只有您完全遵循所有适用的法律和法规，包括所有的美国出口法律和法规，您才可以下载、出口或再出口该信息。

IBM 对这些出版物的内容不作任何保证。这些出版物“按现状”提供，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的关于适销和适用于某种特定用途的保证。

附录 B. 声明

本信息是为在美国提供的产品和服务编写的。有关非 IBM 产品的信息是基于首次出版此文档时的可获信息且会随时更新。

IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节字符集（DBCS）信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
3-2-12, Roppongi, Minato-ku, Tokyo 106-8711 Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation“按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本资料中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是此 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的: （i）允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及（ii）允许对已经交换的信息进行相互使用，请与下列地址联系:

IBM Canada Limited
Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
CANADA

只要遵守适当的条款和条件，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息可能包含在日常业务操作中使用的数据和报告的示例。为了尽可能完整地说明这些示例，示例中可能会包括个人、公司、品牌和产品的名称。所有这些名称都是虚构的，与实际商业企业所用的名称和地址的任何雷同纯属巧合。

版权许可:

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。如果是为按照在编写样本程序的操作平台上的应用程序编程接口（API）进行应用程序的开发、使用、经销或分发，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。此样本程序“按现状”提供，且不附有任何种类的保证。对于使用此样本程序所引起的任何损坏，IBM 将不承担责任。

凡这些样本程序的每份拷贝或其任何部分或任何衍生产品，都必须包括如下版权声明:

©（*贵公司的名称*）（*年份*）. 此部分代码是根据 IBM 公司的样本程序衍生出来的。© Copyright IBM Corp. *（输入年份）*. All rights reserved.

商标

IBM、IBM 徽标和 ibm.com® 是 International Business Machines Corp. 在全球范围许多管辖区域内的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公司的商标。Web 站点 www.ibm.com/legal/copytrade.shtml 上的“版权和商标信息”中提供了 IBM 商标的最新列表。

下列术语是其他公司的商标或注册商标

- Linux 是 Linus Torvalds 在美国和/或其他国家或地区的注册商标。
- Java 和所有基于 Java 的商标和徽标是 Sun Microsystems,Inc. 在美国和/或其他国家或地区的商标。
- UNIX 是 The Open Group 在美国和其他国家或地区的注册商标。
- Intel、Intel 徽标、Intel Inside®、Intel Inside 徽标、Intel® Centrino®、Intel Centrino 徽标、Celeron®、Intel® Xeon®、Intel SpeedStep®、Itanium® 和 Pentium® 是 Intel 公司或其子公司在美国和其他国家或地区的商标或注册商标。
- Microsoft、Windows、Windows NT® 和 Windows 徽标是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

其他公司、产品或服务名称可能是其他公司的商标或服务标记。

索引

[B]

[C]

[D]

[F]

[G]

[H]

[J]

[K]

[L]

[M]

[N]

[P]

[Q]

[R]

[S]

[T]

[W]

[X]

[Y]

[Z]

A

B

C

D

F

G

I

J

L

M

N

O

P

R

S

T

U

V

W

X

[特别字符]

Printed in China

S151-1165-00

Spine information:

IBM DB2 9.7 Linux 版、UNIX 版和 Windows 版 数据库监视指南和参考