IBM DB2 10 for z/OS

Reduce costs with improved performance

TABLE OF CONTENTS

Section I – Introduction of DB2 10 for z/OS

Executive Summary

DB2 10 for z/OS is a tremendous step forward in database technology because of its improvements in performance, scalability, availability, security, and application integration. Its technology provides a clear competitive advantage through its continued improvements in SQL, XML and integrated business intelligence capabilities.

DB2 10 leverages the recent advances in chip technology, storage devices and memory capacities through its extensive exploitation of System z 64-bit architecture to reduce CPU requirements, improve performance, and potentially dramatically reduce the total cost of ownership. DB2 10's CPU reduction features are welcomed around the world as companies analyze every factor to improve the bottom line.

Initial testing of DB2 10 shows that its many enhancements optimize the run-time environment by reducing CPU consumption by up to 5-20%. In addition, DB2 10 handles five to ten times more concurrent users, up to 20,000, providing significant improvements in capacity, scalability and overall performance for any type of large-scale application.

Capacity, scalability and performance continue to be DB2 for z/OS's strengths. Through full exploitation of the System z 64-bit architecture, DB2's enhancements such as shorter optimized processing, leveraging solid-state disk, in-memory workfile enhancements, index insert parallelism improvements and better SQL/XML access paths, provide many reductions in CPU costs and performance improvements without requiring any application changes.

DB2 10 features also enhance continuous business processing, database availability and overall ease of use. DB2 is continuously available due to more options with on-line database changes, more concurrent utilities and easier administration processes. DB2 10 database systems are always available; even while database changes are being done, transactions can be processed. These new database change capabilities combined with more options for parallel utility execution provide a smaller concurrent windows for processing and administration tasks to streamline your overall system operations.

Security, regulatory compliance and audit capability improvements are also included in DB2 10. DB2 10's enhanced security extends the role-based model introduced in DB2 9. DB2 10 provides more granular authorities that separate data access from the administration of the application, database and system. DB2 10 provides administration flexibility for specific security role settings preventing data access exposure to unauthorized applications or administrators. This role based security model combined with the label-based row and column access control and masking or encryption of sensitive information enhances the ultimate secure database environment for your business. All of these features provide tighter controls, allow more security flexibility and provide tremendous regulatory and audit compliance capabilities.

Application integration and portability benefit tremendously from the DB2 10 SQL and XML enhancements. DB2 10 SQL improvements further embrace the DB2 family with more enhancements for porting other DBMS vendor products into the DB2 10 for z/OS environment. Additional enhancements with timestamps with time zones, Java timestamp compatibility and timestamp 12 digit pico-second precision granularity provide unique business transaction timestamps for every transaction in a table. This helps global companies understand all their businesses across all the global events of the day.

Data versioning with current and history tables is also available with DB2 10. Data versioning provides flexibility to migrate current operational data into a historical data table based on set time-periods. This enhances the performance for the application operations and maintains your historical data for audit and better integration into your regulatory compliance architecture.

Also data warehouse and business intelligence capabilities are now built directly into the DB2 10 with the new temporal table capabilities and the SQL capabilities for calculating moving sums and moving averages. This enhances your bottom line by integrating business intelligence capabilities into your front line operational applications.

DB2 10 XML improvements help application flexibility, usability and performance. Through the ability to replace, delete or insert XML document nodes, manage multiple XML schema version documents and utilize the binary pre-tokenized XML format, DB2 10 provides a number of important XML performance and management improvements. Additional enhancements within utilities, date time data types, XML parameter support within SQL functions and procedures provide application flexibility to utilize XML within any application architecture solution.

DB2 10 for z/OS delivers performance, scalability, availability, and security while improving application integration and regulatory compliance. By reducing costs and providing superior technology, IBM's DB2 10 remains the technology leader and best choice database for today and tomorrow's business systems that seek a competitive advantage.

Section II – Performance Availability

Many performance enhancements

For relational database customers, database performance is paramount. DB2 10 for z/OS has raised the bar by reducing CPU demand by up to 5 to 10% immediately out-of-the-box with no application changes. CPU demand can be further reduced up to 20% once all the DB2 10 enhancements are leveraged in New Function Mode (NFM). By pushing the performance limits, IBM and DB2 10 continue to lead the database industry with the state-of-the-art technology and the most efficient database processing available.

IBM DB2 10 for z/OS with its out-of-the-box CPU reduction for applications and many NFM enhancements delivers the best performance improvements since DB2 Version 2.1. DB2 10 emphasis on performance is pervasive throughout the features and enhancements. Availability, scalability, security, compliance, application integration, XML and SQL all contain performance improvements. All of these enhancements together provide an improved operational environment, making administration easier while reducing the total cost of ownership for the business.

Performance is the major emphasis of DB2 10. Many enhancements make direct reduction in CPU time for applications. By migrating to the new version and deploying and binding your applications within the environment, your applications can leverage many of the performance enhancements without changing the application. DB2 10 leverages the full 64-bit architecture, new access paths, optimization of other access paths and provides performance improvements right out of the box.

Rebinding your applications is now even easier. Concerns about regression are easily addressed with Plan Stability, introduced in DB2 9 and further enhanced in DB2 10. DB2 10 Plan Stability with fallback, lockdown and package level zparm settings provides administrators the ability to rebind without risking performance. Plan stability lets administrators examine application performance, choose the best performance and then lock it down within the system. Plan Stability improvements provide parallelism to be turned on for individual packages allowing the administrators more options and better control. These new DB2 10 Plan Stability features provide additional flexibility for fine tuning your application performance and reduce CPU consumption without application changes.

New Optimizer Access Path Range-List Index Scan

DB2 10 offers a new application access path called Range-list Index Scan. This access path improves SQL processing against an index when multiple WHERE clauses can all reference the same index. For example, when the SQL has multiple OR, IN or other predicates that reference the same index the optimizer now recognizes it and scans the index only once instead of multiple times for each

of the WHERE clause predicates. This immediately cuts down the number of RID list entries for the processing, improving I/O and CPU performance.

This new access path can be utilized when all the SQL WHERE clauses with multiple OR statements reference the same table, at least one of them has a matching predicate and all would reference the same index. This type of SQL WHERE clause with multiple OR statements is typical for many types of applications, especially searching, scrolling or pagination application processes.

Optimizer uses more parallelism

DB2 10 also improves several existing access paths through parallelism. These specifically designed enhancements eliminate some previous DB2 restrictions, increase the amount of work redirected to the zIIP processors, and distribute work more evenly across the parallel tasks. All of these enhancements give additional reasons to enable parallelism within your environment.

Parallelism improves your application performance and DB2 10 can now take full advantage of parallelism with several types of SQL queries such as: multi-row fetch, full outer joins, common table expressions (CTE) references, table expression materialization, a table function, a CREATE GLOBAL TEMPORARY table (CGTT), or a work file resulting from view materialization. These new DB2 10 CP parallelism enhancements are active when the SQL Explain parallel mode column contains a “C.”

The new parallelism enhancements can also be active during several other specialized SQL situations. These situations are:

- when the optimizer chooses index reverse scan for a table
- when a SQL subquery is transformed into join
- when DB2 chooses to do a multiple column hybrid join with sort composite
- when the leading table is sort output and the join between the leading table and the second table is a multiple column hybrid join

Additional DB2 10 optimization and access improvements also help many aspects of application performance. In DB2 10, index lookaside and sequential detection help improve referencing parent keys within referential integrity (RI) structures during INSERT processing. This is more efficient for checking the RI dependent data and reduces the overall CPU required for the insert activity.

List prefetch improves index access

List prefetch is utilized more within DB2 10 to access index leaf and non-leaf pages. In previous versions of DB2, when the index became disorganized and had large gaps between non-leaf pages, accessing index entries through sequential reading of non-leaf pages became degraded by huge numbers of synchronous I/Os. DB2 10 improvements use non-leaf page information to perform a list

prefetch of the leaf pages. This eliminates most of the synchronous I/Os and the I/O waits associated to the large gaps in the non-leaf pages during the sequential read. This list prefetch processing especially helps long running queries dependent on non-leaf page access and also helps all the index related utilities such as REORG INDEX, CHECK INDEX and RUNSTATS.

Optimizer no longer changes access path because of RID Pool overflows

DB2 10 also improves the handling of SQL statements that reference a large amount of data through list processing. This list processing uses a large number of record ids (RIDs) and sometimes overflows the RID pool. In previous versions of DB2, the RID pool overflow caused DB2 to change the SQL access method to a table space scan. Now when the RID pool resources are exhausted, the RID list is written to workfile resources and processing continues. This improvement helps avoid the table space scan with its associated elapsed time, locking impact and performance overhead.

Optimizer does more during Stage 1 SQL evaluation

The DB2 10 optimizer can now evaluate scalar functions and non-matching index predicates during the first (Stage 1) evaluation of the SQL access path. The optimizer can apply these previously Stage 2 scalar functions and non-matching predicates early in the optimization process to limit the number of qualifying data pages and rows. This eliminates or reduces the amount of data evaluated in Stage 2, dramatically improving query elapsed time and overall query performance.

Optimizer determines which access is more certain

In previous versions of DB2, the optimizer evaluates the SQL WHERE predicate, the table indexes available, and various statistics to determine the most efficient access path. With the enhancements in DB2 10, the optimizer analyzes additional filter factor variables evaluating the SQL range predicate, the non-uniform distribution of the data, usage of parameter markers, host variables or literals where values are unknown.

When choosing between two different indexes, these unknown filter factor variables make the cost estimate between two different access paths very close. Analyzing which of the different filter factor variables are known, DB2 determines which of the index access paths has a higher degree of certainty. DB2 may choose an index access with a slightly higher cost that will provide a more certain run-time performance. This is especially important to provide the best consistent reliable performance with the different types of programming languages and the application diversity of the parameter markers, literals and host variables available.

Dynamic Statement Cache ATTRIBUTES improvements

One of the most important DB2 10 system improvements is the ability to combine some variations of SQL within the dynamic statement cache. Using the new ATTRIBUTES clause within the PREPARE SQL statement, DB2 is now able to recognize that the SQL is the same except for the WHERE clause literal values. This helps DB2 recognize that these statements already exist within the cache and reuse the cache resources previously generated for the SQL statement, helping avoid additional DB2 catalog activity such as object verification and access path creation for another SQL statement. This also helps free more cache space for other SQL statements for reuse, improving performance and transaction response time.

Improved DDF transaction flow

Application performance and network transaction traffic is optimized when SELECT statements are coded using the FETCH 1 ROW ONLY clause. DB2 now recognizes this SELECT statement FETCH 1 ROW ONLY clause and combines the OPEN cursor, SQL FETCH and CLOSE cursor into a single request instead of three separate messages flowing across the network through the system.

The change to the FETCH 1 ROW ONLY clause also improves the performance of the JDBC and CLI APIs. After the query data is retrieved, the FETCH 1 ROW ONLY clause causes the API's default action for DB2 to close the resources. DB2 closes the resources regardless of whether a CURSOR WITH HOLD was declared and notifies the API driver to cancel any additional FETCH or CLOSE statement requests. This dramatically reduces the number and amount of transaction network messages transmitted while improving DB2 performance and minimizing locking contention.

Workfile enhancements

Several types of SQL statement functionality, such as joins, GROUP BY, and ORDER BY, utilize workfiles to get their results. DB2 10 can now evaluate simple query predicates against these workfiles, improving overall performance and reducing elapsed time.

Also in the WORKFILE database, DB2 10 supports the definition of partition-by-growth table spaces to support in-memory tables for improved performance. This provides a dedicated definition type for in-memory data and relieves the extra administration work of defining a single table space within dedicated buffer pools.

In addition, DB2 10 expands the record length size for workfiles to 65,529 bytes for handling larger record size answer sets. This is especially important for handling XML, larger record sorts and joins.

MEMBER CLUSTER option

DB2 tries to maintain the table space clustering sequence when data is inserted into the table. For data-sharing systems with robust INSERT processing, maintaining the clustering sequence can cause locking contention within the table space across the different data-sharing members. This contention causes extra CPU cycles to negotiate deadlocks and extended response time to maintain the clustering sequence.

DB2 10 partition-by-range and partition-by-growth table spaces have a new MEMBER CLUSTER parameter that allows DB2 to insert data into the first available space and disregard the clustering sequence of table space. While this relieves the contention, the clustering sequence of the table space should be monitored. Poor cluster ratios can negatively impact access paths and sometimes cause additional random I/O for critical high performance applications.

Universal range-partitioned table space

DB2 10 also continues to enhance the universal range-partitioned table space. This table space is the updated version of the classic range-partitioned table space with additional segment size specification, universal settings and capabilities. This universal range-partitioned table space is the migration target for the classic range-partitioned table space that is deprecated and will be only supported for a few more DB2 releases.

Database administrators are being encouraged to use the new universal range-partitioned table space instead of the classic range-partition definitions to leverage all the new utility capabilities, availability and performance features.

Buffer Pool enhancements

System z handles memory allocations better and DB2 10 leverages this capability through its buffer pool management enhancements. In previous versions, DB2 allocated the defined size of the buffer pool at startup for use by all the associated table and index objects. In DB2 10 the system allocates the memory for the buffer pools as the data is brought into the buffer pool. This keeps the buffer pool size down to a minimum and to only what is being used by applications.

In addition, DB2 10 reduces its latch contention and provides the ability to define larger buffer pools from current megabytes to gigabytes of memory for your critical active objects. This will definitely improve the system I/O rates and reduce contention.

The new System z10 1-megabyte page size handles larger buffer pools better. Since buffer pool memory allocations can quickly become very large; prudent definitions management and z10 1-megabyte page size will keep your overall system paging down to a minimum.

New and improved online schema changes

Some of the best new features within DB2 10 are the new online schema change features. First, the list of attributes able to be ALTERed on any DB2 table space, index or table component continues to grow. Within DB2 10, the list has been further enhanced to take care of the most common activities in an online ALTER and then reorg process. Being able to change almost any component of the common database attributes online provides administration flexibility and application availability for almost any changes to your database systems.

A new long list of table space, table, index, and column attributes may now be ALTERed in DB2 10. Some of the new attributes that can be ALTERed are data set size, table/index page size, segment size, and the ability to migrate old style table spaces definitions to the new partition-by-growth (PBG) or partition-by-range (PBR) universal table spaces.

In addition, DB2 10 online schema enhancements improve adoption of the new attributes by replacing the old operation processes of DROP/RECREATE table space/ table and REBUILD indexes process to only an ALTER and then online REORG process. When utilizing online REORGs, the new ALTER and REORG process leaves the database tables available for application activity while the new attribute changes are applied into the system. This new method of enhancing the database eliminates downtime and provides great relief for mission-critical very large database (VLDB) availability issues.

As DB2 databases continue to grow in size and transaction volume, the amount of administration time required for database changes continues to be a challenge. A number of DB2 utility improvements further minimize downtime during normal operational and database-change activities.

Now also within DB2 10 DBAs will be able to create or rebuild a non-unique index against tables with LOBs without any application impact or locking downtime. This online schema change enhancement will especially help newly installed applications where another index can be quickly defined and dramatically improve SQL access or resolve a performance issue. This enhancement alone can help improve performance instantaneously for any installed application.

The new ALTER and REORG method of applying changes introduces a new database exception state of AREOR to signify that attribute changes are pending. This new database AREOR state along with DB2 catalog tables, utility enhancements and the DROP PENDING CHANGES command provides full functionality for managing your ALTER and REORG processes.

INCLUDE non-unique columns within a unique index

One of the schema changes that your applications need right away is the ability to include more columns into unique indexes. This new DB2 10 feature allows non-

unique columns to be included in the definition of a unique index definition. Before this enhancement, multiple indexes were required for indexing, one for the unique constraint and another index for the non-unique columns. Using the new CREATE or ALTER INCLUDE clause a unique index definition can include additional non-unique columns into the definition of the index. This eliminates all the extra I/O spent maintaining the other index along with the additional storage needed for multiple index definitions with similar columns and improves performance for all access to the table.

By combining two indexes into a single index definition using the new INCLUDE non-unique columns, the new range-list index scan access path can be used because the single index criteria is now satisfied. DB2 can reference the new single index definition with INCLUDEd non-unique columns and potentially improve your application performance by referencing the single index only once and using the new more efficient access path.

New Hash space and access method

DB2 10 also introduces a completely new access type called Hash Access. A new Hash space supports this new access method. DB2 uses an internal hash algorithm with the Hash space to reference the location of the data rows. In some cases, this direct Hash Access reduces data access to a single I/O, dramatically decreasing the CPU workload and speeding up application response time. Queries that use full key equal predicates like customer number or product number lookups are good candidates for Hash Access. Additional indexes can be created to support other range, list or keyed access types.

The definition of the Hash space requires column or columns for the direct Hash Access keys. Each table with Hash access has an associated Hash space or Hash space partitions. Hash Access requires some additional storage space for dramatically reducing access CPU workload.

Another advantage of Hash Access is no need to maintain a clustering index or data sequence. This allows for efficient insert processing and avoids data sharing contention for maintaining a clustering sequence or clustering index.

There are tradeoffs for using Hash Access. Parallelism is not available and traditional clustering is not allowed for the hash data. Nevertheless, Hash Access will be beneficial for database designs where unique keys are already using equal predicates on product or customer IDs, object ids, XML document ids and other direct key retrievals.

DB2 Catalog enhancements

The DB2 catalog and directory are restructured removing special structures and links. Through this restructuring, the DB2 catalog now utilizes the new universal

partition by growth table spaces, reordered row format and one table per table space, thus expanding the number of DB2 catalog table spaces. These new universal table spaces are defined with DSSIZE 64 MAXPART 1, row level locking and some CLOB and BLOB data types to handle repeating long strings. These common table space definitions allow the catalog tables to be managed like your application database with online reorganizations and check utilities.

In addition to these enhancements, the UTSERIAL lock that caused lock contention with older versions of DB2 utility processing has been eliminated. This improvement along with a reduction in log latch contention through a new compare and swap logic, the new option of readers to avoid waiting for inserters, and improvements in system thread latching serialization help reduce many types of DB2 thread processing contention. All of these enhancements help tremendously with concurrency of DDL, BIND, utility and overall processing within your application database systems and especially processes referencing the DB2 catalog.

Another enhancement provides the ability to add a new log into the system inventory while the subsystem is active. The newly added log is immediately available without recycling DB2, which should help recovery procedures and application performance.

Section III – Scalability, Simplification, Security

Full 64-bit runtime environment exploited

DB2 10 dramatically improves scalability with more exploitation of the 64-bit System z environment. Exploiting the 64-bit environment and moving 80% to 90% of the DB2 memory that is now below the bar – working storage, EDMPOOL, and even some ECSA – above the 2GB bar eliminates the main memory constraints within the DB2 DBM1 address space for most systems.

Large increase in the number of users--up to 20,000

By addressing this memory constraint in the overall system, virtual memory monitoring is no longer necessary as 5 to 10 times more concurrent threads can be run in a single DB2 10 member. This increase in threads removes one key reason for additional DB2 data sharing members and allows some consolidation of LPARs and members previously built for handling more end users. This allows more available memory on constrained machines where multiple data sharing members were taking real storage causing additional maintenance and operational overhead.

Parallel Inserts into multiple indexes

DB2 10 also improves insert performance by using more parallelism. When INSERT SQL modifies a table with multiple indexes, DB2 10 does the prefetch of multiple indexes in parallel. By initiating parallel I/Os for the multiple indexes, the process is not waiting for the synchronous I/Os, reducing the overall insert process time. This cuts down the timeframe of possible contention within your system and improves performance of all your applications.

Plan Stability - package preservation

The Plan Stability in DB2 9 features offer an enhanced way to handle testing of a new version of a DB2 application package. Plan Stability offers a way to save a DB2 access packages for an application and then REBIND a new one. If the access package is not as efficient, Plan Stability provides the administrator the ability to switch back to the previous version of the package with a simple REBIND SWITCH to the old version of the package.

DB2 10 expands on this functionality by providing the ability to handle many versions of both dynamic SQL and static DB2 packages. Plan stability has the ability to allow management of two or three copies of old static SQL packages controlled through the DB2 PLANMGMT parameter and REBIND parameters.

Through the REBIND SWITCH parameter, administrators can pick whatever version of the package they want. This gives the application team the flexibility to evaluate performance and to get the best performing version of the package implemented. Plan Stability also provides the ability to compare access paths, lockdown an access path and provide an error or warning messages if the access path is changed, protecting your critical packages from being replaced with bad access paths. This DB2 10 enhancement removes the trepidation of rebinding an application and helps administrators protect their critical applications' performance.

RUNSTATS improvements and Auto stats

Real time statistics stored procedures

Since the optimizer access paths are dramatically improving performance, up-to-the-moment statistics are vital. DB2 10 comes with a new set of stored procedures to monitor and collect table and index statistics. These new procedures monitor the current statistics, determine whether new statistics need to be collected and then autonomically perform the collection to ensure good access path optimization.

These procedures especially help volatile environments and can dynamically help improve access path optimization by getting index filtering statistics for SQL WHERE clause predicates to make the best access path decisions. By gathering statistics for you, these DB2 10 stored procedures take the burden off the

administrators for large and especially dynamically created objects to help ensure overall application performance.

Improved and finer grain access control

Security and regulatory compliance

DB2 10 also includes security, regulatory compliance and audit capabilities improvements. DB2 10's enhances the DB2 9 role based security with additional administrative and other finer grained authorities and privileges. This authority granularity helps separate administration and data access only providing the minimum appropriate authority.

The new SECADM authorization level provides the authority to manage access to the tables while prohibiting creating, dropping or altering any access to the tables. The enhanced DBADM authority provides an option to have administration capabilities without data access. These authority profiles provide better separation of duties while limiting blanket authority over all aspects of a table and its data.

In addition, DB2 10 embraces audit and regulatory compliance through a new audit policy that provides a set of criteria for auditing for the possible abuse and overlapping of authorities within a system. This helps management, administrators and the business community understand, configure and audit security policies and data access quickly for any role or user. Many audit policies can be developed to quickly verify audit and document the security compliance across your environment's critical data resources and their application users.

Support for row and column access control

DB2 10 also enhances security through its row and column access control. This access control lets administrators enable security on a particular column or particular row in the database. This security limits the data seen by any end user by not allowing that column or row to be returned in the answer to the user's query unless they have the proper authority based on their security level. This capability provides very fine grained security to be defined against any data. The role-based security model combined with the label-based row and column access control, and masking or encryption of sensitive information enhances the ultimate secure database environment for your business. All of these features provide tighter controls, allow more security flexibility, and provide tremendous regulatory and audit compliance capabilities.

Section IV - Application Enablement

pureXML enhancements

Within DB2 10, XML can be used almost anywhere within SQL variables, scalar functions, SQL table functions and SQL procedures. DB2 10 pureXML

incorporates many new enhancements that improve overall XML performance, provides easier XML schema management and embraces DB2 family compatibility.

These enhancements start with XML schema validation that is now a built into DB2 10. The XML schema no longer needs to be specified because DB2 handles XML schema validation more easily through a built-in function that validates the XML schemas. DB2 uses the timestamp to match up the XML document to the correct schema version. This allows multiple schema versions to coexist and validate new or older XML documents against their appropriate XML schema versions.

Additional functionality enhancements provide the capability to manipulate any part of an XML document. By using SQL statements with XML expressions, any single or multiple XML document nodes can be inserted, updated, deleted or can have their data values updated. This provides tremendous XML document capabilities, overall performance and flexibility for any application process.

A new additional XML type modifier is also available with DB2 10. The XML type modifier enforces and validates the XML document column data against the schema definition information. This new XML type modifier is used when adding or removing XML schemas and can be ALTERed onto older XML schemas so their XML column types can be validated. This helps ensure that the XML schema documents stored elements have only the desired XML content.

Support for XML date and time

DB2 10 also has expanded date and timestamp options with timezone XML data types. These data types are supported within XML indexes and the timestamp is expanded to handle more precision for finer data management. DB2 10 also comes with new XML time and date arithmetic comparison functions to further support application processing.

Binary XML Support

DB2 10 improves XML with new support for binary XML objects. Binary support is important because binary format is better for server and application interaction. Binary support uses pre-tokenized format and length definitions which improve overall performance of binary XML objects and provide additional ease of use for application definitions.

Binary XML also has additional flexibility features such as String IDs for text that represents some or all occurrences of the same text with an integer identifier. This can help limit the size of XML. Working with the same text improves application performance.

Faster streaming of XML and LOBs

DB2 10 provides a new LOB_INLINE_LENGTH installation parameter that sets the default number of bytes for storing inline LOBs. Having a minimized LOB length or

a predefined standard length provides better streaming capabilities and the ability to minimize and optimize the usage of the inline LOB space.

The minimizing of the LOB size or elimination of LOB or XML materialization reduces the memory consumption and improves CPU for LOB operations. This is especially beneficial during LOB LOAD utilities that use a file reference variable for the LOB data, inserting LOB or XML data from remote DRDA server applications and when inserting a single LOB or XML value into a row. Using the new LOB_INLINE_LENGTH options in all of these cases minimizes the number of bytes required and streamlines XML and LOB operations.

XML define (NO) for LOB and XML TS

The administrators now have the option of delaying the definition of the LOB or XML data sets and their indexes. This helps save storage space by defining them into the DB2 Catalog and letting application SELECT and FETCH them. The LOB or XML data sets and their indexes are only allocated when the first insert is done, saving storage and application performance until the data is really saved in the database.

The administrator also can use the CHECK DATA utility to check the consistency between the XML schema, its document data and its XML index data.

Temporal queries and their business advantages

DB2 10 provides new temporal data functionality by using two new BUSINESS_TIME and SYSTEM_TIME table period definitions. These new period definitions are used for new temporal table definitions to provide system-maintained, period-maintained or bi-temporal (both system and period maintained) data stores. These temporal data tables are automatically maintained and when the designated time period criterion is met, the data is archived to an associated history table.

The PERIOD SYSYEM_TIME or PERIOD BUSINESS_TIME definition over two columns defines the temporal period for the data within the table. There are many definition restrictions on these temporal period columns. The SYSTEM_TIME relates to the time the data was put into the system. The BUSINESS_TIME relates to the business transaction or business relevant time period of the data. These definitions control the criteria for which data exists in the table and when it is migrated to the associated history table. By using both definitions, PERIOD SYSYEM_TIME and PERIOD BUSINESS_TIME, a table has bi-temporal criteria that control the data that exists in the table.

With the new BUSINESS_TIME WITHOUT OVERLAPS definition parameter the temporal tables also can make all your transaction time stamps unique. This is done using the new TIMESTAMP picosecond precision (precision 12)

enhancements to provide unique transaction time stamps across the entire temporal table. This is a great advantage for robust global systems previously having issues with the uniqueness of business timestamp transactions.

When SQL is executed against the temporal table, the key WHERE clause predicate “FOR (SYSTEM_TIME or BUSINESS_TIME) FROM columna TO columnb” or other similar predicate can be used in the SQL to get only data that is within your temporal timeframe. This both helps and complicates your SQL processing as INSERT or DELETE statements against temporal tables can result in multiple rows being inserted or modified in the table.

If the period specified by the start value and the end value for the BUSINESS_TIME of a row is only partially contained in the specified SQL WHERE predicate for an executed DELETE SQL statement, that row is deleted and then one or two additional rows are inserted. The inserted rows represent the original row values for the periods not deleted by the delete operation. For the newly inserted rows, the start value and end value for the BUSINESS_TIME are set in such a way to reflect data not affected by the DELETE SQL statement. Therefore, either the start value for the BUSINESS_TIME is the start value for the BUSINESS_TIME of the original row and the end value is the beginning predicate value, or the start value is ending predicate value and the end value is the end value for the BUSINESS_TIME of the original row.

The processing for an SQL UPDATE against a temporal table has similar considerations as an update can also result in additional rows inserted into the table to handle the BUSINESS_TIME configuration.

Example: Create a table, policy_info, that uses a SYSTEM_TIME period and create a history table, hist_policy_info. Then issue an ALTER TABLE statement to associate the policy_info table with the hist_policy_info table. (examples from the IBM SQL Reference manual)

```
CREATE TABLE policy_info
(policy_id CHAR(10) NOT NULL,
coverage INT NOT NULL,
sys_start TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN,
sys_end TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END,
create_id TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID,
PERIOD SYSTEM_TIME(sys_start,sys_end));

CREATE TABLE hist_policy_info
(policy_id CHAR(10) NOT NULL,
coverage INT NOT NULL,
sys_start TIMESTAMP(12) NOT NULL,
sys_end TIMESTAMP(12) NOT NULL,
create_id TIMESTAMP(12));
ALTER TABLE policy_info
ADD VERSIONING USE HISTORY TABLE hist_policy_info;
```

Example: Create a table, policy_info, that uses a BUSINESS_TIME period.

```
CREATE TABLE policy_info
```

```
(policy_id CHAR(4) NOT NULL,
coverage INT NOT NULL,
bus_start DATE NOT NULL,
bus_end DATE NOT NULL,
PERIOD BUSINESS_TIME(bus_start, bus_end));
```

Example: Create a table, policy_info, that uses both a SYSTEM_TIME period and a BUSINESS_TIME period to keep historical rows and track a user-specified time period. A table that specifies both a SYSTEM_TIME period and a BUSINESS_TIME period is sometimes referred to as a bi-temporal table. To enable retention of historical rows, a history table, hist_policy, also needs to be created and associated (using the ALTER TABLE statement) with the policy_info table.

```
CREATE TABLE policy_info
(policy_id CHAR(4) NOT NULL,
coverage INT NOT NULL,
bus_start DATE NOT NULL,
bus_end DATE NOT NULL,
sys_start TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN,
sys_end TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END,
create_id TIMESTAMP(12) GENERATED ALWAYS AS TRANSACTION START ID,
PERIOD BUSINESS_TIME(bus_start, bus_end),
PERIOD SYSTEM_TIME(sys_start, sys_end));

CREATE TABLE hist_policy_info
(policy_id CHAR(4) NOT NULL,
coverage INT NOT NULL,
bus_start DATE NOT NULL,
bus_end DATE NOT NULL,
sys_start TIMESTAMP(12) NOT NULL,
sys_end TIMESTAMP(12) NOT NULL,
create_id TIMESTAMP(12));

ALTER TABLE policy_info
ADD VERSIONING USE HISTORY TABLE hist_policy_info;
```

Timestamp, TIME ZONE and other data type enhancements

Greater timestamp precision for Java applications

DB2 10 enhances the TIMESTAMP data type with greater precision and provides a new TIME ZONE sensitive capability, providing more compatibility and functionality for all types of applications.

The TIMESTAMP precision enhancement supports up to 12 digits of fractional seconds (picoseconds) with the default matching the Java default of 6 digits precision of fractional seconds. The 6 digits default also helps Java functionality and DB2 Family compatibility along with SQL Server compatibility. The enhanced CURRENT TIMESTAMP utilizes a special register so applications can specify their desired fractional precision for their application requirements. The precision of the

timestamp seconds can also be adjusted to 0 or 3 digits if that satisfies the application requirements.

Support for TIMESTAMP with TIME ZONE

The TIMESTAMP with TIMEZONE is a new DB2 data type. This new data type incorporates the new TIMESTAMP 12-digit fractional seconds capabilities and also utilizes the new industry standard UTC (Universal Coordinated Time) replacing the old GMT (Greenwich Mean Time). This provides applications with additional TIMEZONE capabilities to compare business divisions along the same exact timeline across the world which is vital for global financial, retail and banking systems.

Access to currently committed data

With 5 to 10 times more concurrent threads within a single DB2 member, DB2 10 focuses a significant amount of enhancements on application concurrency. DB2 now provides an individual package option for managing concurrency within your applications. This enhancement provides a DB2 package level BIND parameter to let you choose the way your applications should handle data concurrency situations.

DB2 10 introduces the new CURRENTACCESSRESOLUTION parameter with the USECURRENTLYCOMMITTED and WAITFOROUTCOME settings. This parameter setting overrides the DB2 subsystem parameters of EVALUNC and SKIPUNCI and helps the application package quickly perform the desired concurrency action.

The USECURRENTLYCOMMITTED setting instructs the system to ignore rows that are in the process of being inserted and only use currently committed rows. This clause is contingent on the package BIND isolation level settings being either Cursor Stability or Read Stability.

The WAITFOROUTCOME setting instructs the system to wait for the rows that are in-flight to be resolved through a rollback or commit. This causes the application to wait for insert or delete activities to be committed or rolled back before determining the rows that will be included in the application SQL answer set.

These different settings provide the application with the flexibility to handle highly concurrent web transactions, wait or use uncommitted data and give tremendous flexibility to the enterprise architecture. These different parameter settings help provide the desired package level of concurrency and also provide capabilities that mimic some other database vendor's application concurrency settings.

SQL compatibility improvement

Extended indicator variable

DB2 10 introduces a new extended indicator variable that provides a way to specify that there is no value provided for an INSERT, UPDATE and MERGE statement column. As the extended indicator variable name implies this extends the functionality that can be used within an indicator variable for providing values within an SQL statement. For example, a value of -5 within an enabled extended indicator variable specifies the DEFAULT value. If the extended indicator variables are not enabled on the SQL package then the -5 specifies a NULL value.

If the extended indicator value is enabled and given a value of -7, this indicates the variable is to be UNASSIGNED, ignored and treated as if it did not exist within the SQL statement. These extended indicator variables are typically for Java applications and are quite useful for dynamic statements and variable SQL statement coding where the number of possible host variable parameters is unknown until the transaction logic is completed and the SQL is to be executed.

This addresses the application issue that previously required multiple SQL statements coded to match the values that were available for the SQL statements. Now these multiple SQL statements can be consolidated. When the column value is not known, the host variable value can use the new keyword UNASSIGNED for the appropriate column(s).

This is especially important for applications using dynamic statements that are clogging up their system's Dynamic Statement Cache with many copies of essentially the same SQL statements.

Extended support for implicit casting

Implicit casting is the automatic conversion of different types of data to be compatible. DB2 enhances its implicit casting by handling numeric data types that are able to be implicitly cast to character or graphical string data types. It also supports converting the data the other direction from character or graphical string data types to numeric data types.

In previous releases of DB2, this had to be done manually and was a labor intensive application process. Now numeric data, character and graphical string data can be handled, compared and assigned implicitly. DB2 for z/OS is more compatible and enhances portability of SQL from other database vendor systems.

Enhanced scalar function support

DB2 10 enhances its compatibility with other database vendors with improvements in SQL scalar and table functions. These built-in functions are used throughout application SQL making quick work of OLAP functions, calculations such as SUM,

AVG, SIN, COS and many others. As in previous releases, these inline functions with their SQL statements return a single value.

Non-inline SQL scalar functions that contain logic provide additional application functionality and flexibility. This flexibility helps DB2 family compatibility and acceptance of data migrations from other database vendors. DB2 also supports multiple versions and source code management of these functions based on their parameter list, routine options and function body. These functions can be altered, replaced with different versions distributed to multiple servers to assist in testing and overall performance. Function version fallback to a previous version is done instantly without a rebind or recompile when a function version is dropped.

DB2 10 introduces support for SQL user-defined table functions that helps ease application migrations from other database vendors. DB2 table functions are very flexible because they return a single data table result based on the many different type of parameters such as LOBs, distinct types, and transition tables.

SQL procedural language enhancements

DB2 9 provided new native support for the SQL procedural language, eliminating the cumbersome requirement to generate a C program from the SQL procedure that would then execute as an external stored procedure. DB2 9 SQL procedures can be executed natively within the DB2 engine for better runtime execution and stored in the DB2 catalog for better management and version control. Running the native code within the DB2 engine also helps in debugging, deploying and managing SQL procedural versions across multiple servers. Storing the SQL procedures improves the overall change control of this application code so that it can be managed like your other application developer modules.

The SQL procedural language has many enhancements such as SQL Table functions, nested compound SQL statements within a procedure and the RETURN statement that can return the result set of a SELECT SQL statement. These Stored Procedure and the other SQL procedural language enhancements allow all types of processing.

The DB2 10 SQL procedural language enhancements provide needed compatibility with other database vendors. The procedure language enhanced ability to accept many data types and XML as parameters and provide limited use of scrollable cursors provides great compatibility and integration opportunities for your applications. The DB2 10 concurrency improvements along with its SQL procedural language compatibility provide the opportunity to migrate other relational database management solutions to the z/OS environment for a better cost-of-ownership experience while providing the unique performance, availability, and scalability capabilities that can only be found with DB2 for z/OS and System z.

Section V – Data Warehousing

Support for temporal tables and versioning

DB2 10 has a many features and functions that make it a great platform for data warehousing applications. The enhancements such as the new temporal data enhancements, many automatic SQL optimizer enhancements, new Hash Access plan, SQL scalar and table functions, new timestamp nanosecond and no overlap precision, timestamp with time zone and all the other performance, availability and scalability improvements contributed to its superiority for data warehousing and business intelligence applications.

The temporal data enhancements are a major advance which can automatically archive and version data into your history tables as the business time and system time change. This automatic archiving coupled with the partition by growth table spaces allow automatic expansion as the data comes into and is then archived in the system.

Many new SQL optimizer enhancements improve performance for many more ad hoc queries submitted through OLAP third party tools or through ODBC connections from Excel spreadsheets handling many more concurrent users. These ad hoc queries with their business time or system time criteria will properly handle the time aspect of the bi-temporal data warehouse SQL questions and return the users' desired answers.

The new Hash Access plan reduces the retrieval of Dimension information down to a single I/O for great response time while the SQL scalar and table functions aggregate and augment the non-overlapping data into a new Java precision interface within a global time zone aware context for enterprise scope analysis.

All of these features and capabilities coupled with the high availability, most scalable and reliable platform, System z, offer a data warehouse platform for optimum system performance and operational business intelligence systems delivered to the entire enterprise.

OLAP functionality built directly into DB2

Support for OLAP - moving sums, averages, and aggregates

The OLAP capabilities of moving sums, averages and aggregates are now built into DB2. Improvements within SQL, intermediate workfile results, and scalar or table functions provide performance for these OLAP activities.

Moving sums, averages and aggregates are common OLAP functions within any data warehousing application. These moving sums, averages and aggregates are typical standard calculations that are done using different groups of time-period- or location-based data for product sales, store location or other common criteria.

Having these OLAP capabilities built directly into DB2 provides an industry standard SQL process, repeatable applications, SQL function or table functions and robust performance through better optimization processes.

These OLAP capabilities are further enhanced through scalar, custom table functions or the new temporal tables to establish the window of data for the moving sum, average or aggregate to calculate its answer set. By using a partition, time frame or common table SQL expression the standard OLAP functions can provide the standard calculations for complex or simple data warehouse requirements. Also given the improvements within SQL these moving sums, averages and aggregates can be included in expressions, select lists or ORDER BY statements, satisfying any application requirements.

Advanced business analytics

The standard and customizable capabilities that are now available in DB2 10 provide advanced business analytics and fertile ground for customization. Built-in pureXML, LOB and open SQL scalar and table function interfaces provide many functions and capabilities that can be extended to any type of custom functionality for business requirements.

Temporal tables and their ability to automatically archive historical data provide active data management to help improve performance, audit, and compliance capabilities. Coupled with SQL that can now use business time or system time SQL parameters to qualify the answer, DB2 10 provides unique industry leading database advanced business analytical capabilities.

Section VI – Reduced Total Cost of Ownership (TCO)

DB2 10 reduces the total DB2 CPU demand from up to 5 to 20% in many different ways when all the new enhancements are fully leveraged. Many CPU reductions are built directly into DB2 requiring no application changes. Some enhancements are implemented through normal everyday DB2 activities through rebinding, restructuring database definitions, improving your applications, and utility processing. All of these CPU demand reduction features have the potential to provide significant total cost of ownership savings based on your shop's application mix and transaction types.

Improvements in optimization reduce costs by processing your SQL automatically with more efficient data access paths. Improvements through a new range-list index scan access method, in list pre-fetch, more parallelism for select and index insert processing, better workfile usage, better RID pool overflow management, access path certainty evaluation and improved DDF transaction flow all provide more efficiency without any

changes to your applications. All of these enhancements reduce your total CPU enterprise costs because of better efficiency in the new DB2 10 for z/OS.

Other enhancements require new database definitions or application programming techniques to reduce your overall costs. These enhancements such as the Hash Access space and access method, including columns on a unique index, consolidating SQL by using the new Attributes feature, and automatic data statistics significantly reduce CPU costs through improved access paths.

Reduced costs also come through better operations and improved availability with DB2 10. DB2 10 with its better memory management, ability to handle 5 to 10 times more users, ability to skip locked rows, improvements in the DB2 catalog, more online schema change capabilities and more online utilities all eliminate system downtime costs. DB2 10 keeps the application available even while more users, applications, database changes, and utilities are executing in the system.

More automated processes help install, configure and simplify the installation of your new DB2 10 system. Pre-migration installation steps capture existing settings and provide appropriate settings for the new DB2 supplied routines and programs. These new procedures are meant to reduce the installation, configuration, testing and reduce your time to value for your DB2 10 system.

The ability to migrate directly from DB2 V8 also allows shops that are behind in their software versions to skip implementing DB2 9. Even though they will not be available to leverage all the DB2 9 for z/OS performance enhancements highlighted in the Appendix, skipping Version 9 relieves the burden of going through the software installation, testing and implementation of an interim step to DB2 10. Migrating directly from Version 8 to Version 10 provides a quick way also to leverage all the great features within DB2 9 and DB2 10 immediately..

DB2 10 offers tremendous value by reducing operational and maintenance costs while improving performance. Plan to install your version of DB2 10 as soon as possible to take advantage of all of these great features.

Appendix

DB2 9 for z/OS Enhancement List

APPEND option at insert
Automatic creation of database objects
Autonomic index page split
Autonomic re-optimization
Buffer management by WLM
CLONE Table: fast replacement of one table with another
Change SCHEMA & VCAT
Conditional restart: automatic search for appropriate checkpoint
DECIMAL FLOAT, BIGINT enhancements
Database ROLEs
Enhanced CURRENT SCHEMA
Enhanced STOGROUP definition
FETCH CONTINUE
Faster REORG by intra-REORG parallelism
Faster and more automatic DB2 restart
Faster operations for variable length rows
Generalizing sparse index and in-memory data caching method
Global query optimization
IPv6 support
Index Lookaside enhancements
Index on expressions
Index page sizes 8K, 16K, 32K
LOB Lock reduction
LOBs Network Flow Optimization
Logging enhancements
MERGE statement
MODIFY RECOVERY enhancements
Modify early code without requiring an IPL
More online REORG by eliminating BUILD2 phase
NOT LOGGED table spaces
Native SQL Stored Procedures, able to use zIIP
Network Trusted Context
OLAP Zparms new options
ORDER BY and FETCH FIRST n ROWS in sub-select and full-select
ORDER OF extension to ORDER BY
Online CHECK DATA and CHECK LOB
Online REBUILD INDEX
Online RENAME COLUMN
Online RENAME INDEX
Optimization Service Center
Partition-by-growth table spaces
Preserving consistency when recovering individual objects to a prior point in time
RLF improvements for remote application servers such as SAP
Recovery of individual table spaces and indexes from volume-level backups
SELECT FROM UPDATE/DELETE/MERGE
SHRLEVEL(REFERENCE) for REORG of LOB table spaces
Skipping locked rows option

Support for optimistic locking
TRUNCATE TABLE statement
Tape support for BACKUP and RESTORE SYSTEM utilities
Temporary space consolidation
Unified Debugger
Universal Table spaces
Utilities CPU reduction
Utilities improvements
Utility TEMPLATE switching
VARBINARY, BINARY
Various scalar functions
XML support in DB2 engine

About the Author

Dave Beulke and Associates, a division of Pragmatic Solutions, Inc is an independent technology, analysis, education and consulting organization specializing in architecture analysis, database performance, data warehouses designs and internet architectures.

Other research and analysis conducted by Dave Beulke can be found at www.davebeulke.com.

Dave Beulke and Associates
3213 Duke Street
Suite 805
Alexandria, Virginia, 22314
(703) 798-3283
Dave@DaveBeulke.com