Comparing DBA Productivity: An Oracle/DB2 Task Complexity Analysis

Triton Consulting
November 2010

Contents

Abstract

As staff costs continue to consume an ever-increasing proportion of IT budgets, productivity and ease-of-use are becoming more important factors in the overall TCO for a given IT system. The degree of complexity associated with common DBA activities is therefore a key differentiator between the various database vendors' products.

Triton Consulting conducted an objective assessment of the complexity of several routine DBA activities, comparing Oracle Database 11gR2 and IBM DB2 for LUW 9.7. This paper presents the in-depth results and conclusions from the complexity analysis.

Introduction

The cost of IT staff such as database administrators (DBAs) continues to consume an ever-increasing proportion of IT budgets. As a result, the productivity of IT staff and the ease-of-use of the IT systems they work with are becoming more important factors in the overall Total Cost of Ownership (TCO) equation. The degree of complexity associated with common DBA activities is therefore a key differentiator between the various database vendors' products.

Triton Consulting conducted an objective assessment of the complexity of several routine DBA activities, comparing Oracle Database 11gR2 and DB2 9.7. This study examines the following common DBA activities, and assesses the complexity of each within an Oracle Database and DB2 for LUW environment:

- Installation
- Enabling table compression
- Enabling index compression
- Backup and recovery
- Automatic memory tuning
- Data Access Control

Each of these activities has been broken down into a number of more detailed steps, which are subjected to complexity analysis using the methodology described below.

This paper presents the in-depth results and conclusions from the complexity analysis.

Complexity Analysis Methodology

Complexity analysis is a new approach to evaluating and communicating the usability of software and is based on established research as described in this document:
http://www-01.ibm.com/software/ucd/Resources/ComplexityAnalysisDGarticle.pdf.
It is used to quantify and illustrate the usability of specific tasks. Complexity analysis reduces the subjectivity associated with usability claims and injects a strong element of objective transparency into the picture.

The complexity analysis methodology used within this study consists of the following process:

1. Define user roles (such as "operational storage administrator" or "procurement approver"): User roles characterize the users that interact with the software being evaluated in the complexity analysis. User roles need to be defined at the outset because the complexity of performing a task will vary depending on the type of user.

2. Define targeted user tasks / activities (such as "install the database" or "evaluate a procurement request"): Each task represent the user's interaction with the software to accomplish a goal. The user's interactions are broken down into component steps that define a typical path through the software for accomplishing the task. For this study comparable user tasks and corresponding steps were defined for both DB2 9.7 and Oracle Database 11gR2.

3. Rate the complexity of each step: Once the user roles and tasks are defined then the complexity of each step in the tasks is rated. These ratings are assigned based on a standardized and objective set of rating scales for each of the following complexity dimensions: context shifts, navigational guidance, input parameters, system feedback, error feedback, new concepts. An example of a complexity rating for a step is "Navigational Guidance is rated as level 4 (navigation supported by basic documentation)". The complexity dimensions are described in more detail in the latter part of this section.

4. Calculate the complexity metrics: Each complexity rating is then mapped into a complexity metric. (For example, a navigational guidance rating of 4 is mapped into a complexity metric of 6.) This mapping is based on a standardized weighting of the complexity dimensions and their corresponding rating scales. The complexity metrics derived for each complexity dimension of a step are added together to calculate the total complexity metric for the step. Similarly, the complexity metrics for the steps are added together to calculate the total complexity metric for the overall task. Note that as the complexity metric for a step increases then the usability of performing the step decreases. Therefore, lower complexity metric values are better than higher values.

5. Analyse the results: The resulting complexity metrics are analyzed – in the case of this study – to draw conclusions about the relative complexity of performing the user tasks with DB2 9.7 and Oracle Database 11gR2.

Complexity analysis is based on the following complexity dimensions. Each of these dimensions captures a potential source of complexity experienced by the user while interacting with software to perform a step in a user task / activity.

- **Context shifts**. A context shift occurs when the user crosses user interface, tool, or product boundaries in order to perform a step. For example, if a user moves from an eclipse-based application development tool to a web-based monitoring tool to complete a step then the user experiences a context shift.

- **Navigational guidance**. Navigational guidance refers to the support provided to a user for proceeding into a step (from the previous step) and through the step. It is characterized by the user question: “where do I go next, and what do I do when I get there?”.

- **Input parameters**. An input parameter is data supplied by the user to complete the step. For example, a userid and password are two input parameters commonly associated with task steps that involve security authentication.

- **System feedback**. System feedback is the system response to the user actions for a given step (with the exception of error feedback which is addressed in the next point). Examples of system feedback include progress indication dialog boxes, confirmation of command execution, and system-generated reports.

- **Error feedback**. Error feedback is the system response to common error situations the user may encounter. Examples of error feedback include error messages, visual cues that identify incorrect field values, and troubleshooting documentation.

- **New concepts**. A concept is background information on a topic area the user needs to understand in order to perform a step. Concepts include the underlying externals model of the product and user interface objects / metaphors / models. A new concept for a step is a concept that is introduced to the user for the first time in the context of their current task(s).

Complexity Analysis

This section provides a breakdown of each of the six DBA activities addressed by the study, together with the outcome of the complexity analysis for each.

Activity 1: Installation

Activity Overview

The initial task covers the installation process for DB2 9.7 and Oracle Database 11gR2. Each installation is completed in one sequential end-to-end flow.

The customer has a Red Hat Linux 5 environment and will be installing DB2 9.7 Enterprise Server Edition or Oracle Database 11gR2 Enterprise Edition; both of the installations will include pre-installation steps, sample database creation, and post-installation validation.

User Roles

Database administrator who has root authority and is familiar with databases but new to DB2 and Oracle.

Assumptions

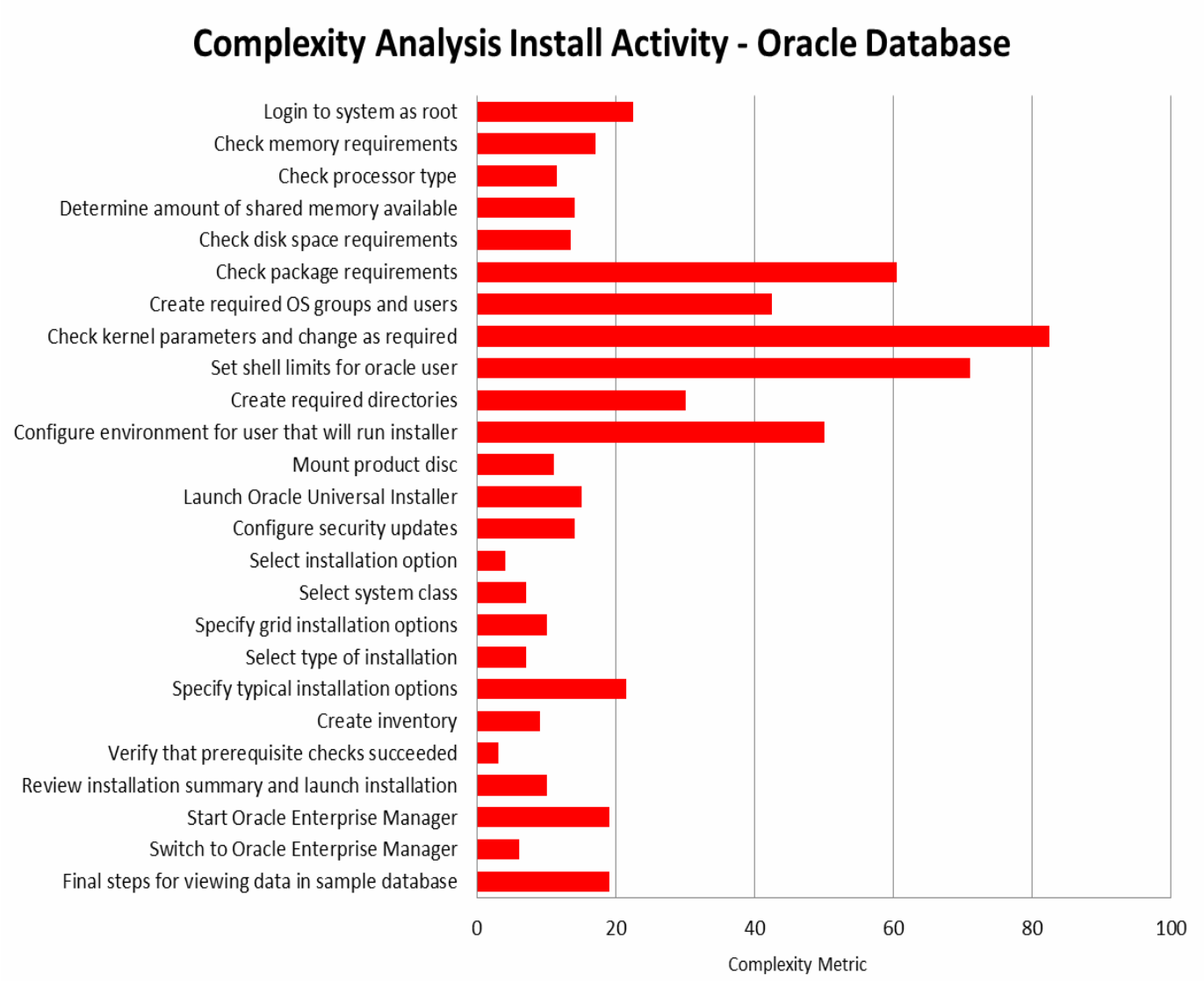
- User has a newly-installed Red Hat Enterprise Linux 5 (RHEL 5) system.
- The Operating System environment (for example, kernel parameters and installed packages) has not yet been customised for DB2 or Oracle.
- Assumptions for the Oracle Database task:
    - User will install Oracle Database Enterprise Edition software and create the sample database.
    - The task flow is based on Oracle 11gR2 “Database Quick Installation Guide for Linux x86” manual.
    - Steps for checking package requirements are included in the task flow even though the Oracle Database installer checks for these package requirements. The rationale for checking package requirements during pre-installation is that if these requirements are not checked / addressed before installation then the Oracle Database installer would flag the missing packages, forcing the user to exit the installer, install these packages, and then restart the installer. Same applies for kernel parameters.
- Assumptions for the DB2 task:
    - User will be installing DB2Version 9.7 Enterprise Server Edition software and creating the sample database. The task flow below is based on the DB2 9.7 manual “Getting Started with DB2 Installation and Administration on Linux and Windows”.
    - The user has root authority and therefore will follow the path for root installation. Note that root install is the typical way of installing DB2 (whereas non-root install is the typical – and only -- way for Oracle). Note also that DB2 does, in fact, support non-root install.
    - System has a supported browser and X-window system installed.

Activity Analysis

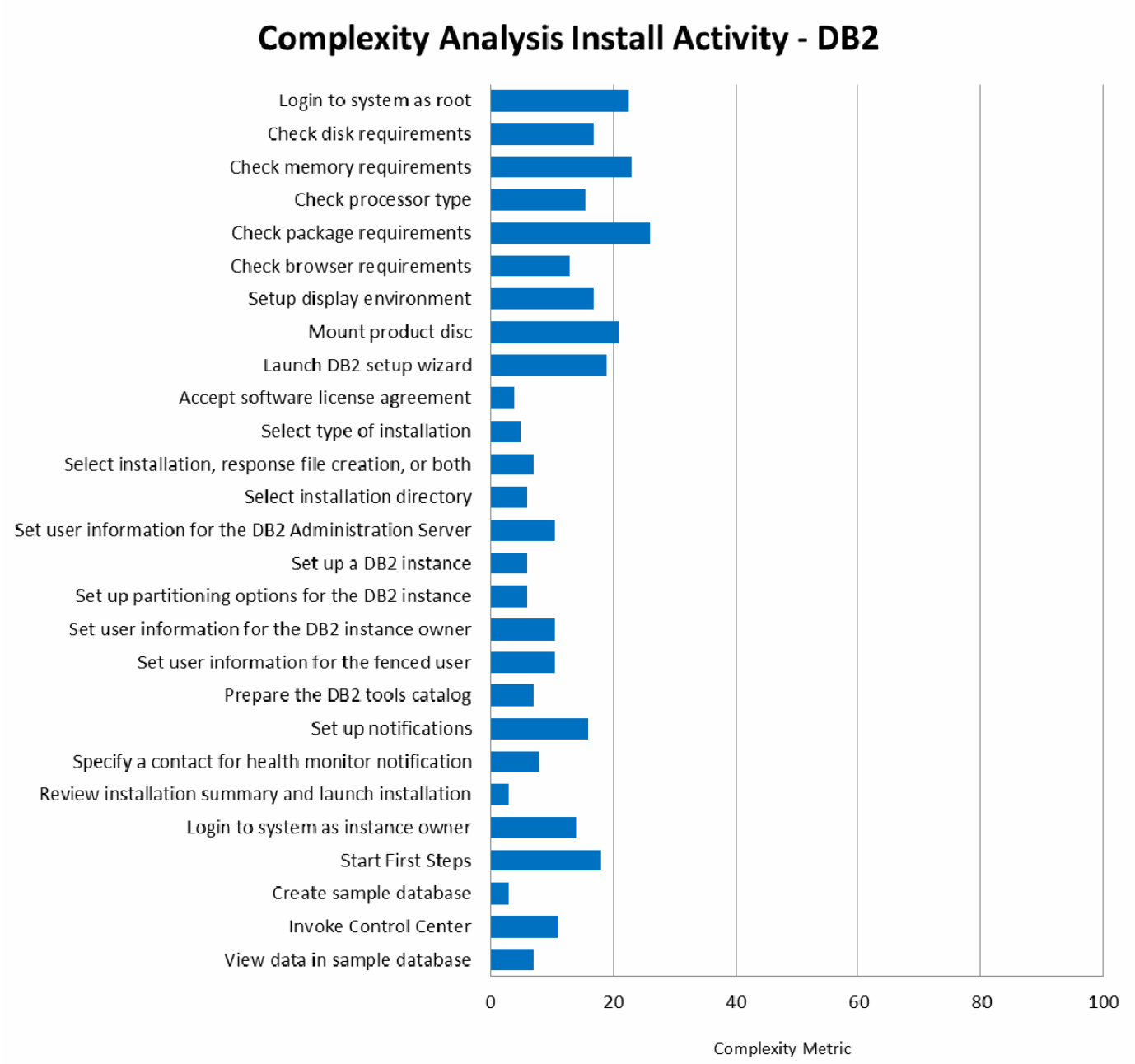
The table below details the complexity rating and derived complexity metric associated with each step in the installation activity.

DBMS	Step	Complexity Rating						Complexity Metric						
		Context shifts (0-4 each)	Navigational guidance (1-5 overall)	Input parameters (0-5 each)	System feedback (0-4 overall)	Error feedback (0-5 overall)	New concepts (0-4 overall)	Context shifts	Navigational guidance	Input parameters	System feedback	Error feedback	New concepts	Step totals
Oracle Database	*Pre-installation - validate pre-reqs*													
	Login to system as root	0	3	15.5	1	2	0	0	4	15.5	1	2	0	22.5
	Check memory requirements	0	3	9.0	1	2	1	0	4	9.0	1	2	1	17
	Check processor type	0	3	4.5	1	2	0	0	4	4.5	1	2	0	11.5
	Determine amount of shared memory available	0	3	6.0	1	2	1	0	4	6.0	1	2	1	14
	Check disk space requirements	0	3	5.5	1	2	1	0	4	5.5	1	2	1	13.5
	Check package requirements	0	3	50.5	1	2	2	0	4	50.5	1	2	3	60.5
	Pre-installation - setup environment													
	Create required OS groups and users	0	3	30.5	2	2	2	0	4	30.5	3	2	3	42.5
	Check kernel parameters and change as required	0	3	69.5	1	2	3	0	4	69.5	1	2	6	82.5
	Set shell limits for oracle user	0	3	58.0	1	2	3	0	4	58.0	1	2	6	71
	Create required directories	0	3	20.0	1	2	2	0	4	20.0	1	2	3	30
	Configure environment for user that will run installer	0	3	40.0	2	2	1	0	4	40.0	3	2	1	50
	Installation - Oracle Universal Installer													
	Mount product disc	0	3	4.0	1	2	0	0	4	4.0	1	2	0	11
	Launch Oracle Universal Installer	4	3	2.0	1	2	0	6	4	2.0	1	2	0	15
	Configure security updates	1	1	6.0	1	2	2	1	1	6.0	1	2	3	14
	Select installation option	1	1	1.0	1	0	0	1	1	1.0	1	0	0	4
	Select system class	1	1	1.0	1	0	2	1	1	1.0	1	0	3	7
	Specify grid installation options	1	1	1.0	1	0	3	1	1	1.0	1	0	6	10
	Select type of installation	1	1	1.0	1	0	2	1	1	1.0	1	0	3	7
	Specify typical installation options	1	1	10.5	1	2	3	1	1	10.5	1	2	6	21.5
	Create inventory	1	1	1.0	1	2	2	1	1	1.0	1	2	3	9
	Verify that prerequisite checks succeeded	1	1	0.0	1	0	0	1	1	0.0	1	0	0	3
	Review installation summary and launch installation	1	2	0.0	2	3	0	1	2	0.0	3	4	0	10
	Post-installation validation													
	Start Oracle Enterprise Manager	4	3	3.0	1	2	2	6	4	3.0	1	2	3	19
		4						6	0	0.0	0	0	0	6
	Final steps for viewing data in sample database	1	4	8.0	1	0	2	1	6	8.0	1	0	3	19
	Totals for task							28	72	347.5	30	38	55	570.5
DB2 9.7	*Pre-installation - validate pre-reqs*													
	Login to system as root	0	3	15.5	1	2	0	0	4	15.5	1	2	0	22.5
	Check disk requirements	0	3	9.0	1	2	1	0	4	9.0	1	2	1	17
	Check memory requirements	0	4	13.0	1	2	1	0	6	13.0	1	2	1	23
	Check processor type	0	4	6.5	1	2	0	0	6	6.5	1	2	0	15.5
	Check package requirements	0	4	14.0	1	2	2	0	6	14.0	1	2	3	26
	Check browser requirements	3	4	0.0	1	2	0	4	6	0.0	1	2	0	13
	Installation - DB2 Setup Wizard													
	Setup display environment	3	4	4.0	1	2	0	4	6	4.0	1	2	0	17
	Mount product disc	0	4	12.0	1	2	0	0	6	12.0	1	2	0	21
	Launch DB2 setup wizard	4	3	6.0	1	2	0	6	4	6.0	1	2	0	19
	Accept software license agreement	1	1	1.0	1	0	0	1	1	1.0	1	0	0	4
	Select type of installation	1	1	1.0	1	0	1	1	1	1.0	1	0	1	5
	Select installation, response file creation, or both	1	1	1.0	1	0	2	1	1	1.0	1	0	3	7
	Select installation directory	1	1	0.0	1	2	1	1	1	0.0	1	2	1	6
	Set user information for the DB2 Administration Server	1	1	2.5	1	2	2	1	1	2.5	1	2	3	10.5
	Set up a DB2 instance	1	1	0.0	1	0	2	1	1	0.0	1	0	3	6
	Set up partitioning options for the DB2 instance	1	1	0.0	1	0	2	1	1	0.0	1	0	3	6
	Set user information for the DB2 instance owner	1	1	2.5	1	2	2	1	1	2.5	1	2	3	10.5
	Set user information for the fenced user	1	1	2.5	1	2	2	1	1	2.5	1	2	3	10.5
	Prepare the DB2 tools catalog	1	1	1.0	1	0	2	1	1	1.0	1	0	3	7
	Set up notifications	1	1	5.0	1	2	3	1	1	5.0	1	2	6	16
	Specify a contact for health monitor notification	1	1	0.0	1	2	2	1	1	0.0	1	2	3	8
	Review installation summary and launch installation	1	1	0.0	1	0	0	1	1	0.0	1	0	0	3
	Post-installation validation													
	Login to system as instance owner	4	3	1.0	1	2	0	6	4	1.0	1	2	0	14
	Start First Steps	4	3	2.0	1	2	2	6	4	2.0	1	2	3	18
	Create sample database	1	1	0.0	1	0	0	1	1	0.0	1	0	0	3
	Invoke Control Center	3	1	2.0	1	0	2	4	1	2.0	1	0	3	11
	View data in sample database	2	3	0.0	1	0	0	2	4	0.0	1	0	0	7
	Totals for task							46	75	101.5	27	34	43	326.5

The derived complexity metric for each Oracle Database step is shown graphically below.



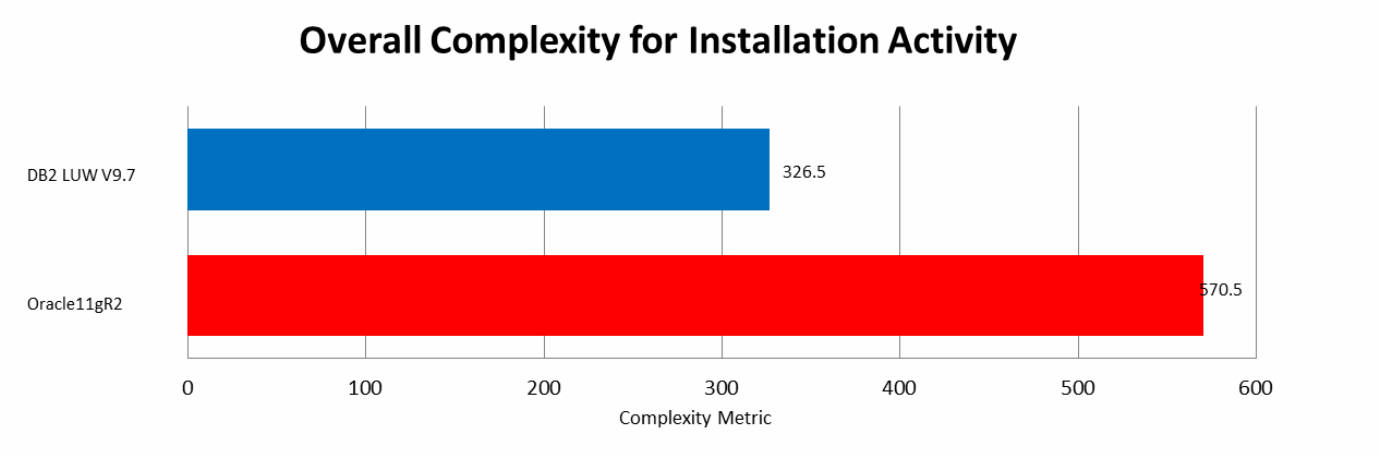
The derived complexity metric for each DB2 step is shown graphically below.



The installations on both platforms are fairly similar, but significant differences are found in the additional steps required to set up the environment for Oracle Database prior to the actual install. This covers creating OS groups and users, checking kernel parameters, setting shell limits for the Oracle Database user, creating required directories and configuring the environment for the user that will run the installer. These tasks add a significant amount of additional complexity to the Oracle Database install task. In contrast, DB2 includes many of these tasks as part of the main installation process which is guided by the DB2 setup wizard.

The following chart compares the overall complexity metrics for the Oracle Database and DB2 installation activities. The complexity metric for DB2 is nearly 43% lower than the complexity metric for Oracle Database. This difference in complexity can have a significant impact in various aspects of total cost of ownership. Our projections show that the Oracle Database installation task for the specified environment could take over 100 minutes of DBA interaction time[1] to complete. In contrast, the same installation task for DB2 would take a little over 60 minutes of interaction time. In addition to these time savings come several additional benefits arising from DB2's lower complexity. This includes DB2 having reduced skill requirements and a reduced risk of errors that can impact quality of service.

[1] Interaction time is defined and discussed in more detail in the "Summary and Conclusions" section.



Based on the results of the quantitative complexity analysis, we have found DB2 9.7 installation to be dramatically easier to implement than the equivalent functionality within Oracle Database 11gR2.

Activity 2: Data Compression

Activity Overview

A customer requires a database that contains both traditional data types (characters, numbers, dates) as well as XML data. To save storage costs and minimize physical I/O to improve overall database performance, the customer wants to compress as much data as possible.

User Roles

Database administrator who has experience with each respective database but is new to compression

Assumptions

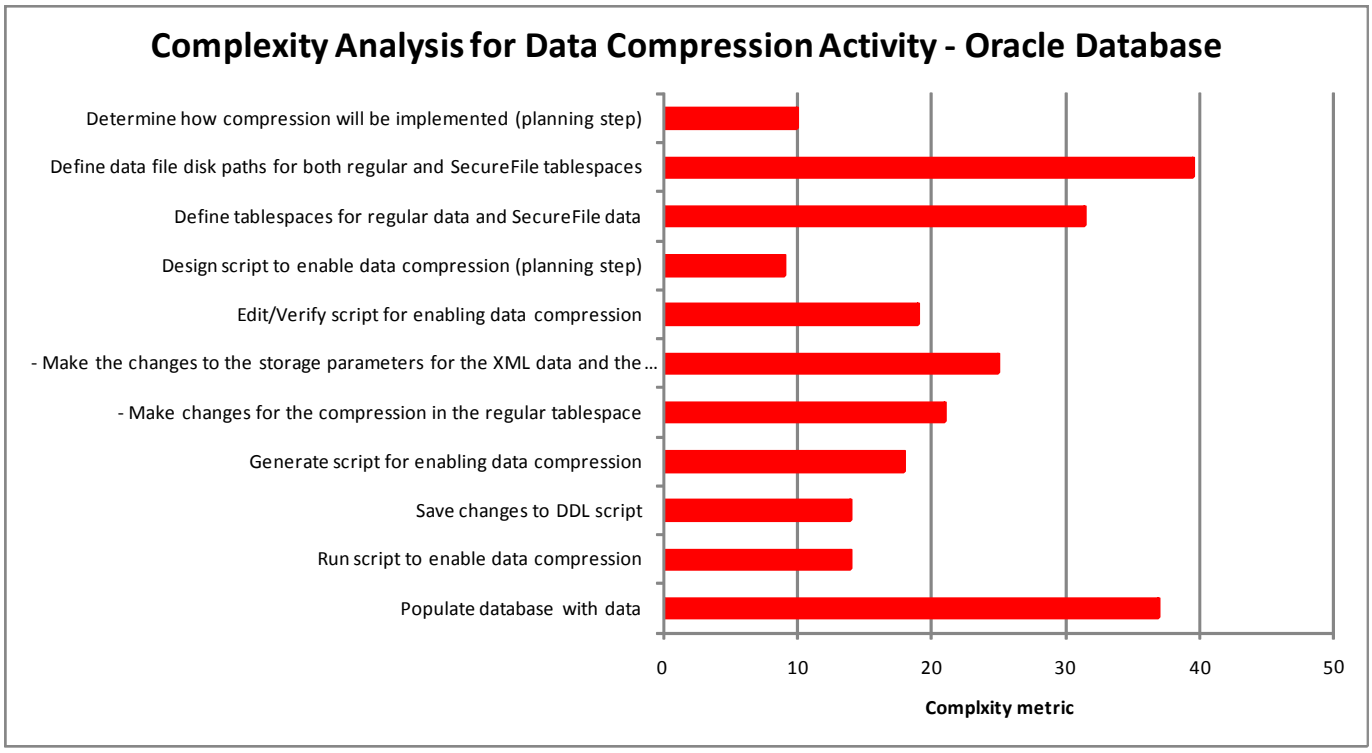
- A customer requires a new database that contains regular data (integer and character data) and XML data.
- The database will have 300 tables, with some tables containing large object (LOB) columns consisting of XML documents. There are no other types of LOB columns in the database.
- The user already has a DDL script (with basic database structure defined) that will be used as a starting point for this task. The DDL script needs to be modified to leverage compression.
- The user already has an ASCII delimited file that was generated by performing an extract from the data source
- The task begins with the user already connected to the database.
- Additional assumptions for Oracle Database task:
    - All LOB (XML) columns will be stored using SecureFiles within Oracle ASSM.
    - Table compression for the SecureFile (XML) columns will be defined as using “MEDIUM” (default) compression for a balance between performance and compression ratio so as to not impact OLTP operation adversely.
    - SecureFiles compression will also utilize the deduplication compression option.
    - XML columns exist with other columns within the tables, so no XMLType tables will be created within Oracle Database.

Activity Analysis

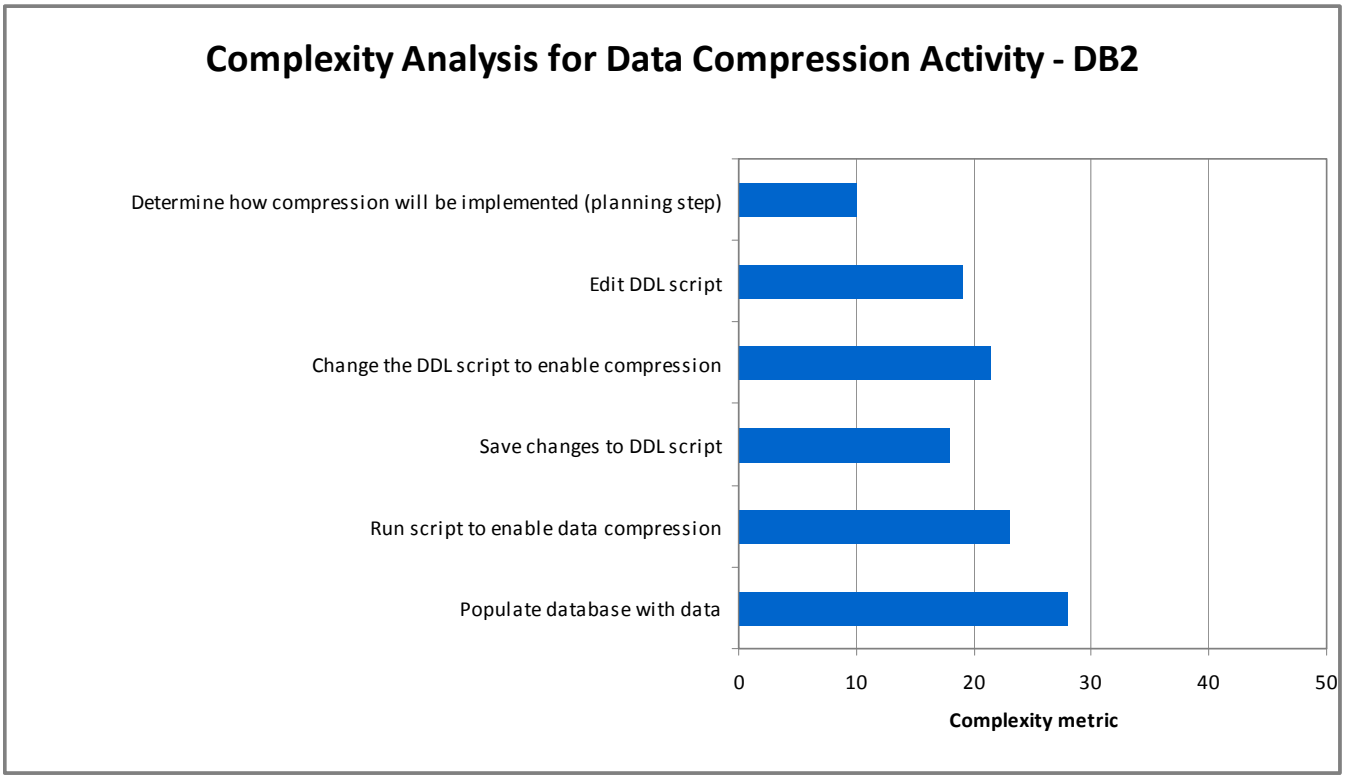
The table below details the complexity ratings and derived complexity metrics associated with each step in the data compression activity.

DBMS	Step	Complexity Rating						Complexity Metric						
		Context shifts (0-4 each)	Navigational guidance (1-5 overall)	Input parameters (0-5 each)	System feedback (0-4 overall)	Error feedback (0-5 overall)	New concepts (0-4 overall)	Context shifts	Navigational guidance	Input parameters	System feedback	Error feedback	New concepts	Step totals
Oracle Database 11gR2	Determine how compression will be implemented (planning step)	0.00	3.00	0.00	0.00	0.00	3.00	0.00	4.00	0.00	0.00	0.00	6.00	**10.00**
	Define data file disk paths for both regular and SecureFile tablespaces	16.00	4.00	8.50	1.00	2.00	3.00	16.00	6.00	8.50	1.00	2.00	6.00	**39.50**
	Define tablespaces for regular data and SecureFile data	0.00	4.00	19.50	1.00	2.00	2.00	0.00	6.00	19.50	1.00	2.00	3.00	**31.50**
	Design script to enable data compression (planning step)	0.00	5.00	0.00	0.00	0.00	0.00	0.00	9.00	0.00	0.00	0.00	0.00	**9.00**
	Edit/Verify script for enabling data compression	3.00	5.00	3.00	1.00	2.00	0.00	4.00	9.00	3.00	1.00	2.00	0.00	**19.00**
	- Make the changes to the storage parameters for the XML data and the table	0.00	5.00	10.00	1.00	2.00	2.00	0.00	9.00	10.00	1.00	2.00	3.00	**25.00**
	- Make changes for the compression in the regular tablespace	0.00	5.00	6.00	1.00	2.00	2.00	0.00	9.00	6.00	1.00	2.00	3.00	**21.00**
	Generate script for enabling data compression	3.00	5.00	2.00	1.00	2.00	0.00	4.00	9.00	2.00	1.00	2.00	0.00	**18.00**
	Save changes to DDL script	0.00	5.00	2.00	1.00	2.00	0.00	0.00	9.00	2.00	1.00	2.00	0.00	**14.00**
	Run script to enable data compression	0.00	5.00	2.00	1.00	2.00	0.00	0.00	9.00	2.00	1.00	2.00	0.00	**14.00**
	Populate database with data	0.00	5.00	24.00	1.00	2.00	1.00	0.00	9.00	24.00	1.00	2.00	1.00	**37.00**
	Totals for activity							**24.00**	**88.00**	**77.00**	**9.00**	**18.00**	**22.00**	**238.00**
DB2 for LUW 9.7	Determine how compression will be implemented (planning step)	0.00	3.00	0.00	0.00	0.00	3.00	0.00	4.00	0.00	0.00	0.00	6.00	**10.00**
	Edit DDL script	3.00	5.00	3.00	1.00	2.00	0.00	4.00	9.00	3.00	1.00	2.00	0.00	**19.00**
	Change the DDL script to enable compression	0.00	5.00	6.50	1.00	2.00	2.00	0.00	9.00	6.50	1.00	2.00	3.00	**21.50**
	Save changes to DDL script	3.00	5.00	2.00	1.00	2.00	0.00	4.00	9.00	2.00	1.00	2.00	0.00	**18.00**
	Run script to enable data compression	0.00	4.00	9.00	1.00	3.00	2.00	0.00	6.00	9.00	1.00	4.00	3.00	**23.00**
	Populate database with data	0.00	5.00	15.00	1.00	2.00	1.00	0.00	9.00	15.00	1.00	2.00	1.00	**28.00**
	Totals for activity							**8.00**	**46.00**	**35.50**	**5.00**	**12.00**	**13.00**	**119.50**

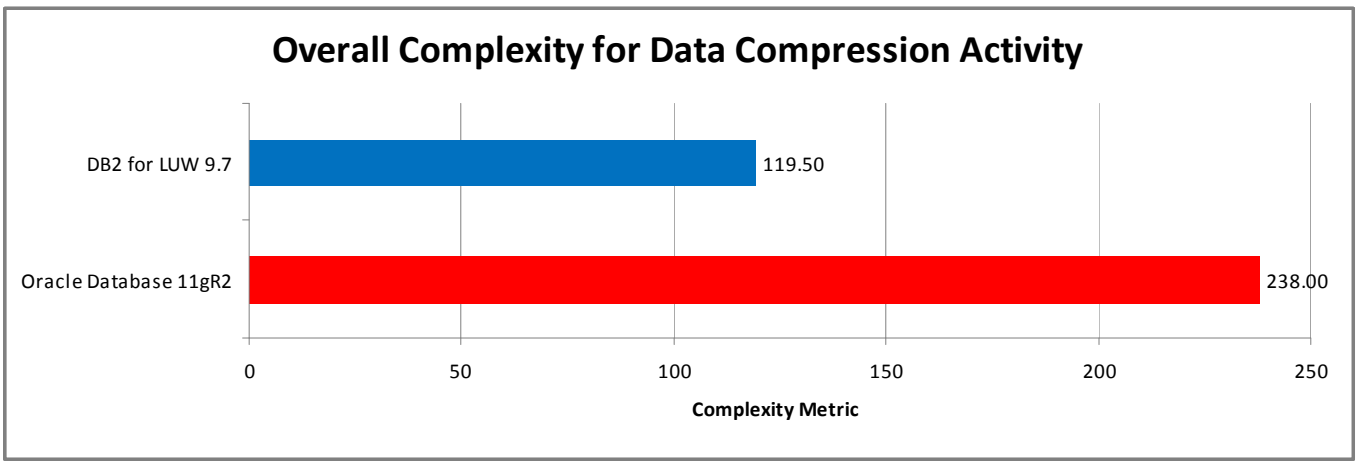
The derived complexity metric for each Oracle Database step is shown graphically below.



The derived complexity metric for each DB2 step is shown graphically below.



The chart below compares the overall complexity metrics for the Oracle Database and DB2 data compression activities. The complexity metric for DB2 is approximately 50% lower than the complexity metric for Oracle Database. This difference in complexity can have a significant impact in various aspects of total cost of ownership. Our projections show that the Oracle Database data compression task for the specified environment could take approximately 46 minutes of DBA interaction time to complete. In contrast, the same data compression task for DB2 would take approximately 23 minutes of DBA interaction time . In addition to these time savings come several additional benefits resulting from DB2's lower complexity. This includes DB2 having lower skill requirements and a lower risk of errors that can impact quality of service.



There are some important functional differences between DB2 and Oracle Database compression which directly impact task complexity. For example DB2 9.7 supports full XML compression regardless of where the XML data is stored. In contrast, Oracle Database requires that XML data be stored externally to use compression, and this requires additional syntax for each column definition within the CREATE TABLE statement. Additionally DB2 9.7 table compression compresses data across the row within the entire table, rather than only compressing duplicates within a single data block page as is the case with Oracle Database.

Activity 3: Index Compression

Activity Overview

A customer has a database and wants to compress all indexes in the database where it makes sense (the goal is to save at least 20% in storage for each compressed index). The underlying customer objective behind compressing indexes is to save storage costs and minimize physical I/O to improve overall database performance.

User Roles

Database administrator who has experience with each respective database but is new to compression.

Assumptions

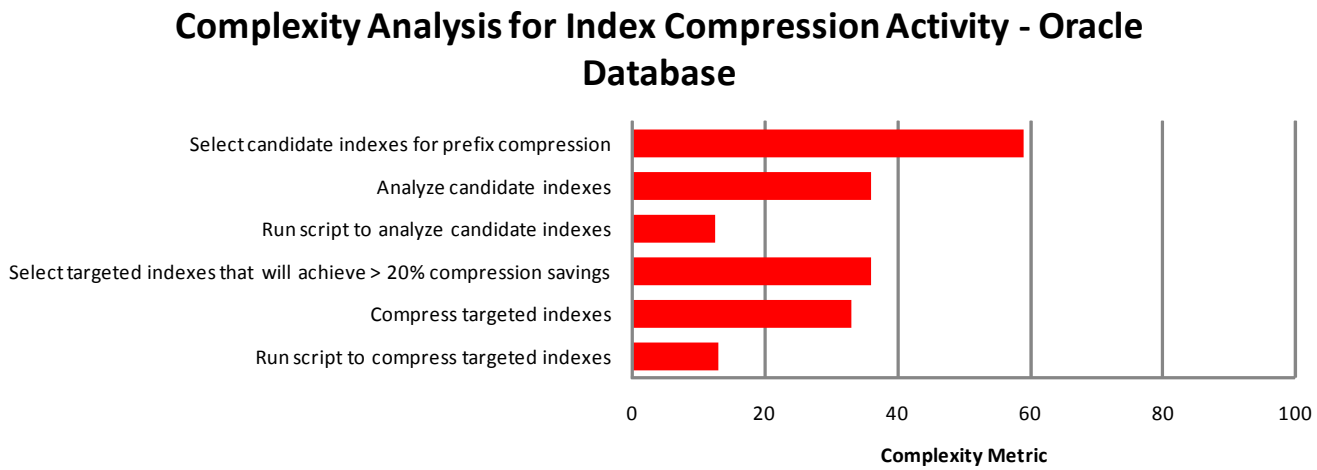
- Same assumptions as for the data compression task apply to this task.
- We do not include testing of the Oracle Database scripts (although DBAs would, in fact, typically carry out such testing). So the Oracle Database complexity metric for this task is a lower bound (i.e., the best-case scenario for Oracle Database).
- Up-front planning for index compression has already been done as part of the planning for data compression.
- The task begins immediately after completion of the data compression task.
- The tables and indexes were created in the data compression task.

Activity Analysis

The table below details the complexity ratings and derived complexity metrics associated with each step in the index compression activity. Note that there are no figures for DB2 in this table as the index compression is done automatically as part of the data compression task flow shown previously on page 10.

DBMS	Step	Complexity Rating						Complexity Metric						
		Context shifts (0-4 each)	Navigational guidance (1-5 overall)	Input parameters (0-5 each)	System feedback (0-4 overall)	Error feedback (0-5 overall)	New concepts (0-4 overall)	Context shifts	Navigational guidance	Input parameters	System feedback	Error feedback	New concepts	Step totals
Oracle Database	Select candidate indexes for prefix compression	0.00	4.00	39.00	3.00	2.00	3.00	0.00	6.00	39.00	6.00	2.00	6.00	**59.00**
	Analyze candidate indexes	0.00	5.00	18.00	1.00	2.00	3.00	0.00	9.00	18.00	1.00	2.00	6.00	**36.00**
	Run script to analyze candidate indexes	0.00	5.00	0.50	1.00	2.00	0.00	0.00	9.00	0.50	1.00	2.00	0.00	**12.50**
	Select targeted indexes that will achieve > 20% compression savings	0.00	5.00	13.00	3.00	2.00	3.00	0.00	9.00	13.00	6.00	2.00	6.00	**36.00**
	Compress targeted indexes	0.00	5.00	18.00	1.00	2.00	2.00	0.00	9.00	18.00	1.00	2.00	3.00	**33.00**
	Run script to compress targeted indexes	0.00	5.00	1.00	1.00	2.00	0.00	0.00	9.00	1.00	1.00	2.00	0.00	**13.00**
	Totals for activity							**0.00**	**51.00**	**89.50**	**16.00**	**12.00**	**21.00**	**189.50**
DB2 9.7	NONE - index compression for DB2 is included as part of the DB2 data compression task flow	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.00**
	Totals for activity							**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**

The derived complexity metric for each Oracle database step is shown graphically below.

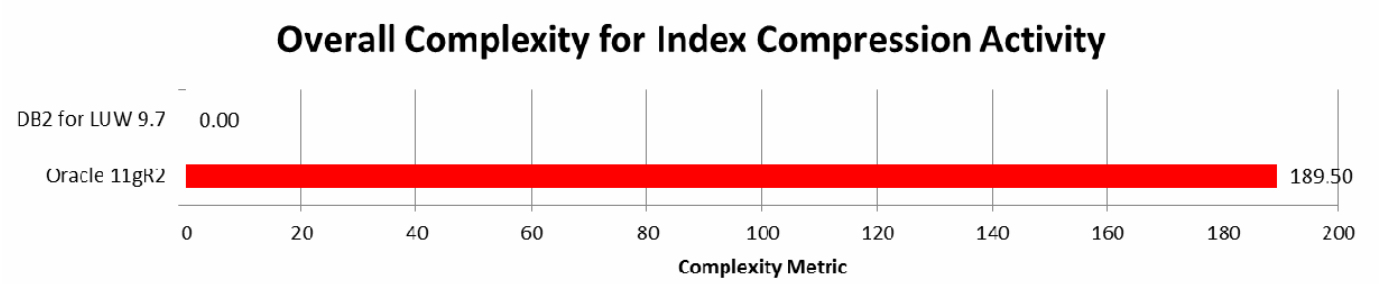


DB2 9.7 does not require the DBA to analyse which tables and indexes would benefit from compression, as is required for Oracle Database 11gR2. Furthermore, Oracle Database also requires separate steps for data and index compression, while DB2 can handle both in a single process. In a large database estate, the time spent in separately assessing and implementing compression for the Oracle Database tables and indexes could be significant.

DB2 9.7 allows automatic index compression using up to three algorithms; variable slot directory, RID (Row ID) list compression, and index prefix compression. DB2 will not compress an index if DB2 detects that the index will not benefit from compression savings. In contrast, Oracle Database only provides index prefix compression and you must setup each index manually and specify the correct number of index columns to include in the prefix compression via index statistics generation and analysis of the system table output. DB2 9.7 provides temporary table compression, data compression, and index compression when compression is enabled at the table level. In addition, index compression can be implemented independently of whether or not table compression is implemented.

Oracle Database allows a multi-column index (1 to N columns for a non-unique index or 1 to N-1 columns for a unique index) or a non-unique single column index to use prefix compression. However, the key point is that Oracle Database bases the unique/non-unique determination on the index definition DDL ONLY – Oracle Database does not factor in the actual index data. Therefore, if you have a single column index or multi-column index in Oracle Database that is defined as non-unique (i.e., not defined using the UNIQUE keyword or the Primary Key syntax) but actually contains unique values then Oracle Database will attempt to compress this index. This will result in a larger index since you have the prefix dictionary and the actual index data with no compression at the conclusion of the index compression operation. Since DB2 provides automatic index compression based on the data values within the index, the potential problems in Oracle Database prefix compression are a non-issue in DB2. DB2 provides variable slot directory and RID list compression for indexes in addition to prefix compression allowing more index compression opportunities than Oracle Database supports.

The chart below shows the total complexity metric for the Oracle Database and DB2 index compression activity. There are no additional tasks required for DB2 index compression, so the DB2 complexity metric is 100% lower than Oracle Database. This difference in complexity can have a significant impact in various aspects of total cost of ownership. Our projections show that the Oracle Database index compression task could take approximately 36 minutes of DBA interaction time to complete. In contrast, the DB2 index compression task requires no time from the DBA because it happens automatically. In addition to these time savings come several additional benefits arising from DB2's lower complexity. This includes DB2 having reduced skill requirements and a reduced risk of errors that can impact quality of service. Another benefit is that when schema changes are implemented then DB2 automatically adjusts index compression whereas with Oracle Database the DBA must carry out the task over again. The impact of such changes is discussed further in the section “Lifecycle Analysis”.



Although the work required to enable index compression in an Oracle Database environment is not particularly complex relative to the other Oracle activities examined in this report, DB2 does not require any additional work once the underlying data is compressed. DB2 therefore has a clear productivity advantage for this activity.

Activity 4: Backup & Recovery

Activity Overview

This task involves creating a backup of a database and then, following an update and load, a restore is performed to restore the database to the point of the backup image. The database consists of four table spaces, which hold several tables within each table space within several sets of containers. The database has also been configured with log archiving.

User roles

A database administrator who is familiar with databases (e.g., general understanding of database utilities, familiar with SQL syntax).

Assumptions

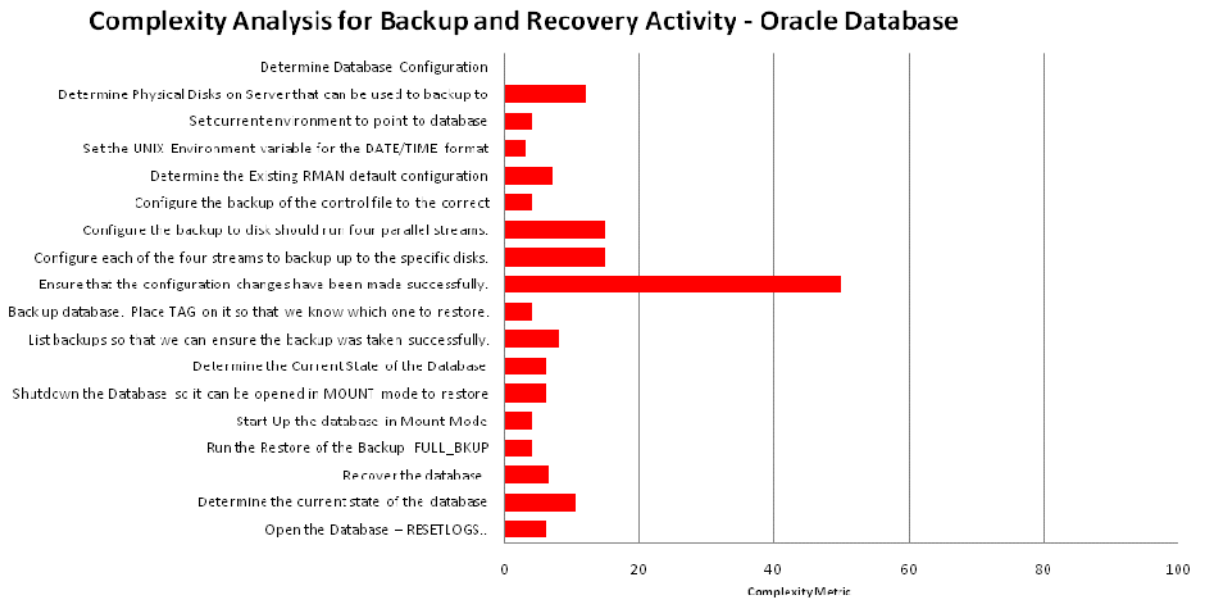
- A database has already been created and populated
- The database consists of the following properties;
    - Database performs log archiving
    - There are 4 tablespaces
    - Database has several tables in each tablespace
    - Database has several files (containers) per tablespace
- The user wants to restore the database back to the last backup point

Activity Analysis

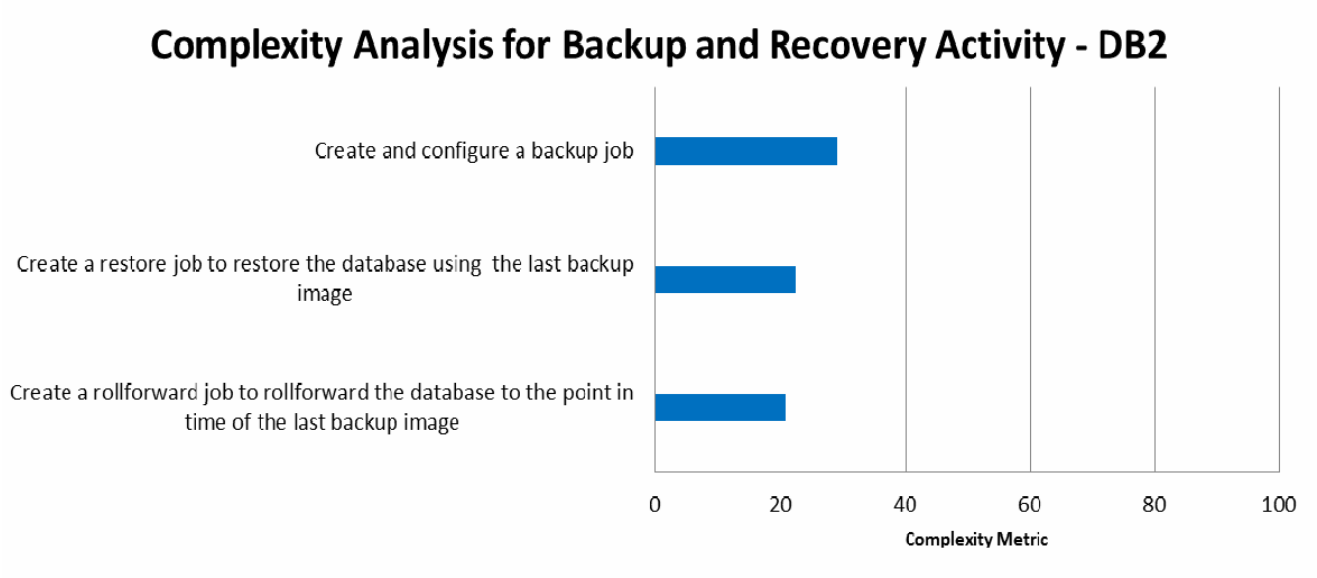
The table below details the complexity ratings and derived complexity metrics associated with each step in the Backup and Recovery activity.

DBMS	Step	Complexity Rating						Complexity Metric						
		Context shifts (0-4 each)	Navigational guidance (1-5 overall)	Input parameters (0-5 each)	System feedback (0-4 overall)	Error feedback (0-5 overall)	New concepts (0-4 overall)	Context shifts	Navigational guidance	Input parameters	System feedback	Error feedback	New concepts	Step totals
Oracle	Determine Database Configuration	0	4	3	1	0	1	0	6	12	1	0	1	**20**
Database	Determine Physical Disks on Server that can be used to backup to	4	4	3	4	2	2	6	6	4	9	2	3	**30**
	Set current environment to point to database	0	4	3	2	0	1	0	6	3	3	0	1	**13**
	Set the UNIX Environment variable for the DATE/TIME format	0	4	3	4	0	1	0	6	7	9	0	1	**23**
	Determine the Existing RMAN default configuration	4	4	3	4	1	2	6	6	4	9	1	3	**29**
	Configure the backup of the control file to the correct	0	4	3	1	1	2	0	6	15	1	1	3	**26**
	Configure the backup to disk	0	4	3	1	1	2	0	6	15	1	1	3	**26**
	Configure each of the four streams to backup up to the specific disks.	0	4	3	1	1	1	0	6	50	1	1	1	**59**
	Ensure that the configuration	0	4	3	4	1	1	0	6	4	9	1	1	**21**
	Back up database. Place TAG on it so that we know which one to restore.	0	4	3	2	1	1	0	6	8	3	1	1	**19**
	List backups so that we can	0	4	3	1	3	1	0	6	6	1	4	1	**18**
	Determine the Current State of the Database	4	4	3	1	1	1	6	6	6	1	1	1	**21**
	Shutdown the Database so it can be opened in MOUNT mode to restore	0	4	3	3	1	1	0	6	4	6	1	1	**18**
	Start-Up the database in Mount	0	4	3	3	1	1	0	6	4	6	1	1	**18**
	Run the Restore of the Backup FULL_BKUP	4	5	3	2	3	1	6	9	6.5	3	4	1	**29.5**
	Recover the database	0	4	3	1	3	1	0	6	10.5	1	4	1	**22.5**
	Determine the current state of the database	4	4	3	1	1	1	6	6	6	1	1	1	**21**
	Open the Database – RESETLOGS..	0	4	3	4	1	1	0	6	8	9	1	1	**25**
	Totals for task							**30**	**111**	**173**	**74**	**25**	**26**	**439**
DB2 9.7	Create and configure a backup job	0	3	3	4	3	1	0	4	11	9	4	1	29
	Create a restore job to restore the database using the last backup image	0	3	3	4	3	1	0	4	4.5	9	4	1	22.5
	Create a rollforward job to rollforward the database to the point in time of the last backup image	0	3	3	4	3	1	0	4	3	9	4	1	21
	Totals for task							**0**	**12**	**18.5**	**27**	**12**	**3**	**72.5**

The derived complexity metric for each Oracle Database step is shown graphically below.



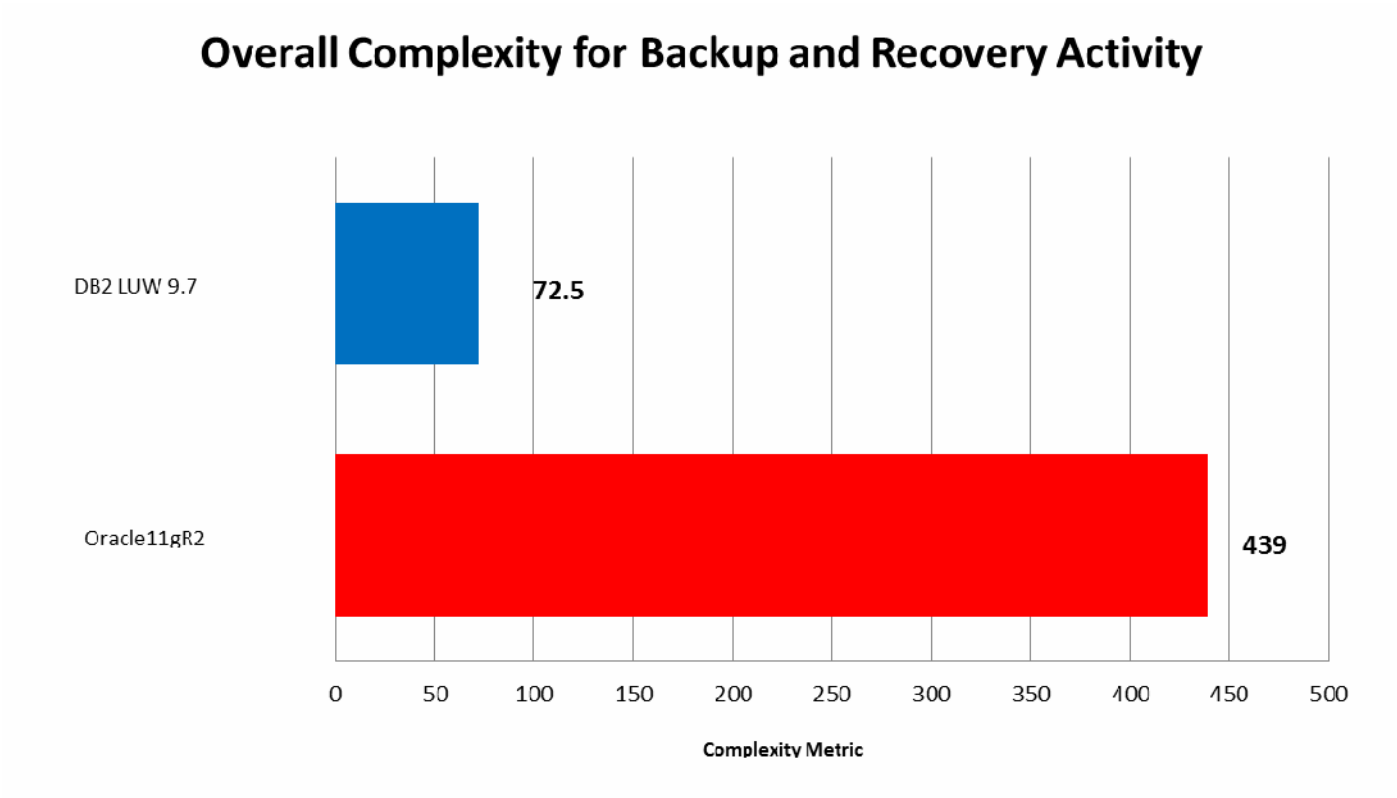
The derived complexity metric for each DB2 step is shown graphically below.



The underlying difference between DB2 9.7 and Oracle Database 11gR2 for this task can be predominantly attributed to the amount of time spent in configuring the environment. With DB2 there is minimal configuration required in order for the backup and restore steps to be completed. However, our findings with Oracle Database show that a significant amount of configuration is required to specify the number of parallel threads and disk locations for the backup. Oracle Database has to rely on recovering its archive redo log files once the database has been restored as the process is not automated within the restore command. Similarly Oracle Database requires the DBA to manually specify a point-in-time for the recovery. Otherwise, the restore will simply work through the archive redo log files and the database will be back to its present state. With DB2 the logs are restored along with the backup image so all that is required is a single additional step to roll forward to the desired recovery point. It is apparent that the additional steps involved with the Oracle Database backup and recovery result in increased complexity and cost.

DB2 backup provides automated configuration for table spaces, backup devices, and parallel I/O capabilities. In contrast Oracle Database backups are complicated in that they require an increased level of manual tuning to specify the number of parallel threads and backup devices.

The chart below compares the overall complexity metrics for Oracle Database and DB2 backup and recovery activities. The complexity metric for DB2 is nearly 83% less complex than for Oracle Database. The complexity metric for Oracle Database would be further increased if the RMAN catalog recovery procedure was used. However, for the activity described in this report the RMAN recovery method is not required.



Based on the results of our analysis, DB2 9.7 backup and recovery is dramatically easier to implement than the equivalent functionality within Oracle Database 11gR2. In our scenario for implementing backup and recovery, the DB2 activity is almost 83% less complex than the Oracle Database equivalent. This difference in complexity can have a significant impact in various aspects of total cost of ownership. Our projections show that the Oracle Database backup and recovery task for the specified environment could take over 80 minutes of DBA interaction time to complete. In contrast, the same backup and recovery task for DB2 would take a little over 10 minutes of DBA interaction time. In addition to these time savings come several additional benefits arising from DB2's lower complexity. This includes DB2 having reduced skill requirements and a reduced risk of errors that can impact quality of service.

Activity 5: Automatic Memory Tuning

Activity Overview

This activity involves configuring the memory-related database parameters for each database platform. This process includes configuration of multiple page size table spaces to mimic tables with larger row lengths, turning on automatic memory, and other tasks specific to DB2 and Oracle Database. Further steps involve creating a larger table than the buffer pool/buffer cache size for each page size, running several queries iteratively, monitoring the buffer pool activity, and observing how DB2 and Oracle Database apply their automation algorithms in adjusting buffer pool sizes.

User Roles

Database administrator who is familiar with databases (e.g., general understanding of database utilities, familiar with SQL syntax).

Assumptions

The task is a continuation of the data compression task and will therefore be configured in the same way. Further assumptions are as follows:

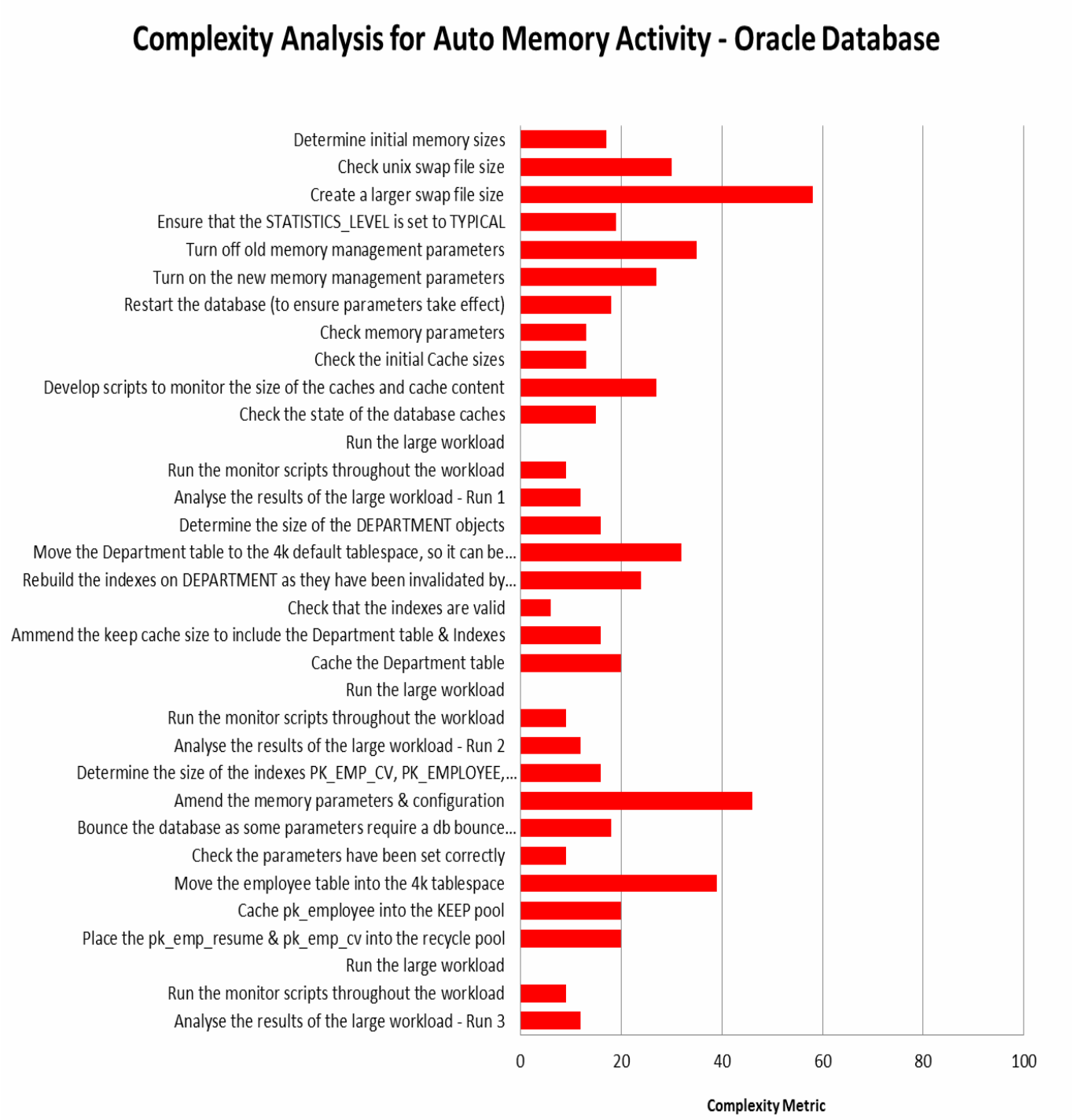
- The database will have 300 tables, with some tables containing large object (LOB) columns consisting of XML documents. There are no other types of LOB columns in the database.
- The task begins with the user already connected to the database
- RedHat Enterprise Linux 5
- Multiple page (block) size (4k and 8k) tablespaces are already in place (to mimic a customer database containing tables with larger row lengths)
- The user has already created 4k (default) and 8k buffer pools
- The automatic sizing feature has been enabled for 4k and 8k buffer pools
- The database tables have already been populated. Assume that at least 2 tables have the following properties
    - Table1: Small table using 4k tablespace for indexing purposes, and an 8k tablespace for data
    - Table2: Large table using 4k tablespace for indexing purposes, and an 8k tablespace for its data
- Workload queries are executed in a mixed sequence so that the queries cause the buffer pool cache to flush
- The initial bufferpool size is about 25% the size of the largest table that will be queried as part of this task

Activity Analysis

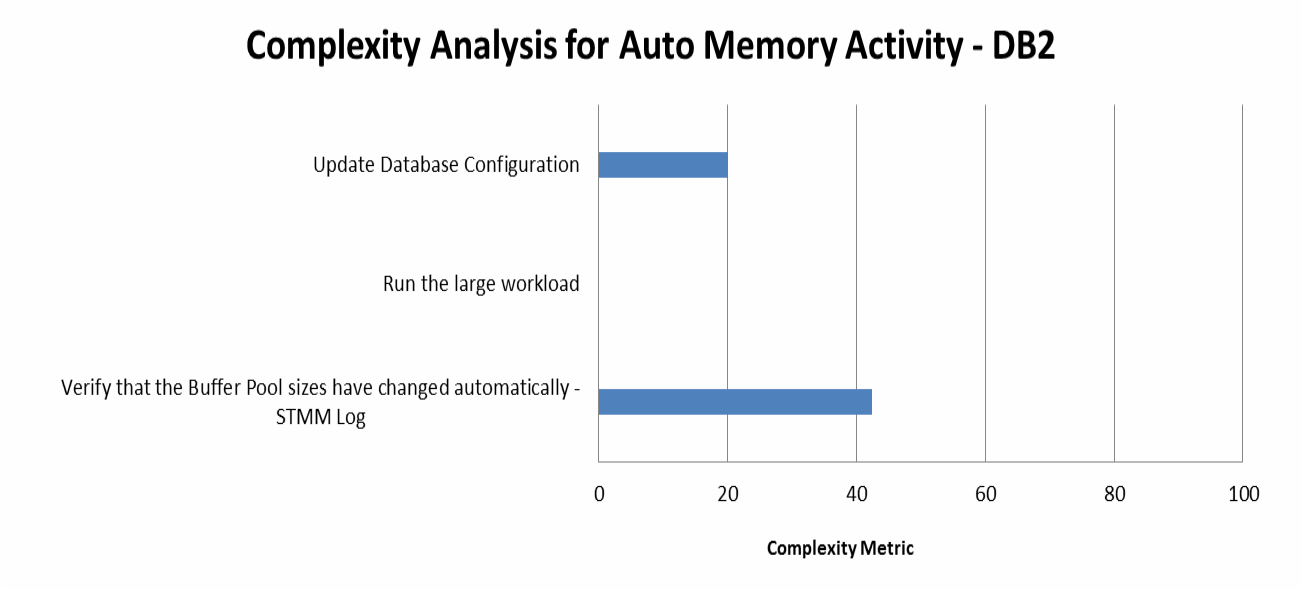
The table below details the complexity ratings and derived complexity metrics associated with each step in the Automatic Memory Tuning activity.

DBMS	Step	Complexity Rating						Complexity Metric						
		Context shifts (0-4 each)	Navigational guidance (1-5 overall)	Input parameters (0-5 each)	System feedback (0-4 overall)	Error feedback (0-5 overall)	New concepts (0-4 overall)	Context shifts	Navigational guidance	Input parameters	System feedback	Error feedback	New concepts	Step totals
Oracle Database 11gR2	Determine initial memory sizes	0	4	3	1	1	1	0	6	8	1	1	1	**17**
	Check unix swap file size	4	4	3	4	2	2	6	6	4	9	2	3	**30**
	Create a larger swap file size	0	5	3	4	5	4	0	9	19	9	9	12	**58**
	Ensure that the STATISTICS_LEVEL is set to TYPICAL	4	4	3	1	1	1	6	6	4	1	1	1	**19**
	Turn off old memory management parameters	0	4	3	2	1	1	0	6	24	3	1	1	**35**
	Turn on the new memory management parameters	0	4	3	2	1	1	0	6	16	3	1	1	**27**
	Restart the database (to ensure parameters take effect)	0	4	3	3	1	1	0	6	4	6	1	1	**18**
	Check memory parameters	0	4	3	1	1	1	0	6	4	1	1	1	**13**
	Check the initial Cache sizes	0	4	3	1	1	1	0	6	4	1	1	1	**13**
	Develop scripts to monitor the size of the caches and cache content	4	5	0	0	0	4	6	9	0	0	0	12	**27**
	Check the state of the database caches	4	3	1	1	2	1	6	4	1	1	2	1	**15**
	Run the large workload	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	Run the monitor scripts throughout the workload	0	3	1	1	2	1	0	4	1	1	2	1	**9**
	Analyse the results of the large workload - Run 1	4	0	0	0	0	3	6	0	0	0	0	6	**12**
	Determine the size of the DEPARTMENT objects	4	4	1	1	1	1	6	6	1	1	1	1	**16**
	Move the Department table to the 4k default tablespace, so it can be cached	0	4	3	4	3	2	0	6	10	9	4	3	**32**
	Rebuild the indexes on DEPARTMENT as they have been invalidated by the move	0	4	3	4	1	1	0	6	7	9	1	1	**24**
	Check that the indexes are valid	0	0	1	1	1	1	0	0	1	3	1	1	**6**
	Ammend the keep cache size to include the Department table & Indexes	0	4	3	2	1	1	0	6	7	1	1	1	**16**
	Cache the Department table	0	4	3	1	1	2	0	6	9	1	1	3	**20**
	Run the large workload	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	Run the monitor scripts throughout the workload	0	3	1	1	2	1	0	4	1	1	2	1	**9**
	Analyse the results of the large workload - Run 2	4	0	0	0	0	3	6	0	0	0	0	6	**12**
	Determine the size of the indexes PK_EMP_CV, PK_EMPLOYEE, PK_EMP_RESUME	4	4	1	1	1	1	6	6	1	1	1	1	**16**
	Amend the memory parameters & configuration	0	4	3	2	1	1	0	6	35	3	1	1	**46**
	Bounce the database as some parameters require a db bounce (db_8k_cache_size)	0	4	3	3	1	1	0	6	4	6	1	1	**18**
	Check the parameters have been set correctly	0	4	0	1	1	1	0	6	0	1	1	1	**9**
	Move the employee table into the 4k tablespace	0	4	3	4	3	2	0	6	17	9	4	3	**39**
	Cache pk_employee into the KEEP pool	0	4	3	1	1	2	0	6	9	1	1	3	**20**
	Place the pk_emp_resume & pk_emp_cv into the recycle pool	0	4	3	1	1	2	0	6	9	1	1	3	**20**
	Run the large workload	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	Run the monitor scripts throughout the workload	0	3	1	1	2	1	0	4	1	1	2	1	**9**
	Analyse the results of the large workload - Run 3	4	0	0	0	0	3	6	0	0	0	0	6	**12**
	Totals for task							**54**	**154**	**201**	**84**	**45**	**79**	**617**
DB2 9.7	Update Database Configuration	0	4	3	1	3	2	0	6	6	1	4	3	**20**
	Run the large workload	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	Verify that the Buffer Pool sizes have	4	4	3	3	3	4	6	6	8.5	6	4	12	**42.5**
	Totals for task							**6**	**12**	**14.5**	**7**	**8**	**15**	**62.5**

The derived complexity metric for each Oracle Database step is shown graphically below.



The derived complexity metric for each DB2 step is shown graphically below.

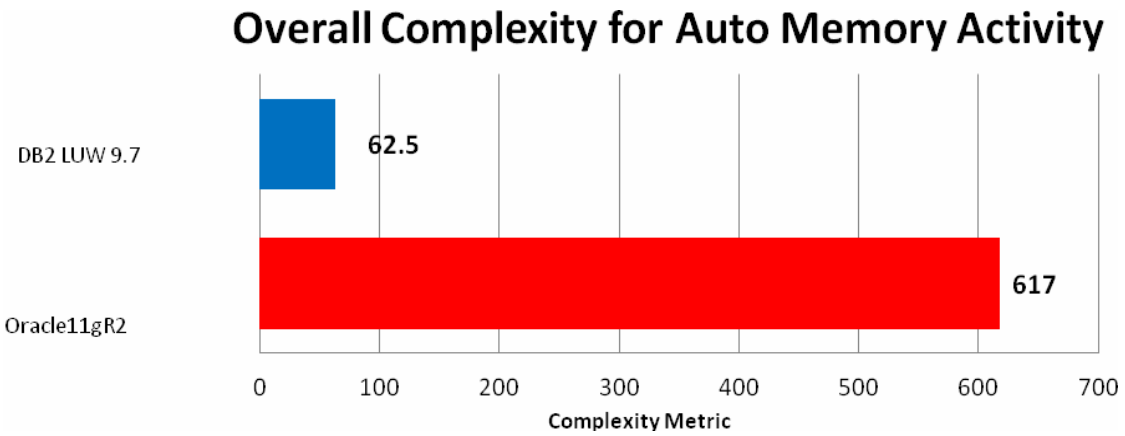


DB2 9.7 provides a major advantage that is orders of magnitude better than it's Oracle Database counterpart where automated memory management is concerned. In today's systems, the workloads placed on a database server have increased significantly and the database server has to effectively manage these workloads as they change over time. Management of buffer pools and other memory areas is a key aspect of this workload management.

DB2 has a Self Tuning Memory Manager (STMM) component, which can be enabled online and is relatively simple to configure (using a two-step process if the default parameters are used). In comparison, undertaking memory tuning in an Oracle Database environment involves multiple operating system checks and several memory parameters that require a full database restart. During our testing we also encountered issues with Oracle Database configuration parameters being reset after tuning.

Overall the restrictive nature of the Oracle Database memory management compared to DB2 9.7 STMM is the result of Oracle Database's inability to be fully automated, with restrictions in both automatic sizing and in buffer cache management. This inevitably leads to manual intervention on the DBA's part when confronted with common tasks such as keeping index pages in the buffer cache in an attempt to prevent a table scan from flushing index pages. Interestingly, Oracle Database is only able to increase memory up to its configured setting, but is unable to lower memory settings based on the workloads. In contrast to Oracle Database, DB2 memory is adjusted by DB2 9.7 STMM as workload requirements change (either an increase or a decrease). These memory adjustments are clearly visible in the DB2DIAG and the STMM logs.

The chart below compares the overall complexity metrics for the Oracle Database and DB2 automatic memory activities. The complexity metric for DB2 is nearly 90% lower than the complexity metric for Oracle Database. This difference in complexity can have a significant impact in various aspects of total cost of ownership. Our projections show that the Oracle Database automatic memory management task for the specified environment could take over 100 minutes of DBA interaction time to complete. In contrast, the same automatic memory management task for DB2 would take a little over 10 minutes. In addition to these time savings come several additional benefits arising from DB2's lower complexity. This includes DB2 having reduced skill requirements, a reduced risk of errors that can impact quality of service, and a higher degree of agility in response to evolving business requirements. (The latter point is discussed in more detail in the "Lifecycle Analysis" section.)



Based on the results of our analysis, we have found DB2 9.7 memory management to be both more effective and significantly simpler to implement than the equivalent activity for Oracle Database 11gR2.

Activity 6: Data Access Control

Activity Overview

A customer has a database and recent compliance changes now require all data access to be removed from individuals with DBA privileges. Due to budgetary constraints, this requirement must be met using base database functionality – no additional product purchases are permitted.

User Roles

Database administrator who is familiar with databases (e.g., general understanding of database utilities, familiar with SQL syntax).

Assumptions

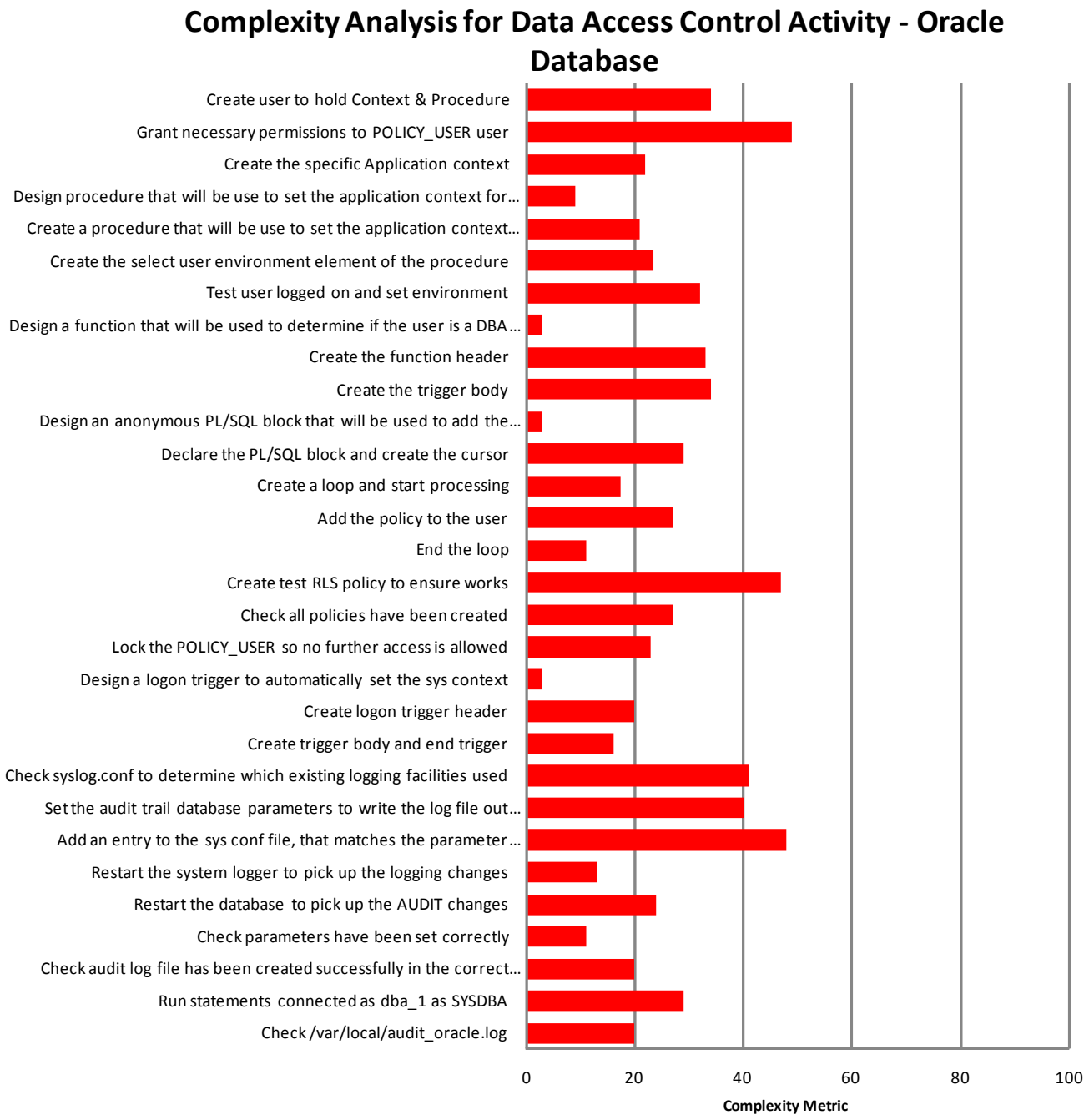
- The user has a level of SECADM authority in DB2 9.7, which is not an available user in Oracle Database
- An existing database that has already been created will be used
- The system has 7 database administrators from whom privileges will be removed
- The operating system is RedHat Enterprise Linux 5

Activity Analysis

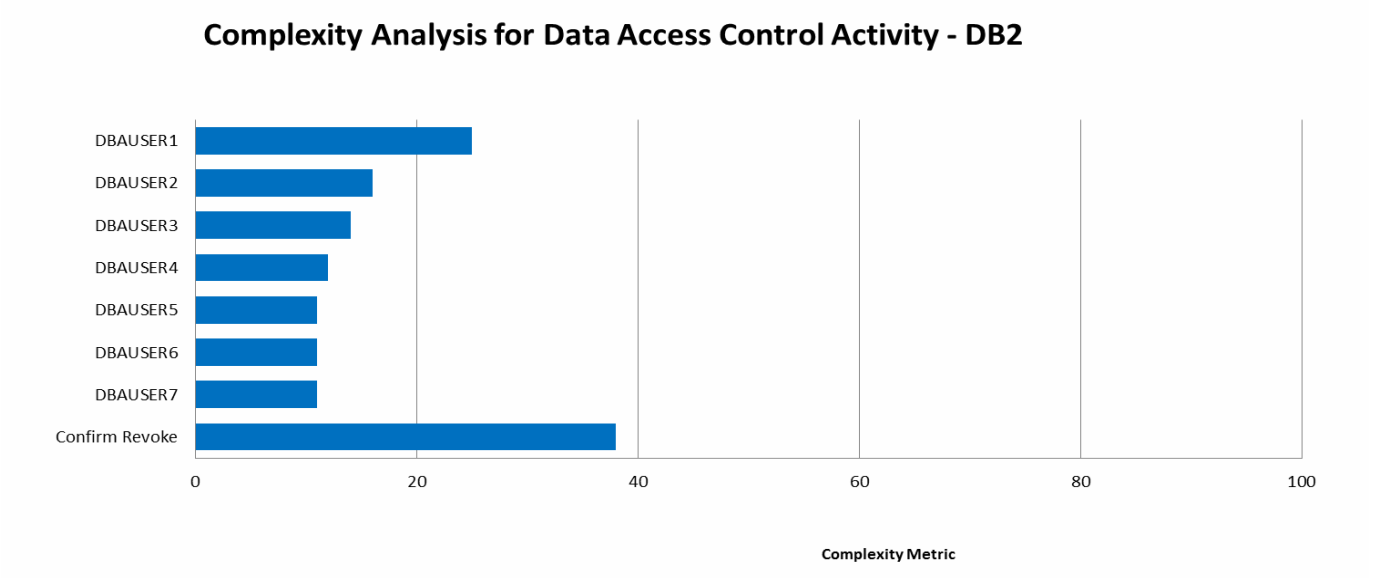
The table below details the complexity ratings and derived complexity metrics associated with each step in the Data Access Control activity.

DBMS	Step	Complexity Rating						Complexity Metric						
		Context shifts (0-4 each)	Navigational guidance (1-5 overall)	Input parameters (0-5 each)	System feedback (0-4 overall)	Error feedback (0-5 overall)	New concepts (0-4 overall)	Context shifts	Navigational guidance	Input parameters	System feedback	Error feedback	New concepts	Step totals
Oracle Database 11gR2	Create user to hold Context & Procedure	0	4	3	1	4	1	0	6	20	1	6	1	**34**
	Grant necessary permissions to POLICY_USER user	0	4	3	1	1	2	0	6	38	1	1	3	**49**
	Create the specific Application context	2	3	3	1	1	2	2	4	11	1	1	3	**22**
	Design procedure that will be use to set the application context for any DBA's that login to the database	4	0	0	0	0	2	6	0	0	0	0	3	**9**
	Create a procedure that will be use to set the application context for any DBA's that login to the database	0	3	3	1	1	1	0	4	14	1	1	1	**21**
	Create the select user environment element of the procedure	0	3	3	1	1	1	0	4	16.5	1	1	1	**23.5**
	Test user logged on and set environment	0	3	3	1	1	1	0	4	25	1	1	1	**32**
	Design a function that will be used to determine if the user is a DBA then add the predicate 1=0 to the where clause which will always return 0 records	2	0	0	0	0	1	2	0	0	0	0	1	**3**
	Create the function header	0	3	3	1	1	1	0	4	26	1	1	1	**33**
	Create the trigger body	0	3	3	1	1	1	0	4	27	1	1	1	**34**
	Design an anonymous PL/SQL block that will be used to add the policys to the user	2	0	0	0	0	1	2	0	0	0	0	1	**3**
	Declare the PL/SQL block and create the cursor	0	3	3	1	1	1	0	4	22	1	1	1	**29**
	Create a loop and start processing	0	3	3	1	1	1	0	4	10.5	1	1	1	**17.5**
	Add the policy to the user	0	3	3	1	1	1	0	4	20	1	1	1	**27**
	End the loop	0	3	3	1	1	1	0	4	4	1	1	1	**11**
	Create test RLS policy to ensure works	0	3	3	2	4	4	0	4	22	3	6	12	**47**
	Check all policies have been created	2	4	3	1	1	2	2	6	14	1	1	3	**27**
	Lock the POLICY_USER so no further access is allowed	2	3	3	1	1	1	2	4	9	3	4	1	**23**
	Design a logon trigger to automatically set the sys context	2	0	0	0	0	1	2	0	0	0	0	1	**3**
	Create logon trigger header	0	3	3	1	1	1	0	4	13	1	1	1	**20**
	Create trigger body and end trigger	0	3	3	1	1	1	0	4	9	1	1	1	**16**
	Check syslog.conf to determine which existing logging facilities used	4	5	3	4	0	4	6	9	5	9	0	12	**41**
	Set the audit trail database parameters to write the log file out using the o/s syslog functions	4	4	3	2	1	2	6	6	21	3	1	3	**40**
	Add an entry to the sys conf file, that matches the parameter audit_syslog_level	4	4	4	4	5	4	6	6	6	9	9	12	**48**
	Restart the system logger to pick up the logging changes	0	3	3	1	1	2	0	4	4	1	1	3	**13**
	Restart the database to pick up the AUDIT changes	4	4	3	3	1	1	6	6	4	6	1	1	**24**
	Check parameters have been set correctly	0	4	3	1	1	1	0	6	2	1	1	1	**11**
	Check audit log file has been created successfully in the correct location	4	4	3	1	1	1	6	6	5	1	1	1	**20**
	Run statements connected as dba_1 as SYSDBA	4	3	3	1	1	1	6	4	16	1	1	1	**29**
	Check /var/local/audit_oracle.log	4	4	3	1	1	1	6	6	5	1	1	1	**20**
	Totals for task							**60**	**127**	**369**	**53**	**46**	**75**	**730**
DB2 for LUW 9.7	DBAUSER1	0	4	3	1	3	2	0	6	11	1	4	3	**25**
	DBAUSER2	0	4	1	1	3	0	0	6	5	1	4	0	**16**
	DBAUSER3	0	3	1	1	3	0	0	4	5	1	4	0	**14**
	DBAUSER4	0	2	1	1	3	0	0	2	5	1	4	0	**12**
	DBAUSER5	0	1	1	1	3	0	0	1	5	1	4	0	**11**
	DBAUSER6	0	1	1	1	3	0	0	1	5	1	4	0	**11**
	DBAUSER7	0	1	1	1	3	0	0	1	5	1	4	0	**11**
	Confirm Revoke	0	5	3	1	3	3	0	9	18	1	4	6	**38**
	Totals for task							**0**	**30**	**59**	**8**	**32**	**9**	**138**

The derived complexity metric for each Oracle Database step is shown graphically below.

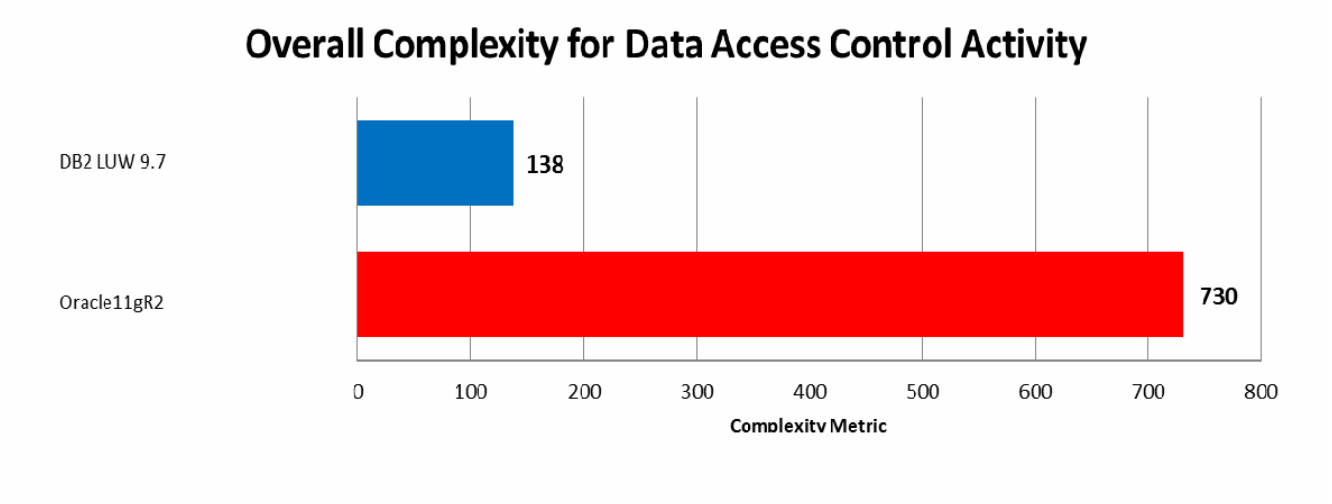


The derived complexity metric for each DB2 step is shown graphically below.



DB2 9.7 allows for the revoking of privileges from users with one command per user, and the overall complexity is minimal in comparison to Oracle Database. DB2 completes the task at the database level whereas its Oracle Database counterpart requires the implementation of a virtual private database to control the data access. This is further complicated with the number of steps required to complete the process, which involves the creation of triggers, Oracle Database re-writing SQL statements and adding a policy to each table. The virtual private database is only available with Oracle Database Enterprise Edition while DB2 provides a single command which is available throughout all DB2 for LUW product editions. Another significant difference between DB2 and Oracle Database is the separation of duties. DB2 uses a specific user (SECADM), who is the security administrator. Oracle Database does not have an equivalent capability.

The chart below compares the overall complexity metrics for the Oracle Database and DB2 data access control activities. The complexity metric for DB2 is nearly 81% lower than the complexity metric for Oracle Database. This difference in complexity can have a significant impact in various aspects of total cost of ownership. Our projections show that the Oracle Database data access control task for the specified environment could take over 130 minutes of DBA interaction time to complete. In contrast, the same data access control task for DB2 would take a little over 25 minutes. In addition to these time savings come several additional benefits arising from DB2's lower complexity. This includes DB2 having reduced skill requirements, a reduced risk of errors that can impact quality of service, and a higher degree of agility in response to evolving business requirements. (The latter point is discussed in more detail in the “Lifecycle Analysis” section.)



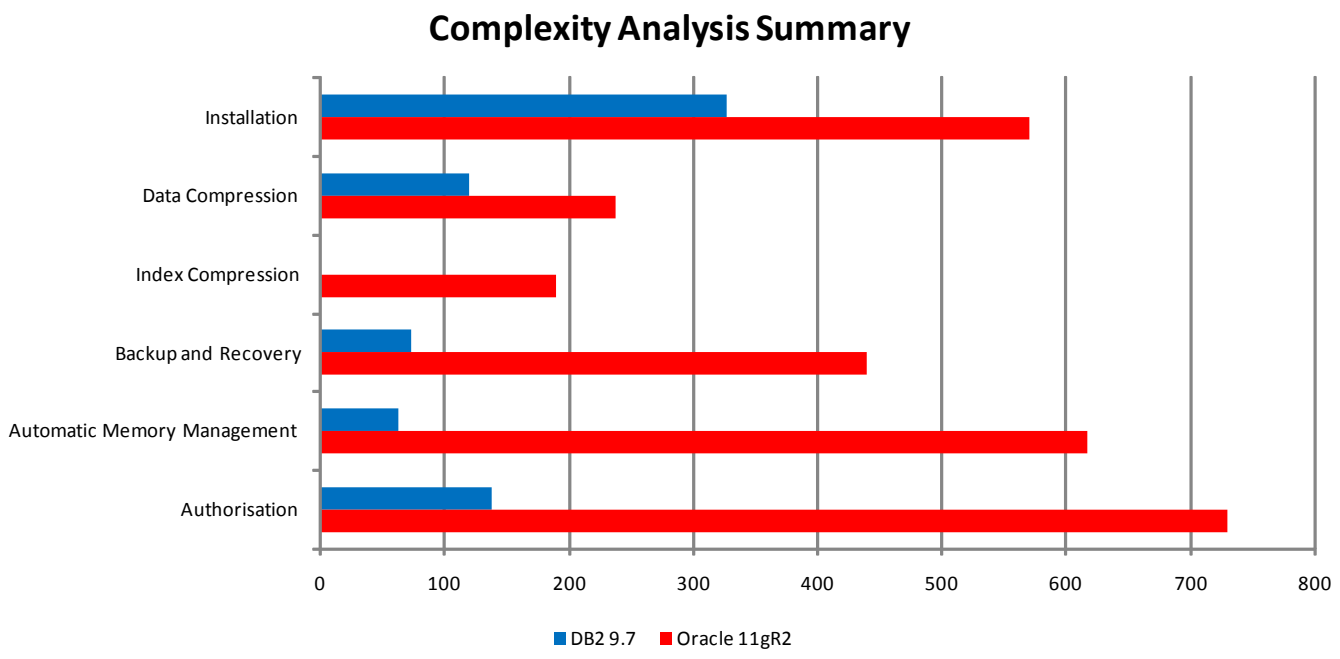
Based on the results of our complexity analysis, we found the data access control activity to be dramatically easier to implement in DB2 V9.7 than the equivalent functionality within Oracle Database 11gR2.

Summary and Conclusions

This study examined six common DBA activities, and assessed the complexity of each within an Oracle Database 11gR2 and DB2 9.7 environment. As shown in the table below, DB2 holds a significant advantage in every single category. Of these, the most significant advantage is shown in the index compression task, but DB2 also has a clear advantage within the installation, data compression, backup and recovery, automatic memory management, and data access control activities.

Task	*Complexity Metric*		*DB2 Advantage*
	DB2 9.7	Oracle Database 11gR2	
Installation	326.5	570.5	**43%**
Data Compression	119.50	238	**50%**
Index Compression	0	189.50	**100%**
Backup and Recovery	72.5	439	**83%**
Automatic Memory Management	62.5	617	**90%**
Data Access Control	138	730	**81%**

The analysis results are also shown graphically below:



A detailed breakdown of the individual task complexity scores for each activity is included in the “Complexity Analysis” section.

Based on the results of our study, DB2 has a clear and overwhelming advantage over all six of the routine DBA activities we evaluated. This DB2 complexity advantage translates into higher DBA productivity for these tasks relative to Oracle Database through factors such as:

- Less DBA time spent on researching and performing these tasks (projections for time savings for these tasks are included in the “Complexity Analysis” section).
- Less time spent in training new staff.

- Lower risk of errors that can impact quality of service.

It is worth noting that lower complexity is directly correlated to interaction time[2] savings. For example, a DBA would need approximately half the interaction time to implement data compression with DB2 9.7 relative to Oracle Database 11gR2 given that DB2 9.7 is approximately 50% less complex than Oracle Database 11gR2 for the data compression activity. Similarly, the complexity advantage for DB2 for automatic memory management means that this activity would require approximately 90% less interaction time with DB2 9.7 in comparison to Oracle Database 11gR2. Therefore, this analysis demonstrates tangible TCO benefits for these common DBA activities.

Furthermore, we have also examined these tasks from a lifecycle analysis perspective. For example, after initial setup of table and index compression it may be necessary – over time – to revisit the compression tasks in response to evolving business requirements that result in schema changes. Through this lifecycle analysis we have found DB2's advantage of lower complexity is propagated beyond the initial setup and configuration stages. Many of the complexity advantages of DB2 cited in this report are part of recurring lifecycle events that add up to a considerable DB2 advantage over time. This analysis is described in more detail in the next section – "Lifecycle Analysis".

[2] Interaction time is the dedicated time spent by the user in actively working with a system to perform a task. It includes the time spent on work activities such as executing commands, writing scripts, and interpreting system output. Interaction time does not include system response time, as in the case where a utility takes a long time to execute and during which the user can be attending to other work. Interaction time also does not account for work interruptions – it assumes the user works in an uninterrupted manner.

Lifecycle Analysis

Introduction

The complexity analyses described in the preceding sections cover the activities:

- Initial installation
- Setup of data and index compression
- Implementation of a backup / recovery strategy
- Configuration of memory
- Changing data access control.

With the exception of the last activity (changing data access control), these activities are focused on up-front setup and configuration. After these up-front activities are completed organizations will typically deal with various changes driven by evolving business requirements. Therefore, we have also looked beyond the up-front setup and configuration stage, examining downstream stages of the application lifecycle. This section describes our findings for these downstream stages.

Memory management

Once DB2's self-tuning memory manager (STMM) is turned on then it will provide complete automated memory management on an ongoing basis. In contrast, Oracle Database Automatic Memory Management (AMM) only partially automates memory management due to the restrictions described earlier. For example, Oracle Database AMM can only manage the buffer cache memory for the default block size buffer cache and not other block size buffer caches. As a result of these restrictions, Oracle Database requires -- on an ongoing basis – manual tuning and database administrator intervention to deal with evolving business requirements.

One example of how evolving business requirements would result in more complexity for Oracle Database is in the case of introducing a new application or changing an existing application. Such changes would be automatically handled by DB2 STMM whereas with Oracle Database such changes would require the database administrator to once again monitor workloads, analyze the results, and implement memory configuration changes. Several steps of the Oracle Database memory management activity described earlier would need to be implemented each time such a change occurs, whereas DB2's automation would eliminate the need for these memory configuration steps. We project that these steps would require at least 21 minutes of uninterrupted interaction time for a database administrator already familiar with Oracle Database given a scenario comparable to the one we described in section “Activity 5: Automatic Memory Tuning”. In contrast, with DB2 there would be no time required of the database administrator to tune memory. Therefore, each time such a change would be introduced Oracle Database would require at least an extra 21 minutes of database administrator time in comparison to DB2. In addition, the manual nature of adapting Oracle Database to such changes not only increases the time demands on the database administrator but also increases the risk of errors which can impact quality of service.

Data compression

In the case of data compression a key factor that emerges in the downstream stages of the application lifecycle is data growth. Operational databases typically grow over time and therefore data growth needs to be considered in a complete lifecycle analysis of data compression.

The initial disk-storage-savings ratios achieved through data compression may degrade over time depending on the type of compression implemented by the database. This degradation is particularly relevant in the case of Oracle Database block-based compression data compression. The Oracle Database block-based compression algorithm requires a table's initial data to be loaded in sorted order by the least unique column values in order to achieve maximum compression ratios. Over time as data is inserted and updated within a table, the data being written to the data blocks is more random in nature (i.e., not grouped by column values). Therefore a block-based compression algorithm will result in the overall compression ratio for a table degrading over time. To achieve maximum data compression, an Oracle DBA would periodically need to unload a table, sort the data by the designated column value(s), and then reload the table. This results in increased database maintenance tasks when comparing DB2 with Oracle Database from an administration viewpoint. DB2's table-level compression algorithm does not require any pre-sorting of the data prior to the initial load operation and adapts to the random distribution pattern of data growth over time. Each time the Oracle Database DBA would need to unload a table, sort the data, and then reload the table we

project that it would easily take 42 minutes given a scenario similar to the one described earlier for Oracle Database data compression, and this is time and effort that would not be imposed on the DB2 DBA.

Index compression

As described earlier in the section “Activity 3: Index Compression”, DB2 index compression is automatically implemented at no additional cost of database administrator time or effort once data compression is implemented. In contrast, Oracle requires a significant amount of additional time and effort from the database administrator to implement index compression. Over time, this fundamental difference in simplicity between DB2 and Oracle Database index compression will surface again each time changes are introduced into the operational environment. Changes to an application may require the Oracle database administrator to re-analyze the indexes, determine which indexes will deliver acceptable compression savings, and then implement the new index compression scheme. Examples of changes that would necessitate this index compression rework in Oracle Database are application changes that involve adding columns and tables, and performance tuning that results in adding new indexes. In contrast, DB2 can undergo all such changes without requiring anything of the database administrator to maintain the effectiveness of index compression.

Based on our evaluation of Oracle Database index compression we project that changes requiring a re-analysis and re-implementation of index compression can take approximately 36 minutes of uninterrupted interaction time for a database administrator already familiar with Oracle Database. In contrast, with DB2 there would be no time required of the database administrator to adapt index compression to the changes – DB2 addresses these changes automatically. In addition to the extra time burden of Oracle Database index compression on database administrators, the manual nature of adapting Oracle Database to such changes also increases the risk of errors that can impact quality of service. Selecting the wrong index to compress in Oracle Database can result in increasing the size of the index.

From an administration viewpoint, DB2 integrates data and index compression, so that if a table is using data compression, index compression is automatic. Within Oracle Database, Advanced Compression (data level) and index compression are two distinctly different items. Oracle Database might claim their index compression is available at no charge in Enterprise Edition, but the old adage "you get what you pay for" certainly applies in this case.

Data access control

The data access control activity described earlier involves revoking all data access from individuals with DBA authority. This activity is an example of what an organization would need to carry out as a result of evolving business requirements after the database is set up and configured for initial operations. As described earlier, Oracle database would be several times more complex and time-consuming for this activity than DB2 because Oracle Database does not provide a separation of duties authorization class equivalent to DB2’s SECADM authorization unless the Oracle Database Vault option (for Enterprise Edition) is purchased. As a result, the base level of Oracle Database requires implementation of the Virtual Private Database (VPD) to provide data access control to prevent DBA-privileged individuals from accessing data, whereas DB2 can accomplish the same goal through a simple set of revoke commands. And even if Oracle Database Vault is purchased then an additional set of installation and configuration steps on separate servers is imposed on the organization.

Changes in security policies and new business requirements can result in significantly more complexity – and administration time – with Oracle Database relative to DB2. We project that changes such as those described in the section “Activity 6: Data Access Control” can take approximately 100 minutes more to implement with Oracle Database relative to DB2. And with the additional complexity of Oracle Database comes the additional risk of errors which, in the case of security administration, increases the risk of security holes and non-compliance with policies.