Final Report

HP: Big Data Analytics
Group 13

Oregon State University
CS 461
2016-2017

Prepared By:
Nic Desilets and Nathaniel Whitlock

June 13, 2017

Abstract

The following document is a summary of our team documentation including the agreed upon deliverables to our client and project outline. Our collective blog posts from all three terms and reflection questions have been included. A research workflow has been documented and is discussed in chronological order based on the steps we took on the way to completion. Additionally, we have included appendices which include essential code that could be used to recreate this project. Both initial experiment design and refactored versions are included to give perspective of code changes. Finally, we conclude the document with slides that cover the results of each of our experiments.

CONTENTS

1 Introduction to the Project 7

2 Software Requirements Specifications 7

2.1 Introduction . . . 7

2.1.1 Purpose . . . 7

2.1.2 Scope . . . 7

2.1.3 Definitions, Acronyms, Abbreviations . . . 8

2.1.4 References . . . 8

2.1.5 Overview . . . 8

2.2 Overall Description . . . 8

2.2.1 Product Perspective . . . 8

2.2.2 Product Functions . . . 10

2.2.3 User Characteristics . . . 10

2.2.4 Constraints . . . 10

2.2.5 Assumptions and Dependencies . . . 10

2.2.6 Apportioning of Requirements . . . 11

2.2.7 Toolkit Environment . . . 11

2.3 Specific Requirements . . . 11

2.3.1 External Interfaces . . . 11

2.3.2 Functions . . . 11

2.3.3 Performance Requirements . . . 11

2.4 Logical Database Requirements . . . 11

2.4.1 Design Constraints . . . 11

2.4.2 Standards Compliance . . . 12

2.5 Software System Attributes . . . 12

2.5.1 Reliability . . . 12

2.5.2 Security . . . 12

2.5.3 Maintainability . . . 12
2.5.4 Portability . . . 12
2.6 Specific Requirements . . . 12
2.6.1 External Interface Requirements . . . 12
2.6.2 Functional Requirements . . . 13
2.6.3 Performance Requirements . . . 13
2.6.4 Other Requirements . . . 14

3 Changes in Requirement Document 14

4 Design Document 15

4.1 Introduction . . . 15
4.2 Glossary . . . 15
4.2.1 Definitions, Acronyms, Abbreviations . . . 15
4.3 Context Viewpoint . . . 15
4.3.1 Design Concern . . . 15
4.3.2 Design Elements . . . 16
4.4 Composition Viewpoint . . . 16
4.4.1 Design Concerns . . . 16
4.4.2 Design Elements . . . 16
4.4.3 Function Attribute . . . 16
4.5 Dependency Viewpoint . . . 16
4.5.1 Design Concerns . . . 16
4.5.2 Design Elements . . . 16
4.6 Information Viewpoint . . . 17
4.6.1 Design Concerns . . . 17
4.6.2 Design Elements . . . 17
4.7 Interface Viewpoint . . . 17
4.7.1 Design Concerns . . . 17

4.7.2 Design Elements . . . 17
4.7.3 Examples . . . 18
4.7.4 Example Languages . . . 18
4.8 Structure Viewpoint . . . 18
4.8.1 Design Concerns . . . 18
4.8.2 Design Elements . . . 18
4.9 Interaction Viewpoint . . . 18
4.9.1 Design Concerns . . . 18
4.9.2 Design Elements . . . 19
4.9.3 Examples . . . 19
4.10 Resource Viewpoint . . . 19
4.10.1 Design Concerns . . . 19
4.10.2 Design Elements . . . 20
4.10.3 Examples . . . 20
4.11 Conclusion . . . 20

5 Design Review Discussion 21

6 Tech Review 22

6.1 Introduction . . . 22
6.2 SQL Development Environment . . . 22
6.2.1 Purpose of a SQL Development Environment . . . 22
6.2.2 Existing Technologies . . . 22
6.2.3 Conclusion . . . 23
6.3 Parallelism and Partitioning . . . 23
6.3.1 Purpose of Parallelism and Partitioning . . . 23
6.3.2 Database Technologies . . . 24
6.3.3 Conclusion . . . 26
6.4 Toolkits . . . 26

6.4.1 Purpose of a Toolkit . . . 26
6.4.2 Existing Toolkits . . . 26
6.4.3 Creating a Toolkit . . . 27
6.4.4 Conclusion . . . 27
6.5 Summary . . . 28

7 Discussion of Tech Review Changes 28

8 Blog Posts 29
8.1 Organization . . . 29
8.2 Fall Term . . . 29
8.2.1 Week Three . . . 29
8.2.2 Week Four . . . 29
8.2.3 Week Five . . . 30
8.2.4 Week Six . . . 30
8.2.5 Week Seven . . . 31
8.2.6 Week Eight . . . 31
8.2.7 Week Nine . . . 32
8.2.8 Week Ten . . . 33
8.3 Winter Break . . . 33
8.4 Winter Term . . . 35
8.4.1 Week One . . . 35
8.4.2 Week Two . . . 36
8.4.3 Week Three . . . 37
8.4.4 Week Four . . . 37
8.4.5 Week Five . . . 38
8.4.6 Week Six . . . 39
8.4.7 Week Seven . . . 39
8.4.8 Week Eight . . . 40

8.4.9 Week Nine . . . 41
8.4.10 Week Ten . . . 41
8.5 Spring Term . . . 42
8.5.1 Week One . . . 42
8.5.2 Week Two . . . 42
8.5.3 Week Three . . . 43
8.5.4 Week Four . . . 44
8.5.5 Week Five . . . 44
8.5.6 Week Six . . . 45
8.5.7 Week Seven . . . 45
8.5.8 Week Eight . . . 46

9 Poster 48

10 Research Draft 50

10.1 Preparation . . . 50
10.2 Initial Learnings . . . 50
10.3 Divide and Conquer . . . 50
10.4 Experimental Topics . . . 51
10.4.1 Parallel Execution and PGA . . . 51
10.4.2 SGA . . . 51
10.4.3 Statement Queuing . . . 52
10.5 Experimentation . . . 53

11 Learning New Technology 54

11.1 List of Helpful Websites . . . 54
11.2 List of Publications Utilized During Research . . . 54

12 Learnings 55

12.1 Nathaniel . . . 55
12.1.1 What technical information did you learn? . . . 55

12.1.2 What non-technical information did you learn? . . . 55
12.1.3 What have you learned about project work? . . . 55
12.1.4 What have you learned about project management? . . . 55
12.1.5 What have you learned about working in teams? . . . 56
12.1.6 If you could do it all over, what would you do differently? . . . 56
12.2 Nic . . . 56
12.2.1 What technical information did you learn? . . . 56
12.2.2 What non-technical information did you learn? . . . 56
12.2.3 What have you learned about project work? . . . 57
12.2.4 What have you learned about project management? . . . 57
12.2.5 What have you learned about working in teams? . . . 57
12.2.6 If you could do it all over, what would you do differently? . . . 57

13 Appendix 1: Essential Code Listings 58

13.1 Test Automation Suite . . . 58
13.2 Parameter Files . . . 61
13.3 Experiment Files . . . 61
13.4 Experimental Design Modifications . . . 63

14 Appendix 2: Results from Experiments 67

References 72

1 Introduction to the Project

This project was requested by the PageWide Web Press (PWP) team at HP in Corvallis. The primary objectives of this project were to identify bottlenecks in their configuration of Oracle database and also to improve the performance of their database instance. Engineers within the PWP division want reports on the health and status of every printer sold around the world, however due to poor database performance these reports would take several hours to generate. Our clients were Andy Weiss, the database administrator, and Kirby Sand, the lead data analyst, who both work in the PWP division. Both clients assisted us in our project throughout the term. Our capstone team consists of Nic Desilets and Nathaniel Whitlock, both of us split responsibility and worked together to complete this project.

2 Software Requirements Specifications

Abstract

This document details the specific requirement agreed upon between our team and our clients. The constraints placed on the project are discussed in detail and the specifications laid out by the clients have be documented. This serves as contract between our our team and client.

2.1 Introduction

2.1.1 Purpose

The purpose of this document is to define the primary objective and details of the toolkit. This document will contain guidelines that describe the purpose, features, general design, interactions with Oracle 12c database, and the constraints that the performance analysis toolkit must operate within. The target audience for this document includes any relevant stakeholders as well as any Database Administrator (DBA) or software developer that works with Oracle database 12c in any capacity.

2.1.2 Scope

The main purpose of this toolkit is primarily to assist with collecting and visualizing performance metrics when running database queries. By doing so, this toolkit will allow us to benchmark database performance based on different configuration schemes in order to find the best solution. Some of the configuration schemes will include modifying memory management, experimenting with varying degrees of parallelism, and altering database table partitioning.

The primary performance metric that the toolkit will be measuring is the Performance Efficiency Index (PEI), which is database time divided by wall clock time, in order to better gauge query efficiency. A higher PEI ratio (e.g. more database time spent per real world elapsed time) indicates higher database query efficiency. Another useful metric that this toolkit will monitor is real world elapsed time which ultimately determines the perceived responsive of queries made against the database.

2.1.3 Definitions, Acronyms, Abbreviations

Term	Definition
Command Line Interface (CLI)	A application which provides a means of interacting with a computer program by entering lines of text
Central Processing Unit (CPU)	The physical processor(s) in the machine that executes instructions.
Database	Software which stores information in an organized, easy to access manner
Database Administrator (DBA)	Ensures availability, reliability, performance, security, and scalability of database systems
Database Time	The amount of real world time the database spends consuming CPU resources
Enterprise Manager (EM)	A set of web-based tools aimed at managing software and hardware produced by the Oracle company
Gigabyte (GB)	A multiple of the storage unit byte, composed of 1024^3 bytes of memory
HP Inc. (HP)	An American multinational information technology company headquartered in Palo Alto, California
Performance Efficiency Index (PEI)	Wall clock time divided by database time. Indicates database query efficiency
Random Access Memory (RAM)	Computer memory where any byte of storage can be accessed without touching the preceding bytes
Software Developer	Performs research, design, implementation, and testing of software products
Stakeholder	Anyone who does not fall under the role of database administrator or developer but is otherwise directly or indirectly involved
Terabyte (TB)	A multiple of the storage unit byte, composed of 1024^4 bytes of memory
Virtual Private Network (VPN)	A method of providing encrypted remote access to a remote computer over the internet
Wall clock time	Real world elapsed time

2.1.4 References

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

2.1.5 Overview

The second section of this document, Overall Description, gives an overview of software dependencies, the intended function, characteristics, constraints, and assumptions of the toolkit. This section is more geared towards general stakeholders given that it is more of a high level examination of the toolkit. The third section of this document, Specific Requirements, is primarily intended for database administrators and software developers who are familiar with the technical aspects and relevant terminology of this toolkit. These two sections are both designed to give a comprehensive overview of the toolkit to two different target audiences, one high level and one low level.

2.2 Overall Description

2.2.1 Product Perspective

The toolkit itself is not designed to be a standalone software application that runs independently of other software systems. Rather, it is a component that is designed to integrate specifically with Oracle 12c in order to give DBA's and software developers quantitative metrics and information with respect to different database configurations. The toolkit will be a compressed folder which will house many SQL scripts each serving it's own utility function. Together these scripts will comprise a set of tools which would we will make available to DBA's and other senior level software engineers.

2.2.1.1 System Interfaces:
The primary system interfaces are Oracle Database 12c and Oracle Linux 7. This toolkit will interact with an Oracle Database 12c instance to gather performance metrics from the database itself. Additionally, the toolkit will be able to make system calls to Oracle Linux 7 in order to gather any other necessary metrics information relating to the systems processes.

2.2.1.2 User Interfaces:

The user interface of the toolkit will be a command line interface in which the user can define parameters and execute commands that are supported by the toolkit. The minimum required screen format must at least support a standard 80 character width by 24 character long terminal.

2.2.1.3 Hardware Interfaces:

The toolkit is purely software based and does not involve any specific piece of hardware. Since the toolkit is comprised of SQL scripts it will be executed by a system which is already configured.

2.2.1.4 Software Interfaces:

This toolkit is intended to be used with Oracle Database 12c running in an Oracle Linux 7 environment. The required system software and additional supporting applications specifications are listed below:

- Oracle Linux 7
  - Mnemonic: OL7
  - Version: 7.2
  - Source: Oracle
- Oracle Database 12c Enterprise Edition
  - Mnemonic: 12c
  - Version: 12.1.0.2.0
  - Source: Oracle
- Oracle Enterprise Manager Database Express 12c
  - Mnemonic: EM
  - Version: 12.1.0.2.0
  - Source: Oracle
- Oracle SQL Developer
  - Mnemonic: N/A
  - Version: 4.1.5.21
  - Source: Oracle
- Command Line Interface
  - Mnemonic: CLI
  - Version: 3.8.13
  - Source: Linux

2.2.1.5 Communications Interfaces:

There will be no communication interfaces that our toolkit will need to take into account any communication interface or network protocols. The toolkit will be a series of scripts that will act as an instruction set for a previously configured system.

2.2.1.6 Memory constraints:

The toolkit we will be developing will consist of a collection of SQL scripts, there are no inherent memory constraints.

2.2.1.7 Operations:

There will be only one mode of operation within the user organization which will be DBA level access. Most of the periods operation will be ran in the background. Initially the user will configure the execution parameters and supply a query, then the system will take over the processing until the results are returned. There will be no data processing or support functions and our toolkit will have no inherent backup or recovery functions.

2.2.1.8 Site Adaptation Requirements:

There will be no site specific requirements for the use of the toolkit. It will be available to anyone who chooses to download it in order to monitor the performance of there system during query execution.

2.2.2 Product Functions

The overall functionality of this toolkit can be divided into four specific components. Each one of the components works in concert to generate the performance metric report. These components are as follows:

2.2.2.1 Querying the Database:
This component will serve as the vehicle in which the user is able to submit a properly formatted SQL query against the database instance.

2.2.2.2 Modify Database Parameters:
This component will allow for the modification of system specific setting in order to optimize query performance. We will use this to change settings within the Oracle 12c environment to increase processing efficiency.

2.2.2.3 Gather System Metrics:
This component will use the dynamic performance views which are native to the Oracle 12c instance to monitor performance specifications during the execution of the provided query. Additionally, this component will make system calls out to the host operation system to isolate process time information that will be used as a performance metric.

2.2.2.4 Generate Performance Report:
This component will compile all of the system performance metrics which have been evaluated into a raw data report. This report can then be utilized by the users to create a visual report.

2.2.3 User Characteristics

The users of our toolkit will be Senior level software developers that are writing queries on a server. These users will have experience writing queries and interacting with a database. These users should be proficient in one or more of SQL, Oracle 12c specifics, Python, shell scripting, Unix system calls, or parallel execution. Users will have some level of formal technical education but this could vary from some college experience to doctorates.

2.2.4 Constraints

The experiments that we will run will be limited to a specific machine, that machine will have 16 real cores, 500GB of memory and 5 TB of storage. The software package that we create will be able to run on any machine that can support Oracle SQL. There will be no regulatory polices, audit functions, safety or security considerations within the scope of the toolkit.

2.2.5 Assumptions and Dependencies

In developing the performance evaluation toolkit we will be assuming that the scripts will be run on an Oracle SQL database. The development of the toolkit will rely on a few specific factors which include:

- Our computers must be connected to the local HP network to access the test database instance
  - Due to company firewall, outside access to the network is not possible without VPN
- Access to physical location limited
  - Corvallis site limited to HP employees and contractors only, so an escort is needed
- No access to server room
  - If the database goes down we have no access and must rely on HP counterpart to assist

2.2.6 Apportioning of Requirements

Refer to the Gantt chart appended to the final page of this document.

2.2.7 Toolkit Environment

The toolkit environment is comprised of four entities: Users which can be DBAs and software developers, the toolkit itself, the Oracle 12c database, and the host operating system that contains the database. Users interact with the toolkit by executing commands which can range from modifying various database parameters to executing queries on the database. While the database is the processing the queries, the toolkit will then begin monitoring critical performance metrics. The sources of the metrics can come from both the database itself and by polling the host operating system for more information about the databases running processes.

2.3 Specific Requirements

2.3.1 External Interfaces

- Input
  - Name: SQL Query.
  - Description: A valid SQL that a user wants a performance report for.
  - Source: Command line input from user.
  - Valid input: Any valid SQL statement.
- Output
  - Name: System performance raw data report.
  - Description: A raw data file of the performance statistics of the machine while running the input SQL query.
  - Destination: A text file in the local directory.

2.3.2 Functions

1) The system shall accept an SQL query as input.
2) The system shall fetch performance database tables to get statistics on the input query.
3) The system shall organize the data and make recommendations on how to increase performance.
4) The system shall output this data in into a data sheet.

2.3.3 Performance Requirements

The system we are developing is a toolkit of scripts that will be easily attributable. This means the software can easily be installed and run on any number of machines simultaneously.

2.4 Logical Database Requirements

Our software package will not be writing any data to a database.

2.4.1 Design Constraints

The only significant design constraint that we face is that our toolkit will be developed in Oracle SQL. That means that developing and using this software will have to be done on a machine that supports Oracle SQL.

2.4.2 Standards Compliance

Because our system is just a toolkit made up of scripts, there are not specific standards that we must comply with.

2.5 Software System Attributes

2.5.1 Reliability

The reliability of this toolkit will be measured in it's ability to take valid SQL statements and generate a performance statistic report.

2.5.2 Security

The toolkit will be available on a git hub repository that will be read-access only to the public. This will protect the integrity of the source code behind the toolkit. Individual users will be using the toolkit at their own risk, because they will already have full database access. This toolkit does not pose any increased security risk.

2.5.3 Maintainability

The toolkit will be organized so that each significant performance test will be in its own script file. These files will be well commented so that users can make changes as per their needs. The names of these scripts should be self explanatory and unique enough to distinguish their function from the name.

2.5.4 Portability

The toolkit that is being created will be portable to any system that supports Oracle SQL 12c. However this toolkit will not be portable to any other type of database.

2.6 Specific Requirements

2.6.1 External Interface Requirements

1) User Interfaces: The user will interact with our software toolkit through the command line. Users will run these scripts on a SQL server to get a performance report.
2) Hardware Interfaces: This software toolkit does not interact with hardware.
3) Software Interfaces:
    - Oracle Linux 7
        - Mnemonic: OL7
        - Version: 7.2
        - Source: Oracle
    - Oracle Database 12c Enterprise Edition
        - Mnemonic: 12c
        - Version: 12.1.0.2.0
        - Source: Oracle
    - Oracle Enterprise Manager Database Express 12c
        - Mnemonic: EM

  - Version: 12.1.0.2.0
  - Source: Oracle
- Oracle SQL Developer
  - Mnemonic: N/A
  - Version: 4.1.5.21
  - Source: Oracle
- Command Line Interface
  - Mnemonic: CLI
  - Version: 3.8.13
  - Source: Linux

4) Communications Interfaces: This toolkit will not interact with any communication or network protocols.

2.6.2 Functional Requirements

2.6.2.1 Database Administrator User Class:

1) The system shall accept an SQL query as input.
2) The system shall fetch performance database tables to get statistics on the input query.
3) The system shall organize the data and make recommend adjustments to increase performance.
4) The system shall output this data in into a data sheet.

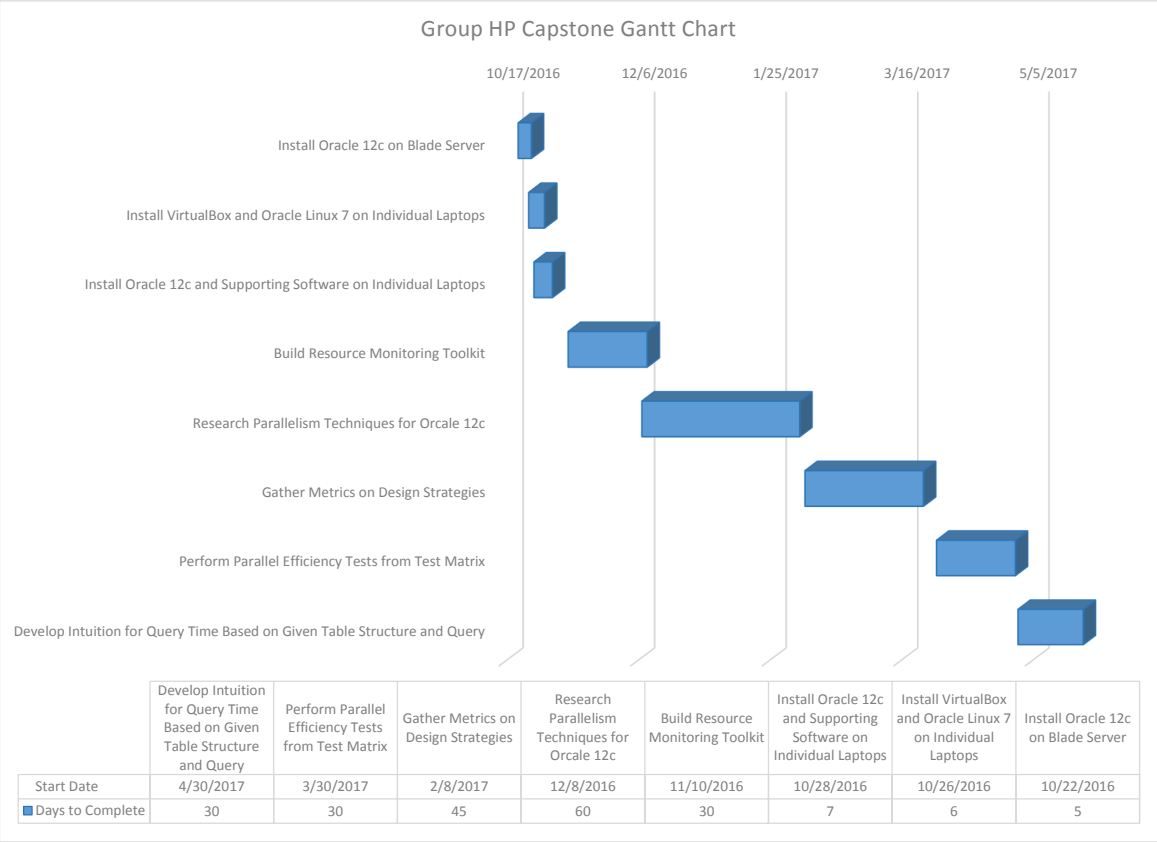
2.6.3 Performance Requirements

The toolkit will be distributed across many machines, meaning it will be run on any number of machines simultaneously.

2.6.3.1 Design Constraints:
This toolkit will be limited to Oracle SQL 12C

2.6.3.2 Software System Attributes:

- Reliability
  - The reliability of this toolkit will be measured in it's ability to take valid SQL statements and generate a performance statistic report.
- Security
  - The toolkit will be available on a git hub repository that will be read-access only to the public. This will protect the integrity of the source code behind the toolkit. Individual users will be using the toolkit at their own risk, because they will already have full database access this toolkit does not pose any increased security risk.
- Maintainability
  - The toolkit will be organized so that each significant performance test will be in its own script file. These files will be well commented so that users can make changes as per their needs. The names of these scripts should be self explanatory and unique enough to distinguish their function from the name.
- Portability
  - The toolkit that is being created will be portable to any system that supports Oracle SQL 12c. However this toolkit will not be portable to any other type of database.



	Develop Intuition for Query Time Based on Given Table Structure and Query	Perform Parallel Efficiency Tests from Test Matrix	Gather Metrics on Design Strategies	Research Parallelism Techniques for Orcale 12c	Build Resource Monitoring Toolkit	Install Oracle 12c and Supporting Software on Individual Laptops	Install VirtualBox and Oracle Linux 7 on Individual Laptops	Install Oracle 12c on Blade Server
Start Date	4/30/2017	3/30/2017	2/8/2017	12/8/2016	11/10/2016	10/28/2016	10/26/2016	10/22/2016
Days to Complete	30	30	45	60	30	7	6	5

Fig. 1: Gantt chart created for our project workflow based on fall term

2.6.4 Other Requirements

There are no additional requirements for this project other than what been outlined above.

3 Changes in Requirement Document

The only changes made to the requirement document were focused on grammatical inconsistencies that were introduced during our initial write up. There were no changes made to the structure of the document. Additionally, there were no additions to any of the specific sections. The agreement that we made with the client in regards to the deliverables of our project did not changed during the course of the project.

4 DESIGN DOCUMENT

Abstract

This document is focuses on a detailed break down of ”how” the individual components of our project will be implemented. Each of the three essential pieces: SQL development environment; parallelization and partitioning; and reporting tools are covered with fine granularity. We chose to focus on the design viewpoints in terms of the IEEE Std 1016-2009 document for Software Design Descriptions. This document outlines the path forward in terms of the viewpoints which were deemed suitable to our project.

4.1 Introduction

The following sections delve into the development design and considerations that need to be outlined prior to the development cycle. James Stallkamp covered the first four sections focusing on context, composition, logical, and dependencies viewpoints. This was followed by three sections from Nic Desilets which covered the information, interface, and structure viewpoints. Finally, Nathaniel Whitlock spent time on the interaction and resources viewpoints.

4.2 Glossary

4.2.1 Definitions, Acronyms, Abbreviations

Term	Definition
Central Processing Unit (CPU)	The physical processor(s) in the machine that executes instructions.
Database	Software which stores information in an organized, easy to access manner
Database Administrator (DBA)	Ensures availability, reliability, performance, security, and scalability of database systems
HP Inc. (HP)	An American multinational information technology company headquartered in Palo Alto, California
Script	Refers to a collection of SQL queries in the context of this document and project.
SQL Developer	An integrated development environment specifically designed for both SQL queries and Oracle Database 12c.
Structured Query Language (SQL)	A programming language specifically designed for relational database interaction.

4.3 Context Viewpoint

4.3.1 Design Concern

This toolkit will be used in the context of database queries that are not utilizing the host machine’s resources. The optimizer analyzes the structure of a query and allocates resources for and executes the query. Some queries are structured such that the optimizer allocates resources in a way that much of the host machine’s resources are not being utilized. The toolkits design would help the user understand the performance of these queries that result in the under utilization of resources. This toolkit will be a package of scripts that can be run in an environment like SQL developer. The primary concern of the toolkit will be to gather and display performance statistics that are stored on internal performance tables within the database. This toolkit will provide a library of scripts that can be used to gather various kinds of information on a query and how the database executed the query. Users will be able to execute these scripts and have the output piped into files or displayed on a web page. A User could then use this information to better understand how the database is executing a query.

4.3.2 Design Elements

This toolkit will largely be standalone, it will interact with one other entity and have one type of user. The only other entity that this toolkit will interact with is the database that it is working with. The SQL scripts will interact with the database by querying internal performance tables for data. The user of this toolkit is a database user that is trying to improve or understand the performance of a query. Users can interact by executing a script or batch of scripts and receive an output. This output will be dependent on the user's input and scripts selected, it will be composed of various performance metrics and information about given query. This data can be viewed by the user in a file or through a web browser.

4.4 Composition Viewpoint

4.4.1 Design Concerns

This toolkit will gather and display performance statistics and will do this through SQL scripts. The purpose of the SQL scripts will be to both run a performance monitoring loop to gather runtime statistics, as well as execute test queries. The combination of all the scripts will function as the toolkit and retrieve data on a specific query. Each of These scripts will retrieve various pieces of data or performance metrics. The data will be used together to display all the required information on the performance of a specific query.

4.4.2 Design Elements

The individual scripts will be the elements of the system and will act like functions. Each script will provide a different function by producing or retrieving a different performance statistic. Most of the scripts or elements will be stand alone, a database user could run one or any number of the various scripts in SQL developer and receive the combined output. The primary attribute of a script is its purpose, each script will have a different purpose, either to retrieve or output a specific performance metric.

4.4.3 Function Attribute

All of the scripts will be accessing various internal performance metric tables on the database. Each script would retrieve and analyze one of various performance metrics like memory usage, CPU usage, I/O, and many other performance metrics. The scripts will be different in that each script will retrieve or output a different kind of statistic. Each script within the toolkit will be providing a different a different data point on attributes like memory usage, CPU usage, etc.

4.5 Dependency Viewpoint

4.5.1 Design Concerns

This toolkit has two significant dependencies, it is however a very versatile tool because almost anyone with access to the database can use it. Any user of the database can use this tool to retrieve information on how their query ran. The toolkit is based on Oracle's Linux SQL technology and as a result the toolkit will only work on a Oracle SQL technology based database. The toolkit requires Oracle SQL because the toolkit uses SQL statements to access data stored on internal performance tables. The SQL statements also refer to entries in internal tables that may be specific to Oracle version 12c, meaning it is strongly recommended to be using this toolkit on a database that has Oracle version 12c or newer.

4.5.2 Design Elements

The scripts in the tool kit require access to the internal performance tables on the database. The user must have access to those database tables for the toolkit to access them.

4.6 Information Viewpoint

4.6.1 Design Concerns

The primary persistent data structure and source of data for the toolkit will be the internally managed performance metrics tables contained in Oracle Database 12c. Users will be able to request information about a particular query ran against the database through the use of this toolkit and these internal performance tables. The data returned from the toolkit regarding a particular query can include information such as time elapsed, processor usage, memory consumption, degrees of parallelism, and other related metrics.

4.6.2 Design Elements

The data items that are retrieved by the toolkit are stored in a permanent data store (Oracle 12c). Access of this data is provided through the use of SQL Developer in which the toolkit scripts are intended to be run with. SQL Developer initiates a connection to the database to retrieve the relevant performance information needed by the toolkit scripts.

More specifically, the information that is retrieved by the toolkit comes from different tables within the database. The records stored in these database tables are generated after the user executes a query. After completion of a query, the toolkit can then be used to query the database again for metrics information about the previous query ran by the user. This information is then processed and returned to the user in a human readable format.

4.6.2.1 Data Attribute:
The metrics information that the database contains spans a broad range of different types of information used for profiling the performance of queries. The specific types of this information ranges from numbers, strings, to date representations. For example, timing information is represented as numbers, metrics descriptions are represented as strings, and start and end times would be date representations.

All of this information is categorized and put into specific tables that relate to a certain domain of information relating to a query. Examples of this can include storing memory usage information into a table that describes how much memory was consumed over time when processing a particular query. The data stored in these tables is intended to be used to offer insights into query performance.

4.7 Interface Viewpoint

4.7.1 Design Concerns

The primary interface for this toolkit is going to be SQL Developer. The toolkit itself consists of SQL queries that are to be executed inside the SQL Developer environment. Users will select which toolkit query they want to run based on what type of information they need. Upon selecting a query, the user will then load it into SQL Developer which sends the query to the database. Results returned by the database will then be presented to the user in a human readable format.

4.7.2 Design Elements

The primary elements of SQL Developer include the SQL Worksheet window. This where queries will be loaded and then sent off to the database for execution. Once the database has completed processing a particular query, the results from the database are then returned and presented to the user. These results are contained in a new window within the SQL Worksheet window. Users will be able to scroll through the results if there is a large amount of information.

4.7.3 Examples

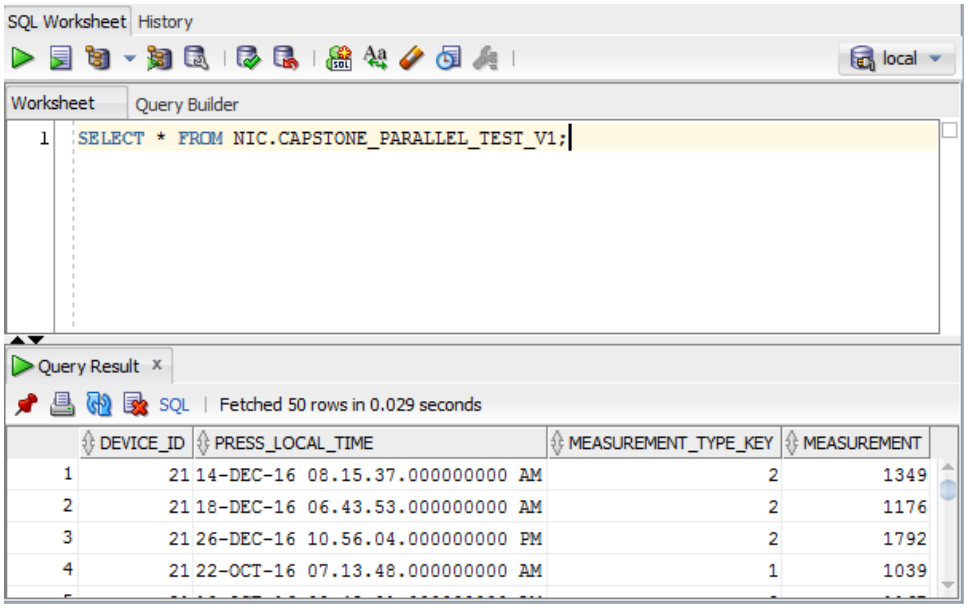


Fig. 2: An example SQL query in the SQL Worksheet window. Query results are shown in the Query Results window

4.7.4 Example Languages

4.8 Structure Viewpoint

4.8.1 Design Concerns

The two primary structures involved with the toolkit are SQL Developer and Oracle Database 12c. Oracle Database 12c is a permanent data store and SQL Developer is an environment that is used to interface and interact with Oracle Database. SQL Developer must first initiate a connection to Oracle Database 12c in order to execute any queries. If the connection is successful, the user can then start loading and executing queries via SQL Developer's database connection.

4.8.2 Design Elements

SQL Developer connects to Oracle Database 12c in order to retrieve information contained in its data store. The purpose of SQL Developer is to serve as a work space for the user to input queries, have the queries executed, and view the output returned by the queries. The database on the other hand is a data store that houses information relating to HP PageWide Press printers and also internal metrics relating to query performance. The connection from SQL Developer to Oracle Database 12c serves as the means of bi-directional communication between the two structures.

4.9 Interaction Viewpoint

4.9.1 Design Concerns

The main design concern in terms of the interaction between system entities is an open line of communication with the Oracle database. If this connections is severed, then there is no way for the user's personal computer to execute a query

on the database. Therefore, the toolkit will not perform and the transaction will be impossible. In terms of allocation of responsibilities in collaborations with the toolkits use at HP, the user's machine will only be responsible for composing the query and sending it through to the database running on the server. The calculations and each procedural step of the toolkit analysis will be processed remotely and the results and statistical report should return to the user.

4.9.2 Design Elements

The entities involved at various levels of action are as follows; Engineer/DBA, personal computer, server, and Oracle database. During the transaction from the user to the server there are multiple wait instances where one of the host systems sits idle waiting for the final result response. Methods used in the communication process are beyond the scope of our toolkit and will be handled by higher order systems.

4.9.3 Examples

The process of interaction between the entities involved in a transaction with our toolkit can be summarized as request from a user and a response from the target Oracle database. In this example the response is composed of the users desired results and a performance statistic report. The report will have information about hardware optimization and run-time analysis. This information can then be used to quantitatively compare queries run with different system parameter settings, by comparing the run-time of multiple experiments a optimal design can be identified.

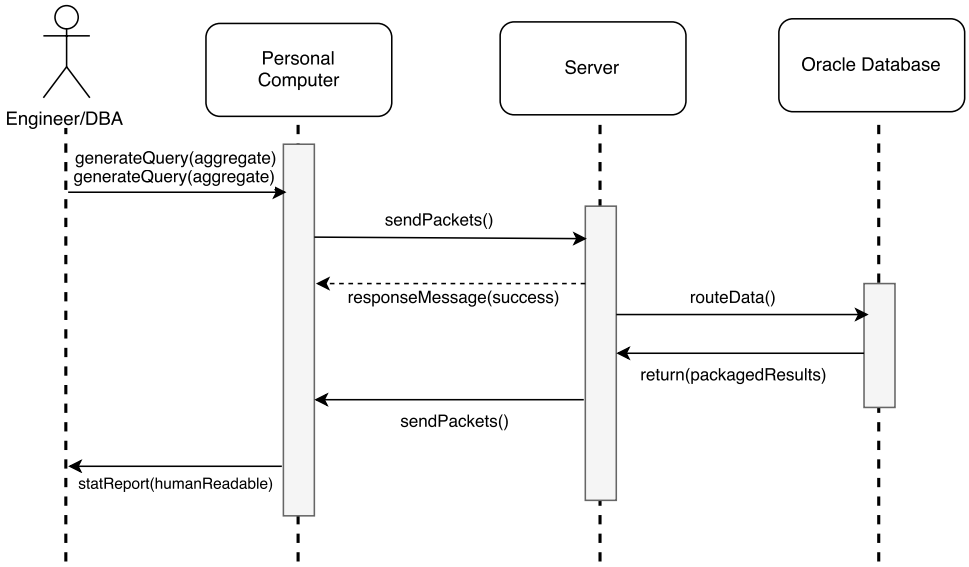


Fig. 3: Sequence of actions once query of interest and toolkit are executed by the user

4.10 Resource Viewpoint

4.10.1 Design Concerns

The purpose of the toolkit is to monitor the system processing a query in order to generate a report which outlines the performance metrics. There should be concern placed on the overhead processing time of the toolkit query. The only inherent performance concerns for our toolkit would be the syntax used to build the toolkit query, which could be evaluated with the SQL explain plan. This toolkit should be portable to all systems capable of hosting Oracle database software so it should be lightweight in terms of size.

4.10.2 Design Elements

The design entities we have considered are modeled after the system properties we are interested in measuring. Time elapsed is the main entity but we also have CPU cores, disk I/O, and parallel coordinator. The time elapsed entity will allow us to track the processing time for the query being evaluated, this will be useful when comparing the run-time of identical queries given different configuration. The CPU cores entity will provide information about how many individual cores were used out of the total available cores and how much work each was assigned. The disk I/o entity will be used to monitor swap space needed due to lack of physical memory resources. Together CPU cores and disk I/O will help us monitor how much of the available hardware is being utilized during SQL processing. Finally, the parallel coordinator entity will provide information about the type and level of parallelism that was used.

In terms of constraints for each entity, the time elapsed entity should ideally be as low as possible to enable near real time access to WebPress performance data. There are a total of 72 core processors available during query execution, so the CPU core entity is limited by that number. The disk I/o entity should ideally have a minimal amount of activity meaning that the SQL plan the optimizer put together was efficient and the initial data fetch was as well. If a query is executed with parallelism enabled and a hint for a specific number (n) of individual processors, then each one of the available cores should be used.

4.10.3 Examples

By isolating each one of the above stated entities we will be able to generate a report with enough information to tune a users queries as well as provide the metrics needed for quantitative comparison. An example of this would be testing the effectiveness of a table partition design in respect to a query that performed calculations on specific values and took a large amount of time. By monitoring these entities, we will be able to see how many cores were made available to the coordinator, how much swap space was used, and how many parallel processes were estimated to be efficient. Additionally, we will have a custom metric calculated from time elapsed from the point of query execution.

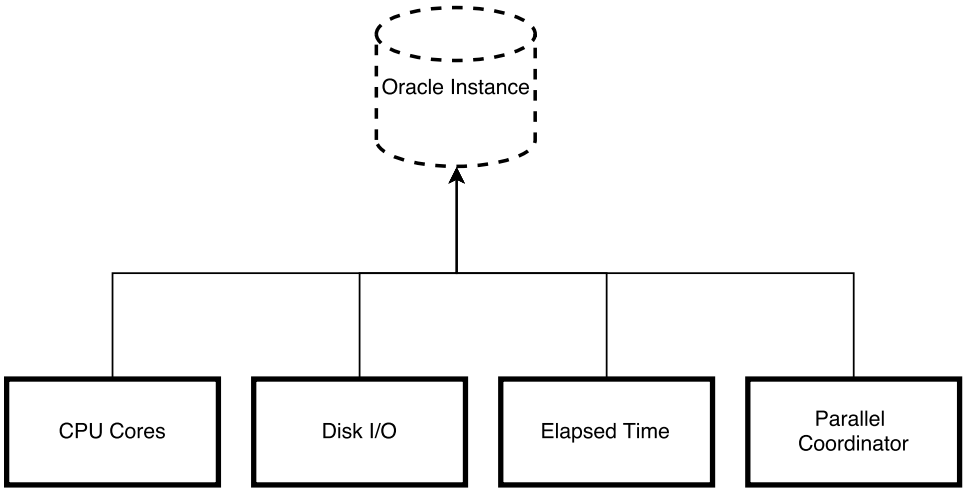


Fig. 4: Simple visualization of the resource entities and the database

4.11 Conclusion

The toolkit that we are creating consists of multiple interacting entities including the end users, Oracle Database 12c, and SQL Developer. During the technology review phase we carefully reviewed our options for these entities and chose technologies that were best suited for this project. This document embodies a solution to bring all of these technologies and entities together in order to assist database administrators with evaluating query performance.

5 DESIGN REVIEW DISCUSSION

Initially we thought we would need to revise the design document. However, after reviewing it, our team had constructed this document in a manor that still fit the scope of our end results. During fall term, our clients were still not entirely sure what they wanted, so writing a design document was somewhat difficult for us. Our clients knew that they wanted a toolkit, or a collection of software components, which would enable them to study the effect of parameter settings and features on their database.

The only changes that were made to this document were modifying the sections in the initial version that discussed the notion of a directory of shell scripts, each acting as an independent part of the requested toolkit, into a description of our toolkit consisting of a test automation suite and sets of series of scripts acting as the experiments and associated tests. All of these were self contained in production file system. This change reflects the agreed upon product requested by our clients at HP.

6 TECH REVIEW

Abstract

This document is a review of three essential roles within our project SQL development environment, Parallel and Partitioning, and Performance Reporting. From each of those integral pieces, three or more technology options were evaluated and compared. The desired SQL development environment was deemed to be Oracle's SQL Developer because it is a free software that provides all the features we might need and more. Additionally, Oracle database software was chosen as the best framework for parallelism and partitioning. Finally, the decision was made to create a reporting tool from scratch in order to allow more customization for data output. The resulting decisions will act as the initial path forward until there is a need to consider one of the other options.

6.1 Introduction

Our project to develop a SQL performance toolkit has been distilled into three different essential pieces: SQL development environment; parallelization and partitioning; and performance reporting. The following sections review technologies for each of these pieces. The goal of this document is to compare available technologies that may be of use to our team during the developmental stages of the project.

The research efforts during this technology review is as follows: James Stallkamp focused on SQL development environments; Nathaniel Whitlock focused on parallelization and partitioning; and Nic Desilets focused on performance reporting Tools. Each section below outlines a comparative analysis of technologies which may be suitable for using within the essential pieces of our project.

6.2 SQL Development Environment

6.2.1 Purpose of a SQL Development Environment

SQL and most programming languages can be developed and executed directly from a command line interface. However since the early 1990s interactive developer environments or IDEs have been used to organize information and provide useful functionality to developers. IDEs provide many useful features such data visualization or manipulation, code completion, usage finding, and much more. It is still possible to develop SQL on the command line but to do so would be a waste because it is ignoring the many tools at ones disposal. The purpose of the IDE is to help the developer visualize data, interact with data if necessary, and to help with code completion. Data visualization and interaction allow developers to easily understand what the application is doing in memory or data space which makes debugging a much less painful process. Code completion means the IDE is aware of table structures and data keys so the IDE can make recommendations as the developer types.

6.2.2 Existing Technologies

There are many different IDEs for SQL, these different IDEs all have their own way tools and ways of organizing data. Comparing the differences between these IDEs comes down to cost, features provided, and personal taste. Many IDEs are very similar and the biggest differences are usually a function of if the IDE is free or paid for. Because many IDEs are very similar personal taste is often a factor in deciding an IDE, each IDE organizes data a little differently and most people have a style they prefer. Three technologies that will be considered for the IDE in this project are SQL plus, Oracle SQL Developer, and Toad.

6.2.2.1 SQL Plus:
SQL plus is an interactive and batch query tool that is installed with every Oracle database installation. Because SQL plus is built in with Oracle this means we could just use SQL plus and not have to install any other packages. This makes SQL plus the easiest to use but as will see it also comes with the least functionality. SQL plus has a command line interface, and a windows graphical user interface, and a SQL plus web based user interface. SQL plus can generate

reports as batch processes and to output the results directly to a text file, screen, or html fie for browsing. SQL plus enables users to execute SQL, PL/SQL, and operating system commands to perform the following operations.

- Format, perform calculations on, store, and print from query results [1].
- Examine table and object functions [1].
- Develop and run batch scripts [1].
- Perform database administration[1].

6.2.2.2 Oracle SQL Developer:
SQL developer is a free standalone graphical tool that can simplify database development [2]. To use SQL developer for free it must not be used for profit, however it can be downloaded for free from the internet. SQL developer is an IDE published by oracle and contains a vast array of features. Some features included are worksheets for running queries, a DBA console, reports interface, and much more. Also because SQL plus is built into oracle, if you are using oracle then you could use SQL plus in SQL developer. SQL developer is a complete IDE that provides all the features need to completely develop a SQL application.

6.2.2.3 Toad:
TOAD is a paid for IDE that offers features and services for databases including those that do not rely on SQL. The main distinction between TOAD and the other IDEs discussed in this document is that TOAD must be paid for and could be run on systems other than SQL. Being that TOAD is not free, it does provide the most features and activly tries to help the developer more than SQL developer or Plus. TOAD does this by providing automated unit testing, performance testing, automatic code consistency improvements, and code optimization [3]. TOAD also has its own version control system that makes collaboration across developers very easy.

6.2.3 Conclusion

After examining the IDEs available we have concluded that we will be mostly using SQL developer. However SQL plus is built into the Oracle installation and can easily be used with SQL developer so mostly likely we will use both. This project will be focused on writing queries and exmaining the runtime of the database to see how it performed. For these experiements the visualization features provided in SQL plus and SQL developer will be extremely important [2]. The last IDE toad, provides more features than the first two combined but we simply dont need the robustness of TOAD. TOAD was included to examine the potential advantages of using a paid for IDE instead of using a free one. Being a paid for service, TOAD can provide a larger number of features that are also more powerful. There are many features that TOAD can provide that neither of the other two options can, such as version control, automatic testing, and data update managment [3]. TOAD is a very powerful IDE, and if we had to actually manage a real database it might be useful, but for these experiements it is more than we need. For the experiments that we are running we do not need to be running unit tests or updating data, and version control can be accomplished other ways. In short SQL developer will be the main IDE that we use to run and queries and examine the database, it makes sense to also use SQL plus because it is built in.

6.3 Parallelism and Partitioning

6.3.1 Purpose of Parallelism and Partitioning

When data sets become large or the aggregate functions used to extrapolate statistics from a table are too complex the database can take a long time to return anything useful. Long periods of latency between query execution and the return of desired data can make the system practically unusable. Techniques such as parallelism and partitioning can be implemented in order to increase computational efficiency and reduce the observed time it takes to return a query.

Parallelism refers to the concept of dividing a large task, such as the processing of a complex query, by breaking it up into specified chunks in order to distribute the work among additional nodes or processes. Before a query is executed the plan for execution can be viewed, this plan lists each of the steps functions and associated cost. During the optimization process, the cost of each step is minimized by tuning the processing options which results in an efficient execution plan

[4]. Logically grouped steps from the execution plan can then be processed on different nodes in order to reduce the time it takes to complete processing. The degree of parallelism is a term which refers to the number of processors that are utilized during program execution [5]. As the number of processors splitting the workload increases, the time needed for computation decreases.

Partitioning refers to the logical separation of ranges in a table, index, or indexed table. The general idea is to make independent subdivisions that would assist in the processing of a query. If you had sales data for several years and wanted to calculate the average sales per month, the table of data could be logically partitioned by month and processed on separate nodes. By dividing the data set among many nodes, each table partition can be processed in concert leading to a quicker return of target data.

Together, parallel processing and partitioning tactics act in a divide an conquer like fashion. This is accomplished by breaking a large problem into workable sub units, computing, followed by a reassembly of data. These techniques are some of the most effective method at tuning SQL queries on large data sets.

Below is a discussion of how parallel queries and partitioning are implemented within the three database technologies of our interest.

6.3.2 Database Technologies

6.3.2.1 Oracle:
The Oracle Corporation has been around for over 39 years and is largely known for database software which got its name from a CIA-funded project that one of the founders had previously worked on. Oracle 12c Enterprise edition has been designed to be used by companies and data intensive users. This means that it is very reliable and there is support available for the service, these benefits come with a license fee that can be quite costly. Oracle database relies on a share everything architecture, meaning that there is no requirement for pre-defined partitions. However, Oracle partitioning offers the same processing potentials as a share nothing architecture [6]. The current version of the Oracle database software is 12.1.0.2 and was released in July 2014.

According to Oracle documentation, every SQL statement undergoes optimization where parallel execution may be selected. If parallel execution is initiated, then the user session takes the role of query coordinator. The coordinator calculates how many parallel processes are needed, then the required read steps are performed in serial and the steps which can be executed in parallel are processed as such. Parallel queries and subqueries are handled by evaluating two components: the degree of parallelism (DOP) and the decision to parallelize. The degree of parallelism is determined most often by the table being modified by an INSERT, UPDATE, DELETE, or MERGE command. In order to use parallelization you must include a statement level hint to indicate the desire to parallel process, the objects referred to need a parallel declaration associated with them, and Auto DOP must be enabled [6].

The Oracle database software has wonderful documentation which outlines their predefined table partition methods. A list of the methods with a brief overview of its function are outlined below:

Partition Methods

- Range Partitioning
  - Maps data based on ranges of values of the partitioning key, this is the most common type of partitioning and is often implemented with dates or times [7].
- Hash Partitioning
  - Maps data based on based on the Oracle hashing algorithm which is applied to the partitioning key. All partitions have even distribution of data making them similar in size [7].
- List Partitioning
  - Maps data by applying a list of specified partitioning key values that should go in each partition [7].
- Composite Partitioning
  - After applying one of the above partitioning schemes, another partition scheme is applied to each partition. The result is subpartitions which represent a logical subset of the data [7].

6.3.2.2 PostgreSQL:

PostgreSQL is an open source object-relational database system that was developed by a team of volunteers in 1996. Although this database software is free it is feature rich and standards compliant. It is also highly customizable allowing stored procedures written in more than a dozen programming languages to be executed. Some potential drawbacks of the software include: installation and configuration difficulties; psql command line different than traditional commands; sheer number of features can take years to learn [8].

PostgreSQL has a feature called ”parallel query” which evaluates an optimizer serial query plan in order to check if the query contains steps which can processed independently of each other. Though this is not true for small simpler queries, more complex queries have steps which can be split off, processed separately, and recombined [9]. Queries that benefit from this feature the most occur when a query reaches out to a large amount of data for calculations and only returns a few rows. Within the PostgreSQL system each separate node or process used during parallelization are referred to as ”workers” . The number of workers the planner has available is determined by an internal variable called max_worker_processes, this variable is initialized at server start with the default value of eight but it can be manually set [9]. It is possible to utilize less than the maximum number of workers and still have an optimized parallel query plan.

Basic table partitioning is supported by PostgreSQL. The partitioning is implemented by creating a master table, as well as multiple child tables that inherit from the master table and act as logical partitions. Table constraints are then set in order to define what data is placed in each partition. The data from the master table is then copied to each appropriate partition and given a partitioning key, the result is smaller tables which are subdivided by a declared expression [10]. Though there was not predefined table partition methods in the PostgreSQL documentation, most of the methods outlined in the Oracle section (Range, List, Composite) could be achieved through the table constraints applied to each child partition.

6.3.2.3 Redshift:

Redshift is a cloud based data warehousing product from Amazon Web Services (AWS) which is based on the 2005 release of PostgreSQL 8.0.2. Since Redshift is cloud based there are no upfront fees from hiring experienced individuals to set up servers or configure the system. According to the AWS site, Redshift is a petabyte scale data warehouse which is fault tolerant with built in encryption. They also claim that there is no need for tuning to maintain performance and speed since the system is self managed, this service is offered at $\$1,000/\text{TB}$ per year [11].

Unlike traditional row-oriented databases, Redshift uses columnar or column-oriented storage. Columnar storage works by reading only the values from a row which are needed to complete the transaction. This means if a query is executed that only needs to evaluate the value of one out of four columns, then only the value from that column needs to be loaded into memory, thus drastically reducing the disk I/O [12]. With row-oriented storage, each of the rows containing the values of interest would have to be read into memory.

Redshift is a classified as a shared nothing or Massively Parallel Processing system. Shared nothing refers to the fact that no single node has a complete view of the database, and therefore cannot communicate with each other by sharing data during processing. A cluster is a set of nodes, each node having an independent operating system and memory. The disk storage of each node is broken down into units referred to as slices, the number of slices per node is dependent on the number of nodes within the cluster. Though there are settings for some system configuration, the distribution of work over the compute nodes is automatically handled in order to create a simple experience for the user [13]. Redshift does not support the partitioning concept, rather it relies on the distribution style option selected by the user as well as distribution and sort keys [14]. There are multiple options for distribution styles which work to redistribute rows to compute nodes when a join or aggregation is needed.

Distribution Styles

- Even Distribution
  - Leader node distributes rows across slices in equal proportions in a circular fashion. This is the default distribution style but is only suitable if the table does not participate in joins [13].
- Key Distribution
  - The rows are distributed among slices based on the value of one column. This process is analogous to range partitioning in terms of this system [13].

- ALL Distribution
  - A copy of the entire table is stored in each slice in order to ensure that every row is collocated for each join. This causes slow function return on the data set [13].

6.3.3 Conclusion

After careful analysis of each option, there are a couple of properties between the three technologies that seem appealing. First, the column-oriented storage mentioned in the Redshift documentation seemed promising given the analytic procedures of our client. However, in 2013 Oracle announced that they we going to support column-oriented storage [15]. Since this feature was not isolated to Redshift alone, Redshift lost some of it's appeal over the Oracle database software the clients already have up and running. Second, was the ability to customize logical table partition designs. Both Oracle and PostgreSQL offered these features through different implementations. However, Redshift relies on the concept of data distribution styles of node slices. Though it is basically the same concept, Redshift has been developed for "ease-of-use" user experience and therefore does not offer as detailed customization as the other two database technologies. Out of the three considered database technologies, the winner is Oracle. The reasons behind this decision are based on Oracles customization and reliability. Though it is a paid service, our client already has a license and hardware setup in order to utilize the service, so the level of granularity in terms of system configuration is more appealing for the Oracle technology than any of the alternatives reviewed.

6.4 Toolkits

6.4.1 Purpose of a Toolkit

The main purpose of a database toolkit is to assist a Database Administrator (DBA) with analyzing the performance of queries made to a database. The different types of toolkits that can be used with Oracle Database 12c vary widely from PL/SQL queries that can be run directly in the database to complex, fully featured web applications. Likewise, the use cases for these toolkits can range from quick and easy query performance diagnostics all the way to automating the analysis and optimization of parameters within the database to a particular SQL query or multiple queries.

When analyzing queries, being able to effectively quantify and visualize multiple aspects of a query statement is important to a DBA. Some of the information gathered might include the time it took for a query to run, how much memory was allocated to a process handling a query, and the amount of disk input and output operations. This information can then be combined and analyzed in order to identify problematic areas in the configuration of the database or with particular queries themselves.

For example, if you have a particular slow running query, you might use a toolkit to analyze what is happening when the query is being run. After analyzing the output of the toolkit you then find out that the database is suboptimally allocating memory for the process that is responsible for the query, thus resulting in a lot of unnecessary swap space usage. With this knowledge you can then adjust parameters within the database to allocate more memory to the processes that handle queries. Alternatively, you may discover that the database parameters are fine and that the structure of the query itself is suboptimal in the context of the database configuration.

6.4.2 Existing Toolkits

6.4.2.1 Oracle Enterprise Manager:
Oracle Enterprise Manager (EM) is already included in the Enterprise edition of Oracle Database 12c. It is intended to be a one stop, all in one application that can be used for analyzing and optimizing query performance as well as a multitude of other database functions such as general administrative tasks. Unlike other toolkits which are comprised of PL/SQL queries that can be executed in the database, EM is a full fledged web application that is integrated with the database itself and is packed with many features beyond analyzing query performance.

Some features that are included in EM are: the ability to manage and maintain several databases from one web application; automating repetitive tasks involved with maintaining databases; testing changes before rolling out to

production; diagnosing performance problems and automated database tuning [16]. Additional examples of what EM can do are it can determine if indexes will be helpful for performance of a particular table and analyzing sql query statement structure for potential performance issues.

One particularly useful feature of EM for analyzing query performance is a subcomponent called Active Session History (ASH). The ASH tool runs in the background and samples each active session within the database once per second. Each sample for each session contains detailed info about what resources are being used by the session and how the session is using it. This is especially useful for identifying bottlenecks and other performance issues of running SQL queries.

6.4.2.2 SQLd360:
SQLd360 is a tool that is written by Mauro Pagano, an ex-Oracle Database Engineer of about ten years, which analyzes performance metrics of SQL queries. Unlike EM, it will only analyze a single specified query and will not perform any database optimization for you. Instead, it is only intended for performance analysis of a query which usually falls under the case of troubleshooting. SQLd360 is very similar to EMs ASH tool since it operates under the same underlying principle of periodically sampling the session that the query is running in to gather performance metrics [17]. This information can be specified to be output in either html, text, csv, or graphical charts. One important difference is that SQLd360 is provided as a completely free tool while EMs ASH requires additional licensing from Oracle. This makes it particularly useful for users who do not have the required licensing but still need to obtain key metrics from session data. After the tool is finished running, it will dump all of the collected information into a zip file for later analysis.

6.4.2.3 Snapper:
Snapper is another toolkit written by Tanel Poder, an Oracle Certified Master DBA, which also analyzes SQL queries. Similar to SQLd360, it is also provided completely free of use and does not require additional licensing from Oracle. It also operates similarly to SQLd360 and ASH in EM in that it uses the database's sessions table to extract performance metrics from the session that represents the currently running query. Like SQLd360, it will also not provide any recommendations about how to optimize queries or database parameters. It's primary use case is basic troubleshooting of a query that seems to be running slowly in a production or development environment [18]. Aside from providing metrics information, everything else is left up to the DBA to figure out. In addition to sampling data from query sessions, it also gathers data from the V$ and X$ tables within the database which both house extra information regarding query performance. Snapper combines this information together in order to provide a data rich visual representation of how exactly a query is running in the database.

6.4.3 Creating a Toolkit

An alternative option to selecting an already existing toolkit would be to design our own toolkit from the ground up. However, this comes at the cost of reinventing the wheel which understandably brings up the question of why there is a need to create yet another database toolkit. One of the primary benefits of designing our own toolkit is that we would have complete control over the provided features and functionality. We could then design a toolkit that is tailored exactly to our needs rather than going with a generalized solution. Some features could include not only analysis of query performance, but also automatic tuning and changing of key database parameters that are directly relevant to our problem. By running a particular query over and over again with each running query having a slightly different database configuration, we would be able to identify what database configurations work and what doesn't work specifically for our use cases. In our case, since this is a research project, it would make sense for us to research the details of the database and reinforce that understanding by designing our own toolkit.

6.4.4 Conclusion

Out of the three toolkits mentioned above, it might sound like Oracles EM is the obvious way to go since it is very feature rich and is even capable of automatically optimizing SQL queries and database parameters for you. However, we are still going to develop a toolkit of our own that will provide similar functionality. While avoiding reinventing the wheel is typically important when it comes to designing and creating software to prevent wasting time, in our case it makes sense for us to design a toolkit of our own that provides similar functionality to the three toolkits described above. The primary motivation for this is the additional knowledge and insight that is to be gained from working with and researching the internals of the database. The expected outcome of developing our own toolkit is that not only will we have a custom made toolkit tailored to our needs but we should come out of it with more knowledge about Oracle 12c.

While we could just use of the existing toolkits to perform query performance analysis for us, it is likely that we would not be as effective in modifying important database parameters when it comes to memory management, database table partition design, and parallelism. Ultimately, creating our own toolkit will force us to dive into the database which will allow us to become more proficient in understanding the internals of the database and also customizing the parameters of the database to increase query performance.

6.5 Summary

Oracle Database 12c seems to be the best fit for our application and so we will be using that for our project. The alternatives did not offer enough customization and were not appealing enough to warrant switching databases. Additionally, Oracle Database 12c is a tried and true enterprise grade database. Given that our project is heavily research based, creating our own toolkit will benefit us the most in terms of knowledge gained and functionality of the toolkit. Researching and understanding the internals of Oracle Database 12c while in turn creating our own toolkit tailored towards our exact needs derived from this research will play a key role in our success with this project.

7 Discussion of Tech Review Changes

No changes were made to the original tech review. This was an exercise conducted to consider alternative technologies, as well as the technologies we were considering. Since our clients already has a group of tools that they intended to use for this project, there was very little that we could have done to persuade the on alternative means of a solution. Some of the technologies concluded as most useful during this process, such as SQL Developer, were used out of convenience and usability even though it was not required by our clients.

8 Blog Posts

8.1 Organization

For the following section, the blog posts of Nic Desliets and Nathaniel Whitlock will be listed by order of week for each term of senior design. Since we lost our teammate James Stallkamp, we will not include his blog posts in this submission.

8.2 Fall Term

8.2.1 Week Three

Nathaniel:
The initial meeting of the client went smooth.Our project mentors are Kirby, he has a background in chemical engineering and does a lot of the data analytic work in the team, and Andy, who is an Oracle database administrator. As a team they support the database and provide summary statistics of WebPress performance data.

During the first official week meeting that we had at the HP campus, we were forced to abandon our conference room. The reason was because another group had previously had a reoccurring room booking through Outlook that must have expired and was not re-booked. Since there were more of them than there were us, we fled the room to a end-of-isle cubicle meeting area for an hour. This was a trivial hiccup in the 3 hour meeting, during the rest of the meeting we discussed an overview of the unique ”areas” located in a Oracle database instance (ie. PGA and SGA). We also talked a lot about the data dictionary, which is an essential concept in an Oracle system. The data dictionary is a read-only set of tables which provides useful information about the current database instance. Some of the important features of the data dictionary we are interested in is:

- Definitions for each schema object, so every table, view, index, ...
- The amount of space allocated for each schema object and how much it is utilizing
- Information about the users logged in and the privileges they have been granted
- General database information

Most of our group efforts so far have been focused on preparing the Problem Statement document which is due Friday October 14th at noon. After that, each of use will install Oracle 12c on our personal computers. We will be given test data and our goal will be to figure out how to write a toolkit, composed of PL/SQL queries that target a specific table in the database (note sure here, we talked about v$ and x$). This will allow us to gather metrics on the server hardware performance after we run a work heavy query.

Nic:
I had pretty much the same experience as Nathan as we all went to the same meeting and got the same information. We were able to further discuss exactly what our project entails with our clients and got a good understanding of what is expected of us. Our clients also gave us a rundown of their current processes, a little bit of how it works, and what isn't currently working/what we are going to fix. Our goal before our next meeting is to get Oracle 12c installed locally on our machines so that we're ready to dive into it during our next meeting. From there, we will be begin working on further familiarizing ourselves with the database and start developing a toolkit to record metrics on database performance.

8.2.2 Week Four

Nathaniel:
During our meeting this week we focused on getting Oracle Linux 7 installed on VirtualBox. Once this was accomplished we installed Oracle 12c database software on the VM's so that we could start doing performance related queries targeted at v$ and x$. After becoming more familiar with this process we will begin to put together our performance toolkit that

we will use during the rest of the project. Moving forward through the week, I plan on finishing up the installing tasks for my laptop and starting the process of looking at performance states of small queries on my local device.

The toolkit development process will be version controlled through this GitHub repo. We as a team will work on building and modifying things as needed. Additionally, we plan on starting the revision of our Problem Statement after we receive our feedback tomorrow 10/20/2016.

After meeting with our T.A., Jon Dodge, I took his advice of sending James a message. The intent of the message was to let James know the concerns which we are having with his attendance and hopefully persuade him to begin participating with the group more. Hopefully this reaches him in a positive light, I don't want to make enemies with anyone and I surely don't want him to be forced to take another year of school.

This weekend I plan on trying to take a higher level approach to the Problem Statement in order to appeal to a broader audience. The feedback that we received will not take to long to address, then some effort will be put on formatting the final LaTeX document so that we do not lose any easy points.

Nic:
Same as last week, Nathan, James, and I generally did the same things at the meeting that we went to. Our goal before attending this meeting was to at least have Oracle Linux 7 installed on a virtual machine so that we would have a development environment to work with. We spent a fair amount of time trying to get Oracle 12c installed onto our machines and have it function correctly. We also discussed how we would gather performance metrics from using our toolkit (one way is to look at the V$ and X$ tables within 12c). By the end of the meeting we were mostly setup and ready to start creating the toolkit for the next meeting. Our goal for the next meeting is to actually start this designing and creating this toolkit.

8.2.3 Week Five

Nathaniel:
On Monday of this week I stayed home due to being ill, but I was able to set aside a lot of time to get my work laptop and personal desktop set up with Oracle Linux 7 workspace set up with Orcale 12c. Getting this set up is essential to moving forward with the toolkit development as well as experimental analysis with the queries in order to get used to writing PL/SQL statements. One of the issues I was having with my work laptop is that when connected from my home network I needed to connect to the HPI network by VPN, before doing this I had no network activity through the VM.

Nic:
Earlier in the week I worked on the problem statement a little bit to remove some of the technical jargon and make it more generally accessible. I also got a start on the requirements document towards the end of the week. Aside from papers, I finally got Oracle 12c configured properly with Oracle Linux 7. Next week we should be able to start building our toolkit.

8.2.4 Week Six

Nathaniel:
Tuesday evening I spent time working on the Requirements Document as well as spent some time getting used to running queries on my VM. Some problems that I have run into include hunting for formatting options for LaTeX. Though I am familiar with getting a basic document laid out in LaTeX, there is a huge amount of scripting functions that exist and sometimes it can be a pain finding examples and then isolating the part that you need. One of the sources which has been extremely helpful is the, "Not So Short Introduction to LaTeX".

It was hard for each of us to find time to work on the Requirements Document at the same time, but as of Friday morning we finished up a working version of the document. There are so many sections that seem redundant so there is a sense of repetition through the document.

Problems that I faced this week were in formatting the LaTeX output. I was trying to force formatting on the document to fit the picture I had in my mind, however, trying to override the IEEEtran.cls defaults was causing nothing but trouble.

Luckily, our T.A. Jon Dodge was quick to respond to an email that I sent out Thursday evening. He mentioned that looking through the .cls file for areas of the code that pertain to a specific element you are trying to modify (ie. section, subsubsection, etc.) and see what is available through the class options. I was trying to force the subsection headings to have arabic values when they had alpha characters, the method I was using was something I found on stackoverflow and I should not have tried to use it without understanding the effects. Thanks to Jon I will be looking through .cls files for formatting commands.

Moving forward we will be reading The Oracle Optimizer Explain the Explain Plan. This should give us a basic understanding of sorting and merge operations within the database environment so that we can build on some hands on examples with Andy next week.

Nic:
As mentioned earlier, it was difficult to find for all of us to work together on the requirements document which was a pretty big undertaking of it's own due to the complexity of the document. Personally, I was busy with a midterm from another class that coincided with the due date of the requirements document so I was unable to devote a whole lot of time to it. Some progress that I did manage to make this week was to setup a Slack channel for all of us, including our clients, in hopes of being able to establish a better means of communications for when we need a place to all discuss things and collaborate concurrently. Email and texting is generally sufficient but sometimes an instant messaging service is more convenient. I was also able to help a little bit with the requirements document earlier in the week. Going forward, I need to go over the paper that Nathan linked above in order to gain a better understanding of how Oracle 12c sets up execution plans when it receives queries. Knowing this will aid in developing the toolkit that we need to create in order to measure performance of queries.

8.2.5 Week Seven

Nathaniel:
This morning I finished up reading the Oracle white paper called "Explaining the Explain Plan". It was an insightful document that discussed methods of viewing a list of estimated procedural steps the database optimizer will perform during query execution. This can be a helpful tool to use in order to gauge the efficiency of the optimizer and/or where some specific bottlenecks in the processing may be.

This morning I ran into an issue remembering what password I set up for the sys user profile within the 12c environment in my VM. I was not sure if it was set to something default because we were not worried about security, but it turned out I made the password the same as the oracle OS user account password. After getting that squared away I was able to use a SQL script provided by Kirby Sand to generate fake data. The next step we should focus on is identifying the various tables within the V$ view. After we locate the tables which contain performance results, we can start developing the performance analysis toolkit.

Nic:
Progress: The primary task for this week was to read Explain the Explain Plan which describes how the database optimizer breaks down and processes queries. This document contains a lot of important information and will prove useful later on when we get to configuring the database to perform more optimally for our given data sets. At our meeting on Wednesday we were all given a script which will generate a data set for us to work with in the future.

Next Week: Our plans for next week will be to continue to read and understand how the database optimizer works. Additionally, we should also become more familiar with all the V$ and X$ tables within the database. These tables are what hold all the information that we will need for generating performance metrics.

Problems: I do not recall running into any significant problems this week.

8.2.6 Week Eight

Nathaniel:
Last night was Sunday 11/13/2016, I spent most of the afternoon and evening finalizing my section of the technical

review as well as the introduction, abstract and general document formatting. I spent additional time citing each of the web based sources that I used in IEEE citation format. It appears that the document which is due today, Monday 11/14/2016, only needs to be a draft. I know that this has been the case with our other documents, however, I assumed that we should be moving away from that option. If all that is due is a draft, then I am feeling good about my section. Before it is turned in tonight by midnight, I would like to do the following:

- Mention in the conclusion why PostgreSQL and Redshift were not chosen. This could mention that Redshift did not support many of the modern features from updated versions of PostgreSQL. Also, mention something about the recent update for PostgreSQL that attempted to patch a data corruption issue (That would not be good for a corporate environment).
- Decide whether or not to scrap the Introduction section. The abstract can probably serve as a document outline, as well as the table of contents. However, it would still be good to have a global conclusion or something.

We were able to finish and clean up the Tech Review document, and I feel that we will do well on it overall. It was refreshing to see that we got another A grade on a large paper (Requirements Document). Since we had class cancelled on Thursday, I made a brainstorm list of design based choices that would dictate our groups path forward in terms of manipulating parallelism and partitioning related setting in the Oracle environment. There were no specific problems which were experienced during this week.

Nic:
Most of the beginning of the week was spent working on the tech review making sure that it was complete and that all the assignment requirements were appropriately met. Our weekly meeting with our client went well and felt much more productive than past meetings since we weren't bogged down with getting input on yet another writing assignment. We were able to start getting acquainted with the V$ tables in the database and start getting basic metrics such as database time and wall clock time for a particular query. I do not recall running into any significant problems this week.

8.2.7 Week Nine

Nathaniel:
This week we did not encounter any problems. During our meeting with our clients we were able to work on my queries on the dynamic performance view table within our Oracle instance. It was nice to have Andy back from hunting so we could ask his questions.

Moving forward we need to begin focusing on the design document. It is difficult to get started, even by consulting the IEEE document, this assignment seems very repetitive. Since we have a research based project, it is hard to plan out the entire design phase because we are not sure what direction we will be taking after the next couple weeks of thorough research.

Hopefully we will be able to finish through the documents we need to write this term and then transition nicely into the SQL tool development. As of now, I am not sure how well this will work out.

Nic:
Problems: Our main problem was just getting a start on the design document since it did not seem to fit our project very well. The IEEE document that we were to base our design document off of had a lot of content and expectations in it for a simple .zip folder of SQL queries for performance monitoring.

Progress: We made some progress during our meetings with our client by working with some more of the V$ tables within Oracle 12c. These will come in handy later on when developing the toolkit.

Plan: Our plan for the week is to continue working on the design document.

8.2.8 Week Ten

Nathaniel:
Progress: We started focusing on the topic of tracing a query. It is basically a process that documents the processing decisions in a log file. This can be used to generate additional statistics about file I/O and other useful metrics. We were able to get the design document done, but we were not able to get signatures, that will be taken care of Monday of finals week.

Problems: The deign document was pretty difficult to finish. I was having a hard time generating content for some of the design viewpoint sections. It is nice that the design document was the last technical writing assignment for a while.

Plan: Moving forward we will be focusing on the progress report and presentation. Since all that we have done is write papers, the paper will likely be pretty boring. In terms of development, I think we are going to focus more on tracing.

Nic:
Progress: This week primarily consisted of working on the design document and getting it finished up. In our meeting with our clients this week we were introduced to tracing which is another method of monitoring database performance. Tracing involves turning on certain parameters in the database to enable verbose logging of different levels of activity depending on which parameters were enabled. Oracle 12c was designed from the ground up with logging in mind so tracing is a very powerful tool for monitoring what exactly the database is doing in addition to the built in performance tables.

Problems: We did not really have any new problems this week. We continued to struggle with creating the design document but we were able to get it done in time.

Plan: The plan for now is to begin compiling our blog entries into our progress reports. After this is done, we are planning on creating a presentation from the compiled progress report.

8.3 Winter Break

Nathaniel:
During the break most of my efforts were spent on implementing trace functionality within our toolkit app. Additionally, I set up an external tablespace within the Oracle DB that launches a preprocesssor script that runs the recent tracefile through a summary tool called tkprof, the result can then be queried. The plan is to use the summarized output to display a diagnostic report within the toolkit app interface.

More work was done that focused on how we can make stored procedures available to the app, options considered include:

- Storing the procedures on the remote file system in a specific directory for utilities
- Create stored procedures that can be executed from the query workspace

We meet with the client one time during break, and all the work that was done in terms of development and research was done remotely. Now that winter term has started, our team needs to hit the ground running. Our clients will be giving a presentation at an Oracle convention in Texas during the window of February 27th to March 2nd. They plan on using some of the research that we have provided during the presentation, we also need to provided quantitative evidence of different system configurations.

Problems: Some of the major problems that I dealt with during break included:

- Issues with JS asynchronous callbacks and wrapping a trace wrapper around the user query
- Figuring out a way to move all the tracefiles from one session into a specific directory and giving the summary a unique stamp

- Figuring out how to use the external tablespace concept to help us display reports to the user

Progress: All of the issues that were run into were resolved. In the current state, the user can submit a query, choose optional trace event codes, and select from a list of performance metrics. The output needs to be reformatted so that each desirable result can be displaying in a dashboard style.

Plan: Moving forward, we are not going to be spending any more time on developing the toolkit app. All efforts will be placed on initial research of configurable parameters that may help optimize PGA usage, in turn reducing the need to swap to temp. These parameters will then be adjusted and the runtime analysis will be used to quantify the improvements from the base configuration with default values.

In terms of the toolkit app, we need to consider:

- Preventing malicious SQL execution (ie. reject DROP/INSERT/UPDATE clauses)
- Update the results page to better display the final output, including the diagnostic reports and statistics
- Move procedures to server file system, or create stored procedures within the DB that can be called from the SQL workspace

Nic:
Progress: At the beginning of Winter break we had met with our clients at HP to discuss topics to research and focus on over the break. I also showed our clients the beginnings of our toolkit that we were going to create as part of our deliverable. Our main area of interest over break was to experiment and get familiar with the various trace events within the database. These trace events will be useful to us in our research since they help detail how the database makes decisions internally when it comes to executing user queries. We were thinking of integrating trace event functionality into the toolkit to facilitate obtaining trace results. Near the end of the meeting I had an idea to help temporarily alleviate the issue of excessive temporary table usage (swapping to disk) for certain types of queries.

I spent the first half of Winter break mostly focusing on making progress in our toolkit and also experimenting with a ramdisk for the database. The ramdisk that I created was to be used for the temp tablespace for the dataspace. The temp tablespace is what is used for when the individual database server processes run out of memory and so they can essentially swap to disk by swapping to the temp tablespace. My idea was to move the temp tablespace off disk and into memory by using a ramdisk as a stopgap solution until we find the real solution through experimenting with internal database parameters. This idea proved to be very effective in improving response time for queries which originally spent a large portion of time swapping to and from temp.

Later on in the break it was determined that we should stop focusing on the toolkit and to really start focusing on database research. This is primarily because our clients are going to be presenting our collective findings (between us and our clients) at an Oracle convention by the end of February. This only leaves us a little less than two months to figure out why the database isnt performing optimally, document our solutions, and create publically reproducible test cases.

Problems: I had some problems with the toolkit early on, mainly stemming from some design issues. Originally the toolkit that I created was meant to be a quick demo and so naturally some aspects of it were rushed and looked over. My primary problem with the toolkit was the server returning non-descriptive results which were displayed in a hard-coded fashion in the actual web application. I fixed this by redesigning how the server returns results back to the web interface so that it is more generalized. The server can now return named datasets each containing some metadata on how the results should be processed and shown to the user. However, after we decided to halt progress on the toolkit, our main problem is now time and the fact that we still dont really know how the database works internally.

Plan: The plan now (as of the end of Winter break) is to focus on figuring out how the database works so that we can tune it to perform more optimally for our client. Some key research points will be figuring out how parallelism works, how database processes use memory, and how to best optimize these two.

8.4 Winter Term

8.4.1 Week One

Nathaniel:
This week we focused on breaking our research efforts into a few high level questions, the three we came up with Wednesday were:

- How is the DOP of a query executions actually determined? (James)
- Explain the relationship of the PGA and its sub-components. (Nic)
- Figure out how to extract clear diagnostic information from a query. (Me)

Problems: There are two trace event codes 10390 and 10391 that generate run time information about parallel processing. Since the system we are working on nearly always runs in parallel, it is important to try to extrapolate this info so that it can be used to validate our run time model between the PGA use, wall time, and DB time. Though I have been able to successfully generate reports for many of the other trace event codes, 10390/391 has it's levels of granularity expressed in hex, where all of the others are expressed in decimal. One of my tasks was to profile all of the individual levels of both problematic trace events. I ran into major issues getting output files from the trace request.

Progress: In order to try and resolve this I converted the hex values to decimal and tried that. The other available trace event codes have a flagged numbering system that resolves to a decimal. A value of 0 is basically off, 1 value initializes the trace, a 2 value sometimes adds a layer of complexity, and 12 value run both of the previous layers. This is not something that I am giving up on, but it is been a major pain.

Plans: Plans going forward will be to finish reading the Oracle whitepaper on parallel execution, the goal is to pull out some important factoids that Kirby and Andy can use in their presentation. Also, I want to experiment with the query that Kirby put together to grab run time statistics from a query. Then I would like to make a couple parameters of choice and run then at various values to track the effect on the query performance, the initialization parameters we are most interested are:

- _pga_max_size
- _smm_max_size
- _smm_px_max_size

and possibly:

- sort_area_size
- hash_area_size

Nic:
Progress: We met twice with our clients during our first week of the term. During our first meeting, we had a discussion on what we all did over the break and presented our findings. We also more formally went over our overall research plan and broke it up into areas that were assigned to each of us (mentioned by Nathaniel above). I was assigned to documenting and creating visual diagrams of the PGA and its relationships with its various subcomponents.

In between our two meeting we all did some beginning research on our respective topics. I was able to document the components within the PGA and identify some internal database parameters that govern them. We each presented our findings during the second meeting. By the end of the second meeting we came up with a list of parameters to focus on that were related to PGA.

These parameters are:

- PGA_AGGREGATE_LIMIT

- PGA_AGGREGATE_TARGET
- _PGA_MAX_SIZE
- _SMM_MAX_SIZE
- _SMM_PX_MA_SIZE

Problems: My primary problem with my assigned area was I was at first a little unsure of the exact requirements of what I was supposed to be doing (as in how to create the diagrams and such) after the end of the first meeting. This turned out to be a non-issue after confirming my results with our clients during the second meeting. The main issue now is still the fact that we dont have much time left to get this all figured out, but at least we have a better idea of what to focus on now.

Plan: The plan now is to further explore the relationship between PGA and its subcomponents and also experiment with the parameters above. I hope to have some material ready for our next meeting detailing how the parameters affect the subcomponents within the PGA and ultimately how they affect performance.

8.4.2 Week Two

Nathaniel:
During this week we had our first T.A. meeting for the term. I brought up the fact that our efforts will be focused on research rather than actual product development. Jon Dodge, our T.A., requested that we get an email from our clients that outline the fact that they don't expect us to deliver them a software product, rather, they want us to put together a presentation with them and produce experimental evidence that supports our claims. After telling Kirby about this, he was happy to send out the email.

This week was somewhat slow. I read through Oracle documentation on Parallelism and Partitioning as well as looked through info on credible Oracle Ace or Oak Table sites. These sites are hosted by individuals with paid or non-paid support from the Oracle company.

Problems: One problem that I ran into was getting false positives for 10390 391 trace events. I tried to wrap a series of trace headers around a query in hopes to generate series of different output, however, the 10046 output was just spread through the neighboring results. Also, on January 11th I had set an initialization parameter, Memory_Max, to a size larger than the available resources. In light of this, once the database was shutdown it would not start back up. Andy, one of our mentors at HP was able to help me get the issue resolved.

Progress: This week I made a graph that outlines that effect setting pga_aggregate_target has on the _pga_max_size, _smm_max_size, and _smm_px_max_size parameters. Though this graph, we were able to derive formulas for how the later three parameters are calculated. This gives us a glimpse on what the desired ratio is in the native Oracle 12c setup.

Plans: Going forward I will be writing two scripts, one that will reset the DB parameters back to the desired baseline values, and another that prompts the user for input. The latter script might be constructed first with substitution variables, but also I might make a command line version that will just accept a series of arguments.

Nic:
Progress: This week I was primarily focusing on cleaning up the slides that I had prepared last week. I was tasked with updating them with our newly found information regarding PGA parameters that Nate had found specifically for Oracle 12.1c. I was also adjusting the coloring a little so that the colors of certain elements were more consistent from slide to slide (e.g. if something is colored red on one slide, that same thing should still be red on the next slide).

Additionally, I started looking into In-Memory Parallel Execution more to get a better idea of how to better set up the database to take advantage of parallel execution without having to resort to swapping to memory (which is a problem at the moment). I was able to find out some parallel execution related parameters that govern whether or not In-Memory Parallel Execution is used. Since I was on that topic I decided to look into other parallel parameters that determine additional parallel execution properties.

Problems: I do not recall encountering any significant problems this week.

Plans: Our plans for next week are primarily to determine how the System Global Area (SGA) works within the database and also how it is used for parallel execution. I will probably get some slides set up detailing the components within SGA.

8.4.3 Week Three

Nathaniel:
This weekend I read through the Oracle whitepaper on In-Memory. Many of the insights and notes that were taken have been placed in our OneNote workbook.

Problems: There were no major problems that I encountered this week.

Progress: Initial research was completed on the topics that will be the focus of experimentation during week 4. As stated above, the notes taken from the research process have been documented on the OneNote workbook.

Plans: Next week I would like to begin some targeted experimentation to test if we can exercise the benefits of In-Memory storage (or In-Memory Parallel Execution). Additionally, it would be nice to experiment with Automatic Big Table Caching.

Nic:
Progress: I spent some time detailing the various components in the SGA in order to identify which components are involved in parallel query processing. One of the components of the SGA that stands out is the Database Cache which contains individual caches for blocks of data based on block size, as well as an optional Big Table Cache. The Big Table Cache can store entire tables and partitions rather than individual blocks. Additionally, I read more into the In-Memory Column Store and performed some testing on that. At first my results were inconclusive and using the In-Memory Column Store did not seem to offer any performance benefits. However, after testing different schemas and queries I was able to get about a 60% performance increase in my preliminary testing.

Problems: I don't recall any significant problems other than initially not seeing any performance increases with the In-Memory Column Store.

Plans: The plans for next week are to hopefully get some early slides made for SGA and continue testing the In-Memory Column Store.

8.4.4 Week Four

Nathaniel:
This week I spent a lot of time reading though In-Memory white papers from Oracle, links to both papers can be found below. My focus was on gaining an understanding of how Oracle 12c In-Memory works under the hood. Besides gaining a basic understanding of how In-Memory works, I also focused on identifying areas to experiment with for potential performance gains.

- In-Memory Overview
- When to use In-Memory

During our Friday meeting we were asked to put together a series of scripts that would facilitate a test suite and resource monitor package. Andy outlined what he had in mind on the board, I started trying to implement his vision with shell scripts and Nic began developing in node.js. By the end of the meeting we both had a basic working system.

Problems: The main problems that I ran into were in getting an initial connection set up to the capstone database with bash. After a bit of troubleshooting the problem was resolved.

Progress: By the end of the meeting I had a three scripts put together driver.sh, runMonitor.sh, and runTest.sh.

- driver.sh: Prepares the environment and calls the two run scripts
- runMonitor.sh: Establishes connection to capstone server and executes resource monitor package loop
- runTest.sh: Establishes connection to capstone server and executes test suite

On Saturday I spent more time cleaning up and commenting what I had originally. I also set up an issue in the issue tracker that outlines my To Do list for this script.

Plans: I plan on getting feedback from the group and our mentors. These test facilitators will be used to run nightly tests to gather data we will use to validate our performance goals/claims.

Nic:
Progress: Read up some on how our queries could potentially benefit from enabling the In-Memory Column Store. The primary advantages are that the IMCS format is optimized to take advantage of SIMD vector processing which allows the CPU to perform operations on multiple values at once (similar to a GPU, but much smaller scale). The most often used columns within the most commonly accessed tables could be set into the IMCS and offer us better query performance. I have done some preliminary testing that showed about 60% of an improvement in query response time when enabling the IMCS for a particular table and query. During our last meeting of the week I got the development server at HP setup with NGINX and NodeJS in order to host our web based toolkit. My plan is to have the web based toolkit wrap around and utilize Nathan's bash and SQL scripts.

Problems: Did not encounter significant problems this week, other than initially having some trouble getting HP's development server setup with NGINX and NodeJS. The server sits behind a proxy which I was not initially aware of, so that caused issues with trying to download anything until I specified the correct proxy to use.

Plans: My plan for now is to put a bit more work into the web based toolkit and then we will probably all start focusing on testing SQL query performance with varying database configurations.

8.4.5 Week Five

Nathaniel:
This week most of my focus was place on implementing the series of shell scripts that work together to facilitate test suites that will be run during the evening. We will be using this tool to run nightly experiments in the following three experiments:

- PGA Experiment: Focused on altering the values of '_pga_max_size' and monitoring the potential spill over to temp during operations such as sort, hash, and window sort
- Big Table Cache Experiment: Focused on altering the percentage of the buffer cache that caches large objects in memory and monitoring potential read performance
- In-Memory Experiment: Focused on enabling In-Memory column store and monitoring the potential performance improvements

Problems: The problems that surfaced this week included the initial version of the test automation scripts ended up not working as planned. One connection is established to run a monitoring loop in the background, when another connection is established and the command is issued to close the pluggable database the monitoring loop was terminated.

Progress: During the meeting we talked about the logic needed to resolve this problem and began modularizing the sql experiment scripts so that they can be run in two phases: if there is a parameter file then run that in a separate connection, run sql file.

Plans: My plans moving forward are to finish and test the test automation script and run all three experiments before Monday so that we can look at the data during the meeting Wednesday during the meeting. The data must be graphed and prepared before that time as well.

Nic:
Problems: No significant problems for me personally, however as a group we ran into issues with previously ran tests where were getting some unexpected results (ex: enabling features to increase performance sometimes led to sharp unexpected decreases in performance or no improvements at all).

Progress: This week I primarily worked with Nathaniel a little bit to help get the script working in some parts. I also refactored one of our tests (IMCS) to fit a newer more modular format to work with the updated testing script. As the team captain I also got the expo registration details sorted out.

Plans: I don't have any immediate plans as of this moment, however as a group our plan is to continue creating, performing, and analyzing tests using our test automation script. After we look over the results of the IMCS test we will probably end up devising further tests to test both the IMCS and Big Table Cache concurrently (since they are designed to work in tandem with each other). Right now we are only testing them separately to help gauge their individual effects before we start looking at their combined effects.

8.4.6 Week Six

Nathaniel:
This week we were able to get all three of our experimental suites to run and collect valid data on all of them. I also spent time troubleshooting the test automation scripts as well as make slides that detail the memory pools within the In-Memory column store.

Problems: Early in the week we ran into issues getting the IMCS experiment to run correctly. During our meeting on Friday we condensed the directory of parameter and sql files into three total files, one parameter file and two sql files. It turned out that all four of the experimental tables could not fit in the IMCS when it was set at 192G and the objects were flagged as NO MEMCOMPRESS.

Progress: Sunday evening I finished up the rough draft of my IMCS pool slides and also made a couple of graphs that detail the effects of the six different compression levels offered in Oracle 12c. Additionally, I made a graph that shows the compression ratio for each of the four experimental tables.

Plans: Moving forward Nic and I will be helping Kirby and Andy prepare anything else they need to make the presentation go smoothly. This will likely include generating more slides on the topics we experimented on, as well as some additional experimentation.

Nic:
Problems: We had some issues with our script where the column store did not finish completing before any other scripts were ran.

Progress: This week we were mainly focused on completing the IMCS test and making sure that the column store was being populated before the IMCS tests were ran.

Plans: Our plan is to get the IMCS test finalized over the weekend, and then from there we can focus on creating some more slides. We are nearing the end of the critical point of our project and so really we just need to finalize everything (mainly presentation details). I'm going to focus on vector processing for slides.

8.4.7 Week Seven

Nathaniel:
This week our clients were in Austin Texas at an Oracle convention, while at this event they gave an hour long presentation that outlined the experimental research that we have been working on. Tuesday evening I got a phone call from Kirby saying that the presentation went really well and they were asked to come back again next year.

Nic and I still met on Wednesday for three hours and Friday for two, in order to work on our Design Document and begin writing up our progress report. This will be the mid point progress report that will outline the end of Fall term to week seven.

Problems: There were not any significant problems this week, rather, after beginning to work though the documentation it became clear that it was written in a way generalized enough that it still fits that scope of our project; therefore, we will only need to make minor adjustments.

Problems: The only problem that we ran into this week was in getting the statement queuing experiment to run correctly. There was some confusion on exactly what to modify within the design document, after reading through it there were only a handful of modifications that were needed.

Progress: The revisions that were made on the Design Document were sent to Jon Dodge towards the end of the working meeting that Nic and I had Wednesday, this should take care of the revisions that we needed to do for our fall documentation. Friday we made bulleted lists in our OneNote that outline our progress up until week seven. Now the plan is to formalize the timeline so that we can submit it.

Nic:
Problems: No problems this week.

Progress: Our clients had given their presentation at HOTSOS and it went well. We did not meet this week with our clients and so not much progress was made since we more or less had this week ”off.” Nathan and I had met to finish up working on our document revisions now that we had time to focus on that.

Plans: Finish up documentation revisions.

8.4.8 Week Eight

Nathaniel:
This week our clients were in Austin Texas at an Oracle convention, while at this event they gave an hour long presentation that outlined the experimental research that we have been working on. Tuesday evening I got a phone call from Kirby saying that the presentation went really well and they were asked to come back again next year.

Nic and I still met on Wednesday for three hours and Friday for two, in order to work on our Design Document and begin writing up our progress report. This will be the mid point progress report that will outline the end of Fall term to week seven.

Problems: There were not any significant problems this week, rather, after beginning to work though the documentation it became clear that it was written in a way generalized enough that it still fits that scope of our project; therefore, we will only need to make minor adjustments.

Problems: The only problem that we ran into this week was in getting the statement queuing experiment to run correctly. There was some confusion on exactly what to modify within the design document, after reading through it there were only a handful of modifications that were needed.

Progress: The revisions that were made on the Design Document were sent to Jon Dodge towards the end of the working meeting that Nic and I had Wednesday, this should take care of the revisions that we needed to do for our fall documentation. Friday we made bulleted lists in our OneNote that outline our progress up until week seven. Now the plan is to formalize the timeline so that we can submit it.

Nic:
Problems: No problems this week.

Progress: Our clients had given their presentation at HOTSOS and it went well. We did not meet this week with our clients and so not much progress was made since we more or less had this week ”off.” Nathan and I had met to finish

up working on our document revisions now that we had time to focus on that.

Plans: Finish up documentation revisions.

8.4.9 Week Nine

Nathaniel:
This week we started looking into two new features of interest for the clients, indexing and partitioning strategies. Our clients are experiencing slow DML operations during the nightly job. Approximately every seven days, a table called the data loader dumps its contents into a table called data reader. Data reader can be a 500GB compressed table with 500GB of indices. The problem is that inserts into an indexed table are extremely costly due to all of the write operations to the redo and undo tables. A rule of thumb is that for a table with two column index, an insert that would take a single instruction will take four on an indexed table, thus the root of the problem.

Problems: The only problems that I ran into this week were in figuring out a reasonable way to try to simulated the data loader and data reader tables within our test environment. I also am still not entirely clear on how many columns in the data reader table are indexed, as each additional index adds a cost factor.

Progress: I have created a data loader and data reader table, the data loader table has all rows from device id 17 from the last two months, and the data reader table had the remaining four months. Based on this quick and dirty experiment, I was able to observe the cost of inserts on an indexed table.

Plans: Moving forward, I need to make sure I have a solid dataset prepared by Wednesday. Emphasis needs to be placed on getting a primary key relationship between the device id and timestamp, however, since our data was randomly generated with a tight range of values one could not be set up initially due to non unique entries.

Nic:
Progress: With the presentation over with and bulk of our research done, we decided to keep on researching more aspects of the database that could further improve data performance for our clients' nightly jobs. Since the tables involved are very large and thus have very large indexes, it takes a while to insert new records into the database (part of the reason is maintaining these indexes). Two of the key areas that we are going to focus on now are partitioning and indexing in the context of bulk inserts (from nightly jobs) on very large tables.

Problems: The main problem that we are focusing on now is how to alleviate performance problems associated with bulk inserts on very large indexed tables.

Plans: My plan is to research database partitioning and try to figure out an optimal way to partition the data. Right now the debate is partition by date and subpartition by device id and vice versa.

8.4.10 Week Ten

Nathaniel:
Most of our efforts this week have been placed on getting our progress reports and poster together for finals. I have not continued setting up the index experiment because I have a lot of overhead with finals this week, once I am done with my exams on Tuesday I will start up again.

Problems: No major problems were encountered this week.

Progress: Our poster rough draft came together quite well. Some of the working will need to be carefully read through and made more succinct.

Plans: Moving forward, I plan on focusing on finals until Tuesday evening, then I will start up on the index experiment again as well as finish my progress report for CS 462.

Nic:
Problems: The problem for my portion of research is still whether or not we should be partitioning by date and then subpartition by device id or partition by device id and subpartition by date. So far we know that the bulk inserts portion of the nightly job would favor one partitioning scheme while the nightly aggregation portion would benefit from the other. Overall, we are looking for the best solution and compromise to suit both.

Progress: This week we had put together and submitted the draft for our poster. We were able to get most of the details that we wanted to in the poster, minus the final results section since we technically still aren't done with our research now that we have started exploring more options. Similar to Nathan I have been busy with finals, mainly with other final projects, that are due at the end weekend of Week 10 and so I haven't had a whole lot of time to devote to research recently. I have made some headway on partitioning over the past couple weeks but not as much as I would like.

Plans: My focus for now is to get my finals out of the way and then I will have more time to dedicated to researching partitioning.

8.5 Spring Term

8.5.1 Week One

Nathaniel:
Most of my efforts this week were focused on transitioning back into school responsibilities. After working more than 40 hours over break, it took some time to switch gears. There was not much done on the project during spring break and we only had a single meeting with our mentors.

Problems: There were no problems during this period.

Progress: There were no obligations during the break, therefore no true progress was made.

Plans: Moving forward I plan on writing up four different experiments that leverage the partitioning strategies that Nic has put together and apply the indexing experiments to each. The indexing strategies I am looking into are:

- Global index
- Local index (Prefix/Non-prefix)

Also, I plan on looking into the effect of different sizes data loaded into the main table (ie. 2%, 5%, 10%) and look into overlapping data that exists in both tables to observe the effect it has on weekly load performance.

Nic:
Problems: We did not officially setup a meeting time for this week and so we did not meet this week. Because of this, we didn't get much done.

Progress: Not much progress was made for me this week.

Plans: Meet next week and go from there.

8.5.2 Week Two

Nathaniel:
This week most of our efforts were focused on finishing the experiments focused on indexing and partitioning strategies. The initial design of experiment that we put together has been complexified by our clients, they would also like us to focus on the different methods of uploading overlapping data.

Problems: The only issues that I ran into this week was not having enough memory on our test server to create the two billion record table that our clients requested. It seems like the issue is a physical memory limitation, rather than a disk space issue.

Progress: During the terminal weekend for this period I spent quite a bit of time writing up the indexing experiments for each of the unique partitioning strategies, which were put together by Nic. I was able to run quick and dirty experiments to log the runtime of the different configurations, however, I will need to configure the resource monitor package to track the data points I am interested in monitoring for these experiments.

Plans: Going forward, I plan on altering the resource monitoring loop and completing the modifications requested by our mentors to the four scripts which represent the different partition strategies that we will be evaluating.

Nic:
Problems: I do not recall running into any significant issues this week.

Progress: Nathan has been doing most of the work involving the script to test our experiments. I started to look into clustering to cluster related blocks of data together on disk. There are options to cluster based on columns or join conditions. Additionally blocks can be clustered linearly with each other or interleaved in a zig-zag type order. However, I feel like that would add too many dimensions to our test script and potentially go beyond our scope, especially since we only have a couple weeks left for ”development” until the code freeze before expo.

Plans: We have our testing plan (what we're testing) more or less complete, now we just need to focus on the monitoring script so that we can gather the relevant data that we need to evaluate the performance of our options.

8.5.3 Week Three

Nathaniel:
This week we spent time working on cleaning up the content in our poster. One of the major changes we made was writing the intro to be more succinct. We still need to wrap up the conclusion and get a picture of us, as well as get a signature from our clients.

Problems: No issues were encountered during this period.

Progress: The last draft of the poster we submitted was also sent to our clients so that they have ample time to look over and suggest any changes. We still need to talk with them about an approved company logo.

Plans: Moving forward I plan on finishing the indexing and partitioning experiments Nic and I put together and collect the results of the database and wall time it took for DML operations. Monday at our meeting Nic and I will get a picture together for the poster, and I will ask Kirby about a company logo. We will likely spend some time finishing up the parts of the poster that still needs work.

Nic:
Problems: No big issues during this week other than fixing a small mistake I made in one of the subpartitioning schemes. Originally I had a table partitioned by device id and then subpartitioned by date using hash buckets which didn't make sense. I fixed this so that the dates were subpartitioned into a predefined list of days spanning from the oldest timestamp to newest timestamp. This should be a much more effective partitioning scheme.

Progress: Aside from fixing one of the partitioning schemes, we mainly just worked on the poster this week to clean some areas of it. The introduction of the poster was a little verbose so we tried to shorten it down some. We also sent the poster to our clients for feedback.

Plans: Plans for next week are to start working on a simple web app that we can use at Expo to help visualize some of our experiments. Also we still need to get a picture for our expo poster and finalize the poster before 4/28.

8.5.4 Week Four

Nathaniel:
This week, I worked on finalizing the series of experiments designed to look at the run time of inserting a 10 million row table into a 200 million row table. The goal is to see how different indexing and partitioning strategies effect performance, in hopes to figure out the most appropriate method of inserts for nightly job performance.

Problems: I began setting up this series of scripts like we had set up the initial experiments (ie. PGA, BT Cache, and IMCS), however, the scripts quickly became to long and likely prone to human introduced error. In order to resolve this I made additional scripts to contain only the various tests I needed to run. Six scripts were needed, the first three for when the data loader table has only rows not existing in data reader, and the other three for overlapping data in data loader which already exists in data reader. The purpose of the test types are defined below:

- No or Non-unique index
- Prefix as date for unique and primary key
- Prefix as ID for unique and primary key

Progress: Friday morning at 12:21am I started up the test automation suite and began executing the index experiments on the server at HP. I confirmed that the resource monitoring loop was writing run time data from the experiment to our collections tables. The experiment completed at 7:15pm, if all the tests completed without any syntax errors the entire experiment took nearly 19 hours to complete.

Plans: Going forward I need to confirm there were no issues with any of the tests which were run, which can be done by comparing the number of entries in the data collection tables to the number of tests which should have completed. Also, I will begin gathering data so that we can visualize the results.

Nic:
Problems: No significant problems this week.

Progress: Started working on a web based visualization tool that serves as an extension to our poster for our Expo presentation. It's something that is outside the scope of our project but would be nice to have. The idea is that we can use it to further elaborate on the experiments that are mentioned on our poster. For each test there will be a description that briefly describes the experiment and also any associated graphs to help visualize the results.

Plans: In our next meeting with our clients we will hopefully have some time to go over what information and graphs we want to show for each test. Once those aspects are decided it should not take long to get the web app completed.

8.5.5 Week Five

Nathaniel:
This week we delivered the results of the index/partitioning experiments that Nic and I put together. Kirby was happy with what he saw from the initial evaluation of the results, however, there still might need to be some general house keeping steps before we can be done with that.

Problems: One of these module names for overlapping data was mislabeled as standard query, though it was using a no duplicates hint. This is not a huge issue, as the no duplicate queries were also run in the named section, but rather it just caused some confusion when Kirby started looking through the results.

Progress: All agreed upon deliverables have been received by the client, we were also able to complete some of the stretch goals purposed by our clients.

Plans: Moving forward, Nic and I plan on putting together a simple web application that will allow us to have additional information at expo. This might include, the plots we have on the poster but able to view at full screen, information about the side projects we worked on during the capstone project (ie. Ramdisk and tracing) as well as more general

information about Oracle 12c and the features we experimented with. In doing so, we will be able to have a digital extension of our poster in hopes to provide a clearer picture to those interested.

Nic:
Problems: No significant problems for me other than I got a little busy during the week and was not able to devote much time towards the Expo web app.

Progress: On Monday we met to discuss how we should present our project at Expo and came up with ways to visualize our project by using Legos to demonstrate parallelism, table structure, etc. Aside from that, our project was more or less completed at this point.

Plans: Moving forward my primary goal is to work on the Expo web app.

8.5.6 Week Six

Nathaniel:
During this week, all of our efforts were directed towards preparing for expo. Our poster had already been submitted, but we still needed to prepare some pitches for people from varying technical background.

Problems: It was difficult to come up with a way to describe our project to a five year old or a grandmother.

Progress: We were able to get around this by creating an analogy comparing the database to a toy chest or a bank. We could focus on the organization aspects of a toy box for efficiency purposes, and focus on transaction latency in terms of the bank. Both of these examples can be used to demonstrate some of the need for our project.

Plans: Moving forward, Nic and I will be finalizing the expo app and further preparing the talk to people about our project by the end of next week.

Nic:
Progress: This week we primarily just focused on getting ready for expo, part of that was coming up with pitches for various audiences.

Problems: Coming up with pitches to describe our project to less technical audiences was challenging.

Plans: We needed to start finishing up the expo app that we would use as a poster extension.

8.5.7 Week Seven

Nathaniel:
This week we finished up the expo app and further prepared mentally for expo, it was important to come up with a loose script that we could follow when giving a brief overview of the project.

Problems: No problems were experienced during this period.

Progress: The app was finalized and we wrote up some pitch lines for multiple situations.

Plans: Friday is expo, the plan are to be ready and properly represent ourselves and the work that we have done.

Nic:
Progress: We finished up the expo app earlier in week by filling out the details of our experiments. Then Expo happened on Friday.

Problems: None.

Plans: None after expo.

8.5.8 Week Eight

Nathaniel:
Expo went great. It was somewhat laborious saying the same thing over and over again, but after the first couple of iterations we were able to get into a flow.

Problems: One thing that was unforeseen this week as that when we made the plots bigger on our poster, we accidentally covered up the bottom half of one of our sentences. This was not a big deal at all, but we did have someone comment about it.

Progress: We have completed all of our clients requested deliverables and made it through expo.

Plans: Before the end of the term we have three, or more, writing assignments. We will get more information later this week during one of our scheduled classes. Our efforts will be concentrated on finishing these promptly and completely.

Questions:

- If you were to redo the project from Fall term, what would you tell yourself?
  - I would probably tell myself to vote for James, our lost comrade, as the student I did not want to work with. The reason being that he did not contribute to our overall project, other than some documentation in fall, and it would have been better to have a full team. Nonetheless, Nic and I work really well together and we were able to take care of everything without another person. I would also tell myself to start a OneNote log of research at an earlier date. Our team could then rely on this for communal knowledge. This document could then be given to the clients for future reference.

- What's the biggest skill you've learned?
  - Proficient use of SQL. Based on the experience from the two database classes I have taken, I would say that I did not have the required knowledge to tackle some of the things we turned out during this project.

- What skills do you see yourself using in the future?
  - Definitely the skills I have gained in terms of SQL, but also the communication skills that were required to interact with our clients at a technical level is something that I am greatly appreciative of.

- What did you like about the project, and what did you not?
  - I really enjoyed the fact that I did not have a strong background in this field, this project was a growth period where I was able to broaden my scope of CS comfortability. Our clients were very supportive and made sure that we had everything we needed, they were also available after work hours. Initially, the project seemed like a big undertaking. The scope of what our clients wanted was quite broad, such as they did not know exactly what they wanted, this caused some initial stress early on. Also, the emphasis on documentation in the fall did not jive that well with what our clients were requesting.

- What did you learn from your teammate?
  - Nic was a wonderful student to work with. He is somewhat of a quite guy, but he brings a lot to the table. I think that the thing I learned most from Nic is trust in a group member. Early on it was quite dicey with communication between our group, this was mainly caused by James' lack of participation. It was great to have confidence in another teammate as someone who can stand on their own and produce equal if not better work. Without Nic on this project, I am not sure how things would have went.

- If you were the client for this project, would you be satisfied with the work done?
  - Absolutely, upon our first interactions with the client it was clear that they knew what there issue was but did not know how exactly to proceed. Nic and I were able to complete all agreed upon deliverables by the tail end of winter term, and then we started working on a completely different issue. Everything that was requested was done well before expo, if I were a client in a similar position I would be completely satisfied with our teams work.

- If your project were to be continued next year, what do you think needs to be working on?
  - Based on the last experiments we worked on, Indexing and Partitioning, there could be some work on correcting the naming convention issues in the test scripts. Additionally, emphasis could be placed on looking into bitmap indexes and bitmap joins, these implementations are said to be effective for data warehousing, however, we did not have time to set anything up to gather results.

Nic:
Progress Expo went well for us and we had finished our project objectives beforehand.

Problems No significant issues.

Plan Survive the rest of the term.

- If you were to redo the project from Fall term, what would you tell yourself?
  - When I first started the project I was taking notes with Google Docs during our client meetings. If I had to do it over again I would definitely use OneNote as far as notes and research goes. My notes on Google Docs weren't well organized and I had a separate document for eaching meeting which made going through all my notes a pain. Later on we switched to OneNote exclusively for our research and that worked out really well for us. Other than that I can't really think of anything that I could have told myself at the start to make the project easier because there were a lot of unknowns early on (e.g. what we would actually be researching, it was a learning process for all of us).

- What's the biggest skill you've learned?
  - When I first started this project, the idea of learning Oracle database seemed daunting since I didn't really know much about relational database management systems aside from basic setup and usage (setting up MySQL, basic queries). However, from just persistently researching and asking questions when needed for guidance we all learned a lot as we progressed through the project. More specifically, my mentors had taught me a lot about how Oracle database works (the architecture of it, how all the caches work, disk I/O, what actually happens when a query runs, different types of sorts/joins, etc.) but a lot of the concepts from Oracle database also apply to other RDMSs as well.

- What skills do you see yourself using in the future?
  - Before when I worked with databases in the context of an application I always treated databases as a black box that I send queries to and it returns data back to me. Now that I have a better understanding of the internal workings of a RDMS, I can use this knowledge going forward to use databases more effectively (how to diagnose slow performance, performance tuning, writing better queries, and so on).

- What did you like about the project, and what did you not?
  - I liked working with and exploring the internals of Oracle database/databases in general. The one downside of this is while Oracle database itself is pretty nice, the licensing is prohibitively expensive unless you're a company pulling in some decent revenue. So it was cool learning Oracle database but it sucks knowing you can't use it and it's cool features for personal projects unless you're willing to pay the average yearly salary several times over. I also feel that the documentation from Fall term did not really go with our project at all especially since we weren't even sure what we would be researching at the time.

- What did you learn from your teammates?
  - This group project was different from ones that I've been involved with in the past in that it was a year long and was a real project. Despite losing one member due to lack of participation, we were still able to pull through successfully and achieve the goals laid out by our clients. A large part of our success was Nathaniel being a proactive team member and a hard worker. We both learned from each other as we took on research topics individually and shared our findings. One thing I learned is if a team is short on members, you can still pull through if you trust your team and if everyone puts in effort.

- If you were the client for this project, would you be satisfied with the work done?
  - I would say so. We started off not knowing much at all about databases and were able to learn a lot about Oracle database throughout the project. Ultimately we were able to increase query performance by at least 4x, and increased overall system utilization (very little was being used before) compared to our baseline configuration.

- If your project were to be continued next year, what do you think needs to be worked on?
  - Some more topics that could be addressed are: bitmap indexing, attribute clustering, schema design, and other techniques that are related to increasing

9 POSTER

On the following page you will find the poster that we created for senior design expo. This poster details some of the main experiments that we conducted for our clients.

BIG DATA ANALYTICS:

Effects of Memory Management and Parallelism

Drive your server like you stole it

Background

Computational efficiency and performance are paramount to being able to quickly and effectively analyze data within a reasonable time frame when working with very large data sets. Oracle 12c performs sub-optimally out of the box when it comes to analyzing the massive amount of telemetry data gathered from HP PageWide Web Press printers. Oracle database is instead setup for Online Transaction Processing (OLTP) workloads by default, which caters to handling many simple queries returning relatively small amounts of data.

Our clients current configuration does not perform optimally for Decision Support System (DSS) workloads, which involve analytical queries performing aggregations over very large sets of data. Because of this, our clients were experiencing very poor database performance for their use case despite having a very high performance server.

Project Description

The primary purpose of this project is to research concepts of Oracle database relating to increasing query performance for DSS workloads and experimenting with turning internal performance knobs in the database. Our goal is to identify what knobs to turn and what features to use to increase query throughput. In turn, this will let the database use the server's resources more efficiently to ultimately reduce query execution time.

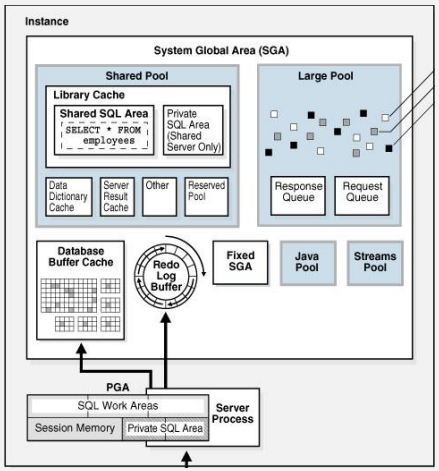


Figure 1: Database Overview

Program Global Area (PGA) Parameters

Emphasis was placed on both documented and undocumented parameters focused on managing work area size. A specified work area provides the allotment of main memory used to complete an operation such as sort or hash. If a work area size is surpassed, the operation spills to disk to complete the work which causes drastic increase in execution time.

Automatic Big Table Cache

Oracle 12c has a new feature which uses a temperature based algorithm to determine which tables considered large should be stored in a cache store located in the SGA. When enabled, stores table after initial scan so benefits are seen upon replicate samples.

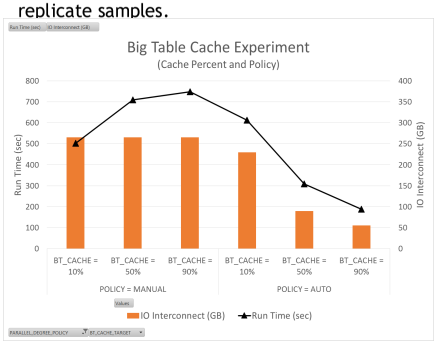


Figure 2: Effect of buffer gets on IO interconnect and runtime

In-Memory Column Store

Oracle 12c also has a new feature which stores data in columnar format. Objects as specific as columns from a table can be flagged for storage. Blocks from In-Memory storage can be scanned without decompression. Figure 3 shows how the run time for highly compressed tables increases when stored in the IMCS.

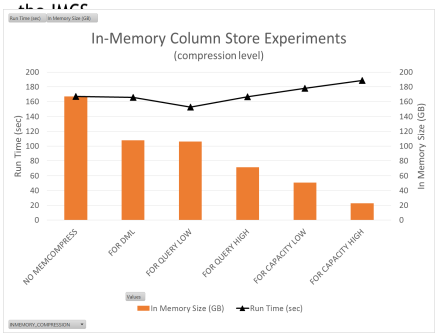


Figure 3: Effect of various compression rates on run time

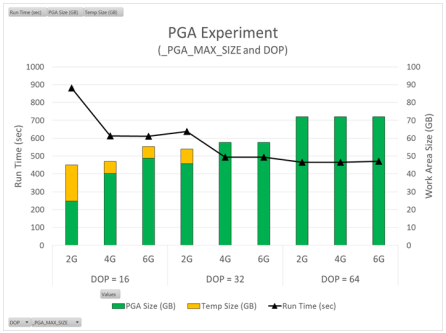


Figure 4: Effect of degree of parallelism and PGA size on run time

Overall Results

- Increase of undocumented parameter _PGA_MAX_SIZE prevented operations from spilling to temporary tablespace.
- Big Table Cache experiment resulted in buffer reads for initial table scans, essentially reducing initial disk IO.
- In-Memory Column Store experiment demonstrated the benefits of in memory execution, with the added benefit of scans on compressed columnar data.

Each column in the plot below represents the summation of individual test run times for each experiment. There is a clear decrease in overall run time for each of the three experiments. The most dramatic increase in performance resulted from utilizing the In-Memory Column Store feature. Through these three experiments we were able to quantitatively demonstrate techniques for increasing DSS workload performance.

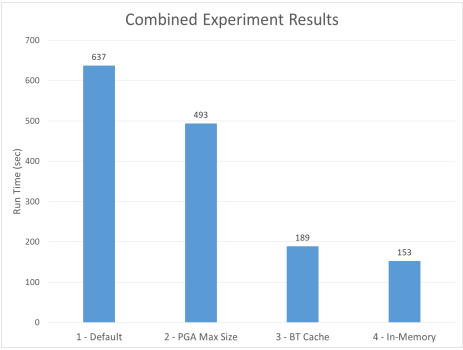
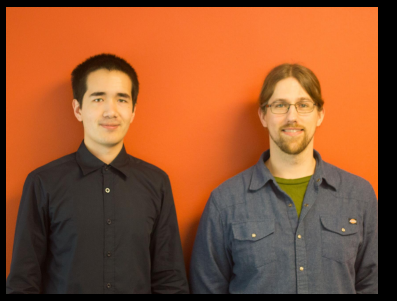


Figure 5: Summary of Results

Research Team



Team Members:

- Nic Desilets
  desiletn@oregonstate.edu
  Applied CS: Web & Mobile Development
- Nathaniel Whitlock
  whitlocn@oregonstate.edu
  Applied CS: Bioinformatics

Clients:

- Kirby Sand
  Data Analyst
- Andy Weiss
  Database Administrator

Affiliation:

- Hewlett-Packard Inc.
  PageWide Web Press

Outcomes:

- 4x speed improvements
- More resource utilization
- Increased query throughput

In conclusion, we saw significant performance increases by optimizing database system resource usage and taking advantage of some new features found in Oracle 12c. This was accomplished by increasing PGA workspace size, enabling In-Memory Column Store, and enabling Automatic Big Table Caching.

Oregon State UNIVERSITY

10 RESEARCH DRAFT

This section of the document is designed to reflect the path of action our team took in order to provide a solution to our clients problem. Not only does it serve as documentation of our efforts, but also has the potential to guide others, in a similar pursuit, to quality information that will help guide their design.

10.1 Preparation

In order to investigate the feature and parameters of interest, our group first needed to focus on configuring a suitable environment on local devices. This entailed installing the following pieces of software:

- **Oracle Linux 7:** Operating system installed on a virtual machine running on the host operating system
- **Oracle Enterprise 12c:** Target database management system (DBMS) for experimentation
- **SQL Developer:** Application mainly used to interact with the database

After installing these software tools, our group was ready to begin learning more about how Oracle 12c works. Before we would be able to make constructive suggestions to our clients, we first needed to spend a couple of weeks reading technical papers.

10.2 Initial Learnings

The optimizer is the governing operator within the Oracle environment. This logical structure determines the most efficient way to execute a SQL statement, many factors are considered during this process. Examples of these factors are: availability of resources, cost of fetching rows, cardinality of objects, and much much more. In order to get a better understanding of the inner working of this system, our clients requested that we read through the Oracle white paper titled, *Explaining the Explain Plan*[3].

An 'Explain Plan' is the set of steps that the optimizer has concluded are best for query execution before the statement is run. This does not necessarily mean that this resulting plan will be the final set of actions, but rather that it is a efficient set of steps to resolve the request, factors like resource restrictions can lead to alterations during execution. After a query is completed, there is another plan that is available which is referred to as the 'Execution Plan'. this is the actual set of steps taken during query execution. This plan details information including: operation, table name, rows accessed, bytes of data, cost in CPU percentage, time for completion, and stats on parallel processing.

Other main points detailed in this paper which were helpful for us include, but not limited to:

- Methods of accessing explain and execution plan
- Detailed description on table cardinality
- Discussion on table access methods and cost
- Join methods and their costs
- Basic intro to parallel processing in regards to the execution plan

These readings provided us the fundamental understanding which was essential in moving forward with research. Without this background, it will be hard to make sense of future readings and experiments.

10.3 Divide and Conquer

After gaining a general understanding of the inner working of Oracle 12c, our team split up to address the following topics presented by our clients:

- **Trace event codes and diagnostic extraction:**
  - Trace event codes can be used to write log files to a designated directory during the query execution. There are specific trace event codes to extract performance data for things like: optimizer decisions, parallel processing, and step by step block level details on query execution. The results from this could be used to perform fine granularity diagnostic work.
- **Relationship between PGA and sub-components:**
  - Understanding the relationship between these memory structures was essential given the fact our clients were experiencing poor resource utilization.
- **Determination of the degree of parallelism (DOP):**
  - The DOP is the variable that determines the number of parallel processes that can be alloted to the execution of a single SQL statement. The goal here was to determine a set of heuristics that would allow us to have a stronger insight on why a specific DOP might be available to a query.

After spitting these tasks among our three team members, we independently focused on providing as much focused and relevant material we could so that we could share it with the group. Our clients requested that we used resources as references which had Oak Table or Oracle Ace badge, these sites are hosted by individuals with paid or non-paid support from Oracle and would have reliable information. During the two weeks this process to place, we had two meetings per week with our clients where we discussed our individual findings.

10.4 Experimental Topics

10.4.1 Parallel Execution and PGA

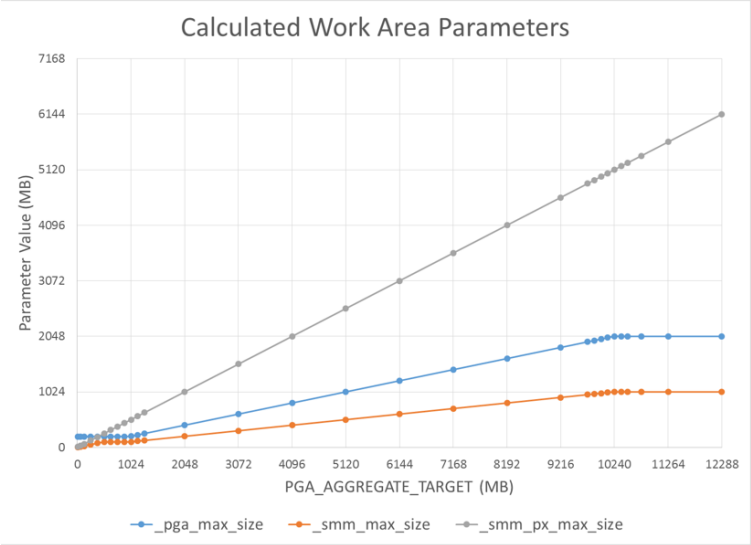
Moving forward in our research time line, we identified the PGA memory structure as a target for experimentation. In order to gain a better understanding of this logical structure we referred to another Oracle white paper titled, *Parallel Execution Fundamentals*[4]. This paper details general concepts about parallel execution and then focuses on much more Oracle specific details. One of the main topics is the producer/consumer model for parallel processing. In order to implement this model there must be a producer for every consumer. As the name implies, the producer's job is to obtain the blocks of data from disk and provide them the the consumer process. The consumer's job is to take the block and perform ¡any necessary operation on the provided blocks, this could be a sort, hash, join, or other aggregation.¿ Based on the finding within this paper, our team decided to experiment with the following list of parameters:

- **PGA_AGGREGATE_LIMIT:** Upper threshold for memory alloted to parallel processes
- **PGA_AGGREGATE_TARGET:** Target value, below threshold, that the system considers safe
- **_PGA_MAX_SIZE:** System defined memory size alloted to each work area (ie. sort and hash)
- **_SMM_MAX_SIZE:** System defined parameter for serial processing work areas
- **_SMM_PX_MAX_SIZE:** System defined parameter for parallel processing work areas

Given the fact that there are multiple system defined parameters, the next point of interested were in defining what conditions lead to what values. Based on Oracle documentation, all three of the system defined parameters were calculated by the user set value of PGA_AGGREGATE_TARGET. In order to provide this information we documented the change in the system defined parameters as the values of PGA_AGGREGATE_TARGET was increased from the default value, 10MB, to 12GB. The trends observed in the resulting plots for each of the system defined parameters were used to derive formulas for calculating the values of these parameters as a function of the value of PGA_AGGREGATE_TARGET. The plot we used for formulation derivation can be viewed below:

10.4.2 SGA

The next step in our process was to look into the shared memory structures in order to see how we could utilize modern features of Oracle 12c. One topic which was briefly covered in the *Parallel Execution Fundamentals*[5] is Oracle In-Memory



$$\textbf{\_PGA\_MAX\_SIZE} = max \begin{cases} 200MB, \\ min(2GB, 20\%\ of PGA\_AGGREGATE\_TARGET) \end{cases}$$

$$\textbf{\_SMM\_MAX\_SIZE} = min \begin{cases} 20\%\ of PGA\_AGGREGATE\_TARGET, \\ 50\%\ of\ \_PGA\_MAX\_SIZE \end{cases}$$

$$\textbf{\_SMM\_PX\_MAX\_SIZE} = \begin{cases} 50\%\ of PGA\_AGGREGATE\_TARGET \end{cases}$$

$$\textbf{Work Area} = min \begin{cases} \_SMM\_MAX\_SIZE, \\ \_SMM\_PX\_MAX\_SIZE/DOP \end{cases}$$

Fig. 5: Sample plot of the relationship of system defined parameters as a function of the PGA_AGGREGATE_TARGET value.

parallel execution. In order to learn more about this feature two Oracle white papers were evaluated: *Oracle Database In-Memory*[3] and *When to use In-Memory*[6].

The In-Memory Column Store (IMCS) is a memory structure within the SGA. The IMCS works independent of the buffer cache memory structure which is designed to support access dependent support for cached blocks in traditional row format. The IMCS acts as an extension to this structure and provides the option for storage of blocks in compressed columnar format. This means that you can store specified columns for a table, or entire tables of columns within this memory structure. There is a lot of flexibility in terms of the columns of data you would like to store, each can be specified by a command.

One of the most attractive features of IMCS use is that the data can be stored in a compressed format, and accessed without decompression. This is starkly different than traditional table compression techniques for row based storage. Additionally, all data in the IMCS are stored in logical subunits that have a header containing pre-aggregated information about the columnar section. This means that the max and min of the housed values are stored in a header block, benefits from this feature can been seen when limiting the blocks needed for access when performing a table scan.

Another important feature that we researched within the SGA is referred to Automatic Big Table Caching (ABTC). When enabled, this feature stores large objects in a separate memory structure, similar to the buffer cache, after they are initially accessed from a query execution. A temperature based algorithm is used to manage the presence of objects with the memory structures, as an object is accessed multiple times, the temperature value of that object increases. This results in lower temperature objects losing there place in this cache memory structure. Since objects cannot be pre-populated in the ABTC, this feature would only be useful in environments where table objects were accessed frequently.

10.4.3 Statement Queuing

This experiment was geared towards investigating the effect of resource distribution on many queries running simultaneously on are test environment. We designed this experiment to launch 48 separate queries composed of the same statement against an environment with both statement queuing enabled and without statement queuing enabled. When statement queuing is not enabled, incoming queries are downgraded, receive less parallel resources than requested, resulting in long run times and near serial execution. Alternatively, when statement queuing is enabled, incoming queries are placed in a first in first out queue and are only executed when requested resources are available. Not enabling statement queuing in an active high throughput parallel environment results in long running queries that have the potential for serial execution. When working with multi billion row tables this is just not acceptable.

10.5 Experimentation

In order to begin experimentation and collect data we first needed to develop the toolkit that we could use to gather data while the experiments were running. This is especially true given the fact that each experiment would likely take from 10-24 hours to complete.

We first met with our clients in order to define what was needed in order to facilitate this process. As a group, we were able to work out the basic details as to what was needed and decided to write up the toolkit in BASH since it would be running on a Linux based server. The initial design which was agreed upon included the following:

- **driver.sh:** Prepares environmental variables and calls individual run scripts
- **runMonitor.sh:** Establishes connection to capstone server and initializes resource monitoring package loop
- **runTest.sh:** Establish alternate connection to server and executes experiment scripts

Together, this sect of shell scripts allowed us to run experiment SQL scripts we had stored in the directories passes in as runtime arguments. This was an important step in being able to collect data in the background while multiple long running tests were being performed.

The next step was to develop SQL scripts that would test the performance of the features we were interested in. The details of each experiment are detailed below:

- **PGA Experiment:** Focused on altering values of '_PGA_MAX_SIZE' and monitoring the potential spill over from work area to temporary tablespace (ie. disk)
- **Big Table Cache Experiment:** Focused on altering the percentage of the buffer cache that caches large objects in memory and documenting the potential read performance changes
- **In-Memory Experiment:** Focused on enabling In-Memory Column Store and documenting the changes in performance at different levels of data compression

After the first iteration of testing our experiments we noticed that the database was being restarted after running out baseline file which was designed to return the database to a controlled state. In order to resolve this issues, we split the experiment SQL scrips into two groups: parameter files, and experiment files. In doing so, we were able to adjust the system parameters and restart the database without terminating the execution of the resource monitoring package loop.

This concludes the pathway taken by our team during the development process for our initially agreed upon requirements. Though there was additional work done for our clients, it has not been documented in this section.

11 LEARNING NEW TECHNOLOGY

11.1 List of Helpful Websites

- Oracle 12c documentation
  - https://docs.oracle.com/database/121/
- Additional Oracle 12c reference
  - http://www.morganslibrary.org/library.html
- List of 12c initialization parameters
  - https://docs.oracle.com/database/121/nav/initora.htm
- How to check degree of parallelism
  - https://jonathanlewis.wordpress.com/2007/03/14/how-parallel/

11.2 List of Publications Utilized During Research

- The Oracle Optimizer Explain the Explain Plan
  - http://www.oracle.com/technetwork/database/bi-datawarehousing/twp-explain-the-explain-plan-052011-393674.pdf
- Parallel Execution with Oracle Database 12c Fundamentals
  - http://www.oracle.com/technetwork/database/bi-datawarehousing/twp-parallel-execution-fundamentals-133639.pdf
- Oracle Database In-Memory with Oracle Database 12c Release 2
  - http://www.oracle.com/technetwork/database/in-memory/overview/twp-oracle-database-in-memory-2245633.html
- When to Use Oracle In-Memory
  - http://www.oracle.com/technetwork/database/in-memory/overview/twp-dbim-usage-2441076.html

12 LEARNINGS

12.1 Nathaniel

12.1.1 What technical information did you learn?

During the course of senior design I learned quite a bit about the functional use of SQL. Our mentors provided us many practical examples of the queries that they would use to interact with their system, this experience was more applicable than much of the time I have spent taking database courses. Andy Weiss, the Oracle DBA for the WebPress team, was a eager to share his knowledge of the internal workings of Oracle's database software. He was also a great resource for troubleshooting issues that we ran into during our project.

Beyond the scope of learning how to better use SQL, I learned quite a bit about experimental design. This is something that I was familiar with before, however, in this field all of the experiments are conducted electronically. By developing experiments that tested the effects the features we were testing, I exercised skills that were later applicable to my spring machine learning course. Creating a design of experiment matrix and tuning hyperparameters have quite a bit in common.

12.1.2 What non-technical information did you learn?

Our clients for this project were full time employees at the Corvallis HP Inc. site. Working closely with these professionals was a bit daunting at first, but then we began to see that they were down to earth. During my time working with our mentors I feel that I have gained experience feeling confident in a meeting with people in a more authoritative position. Additionally, due to the round robin nature of our winter meetings, I became proficient at researching specific topics and bringing results and examples to the team.

12.1.3 What have you learned about project work?

As with many of the team projects I have worked on, our team was not completely put together with individuals that could functionally contribute. Around week six of winter term we officially lost our teammate James Stallkamp. This was something that was foreshadowed by the lack of participation from James, however, it took quite a long time to before he was actually fired.

Though we lost a member, Nic and I pulled together and worked through the rest of the project on our own. We were able to provide all the requested deliverables to our clients well within time. One very important thing that I learned during this project is that there are other students who are passionate about their education, and even when things between the team are falling apart, those remaining can come together to make things happen. All to often in group interactions there are students who either do not care to put in an effort, or they do not have the skills necessary to complete the task.

12.1.4 What have you learned about project management?

In respect to project management, through this project I have learned that organization is crucial and documenting efforts along the way can pay of ten fold. During this project both Nic and I shared responsibility for keeping things on track. While we were still in fall term, there was a growing concern about the lapse in communication that we experienced from James. He would often not be available until the evening before our initial papers were due.

Both Nic and I had to work together to maintain the workflow for our group, this was not to large of an undertaking since James had not been contributing much since the beginning of the project. Nonetheless, we were forced to separate work between the two of us so that we could cover everything needed to accomplish our goal. This relates to project

management in terms of making modifications to a group workflow based on the absence of one of the expected members. Although it is possible to functionally pick up the slack, this is a huge undertaking and needs to be handled in an organized and structured manor.

12.1.5 What have you learned about working in teams?

Working with a team is something that will be required for the remainder of my professional career, however, this is a stark difference in teammates between the industrial and academic space. Many of the group projects that I have worked on during my undergraduate have been problematic. This is often caused by a communication barrier, or general lack of contribution. There are students in our program from many different walks of life and some are more concerned with their academic performance, while others might be having their college paid for by someone else.

In summary, I've learned that if you have team members that are passionate about their education, then you have the chance to trust them to fulfill their contribution. Otherwise, working with a team member who is absent, or non-contributive, can lead to additional stress. Trust is an extremely important factor in functional group interaction. Each person should be able to carry their own self-advocated work load, if they are not able to do so then they should reach out to the team members for support.

12.1.6 If you could do it all over, what would you do differently?

One of the things I would do is spend more time throughly documenting all of the sources that I read through when undergoing the experimental development. Additionally, I would consider probing our clients for more specific details on exactly what they wanted during fall term. There was a time during winter term that I thought we were going to need to re-write the entire design document due to the ambiguity in solution scope provided by the clients. This did not turn out to be an issue since we wrote the initial paper with enough ambiguity that is still fit the scope of our final deliverables.

12.2 Nic

12.2.1 What technical information did you learn?

My experiences with databases prior to this project were relatively simple MySQL projects with a couple tables. During the beginning of the project I was a little unsure of myself because I did not know much about relational database management systems. With that in mind, being told to research Oracle database in depth seemed daunting initially. However, with guidance from our mentors and persistent research I was able to come out of this project feeling much more comfortable working with Oracle database as well as databases in general. More specifically, through my mentors and research I learned about how Oracle database works including the architecture of it, how the different types of caches work, processes involved with executing a query, how different types of sorts and joins are decided, and so on. A lot of these concepts I learned in Oracle database can also be applied to other relational database management systems.

12.2.2 What non-technical information did you learn?

Aside from the technical aspects of researching Oracle database, I also learned more about researching skills as well as soft skills. One aspect of our project was learning how to approach a problem full of unknowns and break it down into pieces that can be researched independently. In our case, we took a top down approach where we first started with taking broad areas of Oracle database and then began further researching each area more in depth. We used OneNote to document our findings every step of the way. In line with this, we also talked about our research progress during our client meetings in a round-table discussion to share any knowledge gained. What I found that worked best for me was to go over my notes that I had taken on a topic and briefly summarize key areas. This allowed me to concisely explain what I had found while also being able to go into detail if necessary.

12.2.3 What have you learned about project work?

The bulk of the work performed in our project was researching concepts and internal parameters. In our case documenting our research effectively was an important operation. OneNote seemed like a good application to document our findings and we found that it worked very well for our use case. We had a tab set up for each project member, and within each tab there were pages dedicated to specific research topics. For each page, we also had a list of sources in each page to keep track of where we were obtaining information. This allowed us to quickly navigate through our collective notes to find information on a research topic.

Another area that we spent a lot of time in was documentation relating to our project and progress made throughout each term. As far as documenting progress goes, OneNote served as a helpful aid in this aspect since it was effectively also a log of work performed. Alongside this, our weekly progress summaries in the three Ps format were also very helpful. When it came time to actually write the documentation we found that Overleaf was a great tool to quickly and effectively collaborate on LaTeX documents.

12.2.4 What have you learned about project management?

The deadline for the majority of our research was at the end of February and so this did not leave us much time to gather and process information. This was further complicated by the fact that we were not able to get much work done initially because we had spent a large portion of our time writing documentation during Fall term. Efficient use of our time was critical in being able to effectively gather and process as much information as possible in a short time window. What worked for us was to brainstorm potential research topics during our meetings with our clients. From there we would decide on the most important topics to research and split up the workload among us. This allowed to quickly go through and learn a concept by splitting the concept up, researching these portions individually, and then later reconvene with our findings.

12.2.5 What have you learned about working in teams?

Most of my experience in working with teams in the context of software comes from my undergraduate class projects. What I have personally experienced in many of these projects is that in many cases people tend to not want to contribute or they simply flat out lack the knowledge or drive to learn to complete a project. I figured that in our senior design project, which is much more significant than a regular class project, perhaps this would change and people would be more willing to contribute in a meaningful way. However, it seems that this problem doesnt seem to go away in larger projects since we had to deal a team member who did not contribute.

On the bright side, my other senior design group member was an excellent team mate to have throughout this project. Communication between team members was an important aspect in our project since we needed to convey information that weve gathered between all of us. Discussing our findings in round table meetings as well as through electronic means were useful for us in order to stay up to speed with each other. This also ties back in to using OneNote which allowed us to glance over information that other people have gathered.

12.2.6 If you could do it all over, what would you do differently?

Fall term was a little problematic for our team between dealing with documentation that did not seem to fit our project well and also the fact that we were unsure of what work our clients wanted to us to perform. This biggest unknown at the time was that we were unsure of what exactly we were even going to be researching since our clients at the time did not know either. The problem was that a database is performing poorly but no one knew exactly why and also how to fix it. The former problem is just a consequence of our project not quite being a conventional capstone project, however the latter could have been mitigated by communicating with our clients more. It probably would have been useful to us if we asked them more questions relating to Oracle database and potential areas of interest that we would be looking into.

13 Appendix 1: Essential Code Listings

13.1 Test Automation Suite

The following code listings consist of the software our team put together to both run our suite of experiments, as well as initialize a resource monitoring loop that gathered runtime statistics and wrote them to a static table that we could later pull from. The following scripts are configured to run on our clients server and will likely not be able to run on any other system. However, the general idea could be utilized to facilitate automated testing on a database. Note that our research focused on Oracle 12c, some of the parameters we experimented with might not be available in earlier versions and will not be available in alternate technologies. However, the general idea implemented though these scripts could be applied to any user configurable database technology.

Listing 1: driver.sh

```
#!/bin/bash

# This script sets up the environment needed to run a stat monitoring
# loop in the background as well as run a series of sql tests

if [ -f lock.txt ]; then
  echo -e "\nThe testing script appears to already be running."
  echo -e "For detals see:  lock.txt\n"
  exit 1
fi

# Create temp output file
touch lock.txt

# Usage statement
if [ ! $# -ge 2 ]; then
    echo 'Usage: ./driver.sh <snapFreq> <exp 1> [<exp2>] ... [<exp n>]'
    exit 1
fi

# Set up oracle env variables
export ORACLE_SID=capstone
export ORAENV_ASK=NO
. oraenv

# Import user credentials
source ./dbconfig.cfg

# Set up local variables
snapFreq=${1}
experiments=${@:2}
expDir='../exp_scripts'
logDir='../logs'
dataDir='../data'

# Create file system
if [ ! -d ${logDir} ]; then
    mkdir ${logDir}
    echo 'Created ' ${logDir}
fi

if [ ! -d ${dataDir} ]; then
    mkdir ${dataDir}
    echo 'Created ' ${dataDir}
fi

# Starting stamp
echo "************************************************************" >> lock.txt 2>&1
echo " BEGIN EXP " `date`                                           >> lock.txt 2>&1
echo "************************************************************" >> lock.txt 2>&1


# Outer loop to iterate through experiments
for entry in $experiments; do
   # Conditional check for directory (entry)
    if [ ! -d ${expDir}/${entry} ]; then
        echo " cannot find experiment ${expDir}/${entry}" >> lock.txt 2>&1
        continue
    fi

    #echo -e "Starting experiment ${entry}\n"

    # Grab content for inner loop
    dirContents=`ls -1 ${expDir}/${entry}/*.SQL`
```

```
    # Run baseline file
    ./runParam.sh ${logDir} "${expDir}/BASELINE/BASELINE.SQL"

    for i in $dirContents; do

        # Grab file name
        name=$(basename -s .SQL $i)

        if [ -f ${expDir}/${entry}/${name}.PRM ]
        then
            echo "${name}.PRM exists"                                >> lock.txt 2>&1
            echo "Modifying DB params for ${name}.PRM"               >> lock.txt 2>&1
            ./runParam.sh ${logDir} "${expDir}/${entry}/${name}.PRM"
        else
            echo "${expDir}/${entry}/${name}.PRM does not exist"     >> lock.txt 2>&1
        fi

        # Branch of to run monitor and test
        ./setMonitorFlag.sh ${logDir} ${dataDir} ${dbUser} ${password} ${db}          >> lock.txt 2>&1

        echo "Starting Monitoring Loop --- ${snapFreq} "                              >> lock.txt 2>&1
        ./runMonitor.sh ${logDir} ${dataDir} ${snapFreq} ${dbUser} ${password} ${db} &>> lock.txt 2>&1

        echo "Starting Experiment Suite -  ${expDir}/${entry}/${name}.SQL"                          >> lock.txt 2>&1
        ./runTest.sh ${logDir} ${dataDir} "${expDir}/${entry}/${name}.SQL" ${dbUser} ${password} ${db} >> lock.txt 2>&1

    done;
done;

# Remove environmental variables
unset ORAENV_ASK

# wait for any outstanding background processes to finish
wait

# Ending stamp
echo "************************************************************"          >> lock.txt 2>&1
echo " END EXP " `date`                                                       >> lock.txt 2>&1
echo -e "************************************************************\n\n\n" >> lock.txt 2>&1

# Email out temp results
mail -s 'CAPSTONE TESTING' -S smtp=smtp3.hp.com daweiss1@gmail.com kirby.sand@hp.com andy.weiss@hp.com nathaniel.whitlock1@hp.com des

# Append temp file and remove
cat lock.txt >> ../logs/output.log
rm lock.txt
```

Listing 2: runMonitor.sh

```
#!/bin/bash

# This script is designed to create connection to Oracle
# database and run the resource usage monitor in the
# background

# Store passed arguments
logDir=${1}
dataDir=${2}
snapFreq=${3}
dbUser=${4}
password=${5}
db=${6}

# Establish connection and run monitoring loop
sqlplus /nolog <<EOF >> ${logDir}/runMonitor.log
CONN ${dbUser}/${password}@${db}

spool ${logDir}/runMonitor.log append
EXEC CAPSTONE_DEMO.RESOURCE_MONITOR.RESOURCE_USAGE_MONITORING_V2(${snapFreq});
spool off

EOF

echo -e 'Ending Monitoring Loop\n' >> lock.txt 2>&1
```

Listing 3: runParam.sh

```
#!/bin/bash

# runParam.sh
# Alters session and system setting according to the parameter (PRM)
# files that are within each passed experiemnt directory.

# Store passed arguments
logDir=$1
expScript=$2

# Establish connection and execute predefined queries
sqlplus / as sysdba <<EOF >> ${logDir}/runTest.log

alter session set NLS_DATE_FORMAT='yyyymmdd hh24:mi:ss';

spool ${logDir}/runParam.log append
select '*********  ' ||  sysdate || '  **********'  from dual;

@${expScript}
spool off

EOF


echo "Parameters Altered -- ${expScript}" >> lock.txt 2>&1
```

Listing 4: runParam.sh

```
#!/bin/bash

# runTest.sh
# Established independent connection and runs the sql experiment
# files. Once finished the monitoring loop is terminated.

# Store passed arguments
logDir=$1
dataDir=$2
expScript=$3
dbUser=$4
password=$5
db=$6

echo -e '\n------------------------------'
echo 'runTest.sh Input Vars'
echo 'logDir: ' $logDir
echo 'dataDir: ' $dataDir
echo 'dbuser: ' $dbUser
echo 'db: ' $db
echo -e '------------------------------\n'

# Establish connection and execute predefined queries
sqlplus /nolog <<EOF >> ${logDir}/runTest.log
CONN ${dbUser}/${password}@${db}
alter session set NLS_DATE_FORMAT='yyyymmdd hh24:mi:ss';

spool ${logDir}/runTest.log append
select '*********  ' ||  sysdate || '  **********'  from dual;
@${expScript}
spool off

spool ${logDir}/runMonitor.log append
@../exp_scripts/END_MONITORING.sql
spool off

EOF


echo 'Test Script Completed -- ' ${expScript}
```

13.2 Parameter Files

The code listing below is an example of the parameter files we used. These files were split from the initial test scripts because the database needed to be restarted after adjusting these parameters, the restart was terminating our resource monitoring loop so we were forced to separate the files and restart the database before initializing the resource monitoring loop.

Listing 5: Initial parameter file design

```
ALTER SESSION SET CONTAINER = CDB$ROOT;
ALTER SYSTEM SET "_PGA_MAX_SIZE" = 2G;
ALTER SYSTEM SET DB_CACHE_SIZE = 100M;
ALTER SYSTEM SET DB_BIG_TABLE_CACHE_PERCENT_TARGET = 0;
ALTER PLUGGABLE DATABASE DBCAP CLOSE IMMEDIATE;
ALTER PLUGGABLE DATABASE DBCAP OPEN;
ALTER SESSION SET CONTAINER = DBCAP;
ALTER SYSTEM SET PARALLEL_DEGREE_POLICY = MANUAL;
```

13.3 Experiment Files

The experiment files that we generated were basically sets of queries that we would run against the database after setting specific system parameters, or enabling experimental features. We intended to use these experiments to gather data that we could use to validate the methods and strategies that we recommended to our clients.

Listing 6: Intial experiment design

```
alter session set container = dbcap;

EXECUTE CAPSTONE_DEMO.RESOURCE_MONITOR.PARAMETERS_HISTORY_INSERT;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PGA_EXPERIMENT',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'SINGLE_TABLE_ANALYTICS_V1');
END;
/
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'PARENT_CHILD_ANALYTICS_V1');
END;
/
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'LEAD_LAG_V1');
END;
/

SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'FILTER_PARENT_CHILD_JOIN_V1');
END;
/

SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;
SELECT /*+ PARALLEL(16) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PGA_EXPERIMENT',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'SINGLE_TABLE_ANALYTICS_V1');
END;
/
```

```
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'PARENT_CHILD_ANALYTICS_V1');
END;
/
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'LEAD_LAG_V1');
END;
/

SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'FILTER_PARENT_CHILD_JOIN_V1');
END;
/

SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;
SELECT /*+ PARALLEL(32) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PGA_EXPERIMENT',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'SINGLE_TABLE_ANALYTICS_V1');
END;
/
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.SINGLE_TABLE_ANALYTICS_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'PARENT_CHILD_ANALYTICS_V1');
END;
/
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.PARENT_CHILD_ANALYTICS_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'LEAD_LAG_V1');
END;
/

SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.LEAD_LAG_V1;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'FILTER_PARENT_CHILD_JOIN_V1');
END;
/

SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;
SELECT /*+ PARALLEL(64) */ * FROM CAPSTONE_DEMO.FILTER_PARENT_CHILD_JOIN_V1;
```

13.4 Experimental Design Modifications

After completing the initial set of experiments, our team quickly noticed that the way that our previous experimental scripts were set up was predisposed to a poor ability for quick modification and upkeep. We decided to produce further experiments with an alternative design, one that would modularize the tests that were run. We created test scripts that contained to a list of queries to run, as well as an experiment script that focused on modifying the needed system parameters and calls to the test scripts. The listings below demonstrate the modified file type we moved to.

Listing 7: Modified experiment script

```
ALTER SESSION SET DB_FILE_MULTIBLOCK_READ_COUNT=32;
ALTER SESSION SET WORKAREA_SIZE_POLICY = MANUAL;
ALTER SESSION SET SORT_AREA_SIZE = 2147483647;
ALTER SESSION SET HASH_AREA_SIZE = 2147483647;
ALTER SESSION ENABLE PARALLEL DML;
ALTER SESSION SET CONTAINER = dbcap;
-- Set up table
DROP TABLE dr_part_date_day_interval;

CREATE TABLE  dr_part_date_day_interval
PARTITION BY RANGE (press_local_time)
  INTERVAL (numtodsinterval(1,'DAY'))
  (PARTITION part_01 VALUES LESS THAN (to_date('01/02/2007', 'MM/DD/YYYY')))
AS (SELECT DISTINCT device_id, press_local_time, measurement_type_key, measurement
  FROM whitlocn.capstone_two_billion
  WHERE (device_id IN ('11', '12','15','16','19','24') AND press_local_time < to_date(20171220, 'yyyymmdd'))
    OR (device_id IN ('14','23','20','25','26') AND press_local_time < to_date(20161212, 'yyyymmdd')
    OR (device_id IN ('13','17','18','21','22') AND press_local_time < to_date(20161221, 'yyyymmdd'))));

-------------------------------------------------------------------------------
-- DL with only new values
-------------------------------------------------------------------------------
-- 0.05% of DR
DROP TABLE data_loader;
CREATE TABLE data_loader AS (SELECT DISTINCT device_id, press_local_time, measurement_type_key, measurement
FROM whitlocn.capstone_two_billion
  WHERE (device_id IN ('11', '12','15','16','19','24') AND press_local_time >= to_date(20171220, 'yyyymmdd'))
    OR (device_id IN ('14','23','20','25','26') AND press_local_time >= to_date(20161212, 'yyyymmdd')
    OR (device_id IN ('13','17','18','21','22') AND press_local_time >= to_date(20161221, 'yyyymmdd'))));

-------------------------------------------------------------------------------
-- No-Index experiment
-------------------------------------------------------------------------------
EXECUTE CAPSTONE_DEMO.RESOURCE_MONITOR.PARAMETERS_HISTORY_INSERT;
BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_NOI',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_NUI.sql dr_part_date_day_interval

-- GLOBAL INDEX EXPERIMENTS
-------------------------------------------------------------------------------
-- Primary key experiment (Prefix)
-------------------------------------------------------------------------------
ALTER TABLE dr_part_date_day_interval DROP CONSTRAINT prefix_key;
ALTER TABLE dr_part_date_day_interval ADD CONSTRAINT prefix_key PRIMARY KEY(press_local_time,device_id, measurement_type_key, measure

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_PK_P',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_Date_Pre_UI.sql dr_part_date_day_interval

ALTER TABLE dr_part_date_day_interval DROP CONSTRAINT prefix_key;
-------------------------------------------------------------------------------
---- Primary key experiment (Non-prefix)
-------------------------------------------------------------------------------
ALTER TABLE dr_part_date_day_interval DROP CONSTRAINT non_prefix_key;
ALTER TABLE dr_part_date_day_interval ADD CONSTRAINT non_prefix_key PRIMARY KEY(device_id, press_local_time,  measurement_type_key, n

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_PK_NP',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
```

```
/

@index_tests/Tests_ID_Pre_UI.sql dr_part_date_day_interval

ALTER TABLE dr_part_date_day_interval DROP CONSTRAINT non_prefix_key;
-------------------------------------------------------------------------------
---- Unique index experiment (Prefix)
-------------------------------------------------------------------------------
DROP INDEX unique_P_index;
CREATE UNIQUE INDEX unique_P_index ON dr_part_date_day_interval(press_local_time,device_id, measurement_type_key, measurement);

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_UI_P',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_Date_Pre_UI.sql dr_part_date_day_interval

DROP INDEX unique_P_index;
-------------------------------------------------------------------------------
---- Unique index experiment (Non-prefix)
-------------------------------------------------------------------------------
ALTER SYSTEM FLUSH SHARED_POOL;
DROP INDEX unique_NP_index;
CREATE UNIQUE INDEX unique_NP_index ON dr_part_date_day_interval(device_id, press_local_time,  measurement_type_key, measurement);

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_UI_NP',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_ID_Pre_UI.sql dr_part_date_day_interval

DROP INDEX unique_NP_index;
-------------------------------------------------------------------------------
-- Nonunique index experiment (Prefix)
-------------------------------------------------------------------------------
DROP INDEX nonunique_P_index;
CREATE INDEX nonunique_P_index ON dr_part_date_day_interval(press_local_time, device_id, measurement_type_key, measurement);

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_NUI_P',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_NUI.sql dr_part_date_day_interval

DROP INDEX nonunique_P_index;
-------------------------------------------------------------------------------
-- Nonunique index experiment (Non-prefix)
-------------------------------------------------------------------------------
ALTER SYSTEM FLUSH SHARED_POOL;
DROP INDEX nonunique_NP_index;
CREATE INDEX nonunique_NP_index ON dr_part_date_day_interval(device_id, press_local_time,  measurement_type_key, measurement);

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_NUI_NP',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_NUI.sql dr_part_date_day_interval

DROP INDEX nonunique_NP_index;

-- LOCAL INDEX EXPERIMENTS
-------------------------------------------------------------------------------
-- Unique index (Prefix)
-------------------------------------------------------------------------------
DROP INDEX local_prefix_unique;
CREATE UNIQUE INDEX local_prefix_unique ON dr_part_date_day_interval(press_local_time, device_id, measurement_type_key, measurement)

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_LUI_P',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/
```

```
@index_tests/Tests_Date_Pre_UI.sql dr_part_date_day_interval

DROP INDEX local_prefix_unique;
----------------------------------------------------------------------------
-- Unique index (Non-prefix)
----------------------------------------------------------------------------
DROP INDEX local_nonprefix_unique;
CREATE UNIQUE INDEX local_nonprefix_unique ON dr_part_date_day_interval(device_id, press_local_time, measurement_type_key, measuremen

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_LUI_NP',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_ID_Pre_UI.sql dr_part_date_day_interval

DROP INDEX local_nonprefix_unique;

----------------------------------------------------------------------------
-- Non-unique index (Prefix)
----------------------------------------------------------------------------
DROP INDEX local_prefix_nonunique;
CREATE INDEX local_prefix_nonunique ON dr_part_date_day_interval(press_local_time, device_id, measurement_type_key, measurement) LOCA

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_LNUI_P',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_NUI.sql dr_part_date_day_interval

DROP INDEX local_prefix_nonunique;
----------------------------------------------------------------------------
-- Non-unqiue index (Non-prefix)
----------------------------------------------------------------------------
DROP INDEX local_nonprefix_nonunique;
CREATE INDEX local_nonprefix_nonunique ON dr_part_date_day_interval(device_id, press_local_time, measurement_type_key, measurement) L

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_MODULE(MODULE_NAME => 'PART_BY_DAY_NEW_DATA_LNUI_NP',ACTION_NAME => 'RESOURCE_MONITORING');
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'STANDARD_QUERY');
END;
/

@index_tests/Tests_NUI.sql dr_part_date_day_interval

DROP INDEX local_nonprefix_nonunique;

DROP TABLE dr_part_date_day_interval;
```

Listing 8: Modified test script

```
----------------------------------------------------------------------------
-- Tests for Prefix or Unique Index Prefix for Date
----------------------------------------------------------------------------

ALTER SYSTEM FLUSH SHARED_POOL;
INSERT /*+ PARALLEL(8) */
  INTO &1 (device_id, press_local_time, measurement_type_key, measurement)
  (SELECT device_id, press_local_time, measurement_type_key, measurement FROM data_loader);
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
INSERT /*+ PARALLEL(8) */
  INTO &1 (device_id, press_local_time, measurement_type_key, measurement)
  (SELECT device_id, press_local_time, measurement_type_key, measurement FROM data_loader);
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
INSERT /*+ PARALLEL(8) */
  INTO &1 (device_id, press_local_time, measurement_type_key, measurement)
  (SELECT device_id, press_local_time, measurement_type_key, measurement FROM data_loader);
ROLLBACK;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'NO_DUPS_HINT');
END;
```

```
/
ALTER SYSTEM FLUSH SHARED_POOL;
INSERT /*+ PARALLEL(8) IGNORE_ROW_ON_DUPKEY_INDEX(&1(press_local_time, device_id, measurement_type_key, measurement))*/
  INTO &1 (device_id,press_local_time,measurement_type_key, measurement)
  (SELECT device_id, press_local_time, measurement_type_key, measurement FROM data_loader);
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
INSERT /*+ PARALLEL(8) IGNORE_ROW_ON_DUPKEY_INDEX(&1(press_local_time, device_id, measurement_type_key, measurement))*/
  INTO &1 (device_id,press_local_time,measurement_type_key, measurement)
  (SELECT device_id, press_local_time, measurement_type_key, measurement FROM data_loader);
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
INSERT /*+ PARALLEL(8) IGNORE_ROW_ON_DUPKEY_INDEX(&1(press_local_time, device_id, measurement_type_key, measurement))*/
  INTO &1 (device_id,press_local_time,measurement_type_key, measurement)
  (SELECT device_id, press_local_time, measurement_type_key, measurement FROM data_loader);
ROLLBACK;

BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'MERGE_STATEMENT');
END;
/
ALTER SYSTEM FLUSH SHARED_POOL;
MERGE /*+ PARALLEL(8)*/
  INTO &1 dr
  USING data_loader dl
  ON (dr.device_id = dl.device_id
      AND dr.press_local_time = dl.press_local_time
      AND dr.measurement_type_key = dl. measurement_type_key
      AND dr.measurement = dl.measurement)
  WHEN NOT MATCHED THEN
    INSERT (device_id, press_local_time, measurement_type_key, measurement)
    VALUES (dl.device_id, dl.press_local_time,dl.measurement_type_key,dl.measurement);
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
MERGE /*+ PARALLEL(8)*/
  INTO &1 dr
  USING data_loader dl
  ON (dr.device_id = dl.device_id
      AND dr.press_local_time = dl.press_local_time
      AND dr.measurement_type_key = dl. measurement_type_key
      AND dr.measurement = dl.measurement)
  WHEN NOT MATCHED THEN
    INSERT (device_id, press_local_time, measurement_type_key, measurement)
    VALUES (dl.device_id, dl.press_local_time,dl.measurement_type_key,dl.measurement);
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
MERGE /*+ PARALLEL(8)*/
  INTO &1 dr
  USING data_loader dl
  ON (dr.device_id = dl.device_id
      AND dr.press_local_time = dl.press_local_time
      AND dr.measurement_type_key = dl. measurement_type_key
      AND dr.measurement = dl.measurement)
  WHEN NOT MATCHED THEN
    INSERT (device_id, press_local_time, measurement_type_key, measurement)
    VALUES (dl.device_id, dl.press_local_time,dl.measurement_type_key,dl.measurement);
ROLLBACK;

DROP TABLE ERR$_TEST_ERROR;
BEGIN
  DBMS_ERRLOG.CREATE_ERROR_LOG(DML_TABLE_NAME => '_TEST_ERROR', ERR_LOG_TABLE_OWNER => 'C##CAPSTONE');
END;
/
BEGIN
  DBMS_LOCK.SLEEP(SECONDS => 30);
  DBMS_APPLICATION_INFO.SET_CLIENT_INFO(CLIENT_INFO => 'MERGE_ERROR_LOG');
END;
/
ALTER SYSTEM FLUSH SHARED_POOL;
MERGE /*+ PARALLEL(8)*/
  INTO &1 dr
  USING (SELECT * FROM DATA_LOADER) dl
  ON (dr.device_id = dl.device_id
      AND dr.press_local_time = dl.press_local_time
      AND dr.measurement_type_key = dl. measurement_type_key
      AND dr.measurement = dl.measurement)
  WHEN NOT MATCHED THEN
    INSERT (device_id, press_local_time, measurement_type_key, measurement)
    VALUES (dl.device_id, dl.press_local_time,dl.measurement_type_key,dl.measurement)
  LOG ERRORS INTO ERR$_TEST_ERROR REJECT LIMIT UNLIMITED;
```

```
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
MERGE /*+ PARALLEL(8)*/
  INTO &1 dr
  USING (SELECT * FROM DATA_LOADER) dl
  ON (dr.device_id = dl.device_id
      AND dr.press_local_time = dl.press_local_time
      AND dr.measurement_type_key = dl. measurement_type_key
      AND dr.measurement = dl.measurement)
  WHEN NOT MATCHED THEN
    INSERT (device_id, press_local_time, measurement_type_key, measurement)
    VALUES (dl.device_id, dl.press_local_time,dl.measurement_type_key,dl.measurement)
  LOG ERRORS INTO ERR$_TEST_ERROR REJECT LIMIT UNLIMITED;
ROLLBACK;

ALTER SYSTEM FLUSH SHARED_POOL;
MERGE /*+ PARALLEL(8)*/
  INTO &1 dr
  USING (SELECT * FROM DATA_LOADER) dl
  ON (dr.device_id = dl.device_id
      AND dr.press_local_time = dl.press_local_time
      AND dr.measurement_type_key = dl. measurement_type_key
      AND dr.measurement = dl.measurement)
  WHEN NOT MATCHED THEN
    INSERT (device_id, press_local_time, measurement_type_key, measurement)
    VALUES (dl.device_id, dl.press_local_time,dl.measurement_type_key,dl.measurement)
  LOG ERRORS INTO ERR$_TEST_ERROR REJECT LIMIT UNLIMITED;
ROLLBACK;
```

14 Appendix 2: Results from Experiments

The plots below detail some of the results from our experimentation. Add more material here.....

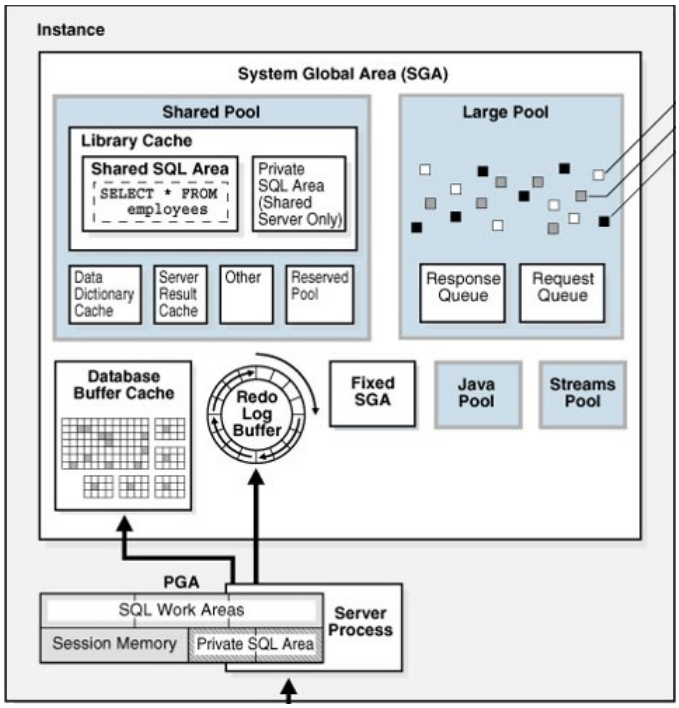
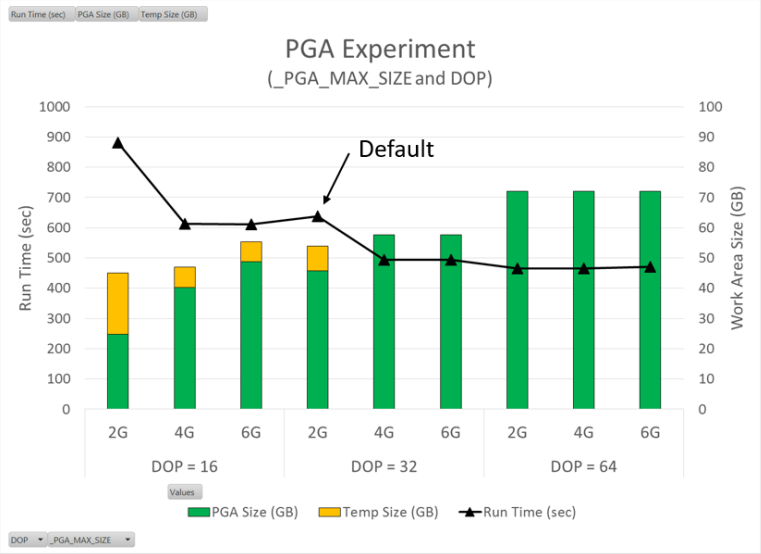


Fig. 6: General overview of the internal working of the Oracle database. Details the majority of the components of the SGA and PGA from a visual perspective. This project focused on the SQL Work Areas within PGA, Database Buffer Cache in SGA, and the In-Memory Column Store which is an optional memory structure in SGA not shown above.

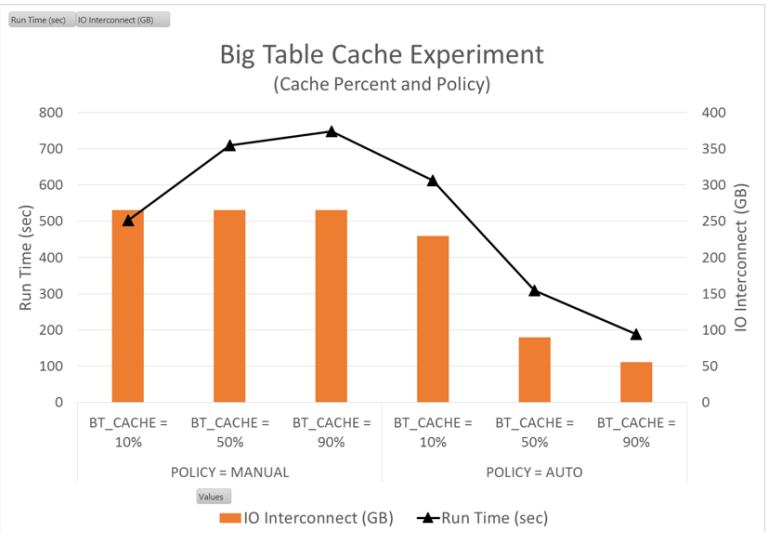
Experiment # 1: Tuning PGA



- Biggest gains happen when we never go to temp.
  - Approx. 1.3X improvement over default
- How do we prevent TEMP?
  - Really high DOP
    - Not ideal, requires lots of CPU allocation without a big improvement in run time.
  - Larger _PGA_MAX_SIZE
    - 4G seems adequate. 6G didn't provide any benefit.
    - A more attractive approach. Allows us to use more PGA and less CPU
    - Should use caution since it's an undocumented parameter
- Things certainly got faster, but with tables this large going to disk is another big bottleneck.

Fig. 7: This experiment demonstrated the benefit of alleviating the need to spill to disk during memory intensive operations such as sorts and merges. This was the first topic of interest to our clients. Notice how the runtime, seen as the black line with triangle nodes, decreases as the amount of Temp, seen as yellow, decreases. However, there is no added benefit after the necessary threshold is reached.

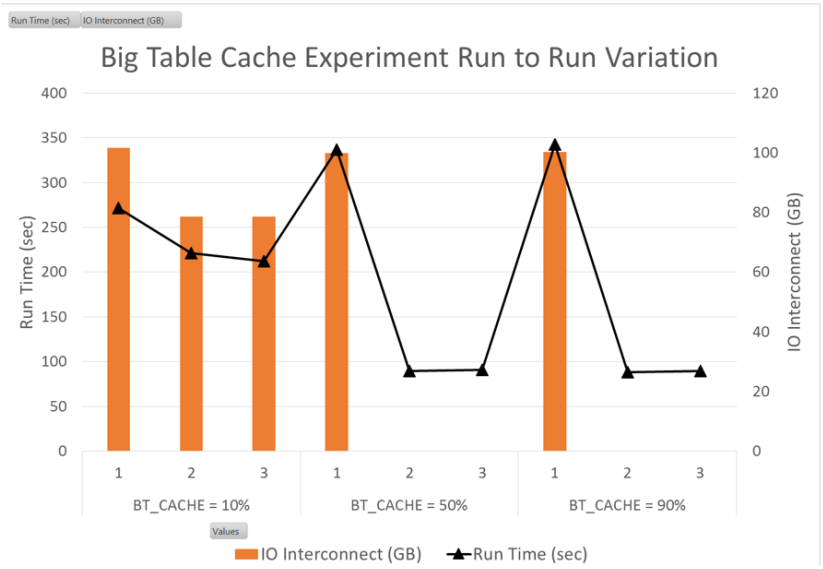
Experiment # 2: Big Table Caching



- Policy = Manual
  - Notice that as the cache percent target gets large, the queries slow significantly.
- Policy = Auto
  - As you increase the big table cache percent you get significant query speed improvements.
    - 3.4X speed improvement vs the original default
    - 2.6X speed improvement from the new default
  - However, you still have to go disk for some of the data.

Fig. 8: Details the effect of resource management when BT Cache is not enabled. When memory is allocated from buffer cache and parallel degree policy is set to manual, there is observed resource contention. When parallel degree policy is set to auto, BT Cache is enabled runtime benefits are observed for subsequent queries.

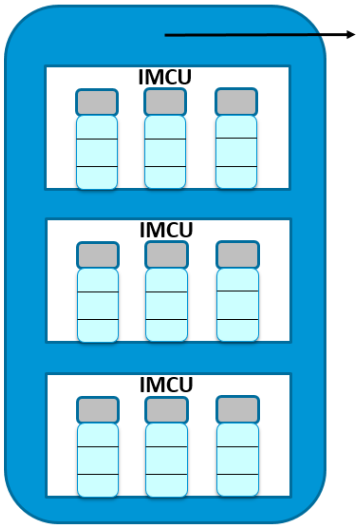
Experiment # 2: Big Table Caching



- First run penalty?
  - It appears there is some overhead to filling up the big table cache
  - Significant improvements in subsequent runts
- Use Cases
  - Ad-hoc queries
    - Something where you spend a day/week crunching on the same big tables
  - Nightly Jobs
    - Probably not, unless you reference the same big many times.

Fig. 9: Demonstrates queries involving data not already in the big table cache will incur IO penalties, such as in a freshly booted database. Once the data is loaded into the cache after the first query, subsequent queries involving the same data will see large performance increases.

In-Memory Pools



1MB Pool

- Used to store actual data populated in columnar format
- Subunits knows as In-Memory Compression Units (IMCU)
- Column stores composed of 1MB extents
- Data can be stored over one or more IMCU
- 1MB Pool Size = (0.8 * INMEMORY_SIZE)

Dynamic Performance Views

- v$inmemory_area
- v$im_header
- v$im_smu_head
- V$im_segments

64KB Pool

- Contains IMCU metadata and transactional information
- Individual subunits known as Snapshot Metadata Units
- 64KB Pool Size = (0.2 * INMEMORY_SIZE)

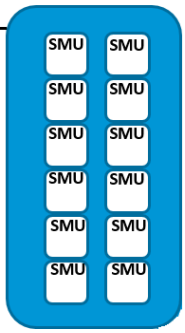


Fig. 10: Structure of the optional In-Memory Column Store memory structure within SGA. There are two separate memory pools, the 1MB pool which stores data in columnar format and the 64KB pool which stores IMCS metadata.

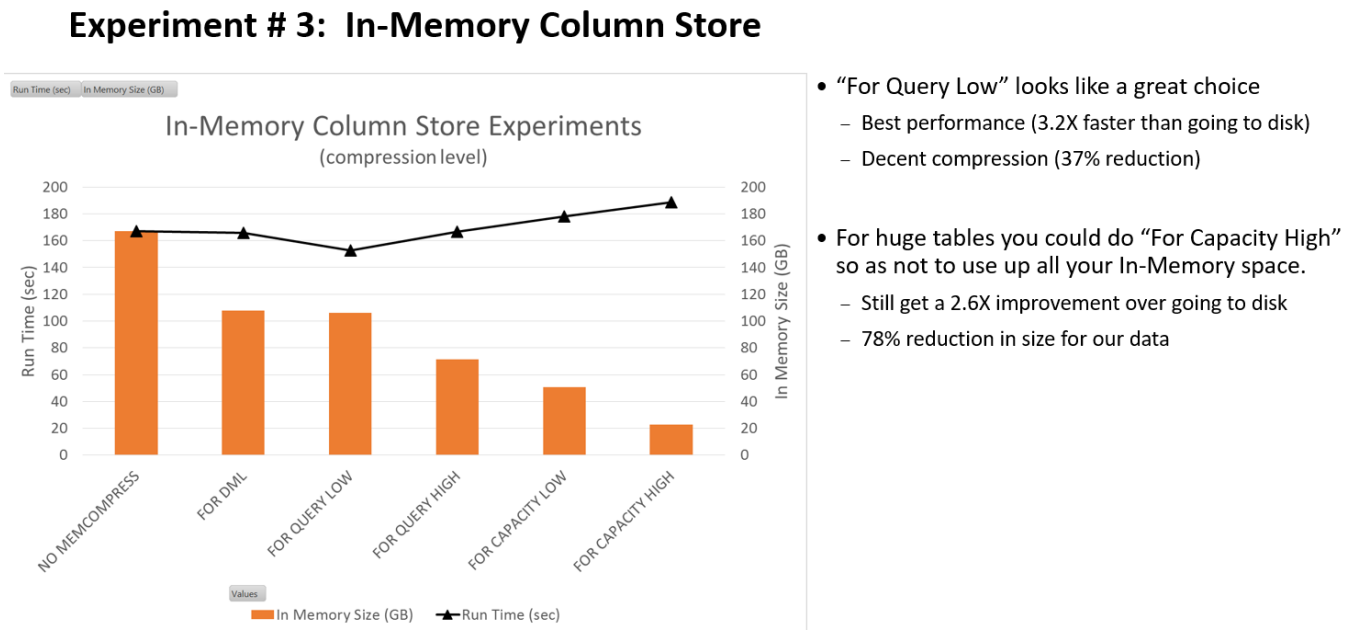


Fig. 11: Our highest performing feature, the IMCS has various compression ratios available for stored objects. This plot represents the effect of both compression ratio of the initial object, as well as the effect of runtime performance on our test queries. Note that the non-compressed table was roughly 168GB and it can be stored in-memory at a size of approximately 22GB. Additionally, the increase in access time to this compressed object is trivial based on the storage benefit.

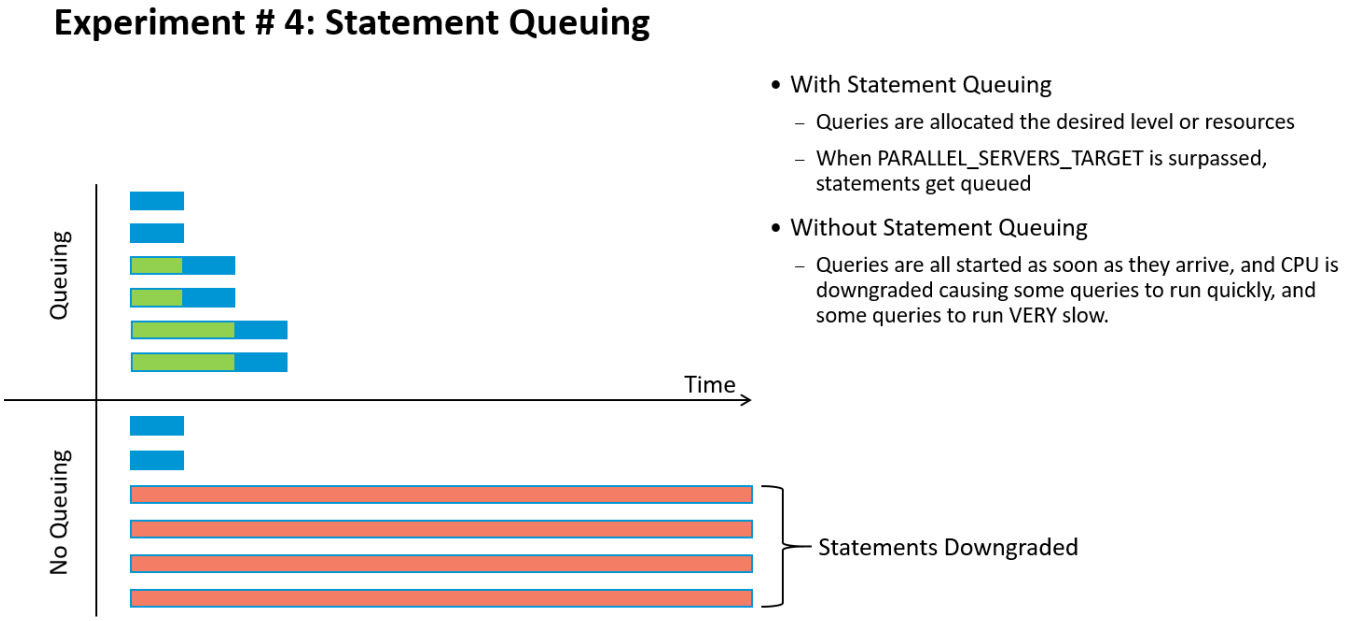


Fig. 12: Visualization of hypothesized runtime differences between two batches of queries where one batch has statement queuing enabled (Queuing) and the other has statement queuing disabled (No Queuing). Time spent waiting to run (queued) is marked with a green bar while execution times are marked with blue bars. Downgraded query run times that are executed with a sub-optimal amount of resources are marked with red bars.

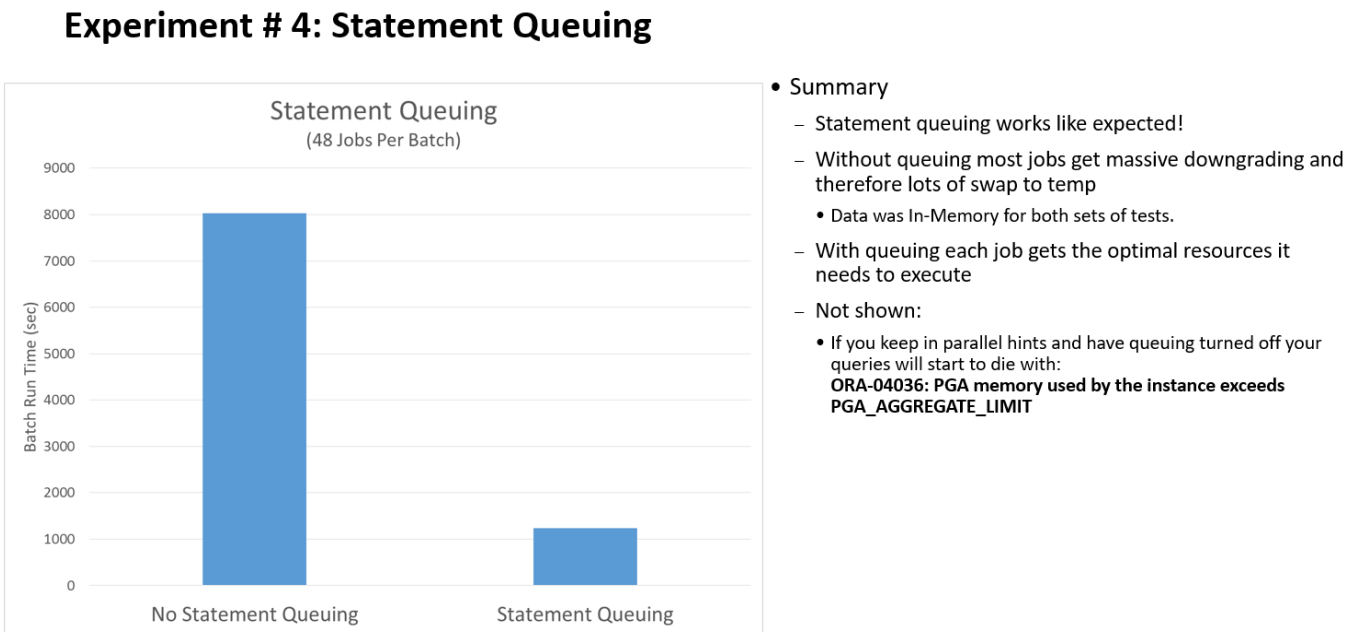


Fig. 13: Runtime differences of a large batch of queries with statement queuing enabled and the other with statement queuing disabled. In this case it is better to queue queries and make them wait for optimal resource allocation rather than try to process every query at once with sub-optimal resource allocation.

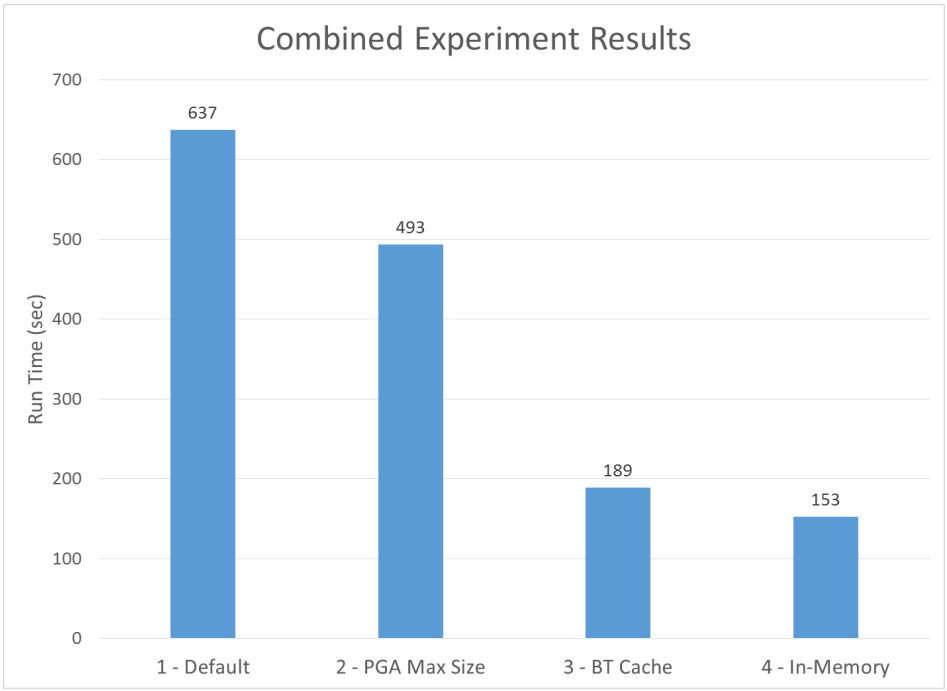


Fig. 14: Each column represents the summation of average runtime for each of the four tests we ran for each of our three main experiments, as compared to the default configuration. Of all the experiments, the In-Memory Column Store gave us the largest performance benefit with a runtime improvement of $\tilde{4}$.1x.

REFERENCES

[1] Oracle. *SQL Plus Users Guide and Reference*. [Online]. Available: https://docs.oracle.com/cd/...
[2] Oracle. SQL Developer Documentation and Release 4.1 [Online]. Available:https://docs.oracle.com/cd/...
[3] TOAD. *Toad For Oracle*. [Online]. Available: https://www.quest.com/documents/...
[4] Oracle Company. *Query Optimizer Concepts*. [Online]. Available: https://docs.oracle.com/...
[5] Surajit Chaudhuri. *An Overview Of Query Optimization In Relational Systems*. [Online]. Available: http://web.stanford.edu/...
[6] Oracle Company. *How Parallel Execution Works*. [Online]. Available: https://docs.oracle.com/...
[7] Oracle Company. *Partitioning Concepts*. [Online]. Available: https://docs.oracle.com/...
[8] Postgresql Volunteers. *General Information*. [Online]. Available: https://www.postgresql.org/about/
[9] Postgresql Volunteers. *Parallel Query*. [Online]. Available: https://www.postgresql.org/docs/...
[10] Postgresql Volunteers. *Documentation: 9.1: Partitioning*. [Online]. Available: https://www.postgresql.org/docs/...
[11] Amazon. *Tuning Query Performance*. [Online]. Available: https://aws.amazon.com/redshift/
[12] Amazon. *Tuning Query Performance*. [Online]. Available: http://docs.aws.amazon.com/redshift/...
[13] Amazon. *Choosing A Data Distribution Style*. [Online]. Available: http://docs.aws.amazon.com/redshift/...
[14] Amazon. *Unsupported Postgresql Features*. [Online]. Available: http://docs.aws.amazon.com/redshift/...
[15] Alex Woodie. *Oracle Gives 12c Database a Column-Oriented Makeover*. [Online]. Available: https://www.datanami.com/...
[16] Deba Chatterjee. *Manageability with Oracle Database 12c*. [Online]. Available: http://www.oracle.com/technetwork/...
[17] Mauro Pagano. *Mauro Pagano's Blog*. [Online]. Available: https://mauro-pagano.com/2015/02/16/...
[18] Tanel Poder. *tech.E2SN*. [Online]. Available: http://tech.e2sn.com/oracle-scripts-and-tools/...