Claremont Graduate University

**Final Project Database**

Class: Databases

Professor: Yan Li

Group 3:

Ming Ting

Zhengming Song

Silviana Munayco

**Table of Contents**

Executive Summary

1. Business Understanding 1
2. Data Exploration
3. Conceptual Data Modeling
4. Constraints and Business Understanding 2
   * + 1. Business User Identification
       2. Business Views
5. Business Views Security Requirements
6. Relational Database Design
   * + 1. Data Normalization
       2. Database Logical Design
7. Tables Creation and Data Loading
   * + 1. Parameter Values
       2. Storage Calculation
       3. Creation and Population of Tables
       4. Constraint Violations and Data Issues
8. Access Structures for Database System
9. Security Scripts and Tests
   * + 1. Table Level Security Scripts and Testing
       2. Business View Level Security Scripts and Testing

Executive Summary

This project’s main goal was to take a business’s unorganized data and tables to transform it into a logical and strategic relational database in which users of the company can run queries and obtain reports in the most efficient way possible.

The whole project’s process was implemented in the following steps:

1. Determine business needs
   1. Identify users of the company
   2. Identify potential sales transactions information needed for each user
2. Conceptual data modeling
   1. Separated tables as new entities
   2. Created entity relations diagram
3. Design relational database
   1. Normalize data types to make sure it is all consistent and appropriate
   2. Consider and implement acceptable data integrity constraints
   3. Create finalized tables with appropriate storage clauses in new schema
4. Security and testing
   1. Assign specific privileges depending on user type and role
   2. Testing to make sure all users have appropriate accesses and denials
5. Query Optimization
   1. Query cost testing based on user’s potential reports
   2. Implement indexes and views to ensure lower running costs

A. Business Understanding 1

A transaction processing system consists of computer hardware and software hosting a transaction-oriented application that performs the routine transactions necessary to conduct business. A transaction-oriented database is critical to all transactional management business application. In order to help our client, a small-mid size retail store, build a reliable transaction-oriented relational model database, we studied the raw data provided from them, conducted business interview with a business manager to verify that our database design complies with their business rules and requirements, keep only useful information, and embed useful reporting functionality/views in the database to help them better understand the business.

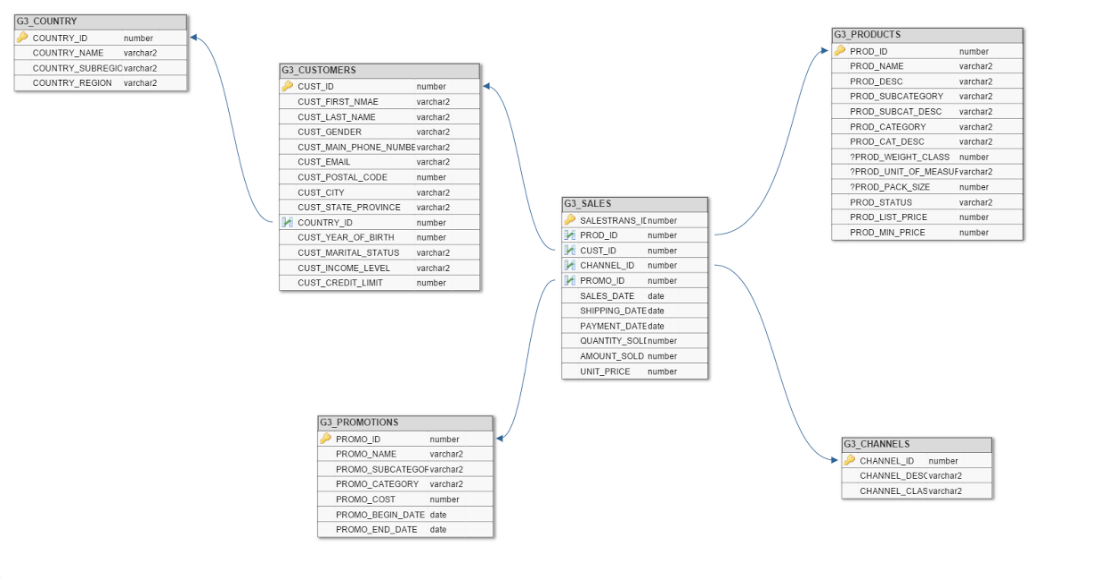
1. Data Exploration

There are initially 7 data tables:

* CHANNELS: channels table stores the channels information about source of sales, and channels description.
* CUSTOMERS\_INTX: customer internal table have the customer information collected by the company.
* CUSTOMERS\_EXT: customer external table stores the customer information the company get from external source such as survey and third party data providers. And there are a lot of null values in this table.
* PRODUCTS: product table stores the properties and characteristics of different products from this company.
* PROMOTIONS: promotion table records historical and current promotions information.
* SALES\_12\_13 and SALES\_14: the two sales table stores all the sale transaction with last three years.

C. Conceptual Data Modeling

Based on our initial understanding of the data, we think it is reasonable to merge (inner join) two customer tables since they are both contain customer information, and have the customer ids in common. And it is logical to merge (append) the two sales tables since they both store the same type of information. In addition, we think it is justifiable to take country and related information out of customer table since there are several filed that all describe countries. Hence, we have our initial conceptual design as below:



D. Constraints and Business Understanding 2

Next, we conducted several interviews with the company’s business manager. From the conversation with her, we confirmed and obtained her approval to discard the columns/fields that are not used anymore, and we established certain business rules they require, which are listed below:

* Product display price must be no less than the product cost (PRODUCT\_LIST\_PRICE>= PRODUCT\_MIN\_PRICE)
* Negative product price (UNIT\_PRICE) and sales quantity (QUANTITY\_SOLD) are not allowed, meaning (UNIT\_PRICE>=0, QUANTITY\_SOLD>=0)
* Customers’ gender should be inputted as 'M ' (male), 'F ' (female), or 'Null' (no entry)
* The incurred cost of promotions cannot be negative (PROMO\_COST>=0)
* The date of the beginning of a promotion cannot be later than the date of the end of said promotion (PROMO\_END\_DATE>=PROMO\_BEGIN\_DATE)
* The date for selling a product cannot be later than the date of the payment for said product (SALE\_DATE<=PAYMENT\_DATE)
* The date for selling a product cannot be later than the date for shipping said product (SALE\_DATE<=SHIPPING\_DATE)

Then we finalized the fields that we will keep, take out, and corresponding datatype:

1. CHANNELS:

CHANNEL\_ID (primary key), CHANNEL\_DESC, CHANNEL\_CLASS made not nullable due to business manager’s decision.

CHANNEL\_TOTAL is taken out because of low cardinality and data ambiguity.

1. CUSTOMERS\_INTX:

CUST\_ID (primary key), CUST\_FIRST\_NAME, CUST\_LAST\_NAME, CUST\_MAIN\_PHONE\_NUMBER, CUST\_STREET\_ADDRESS, CUST\_POSTAL\_CODE, CUST\_CITY, CUST\_STATE\_PROVINCE are made not nullable due to business manager’s decision.

CUST\_TOTAL, COUNTRY\_TOTAL are taken out because of low cardinality and data ambiguity.

1. CUSTOMERS\_EXT:

CUST\_ID (primary key) made not nullable

1. PRODUCTS:

PROD\_ID (primary key), PROD\_NAME, PROD\_DESC, PROD\_SUBCATEGORY, PROD\_SUBCAT\_DESC, PROD\_CATEGORY, PROD\_CAT\_DESC are made not nullable due to business manager’s decision.

PROD\_WEIGHT\_CLASS, PROD\_UNIT\_OF\_MEASURE, PROD\_PACK\_SIZE, PROD\_STATUS, PROD\_TOTAL are taken out because of low cardinality and data ambiguity.

1. PROMOTIONS:

PROMO\_ID (primary key), PROMO\_NAME, PROMO\_SUBCATEGORY, PROMO\_CATEGORY, PROMO\_COST, PROMO\_BEGIN\_DATE, PROMO\_END\_DATE are made not nullable due to business manager’s decision.

PROMO\_TOTAL is tabken out because of low cardinality and data ambiguity.

1. SALES\_12\_13, SALES\_14:

SALESTRANS\_ID (primary key), PROD\_ID, CUST\_ID, CHANNEL\_ID, PROMO\_ID, SALE\_DATE, QUANTITY\_SOLD, AMOUNT SOLD, UNIT\_PRICE are made not nullable due to business owner’s decision.

PROD\_ID, CUST\_ID, CHANNEL\_ID,PROMO\_ID are made to number datatype instead of varchar2 to match the primary keys in other related table.

Please see corresponding .txt files for implementation SQL code.

D1. Business User Identification

Based on business domain knowledge learnt from interview, we designed three major business users into this database:

1. User 1 (ZSONG) Accounting/Financial manager, who is in charge of accounting/financial department. This is User 1 (ZSONG). He/She will need business views to help him understand the profitability of different product and different promotions.
2. Customer manager, who takes care of customer relationship management, and needs to know who are the valuable customers to the firm and makes sure to send promotions in time to them. This is User 2 (IMUNAYCO).
3. Internet Sales manager, who needs access to the business views related to internet sales and products. This is User 3 (MTING).

D2. Business Views

*Business View #1:*

1. For user: internet sales manager
2. The purpose is to show the total number of items sold through the internet, and also the amount sold by category and to which country the items were sold to.
3. The columns seen in the business view are
   1. Year
   2. TotalSold
   3. Prod\_category
   4. country\_name
4. The tables and their columns used in this business view are
   1. G3\_Sales: Sales\_date, quantity\_sold, prod\_id, channel\_id, cust\_id
   2. G3\_Products: prod\_id, prod\_cat\_id
   3. G3\_Channels: channel\_id, channel\_desc
   4. G3\_Customers: cust\_id, country\_id
   5. G3\_Prod\_Cat\_Desc: product\_category, prod\_cat\_id
   6. G3\_country: country\_id, country\_name

*Business View #2:*

1. For user: internet sales manager
2. The purpose is to show the number of items sold by each category through the internet sales promotion
3. The columns seen in this business view are
   1. TotalSold
   2. Prod\_category
   3. Promo\_name
4. The tables used in this business view are
   1. G3\_Sales: quantity\_sold, prod\_id, channel\_id, promo\_id
   2. G3\_channels: channel\_desc, channel\_id
   3. G3\_Promotions: promo\_id, promo name
   4. G3\_Products: prod\_cat\_id, prod\_id
   5. G3\_Prod\_cat\_desc: prod\_category, prod\_cat\_id

*Business View #3:*

1. For user: internet sales manager
2. The purpose is to show the items bought through the internet by a specific customer, and how long it takes to receive payment and to ship the item out to the customer’s country.
3. The columns seen in this business view are
   1. Cust\_id
   2. Cust\_name
   3. Country\_name
   4. Prod\_name
   5. Daystopay
   6. Daystoship
4. The tables/columns used in this table are
   1. G3\_Sales: payment\_date, shipping\_date, sale\_date, prod\_id, cust\_id
   2. G3\_Customers: cust\_first\_name, cust\_last\_name, cust\_id, country\_id
   3. G3\_country: country\_id, country\_name
   4. G3\_Products: prod\_name, prod\_id

*Business View #4*

1. For user: Customer relations/Strategic manager
2. The purpose is to show the number of items sold in each category by customers of different income level
3. The columns seen in this business view are
   1. TotalSold
   2. Prod\_category
   3. Cust\_income\_level
4. The tables used in this business view are
   1. G3\_Sales: quantity\_sold, prod\_id, cust\_id
   2. G3\_Customers: cust\_income\_level, cust\_id
   3. G3\_products: prod\_id, prod\_cat\_id
   4. G3\_Prod\_cat\_desc: prod\_cat\_id, prod\_category

*Business View #5*

1. For user: Customer relations/Strategic manager
2. The purpose is to show customers with the most items bought from the company
3. The columns seen in this business view are
   1. Cust\_id
   2. Cust\_name
   3. TotalSold
   4. Cust\_main\_phone\_number
4. The tables used in this business view are
   1. G3\_Sales: quantity\_sold, prod\_id,
   2. G3\_Customers: cust\_main\_phone\_number, cust\_first\_name, cust\_last\_name, cust\_id

*Business View #6*

1. Annual\_product\_cat\_profit report
   1. User: Accounting/financial
2. Purpose: this view will help accounting/financial managers realize how much raw profit each type of product makes annually
3. Columns in the view: year, product\_category, product\_category\_desc, amount\_sold, min\_price, raw profit
4. Tables: columns

a. G3\_SALES: QUANTITY\_SOLD,UNIT\_PRICE,PROD\_ID

b. G3\_PRODUCTS: MIN\_PRICE

c. G3\_PROD\_CAT\_DESC: PROD\_CATEGORY,PROD\_CAT\_DESC

*Business View #7*

1. Promotion profitability
   1. User: accounting/financial manager
2. Purpose: display the profitability of each promotion type by showing the total revenue, total costs, and total discount amount under each promotion
3. Columns in the view: promo\_id, promo\_name, days\_of\_promo, amount\_sold,promo\_cost, prod\_min\_price,profit, discount
4. Tables: columns
   1. G3\_SALES: quantity\_sold, unit\_price,
   2. G3\_PRODUCTS: prod\_min\_price,prod\_list\_price
   3. G3\_PROMOTIONS:promo\_id,promo\_name,promo\_cost,promo\_begin\_date,promo\_end\_date

\* Please refer to .txt file for the scripts for each business view, which includes the type of view created, and the formulas used in the creation of the reports. Each of the queries were run after the creation of the tables and the populating the data for each table, as well as creating the appropriate indexes, which are discussed later in the report.

E. Business Views Security Requirements

Based on our query profile developed for the business views, the following table represents the initial access permission interaction matrix for each user and business view.

*Table 1. Initial Access Permission Interaction Matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **BV01** | **BV02** | **BV03** | **BV04** | **BV05** | **BV06** | **BV07** |
| **U01** |  |  |  |  |  | x | x |
| **U02** |  |  |  | x | x |  |  |
| **U03** | x | x | x |  |  |  |  |

Based on our query profile developed for the business views, the following table represents the access prevention interaction matrix, in which we show which users have no access to which business views.

*Table 2. Access Prevention Interaction Matrix*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **BV01** | **BV02** | **BV03** | **BV04** | **BV05** | **BV06** | **BV07** |
| **U01** |  |  |  | x | x |  |  |
| **U02** | x | x |  |  |  | x | x |
| **U03** |  |  |  |  |  | x | x |

\* Please refer to security .txt file for specific business view security scripts.

F. Relational Database Design

F1. Data Normalization

We implemented 3 Normal Form into the design of this database.

1. First Normal Form

Column values should be atomic, scalar or should be holding single value. Nor repetition of information or values in multiple columns.

All the tables are in first normal form.

1. Second Normal Form

No partial dependencies on a concatenated key.

* LI\_CHANNELS: primary key-> CHANNELS ID
* LI\_CUSTOMERS\_INTX: primary key -> CUST\_ID
* LI\_CUSTOMER\_EXT: primary key ->CUST\_ID
* LI\_PRODUCTS: primary key -> PROD\_ID
* LI\_PROMOTIONS: primary key -> PROMO\_ID
* LI\_SALES\_12\_13: primary key -> SALESTRANS\_ID
* LI\_SALES\_14: primary key -> SALESTRANS\_ID

Given all the primary key design above, there is no violation of 2NF

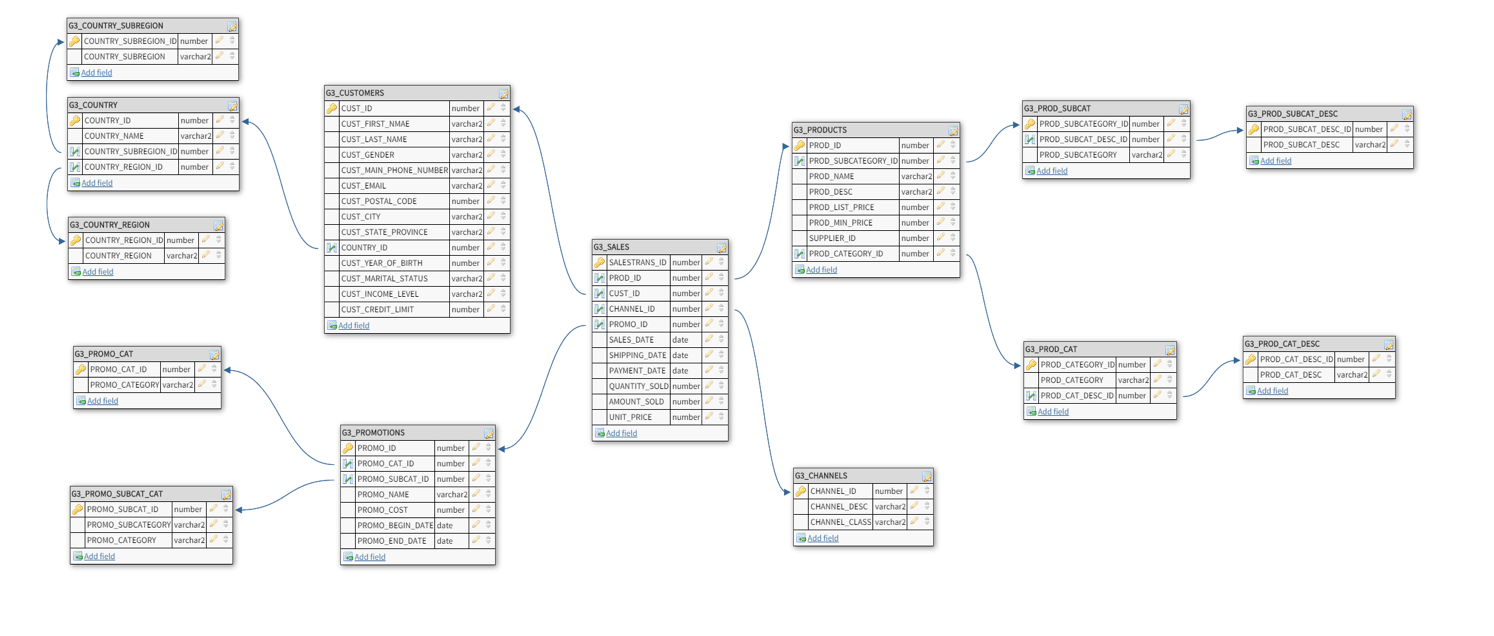
1. Third Normal Form

No dependence on No-Key Attributes.

* COUNTRY\_NAME, COUNTRY\_SUBREGION, COUNTRY\_REGION depend on COUNTRY\_ID, so we put those four columns out separately as a table (entity).
* PROD\_SUBCATE\_DESC depends on PROD\_SUBCATEGORY, so we take those columns out as a table, create ID columns for PROD\_SUBCATEGORY and PROD\_SUBCATE\_DESC
* PROD\_CATE\_DESC depends on PROD\_CATEGORY, so we take those columns out as two tables and create IDs for them, then link those tables using new created IDs.

F2. Database Logical Design

Finally, we reached our final logical design of our database, which is presented in the image below.



G. Relational Database Design

G1. Parameter Values (for each table)

We grouped certain tables based on the values of the parameters we expect. The parameters we are using are PCTFree, which is the amount of free space needed for being able to add extra columns in a table, PCTUsed, which is the amount of free space needed for being able to add extra rows in a table, and PCTIncrease, which is the growth rate of the database per year. Additional parameters used are “i”, which is the INITRANS value and that was set to 4 for each table, as we are not dealing with too large volumes of data; MINEXTENTS, which is used to compute the initial amount of space allocated, and is typically 1, therefore was set to 1 for all tables; B for the BD block size was set to 8k for each table, since it is the typical block size in Oracle.

The following are our justifications for the 3 PCTs parameter values, based on our schema presented above. (all tables start with G3)

|  |  |  |
| --- | --- | --- |
| Customers | Values | Explanation |
|  |  |  |
| PCTFree | 20% | based on columns that can be additionally obtained about customers |
| PCTUsed | 20% | based on differences found from year to year in amount of rows |
| PCTIncreate | 30% | based on discussion with business manager |

|  |  |  |
| --- | --- | --- |
| Promotions | Values | Explanation |
|  |  |  |
| PCTFree | 15% | based on information that can be added about promotions |
| PCTUsed | 20% | based on differences found from year to year in amount of rows |
| PCTIncreate | 20% | Based on discussion with business manager |

|  |  |  |
| --- | --- | --- |
| Sales | Values | Explanation |
|  |  |  |
| PCTFree | 20% | based on information that can be added about sales |
| PCTUsed | 20% | based on differences found from year to year in amount of rows |
| PCTIncreate | 30% | based on discussion with business manager |

|  |  |  |
| --- | --- | --- |
| Products | Values | Explanation |
|  |  |  |
| PCTFree | 15% | based on information that can be added about products |
| PCTUsed | 20% | based on differences found from year to year in amount of rows |
| PCTIncreate | 20% | based on comparison of previous years |

|  |  |  |
| --- | --- | --- |
| Channels, Country,  and Related | Values | Explanation |
|  |  |  |
| PCTFree | 5% | based on information that can be added about channels, country, and Related |
| PCTUsed | 10% | based on differences found from year to year in amount of rows |
| PCTIncreate | 10% | assumed to be the same |

|  |  |  |
| --- | --- | --- |
| Promo\_Cat,  Promo\_Subcat\_Cat | Values | Explanation |
|  |  |  |
| PCTFree | 10% | based on information that can be added about cat, subcat\_cat |
| PCTUsed | 20% | based on differences found from year to year in amount of rows |
| PCTIncreate | 20% | assumed to be the same |

|  |  |  |
| --- | --- | --- |
| Prod\_Cat, Promo\_Subcat, and Related | Values | Explanation |
|  |  |  |
| PCTFree | 10% | based on information that can be added about cat, subcat |
| PCTUsed | 15% | based on differences found from year to year in amount of rows |
| PCTIncreate | 15% | assumed to be the same |

G2. Storage Calculation (for each table)

For each table in the schema, we used the following calculations and syntax:

* For the number of rows we used the “select count(\*) from table” syntax. In the case of tables that were split, we added the “unique” attribute to make sure we are occupying the correct amount of space.
* To determine whether to use 1 or 3 in the calculation of the row average, we used the “select max(length(column)) from table” syntax. There was no column length that was larger than 255, therefore we always used the value 1 for each “vsize”.
* The average row length R was calculated using the “select avg(3+1+vsize(column)+…) from table” syntax. However, in the tables in which we had to create an incremental id column, we estimated that the “vsize” should be 5. By comparison the maximum we have observed has been 3 (for fairly long string of numbers), therefore we deemed our estimate to be appropriate. Also, for tables that were merged together the average row length was calculated separately, one of which did not have the header size in the formula, and then averaged the two calculations.
* The Blocking Factor was calculated in excel using the formula:  
   FLOOR((B-61-23\*i)/(2+(R\*100)/(100-PCTFREE))
* The Number of Blocks was calculated in excel using the formula:  
  Ceiling(Number of Rows/Blocking Factor)
* The Storage Estimate was calculated in excel using the formula:  
  Number of Blocks Required\*B (block size)/1024
* Some of the calculations were based on the tables we had access to from the liy26 schema, whereas some of the calculations were based on views we created for merging certain tables.

\* For full scripts for the storage calculation please refer to storage .txt file

The following tables give the storage requirements for each of the tables in our schema.

|  |  |
| --- | --- |
| ***G3\_COUNTRY\_SUBREGION*** |  |
|  |  |
| Number of Rows | 10 |
| average row length R | 27.19445045 |
| PCTFREE | 5 |
| PCTUSED | 10 |
| PCTINCREASE | 10 |
| MINEXTENTS | 1 |
| Blocking Factor | 262 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_COUNTRY\_REGION*** |  |
|  |  |
| Number of Rows | 5 |
| average row length R | 16.73133333 |
| PCTFREE | 5 |
| PCTUSED | 10 |
| PCTINCREASE | 10 |
| MINEXTENTS | 1 |
| Blocking Factor | 409 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_COUNTRY*** |  |
|  |  |
| Number of Rows | 19 |
| average row length R | 40.1163964 |
| PCTFREE | 5 |
| PCTUSED | 10 |
| PCTINCREASE | 10 |
| MINEXTENTS | 1 |
| Blocking Factor | 181 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PROMO\_SUBCAT\_CAT*** |  |
|  |  |
| Number of Rows | 22 |
| average row length R | 25.7554672 |
| PCTFREE | 10 |
| PCTUSED | 20 |
| PCTINCREASE | 20 |
| MINEXTENTS | 1 |
| Blocking Factor | 262 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PROMO\_CAT*** |  |
|  |  |
| Number of Rows | 9 |
| average row length R | 15.97614214 |
| PCTFREE | 10 |
| PCTUSED | 20 |
| PCTINCREASE | 20 |
| MINEXTENTS | 1 |
| Blocking Factor | 407 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PROMOTIONS*** |  |
|  |  |
| Number of Rows | 503 |
| average row length R | 63.50695825 |
| PCTFREE | 15 |
| PCTUSED | 20 |
| PCTINCREASE | 20 |
| MINEXTENTS | 1 |
| Blocking Factor | 104 |
| Number of Blocks | 5 |
| Storage Estimate | 40 |

|  |  |
| --- | --- |
| ***G3\_CUSTOMERS*** |  |
|  |  |
| Number of Rows | 55500 |
| average row length R | 140.041062 |
| PCTFREE | 20 |
| PCTUSED | 20 |
| PCTINCREASE | 30 |
| MINEXTENTS | 1 |
| Blocking Factor | 45 |
| Number of Blocks | 1234 |
| Storage Estimate | 9872 |

|  |  |
| --- | --- |
| ***G3\_SALES*** |  |
|  |  |
| Number of Rows | 540564 |
| average row length R | 52.46110078 |
| PCTFREE | 20 |
| PCTUSED | 20 |
| PCTINCREASE | 30 |
| MINEXTENTS | 1 |
| Blocking Factor | 118 |
| Number of Blocks | 4582 |
| Storage Estimate | 36656 |

|  |  |
| --- | --- |
| ***G3\_CHANNELS*** |  |
|  |  |
| Number of Rows | 5 |
| average row length R | 22.8 |
| PCTFREE | 5 |
| PCTUSED | 10 |
| PCTINCREASE | 10 |
| MINEXTENTS | 1 |
| Blocking Factor | 309 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PROD\_SUBCAT\_DESC*** |  |
|  |  |
| Number of Rows | 21 |
| average row length R | 22.5 |
| PCTFREE | 10 |
| PCTUSED | 15 |
| PCTINCREASE | 15 |
| MINEXTENTS | 1 |
| Blocking Factor | 297 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PROD\_SUBCAT*** |  |
|  |  |
| Number of Rows | 21 |
| average row length R | 28.5 |
| PCTFREE | 10 |
| PCTUSED | 15 |
| PCTINCREASE | 15 |
| MINEXTENTS | 1 |
| Blocking Factor | 238 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PROD\_CAT\_DESC*** |  |
|  |  |
| Number of Rows | 5 |
| average row length R | 25.83333333 |
| PCTFREE | 10 |
| PCTUSED | 15 |
| PCTINCREASE | 15 |
| MINEXTENTS | 1 |
| Blocking Factor | 261 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PROD\_CAT*** |  |
|  |  |
| Number of Rows | 6 |
| average row length R | 31.91666667 |
| PCTFREE | 10 |
| PCTUSED | 15 |
| PCTINCREASE | 15 |
| MINEXTENTS | 1 |
| Blocking Factor | 214 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

|  |  |
| --- | --- |
| ***G3\_PRODUCTS*** |  |
|  |  |
| Number of Rows | 72 |
| average row length R | 84.52777778 |
| PCTFREE | 15 |
| PCTUSED | 20 |
| PCTINCREASE | 20 |
| MINEXTENTS | 1 |
| Blocking Factor | 79 |
| Number of Blocks | 1 |
| Storage Estimate | 8 |

G3. Creation and Population of Tables

­\* For the full syntax creation of all views, of all the tables, and the population of all the tables, please refer to tablegen .txt file.

The following are the steps taken for the creation of each of the tables:

* Inserting data into a table was done with from a table that we had access to from the liy26 schema or from a view we created, for those tables that had columns from a number of different tables. The syntax used was:  
  “insert into table (columns)   
   select columns   
   from table (or view)”.  
  On the G3\_SALES table the insertion was done twice from 2 different tables in the liy26 schema, since they had nothing in common and did not violate any of the constraints implemented. The syntax was used in the exact same method for every table in our schema.
* Creating views was necessary for those tables which contained columns from different tables in the liy26 schema, and in which we had to use one or more “inner join” statements. The tables G3\_PROMOTIONS, G3\_CUSTOMERS, G3\_PROD\_SUBCAT, G3\_PROD\_CAT, AND G3\_PRODUCTS required the creation of a virtual view. The syntax the we used was:   
    
  “create view name as  
   select columns  
   from table  
   inner join table   
   on column equality condition …”
* In creating each table in the schema, we encountered the following situations:
  + Creating a table in which we had to create an incremental id column. The tables which contained this situation are G3\_COUNTRY\_SUBREGION, G3\_COUNTRY\_REGION, G3\_PROMO\_SUBCAT\_CAT, G3\_PROMO\_CAT, G3\_PROD\_SUBCAT\_DESC, G3\_PROD\_SUBCAT, G3\_PROD\_CAT\_DESC, G3\_PROD\_CAT. These new ID columns are always number, and have estimated length of maximum 5. The syntax used was “col\_name\_id number GENERATED AS IDENTITY”.
  + Creating a table which contained primary key only, primary key and foreign key(s), and primary key, foreign key(s), and other business related constraints.
  + The syntax for primary key used was “Constraint col\_name\_id\_pk primary key(col\_name\_id)” .
  + The syntax for foreign key used was:  
    “CONSTRAINT fk\_col\_name\_id FOREIGN KEY (col\_name\_id)  
    REFERENCES parent\_table(col\_name\_id)”
  + The syntax for business related constraints used was “CONSTRAINT check\_name CHECK (condition)”
* In the creation of each table, we also had to define each column, what data type it included, and whether it was nullable or not.
* At the end of each of the creation statements, we included the storage based on the calculations and explanations previously given.

Here is an example of a full creation statement, which includes most the mentioned above.

create table G3\_PRODUCTS (

prod\_id number(6) not null,

prod\_subcategory\_id number not null,

prod\_category\_id number not null,

prod\_name varchar2(50) not null,

prod\_desc varchar2(4000) not null,

prod\_list\_price number(8,2) not null,

prod\_min\_price number(8,2) not null,

supplier\_id number(6) not null,

Constraint prod\_id\_pk primary key(prod\_id),

CONSTRAINT fk\_prod\_subcategory\_id FOREIGN KEY (prod\_subcategory\_id)

REFERENCES g3\_prod\_subcat(prod\_subcategory\_id),

CONSTRAINT fk\_prod\_category\_id FOREIGN KEY (prod\_category\_id)

REFERENCES g3\_prod\_cat(prod\_category\_id),

CONSTRAINT check\_prod\_list\_price CHECK (prod\_list\_price>=0),

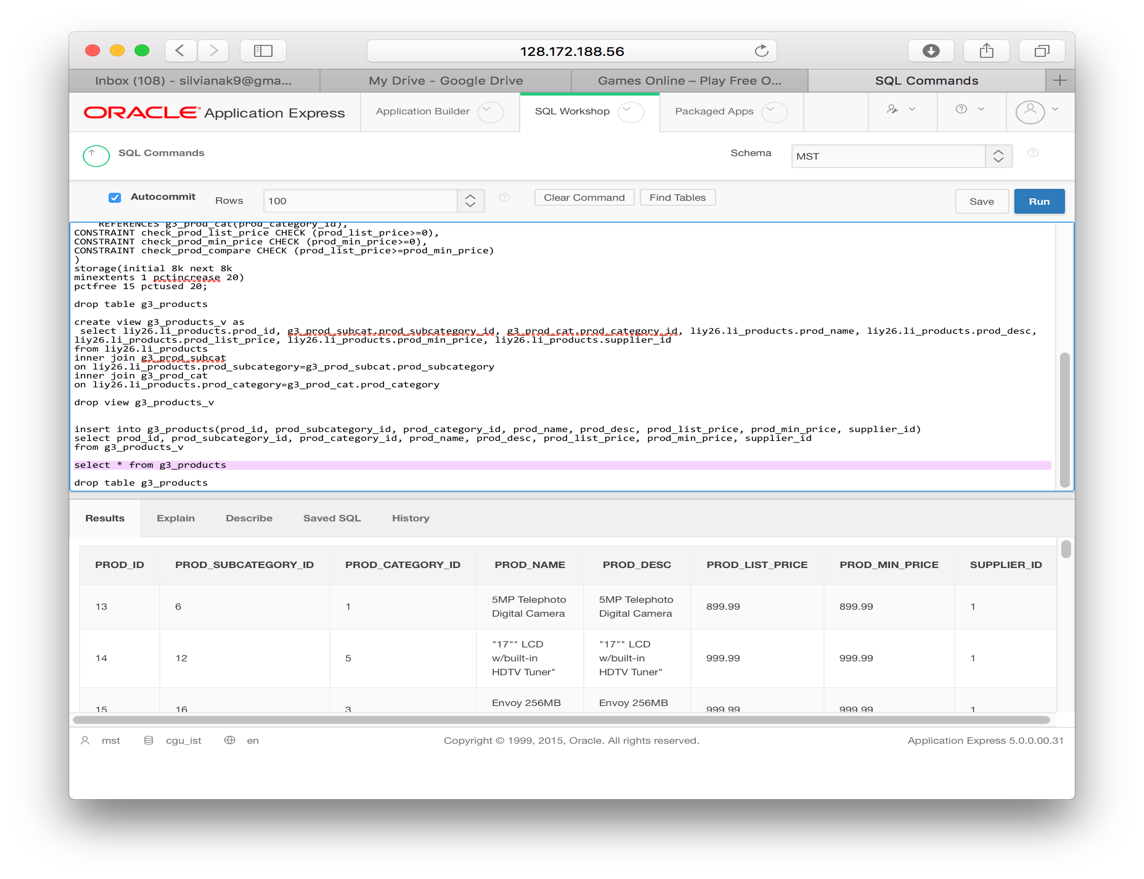
CONSTRAINT check\_prod\_min\_price CHECK (prod\_min\_price>=0),

CONSTRAINT check\_prod\_compare CHECK (prod\_list\_price>=prod\_min\_price)

)

storage(initial 8k next 8k

minextents 1 pctincrease 20)

pctfree 15 pctused 20;

We also included proof of the creation of the table G3\_PRODUCTS.

G4. Constraint Violations and Data Issues

The constraint violations and nullable changes were mentioned in the part D of the report. In terms of data issues, the only case in which they occurred are in the two original Sales tables. The differences in data type between sales\_12\_13 and sales\_14: prod\_id , cust\_id, channel\_id, promo\_id in sales\_12\_13’s data type changed to number (22) from varchar (225) to match sales\_14.

H. Access Structures for Database System

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Costs | 1. Primary Index (automatically created) | 1. Special Index | 1. Index Hint | 1. Materialized View | Best Plan |
| Bv01 | 18692 | 3966 | 3967 | 4154 | 2 |
| Bv02 | 18692 | 1521 | 1522 | x | 2 |
| Bv03 | 2585 | 2584 | 2585 | x | 2 |
| Bv04 | 2585 | 2814 | 2815 | x | 2 |
| Bv05 | 2585 | 14198 | 14199 | x | 1 |
| Bv06 | 2585 | 1525 | 1526 | x | 2 |
| Bv07 | 18692 | 18692 | 18908 | 21298 | 1 |

After running all 7 business view queries, we found that not all indexes created had a positive effect on all of them in terms of cost. Most of the queries had a significant decrease in cost after creating bitmap indexes and regular indexes on a column that we felt would help the query time and cost. For the internet sales manager, we created a bitmap index on the g3\_channels table channel\_desc column as it had low cardinality and will be used all the time for that specific user. The changes for bv01 and bv02 were very significant, as it went from 18692 cost of just using automatically created primary key indexes to just 3966 and 1522 after using the bitmap indexes. Bv06 also had a lower cost at 2585 to 1536, which is not as large of a difference as others but still beneficial.

Then, there are some queries that did not benefit at all from the bitmap index, regular index or index hints, such as bv03, and bv04 as they stayed the same cost. Then, there is bv05 where the special indexes should not be used as it had an adverse effect on the cost significantly. It went from 2585 on just the primary indexes to 14198 after the special indexes.

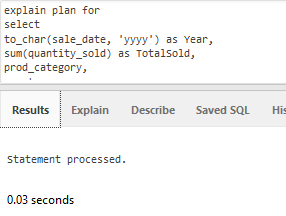
Finally, we decided to use materialized view on BV01 and BV07 as they have the highest initial cost. Bv01’s materialized view helped with the cost significantly from 18692 to 4154. But, bv07 had an adverse effect in which the cost increased from 18692 to 18908.

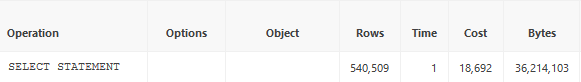
The following are indexes and 2 materialized that were strategically placed in search of lower costs and faster query times.

* bitmap index "internetindex" created on g3\_channels table, on channel\_desc column
* bitmap index "promoindex" created on g3\_promotions table, on promo\_name column
* index "pbeginindex" created on g3\_promotions table, on promo\_begin\_date column
* index 'pendindex' on g3\_promotions table, on promo\_end\_date column
* index hint was also used on each of these index
* materialized view BV1MView created using select statement from BV01
* materialized view BV7MView created using select statement from BV07

BV01:

Internet sales manager is looking for total quantity sold of items sold through the internet in their specific category, which country bought the most of this category of products and organized by year.





Cost without any indexes: 18692



Cost after indexes: 3966

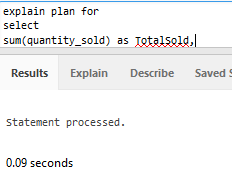


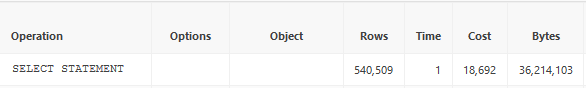
Cost after materialized view: 4154

BV02:

Internet sales manager looking for the quantity of items sold through the internet in their specific category that was sold during the internet promotion.

Explain plan for:





Cost without any created indexes: 18692

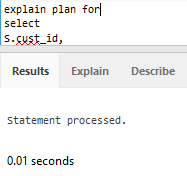


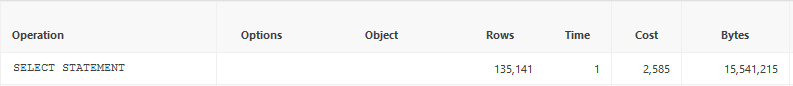
Cost after bitmap indexes: 1521

BV03:

Internet sales manager looking for the customers and their products and seeing how long it took to ship their item and for the payment to come through from the customer’s respective countries.

Explain plan for:





Cost without any indexes: 2585

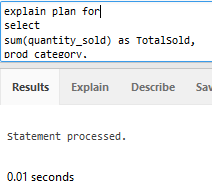


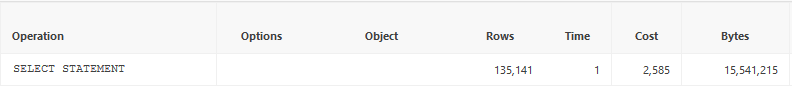
Cost after bitmap indexes: 2584

BV04:

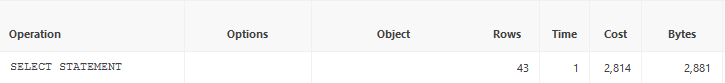
Customer relations manager seeing which category of items were bought most from customers of different income level.

Explain plan for





Cost without any indexes: 2585

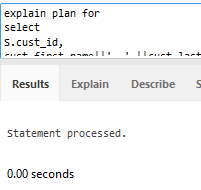


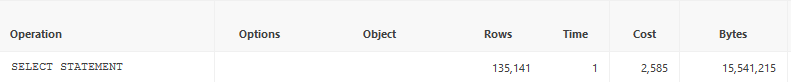
Cost after index: 2814

BV05:

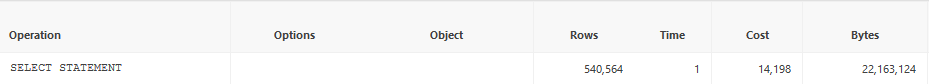
Customers relations manager seeing who are the best customers for the company and their contact information.

Explain plan for





Cost without any indexes: 2585

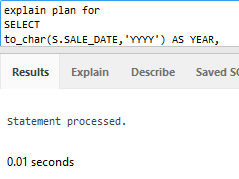


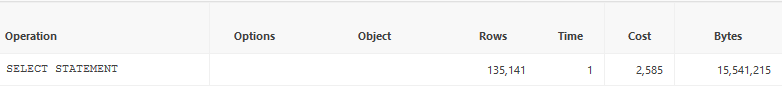
Cost after indexes: 14198

BV06:

Financial/accounting employee looking to report the product categories that gave the most raw profit and organized by year.

Explain plan for





Cost without creating indexes: 2585

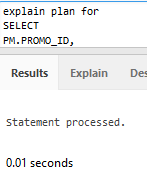


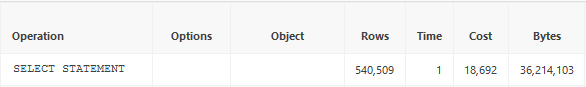
Cost after indexes: 1525

BV07:

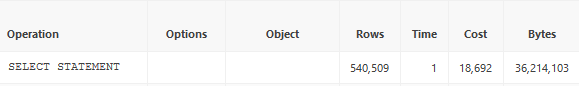
Finance/accounting employee looking to report profitability on each promotion category within a certain period of time.

Explain plan for

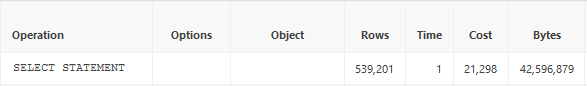




Cost without any indexes: 18,692



Cost after indexes: 18692



Cost after materialized view: 2129.

I. Security Scripts and Tests

\* Please refer to the security tables and views .txt file for full scripts for both security and testing.

I1. Table Level Security Scripts and Testing

We decided that first we will deal with the security scripts at the table level, for users to be able to create their own views at leisure, based on the two matrices created in part E of the report. The scripts were run through the MST account, and the tests were done from the users accounts. The following are the accesses granted to certain tables, some of which are under specific conditions, based on the user.

*Business User 1 – Finance and Accounting Manager (zsong)*

Insert privileges were not allowed on any of the tables.

* **Positive Test** is not possible since the user has no access to any of the tables.
* **Negative Test** was done by attempting to insert a row into the PROMOTIONS table, which of course was unsuccessful.

Update privileges were not allowed on any of the tables.

* **Positive Test** is not possible since the user has no access to any of the tables.
* **Negative Test** was done by attempting to update a row with a specific promo\_id into the PROMOTIONS table, which of course was unsuccessful.

Delete privileges were not allowed on any of the tables.

* **Positive Test** is not possible since the user has no access to any of the tables.
* **Negative Test** was done by attempting to delete an entire row with a specific promo\_id from the PROMOTIONS table, which of course was unsuccessful.

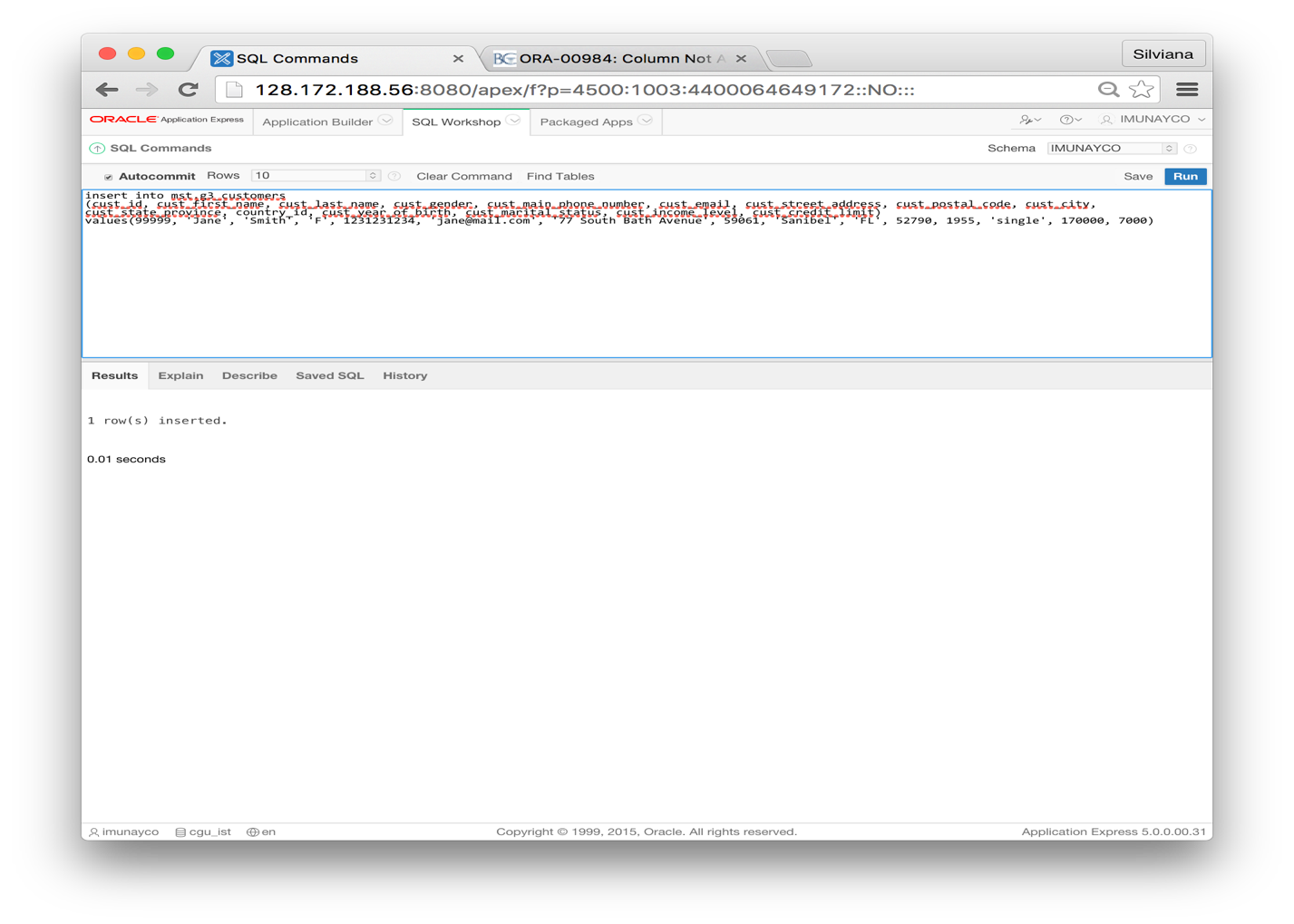
Select privileges were given to the user on the tables G3\_SALES, G3\_PRODUCTS, G3\_PROD\_SUBCAT, G3\_PROD\_CAT, G3\_PROMOTIONS, G3\_PROMO\_CAT, G3\_PROMO\_SUBCAT\_CAT, G3\_CHANNELS with no restrictions.

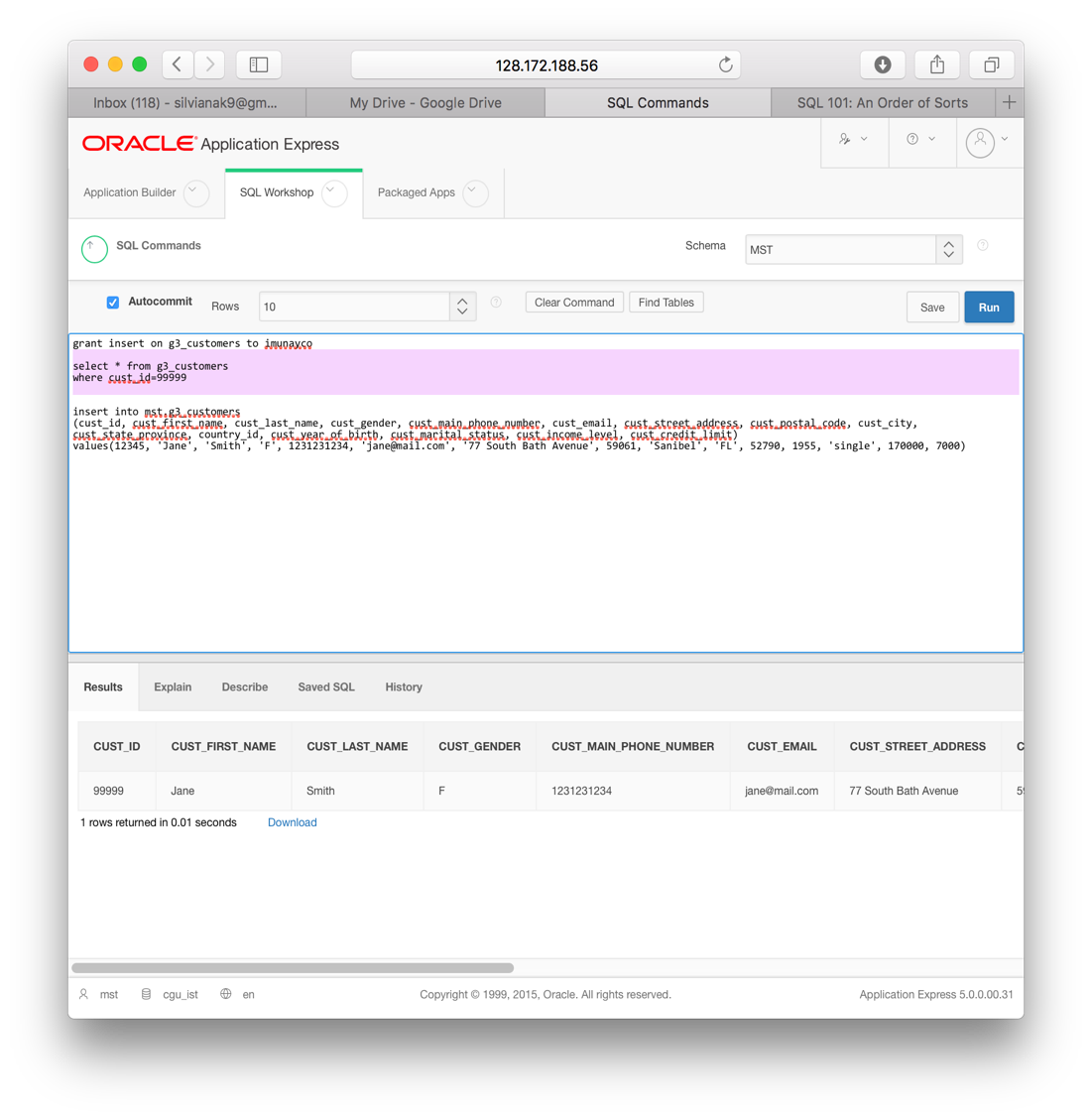
* **Positive Test** was done by selecting columns from one of tables, successfully.
* **Negative Test** was done by attempting to select columns from the table G3\_COUNTRY which the user does not have access to, which of course was unsuccessful.

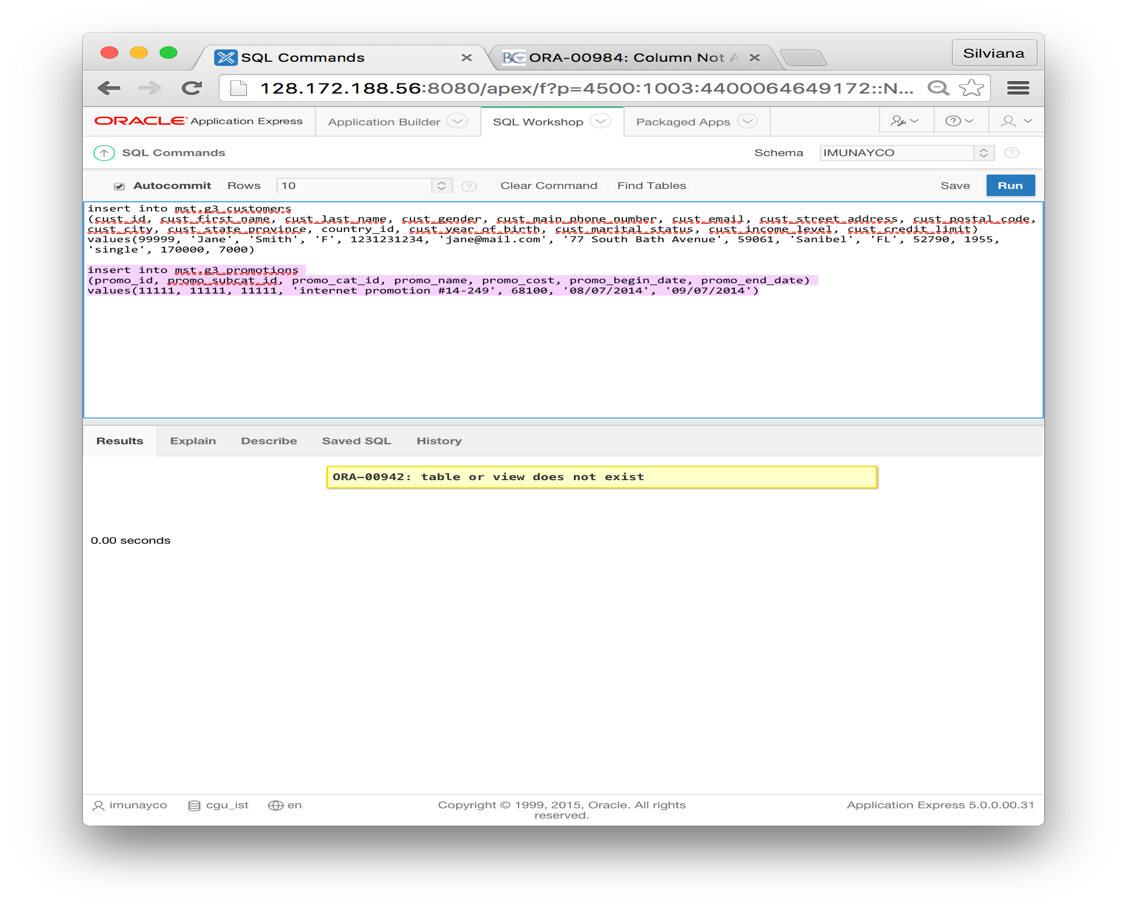
*Business User 2 – Customer Relations Manager (imunayco)*

Insert privileges were given on table G3\_CUSTOMERS only, with no restrictions.

* **Positive Test** inserted an entire row into the customer column, which had an id that did not conflict with the unique clause of the CUSTOMERS table (proof was given by selection of the column with the specific id created).
* **Negative Test** was done by attempting to insert a row into the PROMOTIONS table, which the user does not have access to, which of course was unsuccessful.

We have inserted the following screen shots as proof the positive and negative test.

PositiveTest

****

Negative

Test

Update privileges were given on table G3\_CUSTOMERS only, with no restrictions.

* **Positive Test** was done by changing the marital status for a specific customer id.
* **Negative Test** was done by attempting to update a row with a specific promo\_id of the PROMOTIONS table, which of course was unsuccessful.

Delete privileges were given on table G3\_CUSTOMERS only, with no restrictions.

* **Positive Test** was done by deleting a row with a specific customer id.
* **Negative Test** was done by attempting to delete an entire row with a specific promo\_id from the PROMOTIONS table, which of course was unsuccessful.

Select privileges were given to the user on the tables G3\_CUSTOMERS, G3\_COUNTRY, G3\_SALES, G3\_PRODUCTS, G3\_PROD\_SUBCAT, G3\_PROD\_CAT, G3\_PROMOTIONS, G3\_PROMO\_CAT, G3\_PROMO\_SUBCAT\_CAT with no restrictions.

* **Positive Test** was done by selecting columns from one of tables, successfully.
* **Negative Test** was done by attempting to select columns from the table G3\_COUNTRY\_region which the user does not have access to, which of course was unsuccessful.

*Business User 3 – Internet Sales Manager (mting)*

The insert, update, and delete privileges given to this user were based on the condition that they can only be used based on Internet related columns and rows, in the SALES table. Therefore, we created a virtual view of the SALES table based on the requirement that the channel\_id needs to be 4, which is the only internet related column. The rest of the tables the user was given access to have no restrictions.

Insert privileges were given on G3\_SALES\_VIEW (internet only), G3\_PRODUCTS, G3\_PROD\_SUBCAT, G3\_PROD\_CAT.

* **Positive Test** was done by inserting a row into the sales view, which no conflict of the unique clause of the id column, and channel id 4.
* **Negative Test** was done by attempting to insert a row with channel id not 4 into the sales view table, which of course was unsuccessful.

Update privileges were given on G3\_SALES\_VIEW (internet only), G3\_PRODUCTS, G3\_PROD\_SUBCAT, G3\_PROD\_CAT.

* **Positive Test** was done by changing the sale date with a specific salestrans id.
* **Negative Test** was done by attempting to update the sale date with channel id not 4, which of course was unsuccessful.

Delete privileges were given on G3\_SALES\_VIEW (internet only), G3\_PRODUCTS, G3\_PROD\_SUBCAT, G3\_PROD\_CAT.

* **Positive Test** was done by deleting a row with a specific salestrans id.
* **Negative Test** was done by attempting to delete an entire row with a channel id not 4 from the sales view, which of course was unsuccessful.

Select privileges were given to the user on the tables on G3\_CUSTOMERS, G3\_COUNTRY, G3\_SALES\_VIEW, G3\_PRODUCTS, G3\_PROD\_SUBCAT, G3\_PROD\_CAT, G3\_PROMOTIONS, G3\_PROMO\_CAT, G3\_PROMO\_SUBCAT\_CAT, G3\_CHANNELS with no restrictions.

* **Positive Test** was done by selecting columns from one of tables, successfully.
* **Negative Test** was done by attempting to select columns from the table G3\_COUNTRY\_SUBREGION, which of course was unsuccessful.

I1. Business View Level Security Scripts and Testing

Based on the business views created and our security matrices discussed in part E of the report, we gave viewing (select) privileges to the users on the appropriate views. The following is a list of views access based on the user, together with positive and negative testing. Note that MVIEW refers to a materialized view, and VVIEW refers to a virtual view.

*Business User 1 – Finance and Accounting Manager (zsong)*

Select privileges were given on the BV1MVIEW, BV2VVIEW, BV3VVIEW, BV6VVIEW, BV7MVIEW.

* **Positive Test** was done by selecting columns from BV1MVIEW, successfully.
* **Negative Test** was done by attempting to select columns from the BV4MVIEW, unsuccessfully.

*Business User 2 – Customer Relations Manager (imunayco)*

Select privileges were given on the BV3VVIEW, BV4VVIEW, BV5VVIEW.

* **Positive Test** was done by selecting columns from BV3VVIEW, successfully.
* **Negative Test** was done by attempting to select columns from the BV1MVIEW, unsuccessfully.

*Business User 3 – Internet Sales Manager (mting)*

Select privileges were given on the BV1MVIEW, BV2VVIEW, BV3VVIEW, BV4VVIEW, BV5VVIEW.

* **Positive Test** was done by selecting columns from BV1MVIEW, successfully.
* **Negative Test** was done by attempting to select columns from the BV6MVIEW, unsuccessfully.