Claremont Graduate University

Final Project Database

Class: Databases
Professor: Yan Li
Group 3:
Ming Ting
Zhengming Song
Silviana Munayco

Table of Contents

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

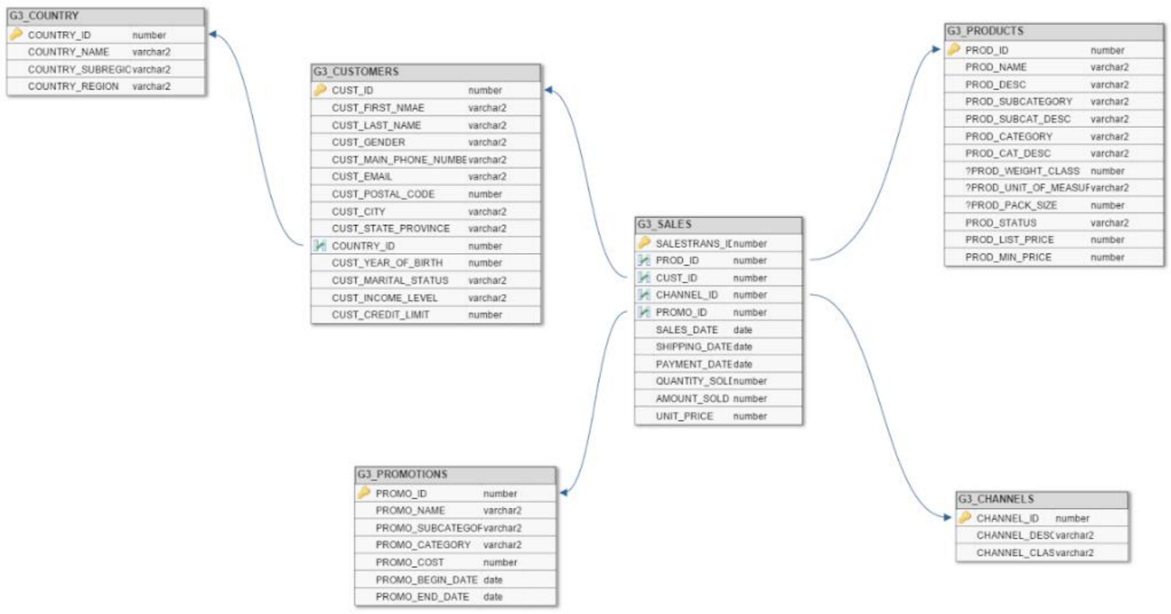
B. Data Exploration

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:

a. 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.

b. 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.

c. CUSTOMERS_EXT:

   CUST_ID (primary key) made not nullable

d. 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.

e. 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.

f. 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:

a. 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.
b. 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).
c. 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:

A. For user: internet sales manager
B. 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.
C. The columns seen in the business view are
    a. Year
    b. TotalSold
    c. Prod_category
    d. country_name
D. The tables and their columns used in this business view are
    a. G3_Sales: Sales_date, quantity_sold, prod_id, channel_id, cust_id
    b. G3_Products: prod_id, prod_cat_id
    c. G3_Channels: channel_id, channel_desc
    d. G3_Customers: cust_id, country_id
    e. G3_Prod_Cat_Desc: product_category, prod_cat_id
    f. G3_country: country_id, country_name

Business View #2:

A. For user: internet sales manager
B. The purpose is to show the number of items sold by each category through the internet sales promotion
C. The columns seen in this business view are
    a. TotalSold
    b. Prod_category
    c. Promo_name
D. The tables used in this business view are
    a. G3_Sales: quantity_sold, prod_id, channel_id, promo_id
    b. G3_channels: channel_desc, channel_id
    c. G3_Promotions: promo_id, promo name
    d. G3_Products: prod_cat_id, prod_id
    e. G3_Prod_cat_desc: prod_category, prod_cat_id

Business View #3:

A. For user: internet sales manager
B. 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.
C. The columns seen in this business view are
    a. Cust_id
    b. Cust_name
    c. Country_name
    d. Prod_name
    e. Daystopay
    f. Daystoship
D. The tables/columns used in this table are
    a. G3_Sales: payment_date, shipping_date, sale_date, prod_id, cust_id
    b. G3_Customers: cust_first_name, cust_last_name, cust_id, country_id
    c. G3_country: country_id, country_name
    d. G3_Products: prod_name, prod_id

Business View #4

A. For user: Customer relations/Strategic manager
B. The purpose is to show the number of items sold in each category by customers of different income level
C. The columns seen in this business view are
    a. TotalSold
    b. Prod_category
    c. Cust_income_level
D. The tables used in this business view are
    a. G3_Sales: quantity_sold, prod_id, cust_id

    b. G3_Customers: cust_income_level, cust_id
    c. G3_products: prod_id, prod_cat_id
    d. G3_Prod_cat_desc: prod_cat_id, prod_category

Business View #5

A. For user: Customer relations/Strategic manager
B. The purpose is to show customers with the most items bought from the company
C. The columns seen in this business view are
    a. Cust_id
    b. Cust_name
    c. TotalSold
    d. Cust_main_phone_number
D. The tables used in this business view are
    a. G3_Sales: quantity_sold, prod_id,
    b. G3_Customers: cust_main_phone_number, cust_first_name, cust_last_name, cust_id

Business View #6

A. Annual_product_cat_profit report
    a. User: Accounting/financial
B. Purpose: this view will help accounting/financial managers realize how much raw profit each type of product makes annually
C. Columns in the view: year, product_category, product_category_desc, amount_sold, min_price, raw profit
D. 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

A. Promotion profitability
    a. User: accounting/financial manager
B. Purpose: display the profitability of each promotion type by showing the total revenue, total costs, and total discount amount under each promotion
C. Columns in the view: promo_id, promo_name, days_of_promo, amount_sold,promo_cost, prod_min_price,profit, discount
D. Tables: columns
    a. G3_SALES: quantity_sold, unit_price,
    b. G3_PRODUCTS: prod_min_price,prod_list_price
    c. 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.

a. 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.

b. 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

c. Third Normal Form

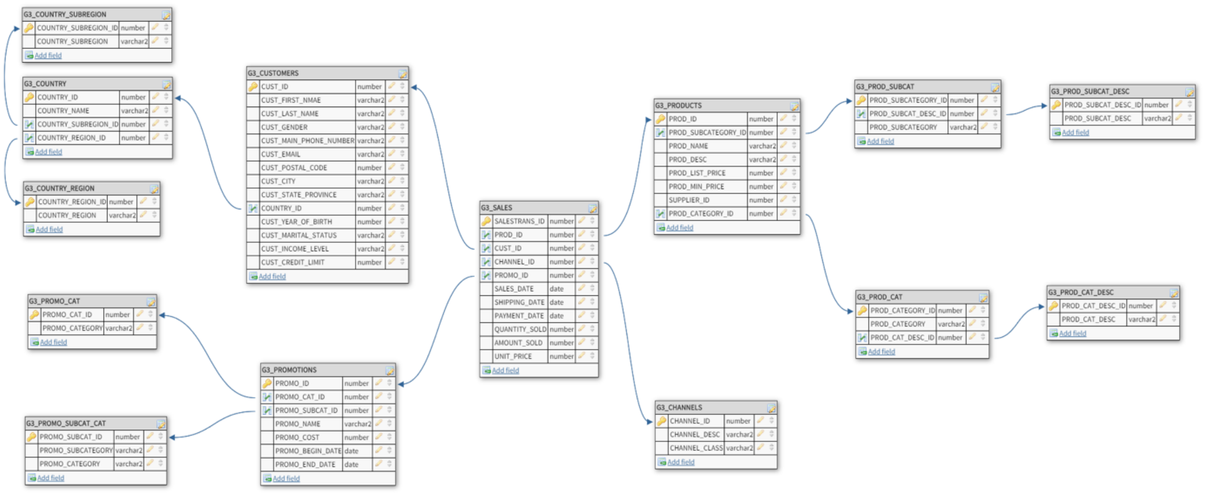
No dependence on No-Key Attributes.

- 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)”.
  ```

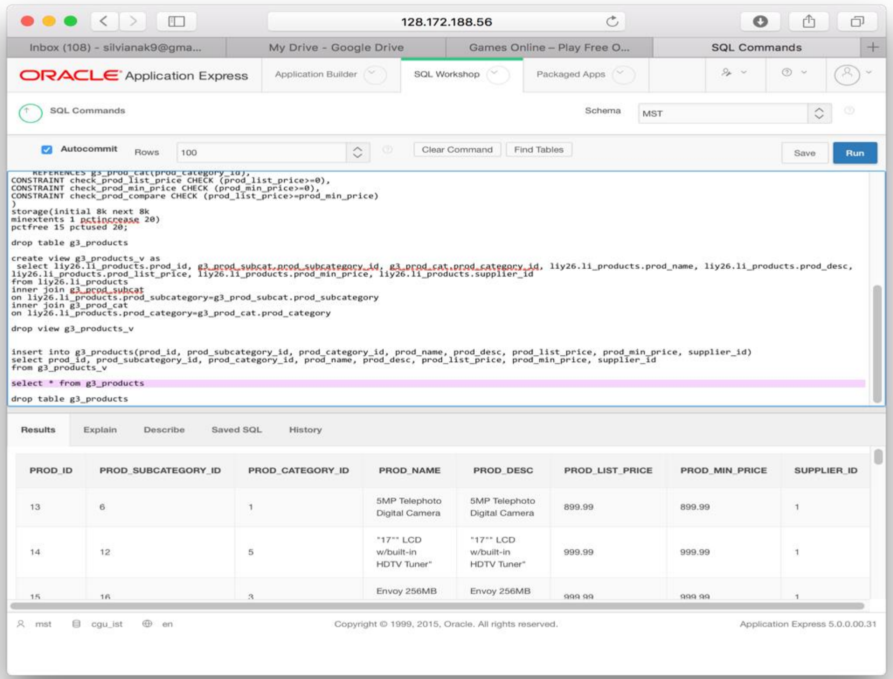
  On the G3_SALES table the insertion was done twice from 2 different tables in the liy26 schema, since they had nothing in common and did not violate any of the constraints implemented. The syntax was used in the exact same method for every table in our schema.
- Creating views was necessary for those 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;
```



Schema: MST

Rows: 100

```
    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;

drop table g3_products

create view g3_products_v as
 select liy26.li_products.prod_id, g3_prod_subcat.prod_subcategory_id, g3_prod_cat.prod_category_id, liy26.li_products.prod_name, liy26.li_products.prod_desc,
liy26.li_products.prod_list_price, liy26.li_products.prod_min_price, liy26.li_products.supplier_id
from liy26.li_products
inner join g3_prod_subcat
on liy26.li_products.prod_subcategory=g3_prod_subcat.prod_subcategory
inner join g3_prod_cat
on liy26.li_products.prod_category=g3_prod_cat.prod_category

drop view g3_products_v


insert into g3_products(prod_id, prod_subcategory_id, prod_category_id, prod_name, prod_desc, prod_list_price, prod_min_price, supplier_id)
select prod_id, prod_subcategory_id, prod_category_id, prod_name, prod_desc, prod_list_price, prod_min_price, supplier_id
from g3_products_v

select * from g3_products

drop table g3_products
```

PROD_ID	PROD_SUBCATEGORY_ID	PROD_CATEGORY_ID	PROD_NAME	PROD_DESC	PROD_LIST_PRICE	PROD_MIN_PRICE	SUPPLIER_ID
13	6	1	5MP Telephoto Digital Camera	5MP Telephoto Digital Camera	899.99	899.99	1
14	12	5	"17"" LCD w/built-in HDTV Tuner"	"17"" LCD w/built-in HDTV Tuner"	999.99	999.99	1
15	16	3	Envoy 256MB	Envoy 256MB	999.99	999.99	1

mst cgu_ist en

Copyright © 1999, 2015, Oracle. All rights reserved.

Application Express 5.0.0.00.31

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)	2. Special Index	3. Index Hint	4. 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.

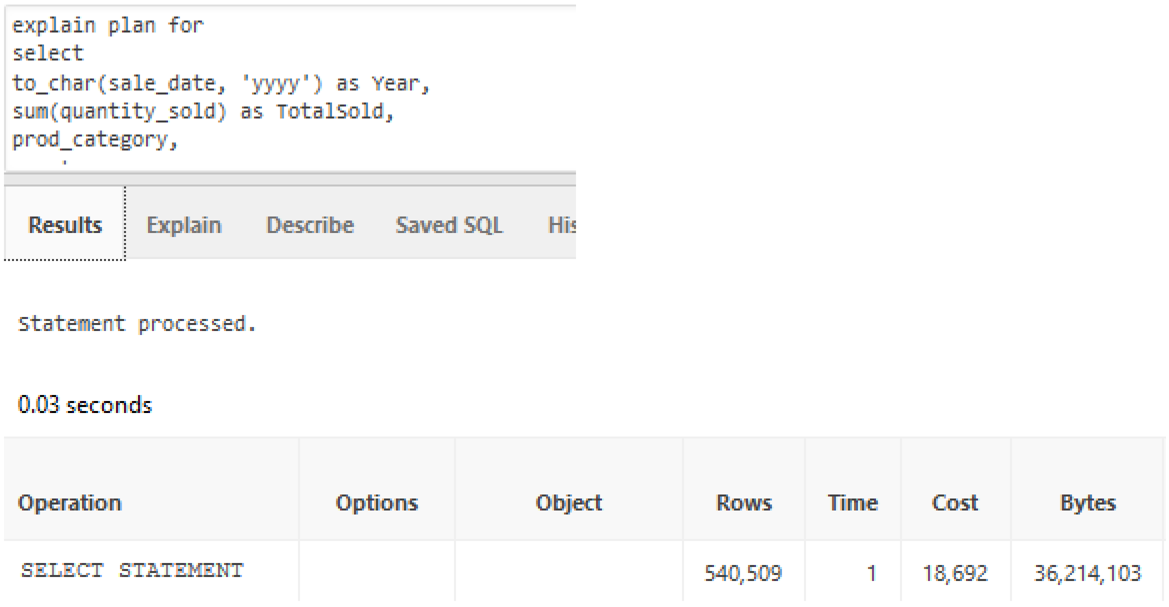
Finally, we decided to use materialized view on BV01 and BV07 as they have the highest initial cost. Bv01's materialized view helped with the cost significantly from 18692 to 4154. But, bv07 had an adverse effect in which the cost increased from 18692 to 18908.

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.

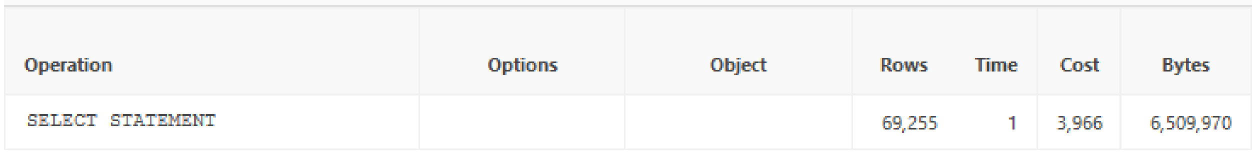

```
explain plan for
select
to_char(sale_date, 'yyyy') as Year,
sum(quantity_sold) as TotalSold,
prod_category,
```

```
Statement processed.
```

0.03 seconds

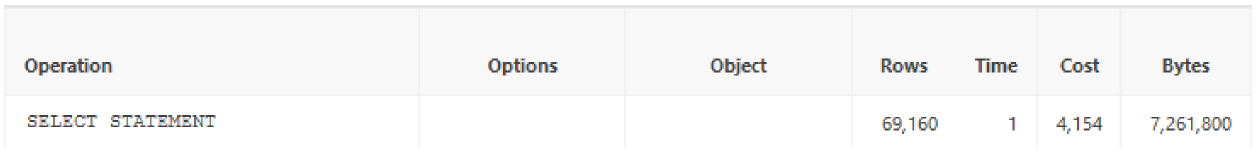
Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			540,509	1	18,692	36,214,103

Cost without any indexes: 18692



Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			69,255	1	3,966	6,509,970

Cost after indexes: 3966

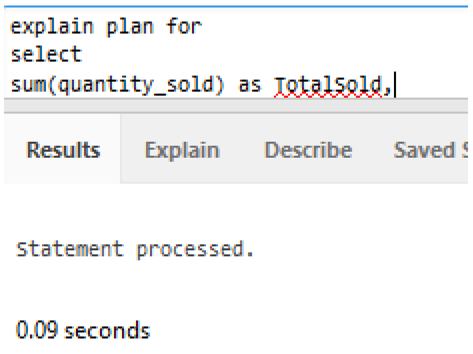


Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			69,160	1	4,154	7,261,800

Cost after materialized view: 4154

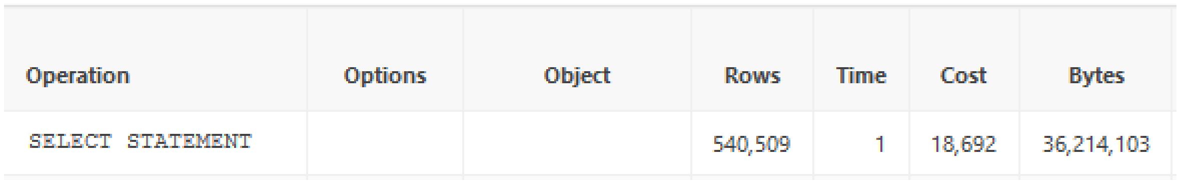
BV02:

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:



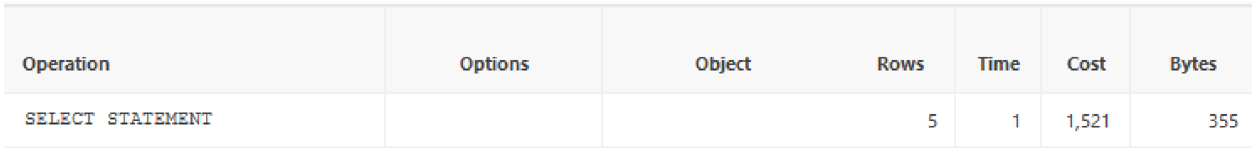
```
explain plan for
select
sum(quantity_sold) as TotalSold,
```

```
Statement processed.

0.09 seconds
```



Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			540,509	1	18,692	36,214,103

Cost without any created indexes: 18692



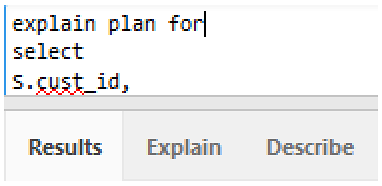
Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			5	1	1,521	355

Cost after bitmap indexes: 1521

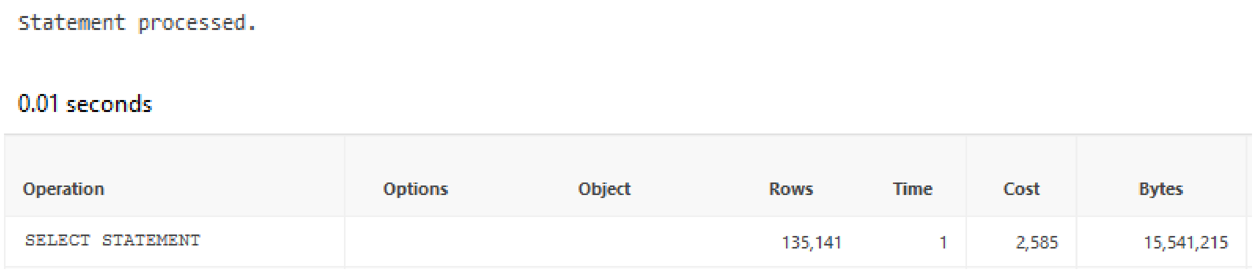
BV03:

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:



```
explain plan for
select
s.cust_id,
```

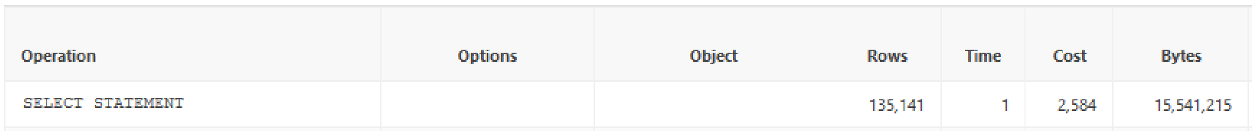


```
Statement processed.
```

0.01 seconds

Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			135,141	1	2,585	15,541,215

Cost without any indexes: 2585



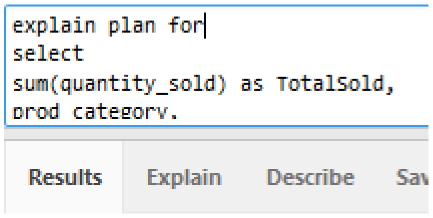
Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			135,141	1	2,584	15,541,215

Cost after bitmap indexes: 2584

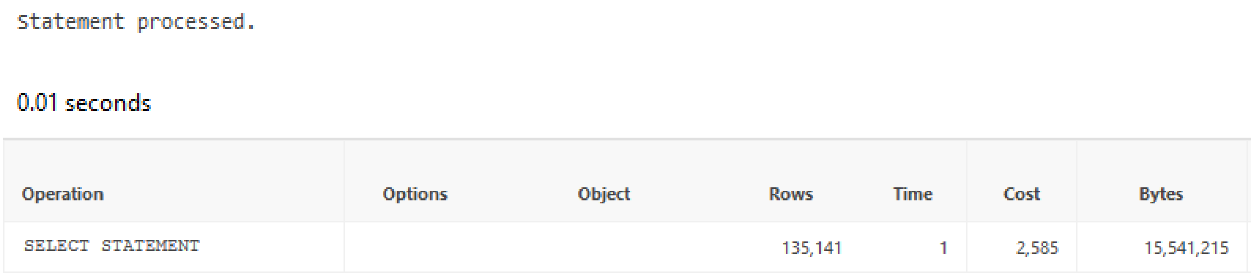
BV04:

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

Explain plan for



```
explain plan for
select
sum(quantity_sold) as TotalSold,
prod category,
```

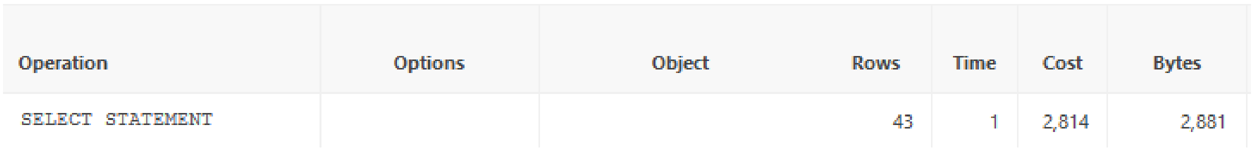


```
Statement processed.
```

0.01 seconds

Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			135,141	1	2,585	15,541,215

Cost without any indexes: 2585



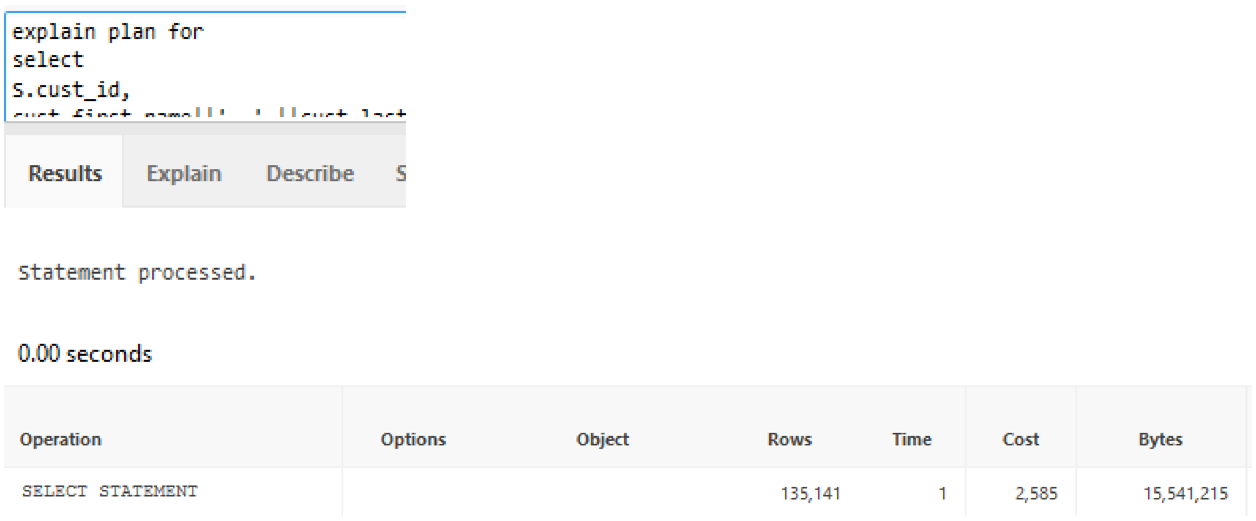
Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			43	1	2,814	2,881

Cost after index: 2814

BV05:

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

Explain plan for

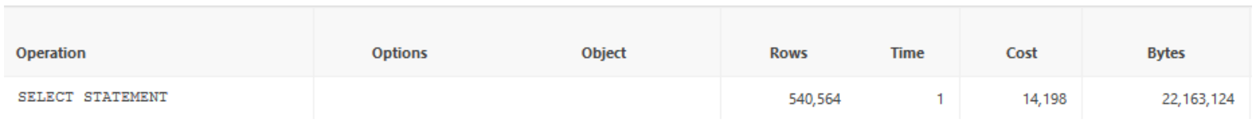


```
explain plan for
select
s.cust_id,
cust_first_name||' '||cust_last
```

```
Statement processed.

0.00 seconds
```

Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			135,141	1	2,585	15,541,215

Cost without any indexes: 2585



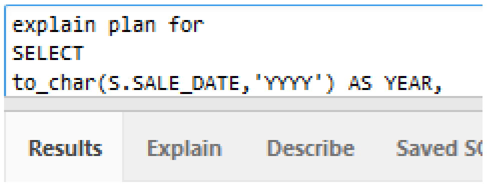
Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			540,564	1	14,198	22,163,124

Cost after indexes: 14198

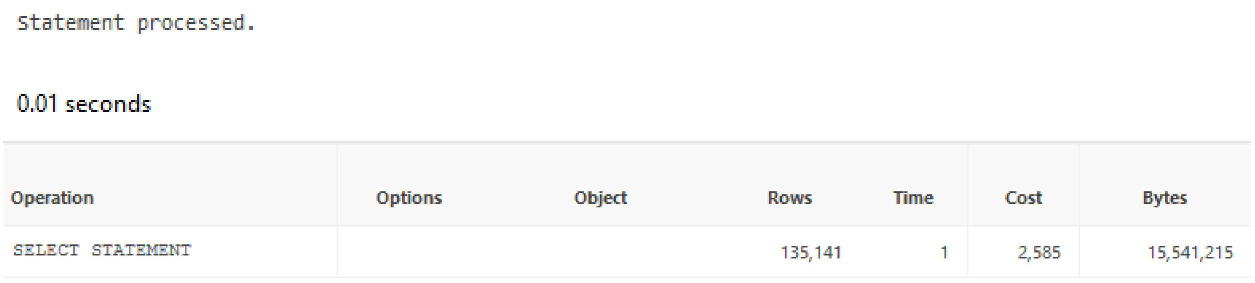
BV06:

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

Explain plan for



```
explain plan for
SELECT
to_char(S.SALE_DATE,'YYYY') AS YEAR,
```

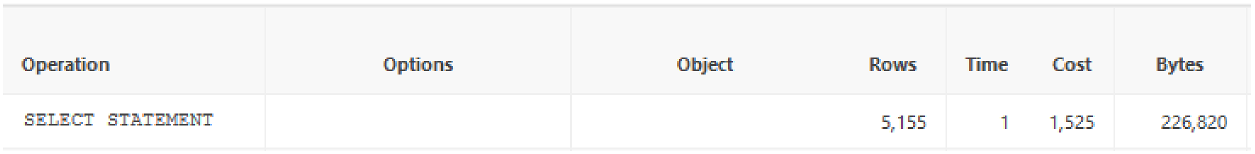


```
Statement processed.
```

0.01 seconds

Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			135,141	1	2,585	15,541,215

Cost without creating indexes: 2585



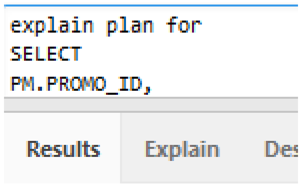
Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			5,155	1	1,525	226,820

Cost after indexes: 1525

BV07:

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

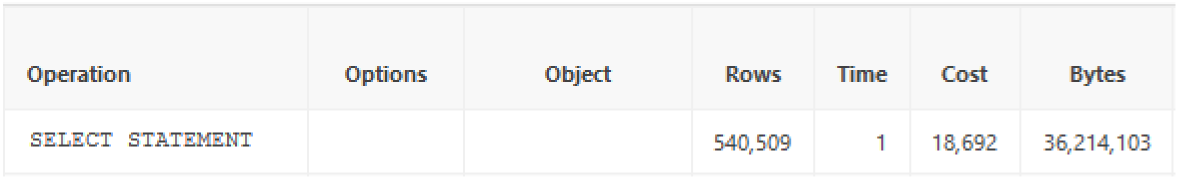
Explain plan for



```
explain plan for
SELECT
PM.PROMO_ID,
```

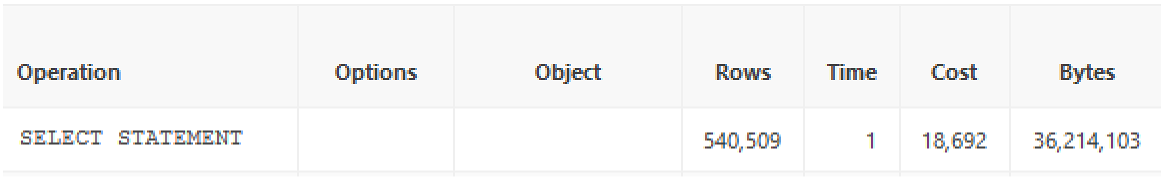
```
Statement processed.
```

0.01 seconds



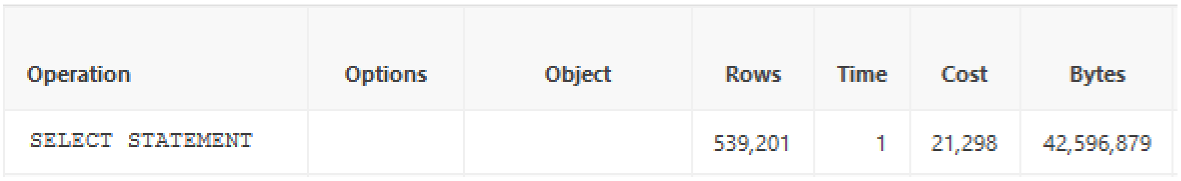
Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			540,509	1	18,692	36,214,103

Cost without any indexes: 18,692



Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			540,509	1	18,692	36,214,103

Cost after indexes: 18692



Operation	Options	Object	Rows	Time	Cost	Bytes
SELECT STATEMENT			539,201	1	21,298	42,596,879

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.

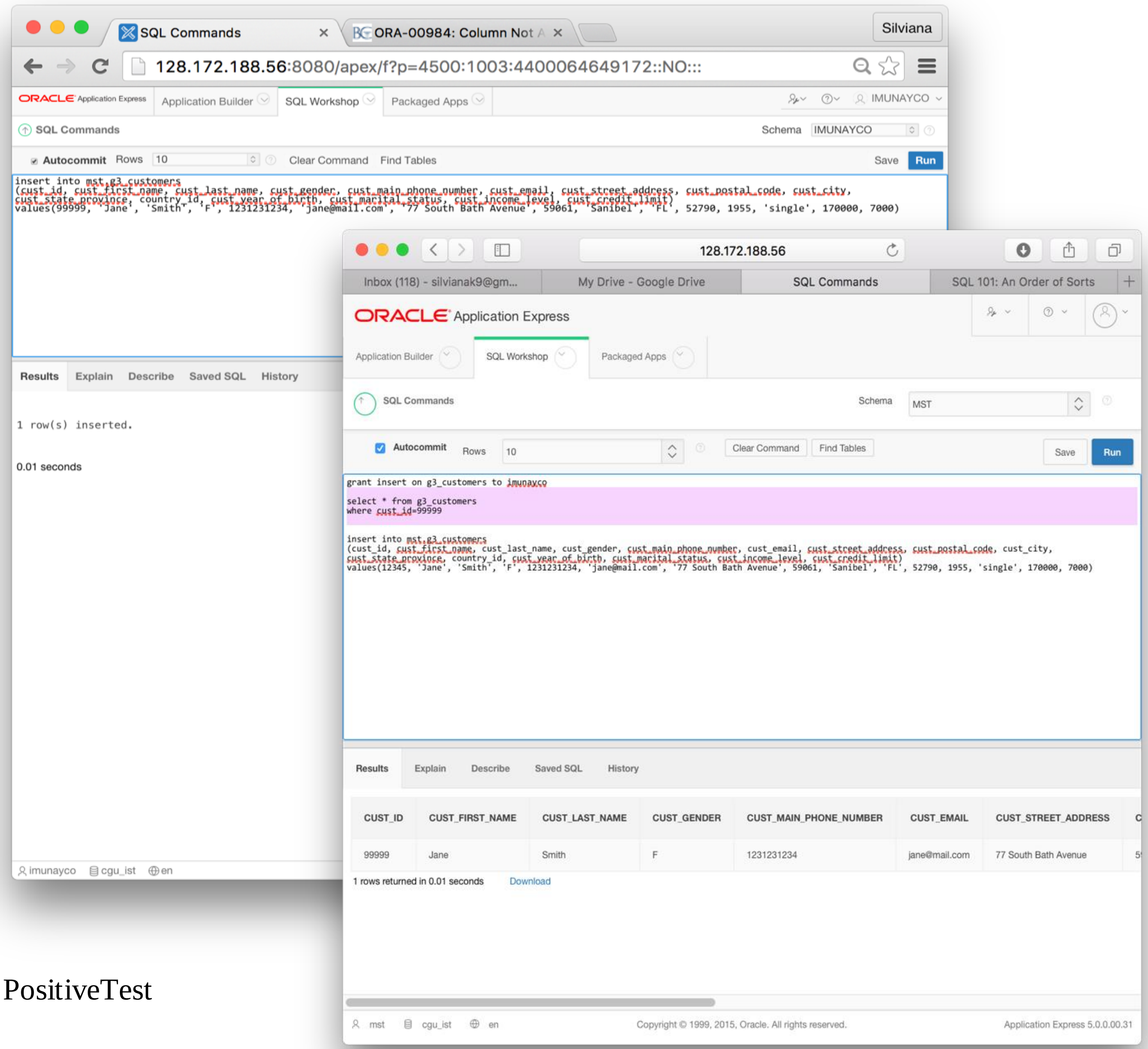
- **Positive Test** was done by selecting columns from one of tables, successfully.
- **Negative Test** was done by attempting to select columns from the table G3_COUNTRY which the user does not have access to, which of course was unsuccessful.

Business User 2 – Customer Relations Manager (imunayco)

<u>Insert privileges</u> 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.



```
insert into mst.g3_customers
(cust_id, cust_first_name, cust_last_name, cust_gender, cust_main_phone_number, cust_email, cust_street_address, cust_postal_code, cust_city,
cust_state_province, country_id, cust_year_of_birth, cust_marital_status, cust_income_level, cust_credit_limit)
values(99999, 'Jane', 'Smith', 'F', 1231231234, 'jane@mail.com', '77 South Bath Avenue', 59061, 'Sanibel', 'FL', 52790, 1955, 'single', 170000, 7000)
```

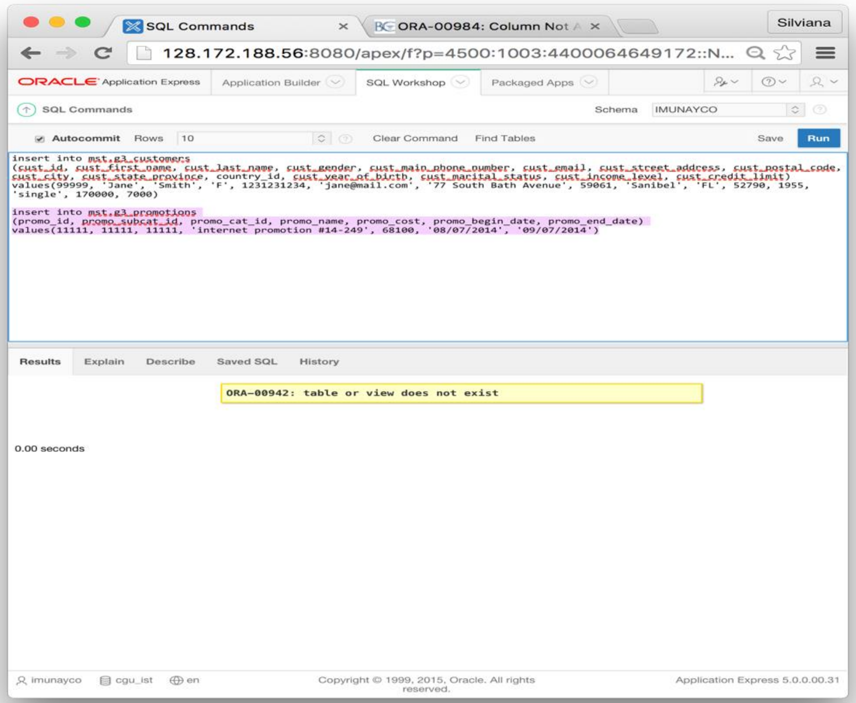
```
1 row(s) inserted.

0.01 seconds
```

```
grant insert on g3_customers to imunayco

select * from g3_customers
where cust_id=99999

insert into mst.g3_customers
(cust_id, cust_first_name, cust_last_name, cust_gender, cust_main_phone_number, cust_email, cust_street_address, cust_postal_code, cust_city,
cust_state_province, country_id, cust_year_of_birth, cust_marital_status, cust_income_level, cust_credit_limit)
values(12345, 'Jane', 'Smith', 'F', 1231231234, 'jane@mail.com', '77 South Bath Avenue', 59061, 'Sanibel', 'FL', 52790, 1955, 'single', 170000, 7000)
```

CUST_ID	CUST_FIRST_NAME	CUST_LAST_NAME	CUST_GENDER	CUST_MAIN_PHONE_NUMBER	CUST_EMAIL	CUST_STREET_ADDRESS
99999	Jane	Smith	F	1231231234	jane@mail.com	77 South Bath Avenue

1 rows returned in 0.01 seconds Download

PositiveTest



```
insert into mst.g3_customers
(cust_id, cust_first_name, cust_last_name, cust_gender, cust_main_phone_number, cust_email, cust_street_address, cust_postal_code,
cust_city, cust_state_province, country_id, cust_year_of_birth, cust_marital_status, cust_income_level, cust_credit_limit)
values(99999, 'Jane', 'Smith', 'F', 1231231234, 'jane@mail.com', '77 South Bath Avenue', 59061, 'Sanibel', 'FL', 52790, 1955,
'single', 170000, 7000)

insert into mst.g3_promotions
(promo_id, promo_subcat_id, promo_cat_id, promo_name, promo_cost, promo_begin_date, promo_end_date)
values(11111, 11111, 11111, 'internet promotion #14-249', 68100, '08/07/2014', '09/07/2014')
```

```
ORA-00942: table or view does not exist

0.00 seconds
```

Negative Test

<u>Update privileges</u> 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.

<u>Delete privileges</u> 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.

<u>Select privileges</u> 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.