CSE

4126

DISTRIBUTED DATABASE MANAGEMENT SYSTEM

PROJECT REPORT

GO Cart

SUBMITTED BY

AREFEEN SULTAN | 15-01-04-111

GROUP MEMBERS

TONMOY SINHA | 15-01-04-129
SAYED HOSSAIN KHAN | 15-01-04-133

Project Summary:

People don't have enough information and resources about products which are beneficial to them when shopping on local stores. Local shops in our country are not yet developed like the other countries. Our goal is to make shopping much more reliable and serviceable. We are using distributed database management system for super shop management.

1. Customer buys product, local store sells product, and therefore transaction happens.
2. Customer can be approved for membership.
3. Customer can search item for information and reliability.
4. Warehouse has product inventory.
5. Warehouse can distribute products to local store.
6. There will be monthly, weekly or daily sales report.
7. Customer can search in different stores for product availability.
8. Customer can see top selling products of specific category.

Platforms:

- Programming Language : PL/SQL
- IDE : Oracle 10g

Entity Relationship Diagram (ERD):

Entity Sets:

- Customer
- Branch
- Membership
- Product
- Transaction
- Warehouse
- Category

Entity and Table Set Name with Attributes and data types

Attributes of Branch:

- Branch_ID(primary key) - int
- Location - varchar(11)
- Phone - int

Attributes of Category:

- C_ID(primary key) – int
- C_Name – varchar(20)

Attributes of Customer:

- Cust_ID(primary key) - int
- M_ID(foreign key) - int
- C_Name - varchar(50)
- Email - varchar(50)

Attributes of Membership:

- M_ID(primary key) - int
- P_Range_From - number
- P_Range_To - number
- Discount_Rate – number
- Type - varchar(50)

Attributes of Product:

- P_ID(primary key) - int
- C_ID(foreign key)- int
- Selling_Price – number
- P_Name – varchar(50)

Attributes of Transaction:

- T_ID(primary key) - int
- Cust_ID(foreign key) – int
- Branch_ID(primary key) – int
- Total_Price – number
- Date – date

Attributes of Warehouse:

- S_ID(primary key) - int
- S_Date - date
- P_ID(foreign key) – int
- Buying_Price - number
- P_Quantity – int

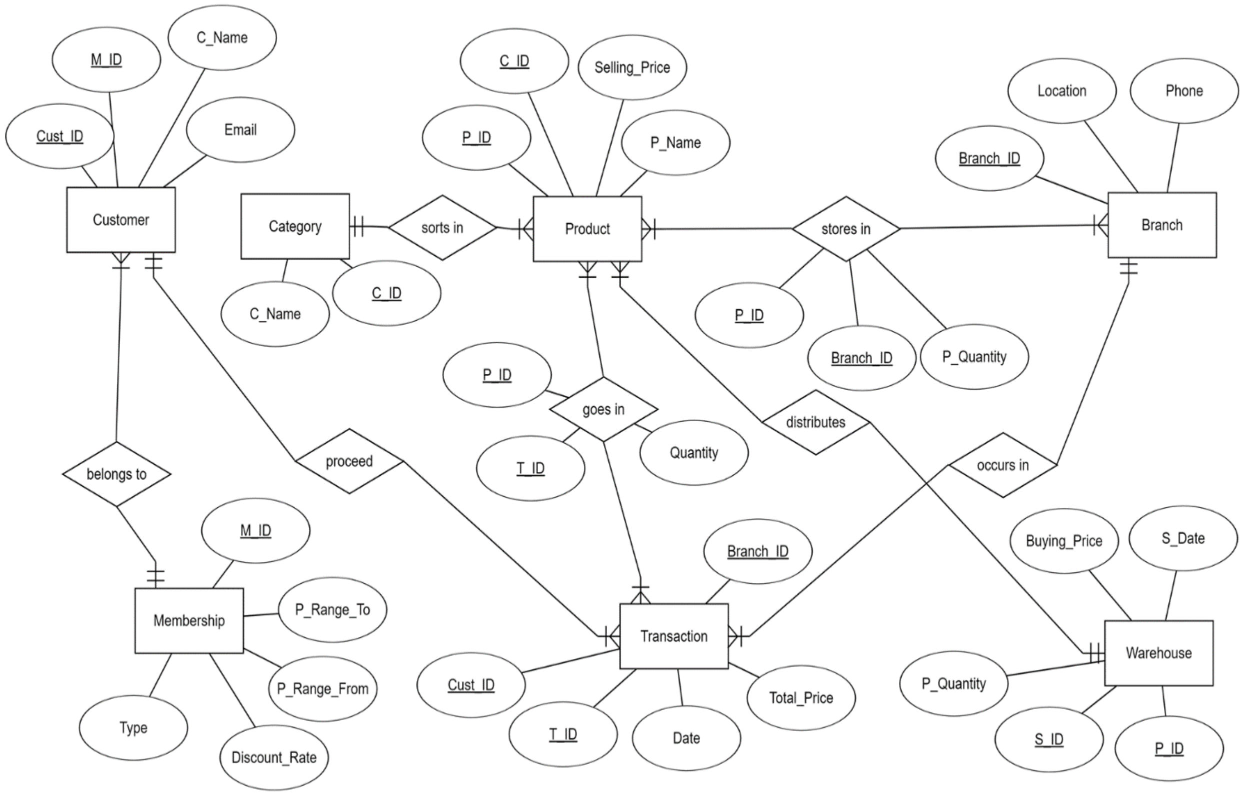
Attributes of goes in:

- P_ID(foreign key) - int
- T_ID(foreign key) – int
- Quantity – int

Attributes of stores in:

- P_ID(foreign key) - int
- Branch_ID(foreign key) – int
- P_Quantity – int

Entity Relationship Diagram (ERD):



Relational Model with Fragmentation Schema:

Global Schema:

- Branch(branch_ID, Location, Phone)
- Category(C_ID, C_Name)
- Membership(M_ID, P_Range_From, P_Range_To, Discount_Rate, Type)
- Product(P_ID, Selling_Price, P_Name, C_ID)
- Customer(C_ID, C_Name, Email, M_ID)
- Transaction(T_ID, Cust_ID, Branch_ID, Total_Price, T_Date)
- goes_in(P_ID, T_ID, Quantity)
- occurs_in(Branch_ID, T_ID)
- proceed(T_ID,C_ID)
- stores_in(P_ID, P_Quantity, Branch_ID)
- warehouse(S_ID, S_Date, P_ID, P_Quantity, Buying_Price)
- Sales(Serial_no, Month, Sales)

Fragmentation Schema with allocated site:

- $\text{Branch}_1 = \text{PJ}_{\text{branch\_ID,Location,Phone}}\text{SL}_{\text{Location=’Dhanmondi’}}\text{Branch}$
- $\text{Branch}_2 = \text{PJ}_{\text{branch\_ID,Location,Phone}}\text{SL}_{\text{Location=’Mohammadpur’}}\text{Branch}$
- $\text{Stores\_in}_1 = \text{PJ}_{\text{P\_sID, P\_Quantity, Branch\_ID}}\text{SL}_{\text{Branch\_ID = 1}}$ stores_in
- $\text{Stores\_in}_2 = \text{PJ}_{\text{P\_ID, P\_Quantity, Branch\_ID}}\text{SL}_{\text{Branch\_ID = 2}}$ stores_in
- $\text{Transaction}_1 = \text{PJ}_{\text{T\_ID,Cust\_ID,Branch\_ID,Total\_Price,T\_Date}}\text{SL}_{\text{Branch\_ID = 1}}$ Transaction
- $\text{Transaction}_2 = \text{PJ}_{\text{T\_ID,Cust\_ID,Branch\_ID,Total\_Price,T\_Date}}\text{SL}_{\text{Branch\_ID = 2}}$ Transaction
- Branch_1 @ site1, Stores_in_1 @ site1, Transaction_1 @ site1
- Branch_2 @ site2, Stores_in_2 @ site2, Transaction_2 @ site2

Database Profile:

For relation Branch (branch_ID, Location, Phone) with fragments Branch_1, Branch_2 the database profile contains the following information:

$\underline{\text{Branch}_1}$

$\text{card}(\text{Branch}_1) = 1$

$\text{site}(\text{Branch}_1) = 1$

	Branch_ID	Location	Phone
Size	16 bits	11 bits	16 bits
Val	1	1	1

$\underline{\text{Branch}_2}$

$\text{card}(\text{Branch}_2) = 1$

$\text{site}(\text{Branch}_2) = 2$

	Branch_ID	Location	Phone
Size	16 bits	11 bits	16 bits
Val	1	1	1

For relation Stores_in(P_ID, P_Quantity, Branch_ID) with fragments Stores_in_1, Stores_in_2 the database profile contains the following information:

Stores_in$_1$

card(Stores_in$_1$): 1

site(Stores_in$_1$): 1

	P_ID	P_Quantity	Branch_ID
Size	16 bits	16 bits	16 bits
Val	1	1	1

Stores_in$_2$

card(Stores_in$_2$): 2

site(Stores_in$_2$): 2

	P_ID	P_Quantity	Branch_ID
Size	16 bits	16 bits	16 bits
Val	2	2	1

For relation Transaction(T_ID, Cust_ID, Branch_ID, Total_Price, T_Date) with fragments Transaction$_1$, Transaction$_2$ the database profile contains the following information:

Transaction$_1$

card(Transaction$_1$): 2

site(Transaction$_1$): 1

	T_ID	Cust_ID	Branch_ID	Total_Price	T_Date
size	16 bits	16 bits	16 bits	21 bytes	7 bytes
val	2	2	1	2	2

Transaction$_2$

card(Transaction$_2$): 2

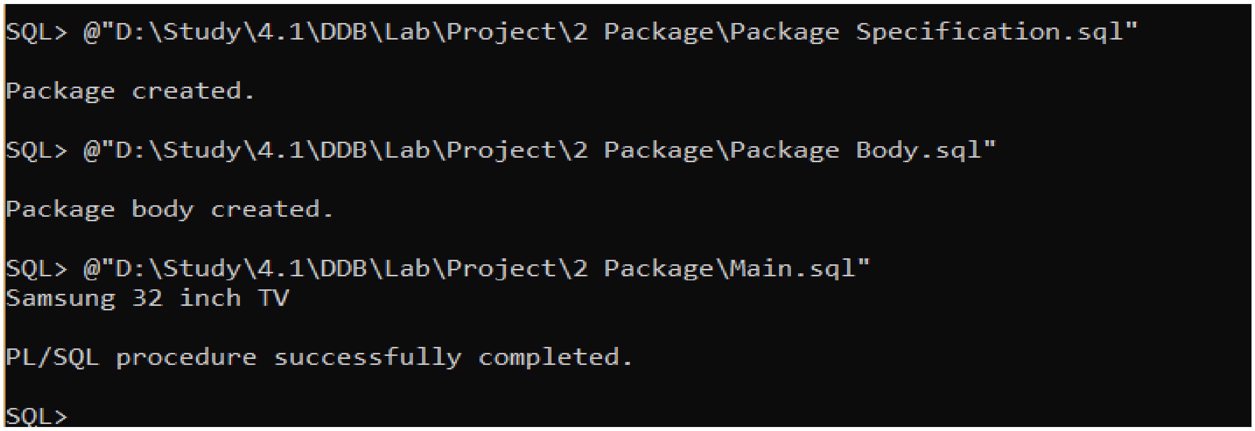
site(Transaction$_2$): 2

	T_ID	Cust_ID	Branch_ID	Total_Price	T_Date
size	16 bits	16 bits	16 bits	21 bytes	7 bytes

val	2	2	1	2	2

Functions or Procedures: All the functions and procedures are implemented in package.

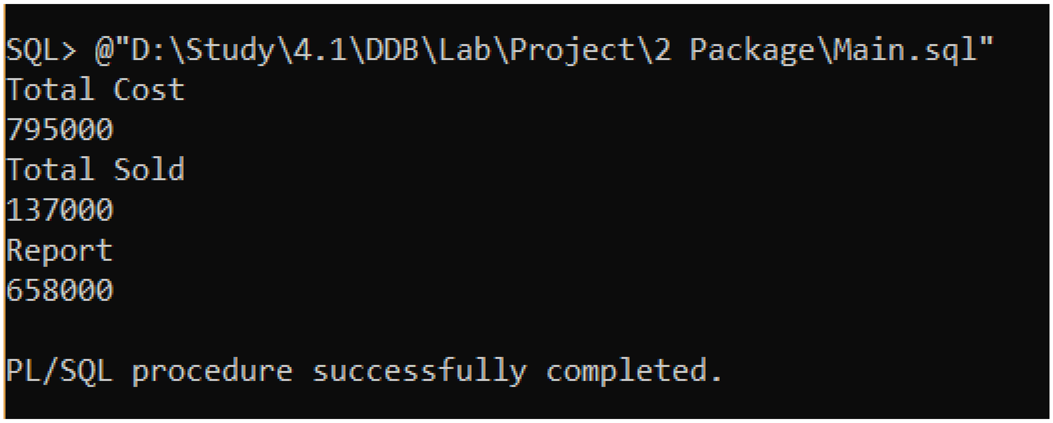
1. **Top Selling Product:** We can see the top selling products of a week, a month or a year.



```
SQL> @"D:\Study\4.1\DDB\Lab\Project\2 Package\Package Specification.sql"

Package created.

SQL> @"D:\Study\4.1\DDB\Lab\Project\2 Package\Package Body.sql"

Package body created.

SQL> @"D:\Study\4.1\DDB\Lab\Project\2 Package\Main.sql"
Samsung 32 inch TV

PL/SQL procedure successfully completed.

SQL>
```

Fig: Output of Top Selling procedure

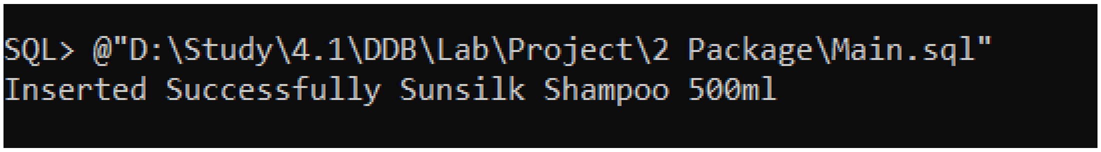
2. **Sales Report:** Sales report will be produced based on buy and sales.



```
SQL> @"D:\Study\4.1\DDB\Lab\Project\2 Package\Main.sql"
Total Cost
795000
Total Sold
137000
Report
658000

PL/SQL procedure successfully completed.
```

Fig: Output of Sales from 1 January to 1 February 2018

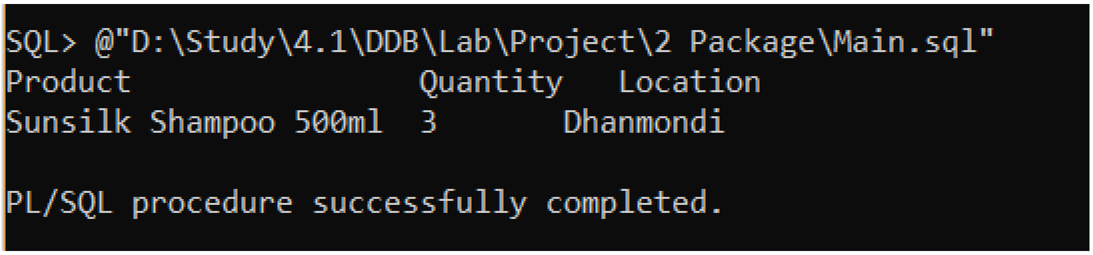
3. **Warehouse Distribution:** Distribution of data of the products on each branch will be saved.



```
SQL> @"D:\Study\4.1\DDB\Lab\Project\2 Package\Main.sql"
Inserted Successfully Sunsilk Shampoo 500ml
```

Fig: Output of Sunsilk Shampoo 500ml being distributed to branch$_2$ @site2

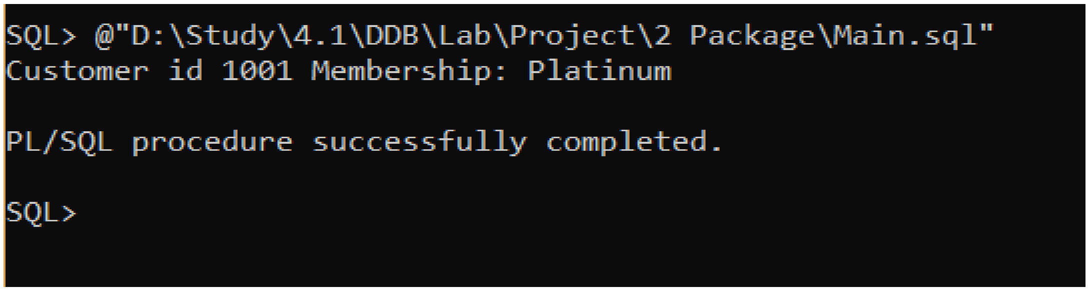
4. **Product availability:** Product availability of nearby branches will be informed to the customer.



```
SQL> @"D:\Study\4.1\DDB\Lab\Project\2 Package\Main.sql"
Product                Quantity   Location
Sunsilk Shampoo 500ml  3       Dhanmondi

PL/SQL procedure successfully completed.
```

Fig: Output of Product’s location branch

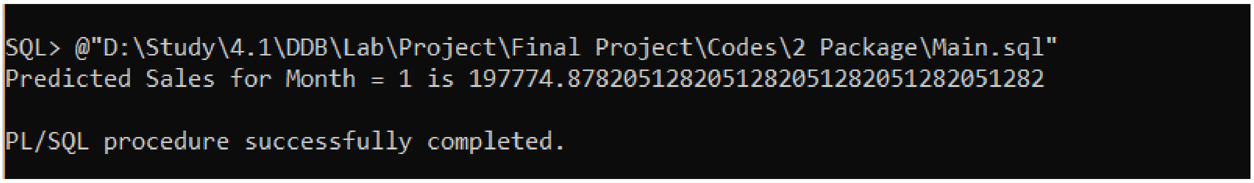
5. **Membership:** Customer will get discount based on membership rank.



```
SQL> @"D:\Study\4.1\DDB\Lab\Project\2 Package\Main.sql"
Customer id 1001 Membership: Platinum

PL/SQL procedure successfully completed.

SQL>
```

Fig: Output of customer 1001’s membership type

6. **Sales Prediction:** Manager can see the sales forecasting of any month. [Note: Details of the implementation are at the last page of this report]



```
SQL> @"D:\Study\4.1\DDB\Lab\Project\Final Project\Codes\2 Package\Main.sql"
Predicted Sales for Month = 1 is 197774.8782051282051282051282051282051282

PL/SQL procedure successfully completed.
```

Fig: Output of forecasted sale of month, 1 = January

Effect of Update: From our query of updating we can show that,

Update Branch_ID = 2 where product ID = 10001:

- 1st step:

Stores_in$_1$

Product ID	Product Quantity	Branch ID
10001	3	1

Stores_in$_2$

Product ID	Product Quantity	Branch ID
10002	1	2
10003	2	2

- 2nd Step:

Deletes 10001 product id from Stores_in$_1$

Stores_in$_1$

Product ID	Product Quantity	Branch ID
10001	3	1

Stores_in$_2$

Product ID	Product Quantity	Branch ID
10002	1	2
10003	2	2

- 3rd Step:

Insert into Stores_in$_2$ @ site 2:

Stores_in$_1$

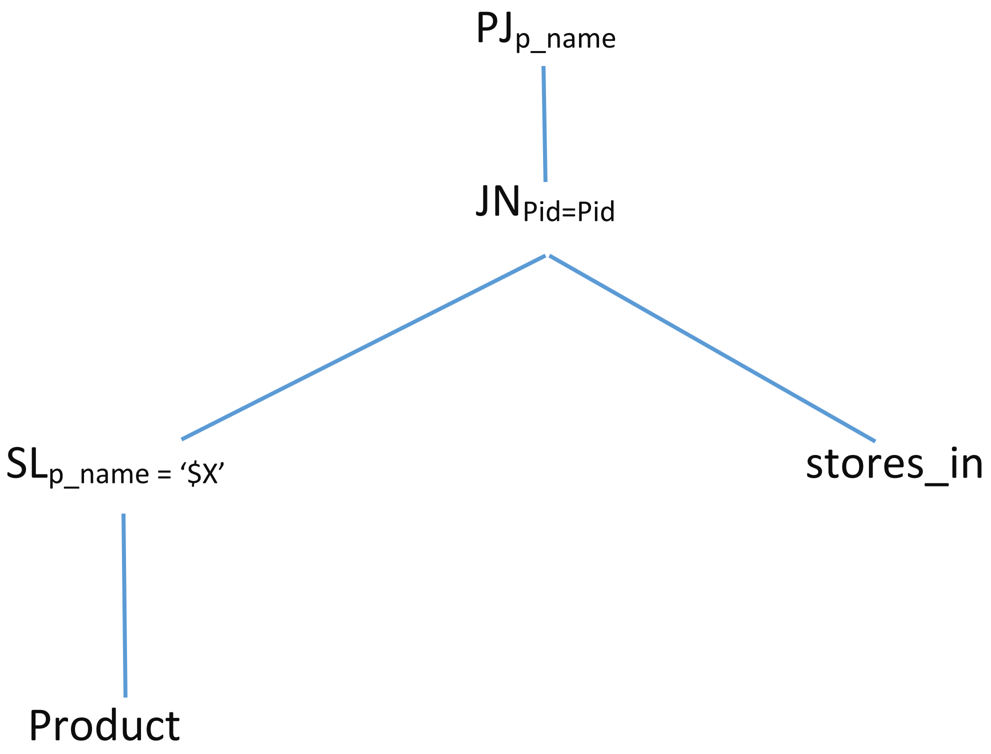
Product ID	Product Quantity	Branch ID

Stores_in$_2$

Product ID	Product Quantity	Branch ID
10002	1	2
10003	2	2
10001	3	2

<u>Operator Tree on Join:</u>

Query: PJ$_{p\_name}$(SL$_{p\_name=’\$X’}$ Product JN$_{Pid=Pid}$ stores_in)



⇨ Using Canonical Expression,

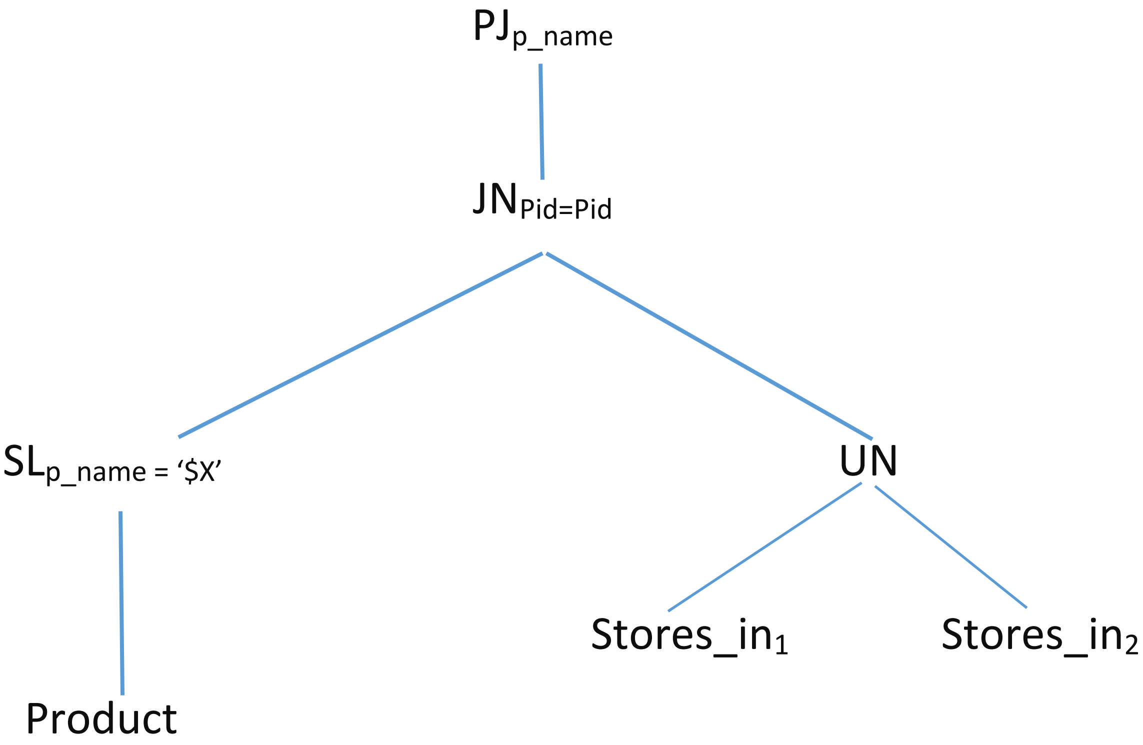
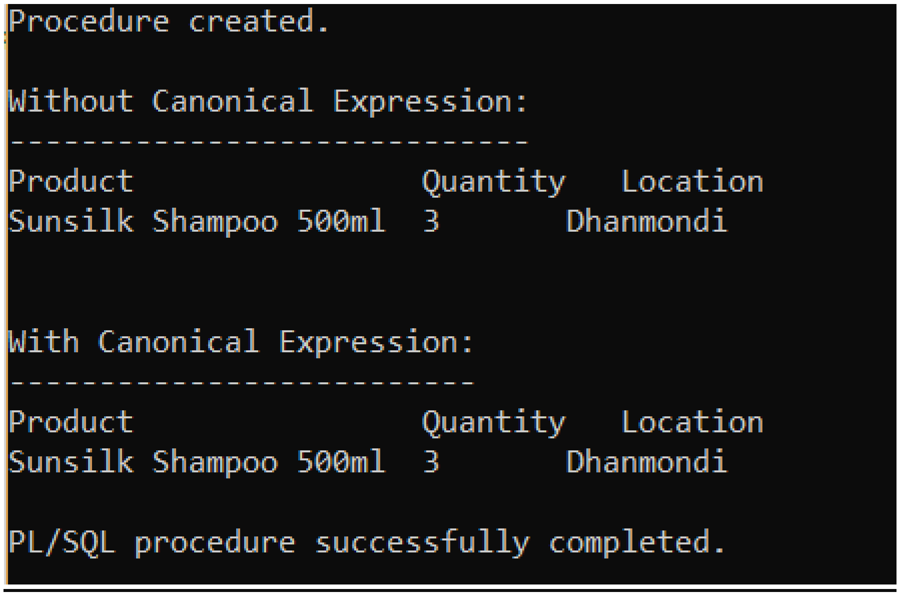


Fig: Operator Tree

We can see that the output without canonical expression and with canonical expression remains constant.



```
Procedure created.

Without Canonical Expression:
-----------------------------
Product                 Quantity   Location
Sunsilk Shampoo 500ml   3       Dhanmondi


With Canonical Expression:
--------------------------
Product                 Quantity   Location
Sunsilk Shampoo 500ml   3       Dhanmondi

PL/SQL procedure successfully completed.
```

Fig: Output for operator tree (With and without canonical expression)

Qualified Relation: From procedure product search we can write,

[Stores_in$_1$: Branch_ID = 1] JN$_{Branch\_ID=Branch\_ID}$ [Branch$_1$: Branch_ID = 1]

Applying Rule 6,

⇨ [Stores_in$_1$ JN$_{Branch\_ID=Branch\_ID}$ Branch$_1$: Branch_ID = 1 and Branch_ID=Branch_ID]

Stores_in$_1$

P_ID	P_Quantity	Branch_ID
10001	3	1

Branch$_1$

Branch_ID	Location	Phone
1	Dhanmondi	0168552341

After Joining

P_ID	P_Quantity	Branch_ID	Location	Phone
10001	3	1	Dhanmondi	0168552341

Semi-join Program:

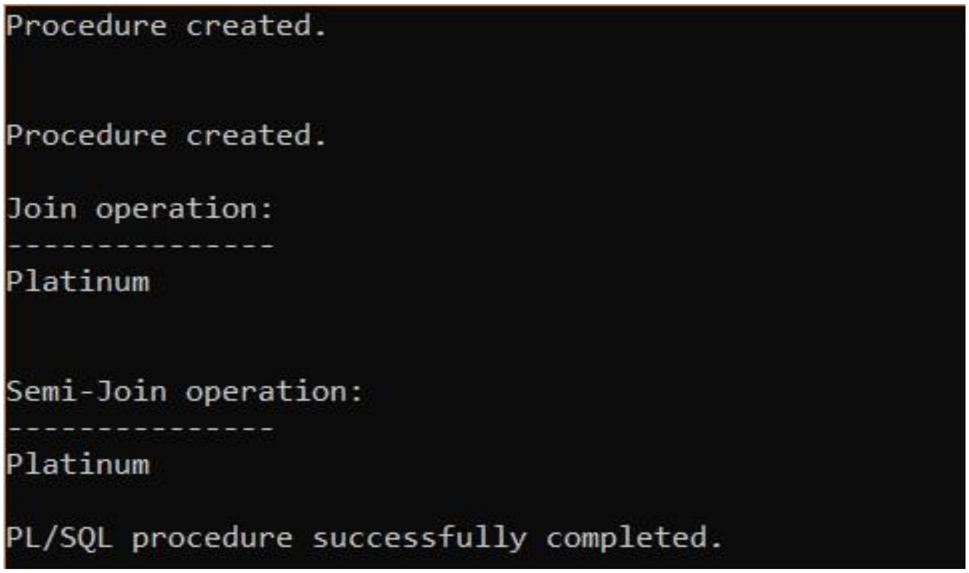
In our query with join was,

PJ$_{Type}$ SL$_{C\_ID='\$x'}$ (Membership JN$_{M\_ID = M\_ID}$ Customer).

So for the equivalence, semi-join program would be:

⇨ PJ$_{Type}$ SL$_{C\_ID='\$x'}$ ((Membership SJ$_{M\_ID = M\_ID}$ PJ$_{M\_ID}$ Customer) JN$_{M\_ID=M\_ID}$ Customer).

We have ensured that the semi-join program outputs the same result as join program.



```
Procedure created.


Procedure created.

Join operation:
---------------
Platinum


Semi-Join operation:
---------------
Platinum

PL/SQL procedure successfully completed.
```

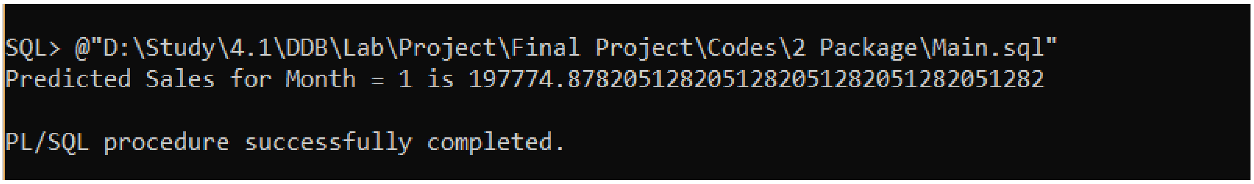
Fig: Output for Join and Semi-join Program

Machine Learning Technique:

In procedure 6, we used a machine learning technique called linear regression to predict the sales outcome for any month.

Our dataset includes 2 columns (Month = x, Sales = y). So for month, x = 1, 2…12 we inserted sales value y.

Then we calculated values by using the regression formula for a and b. Then for any month, x = 1 = January we calculated y = a + bx and found the predicted score.



```
SQL> @"D:\Study\4.1\DDB\Lab\Project\Final Project\Codes\2 Package\Main.sql"
Predicted Sales for Month = 1 is 197774.8782051282051282051282051282051282

PL/SQL procedure successfully completed.
```

Fig: Output of forecasted sale of month 1 = January

Contribution:

As a member of my team my contribution to this project was:

1. Implementing some of the procedures of our project.
2. Organizing all the procedures in package.
3. Implementing the site1 and site2 database and inserting values into them.
4. Showing database profiles in the database.
5. Implementing effect of update, operator tree and qualified relation on fragmentation query.
6. Implementing the machine learning technique linear regression for sales prediction.