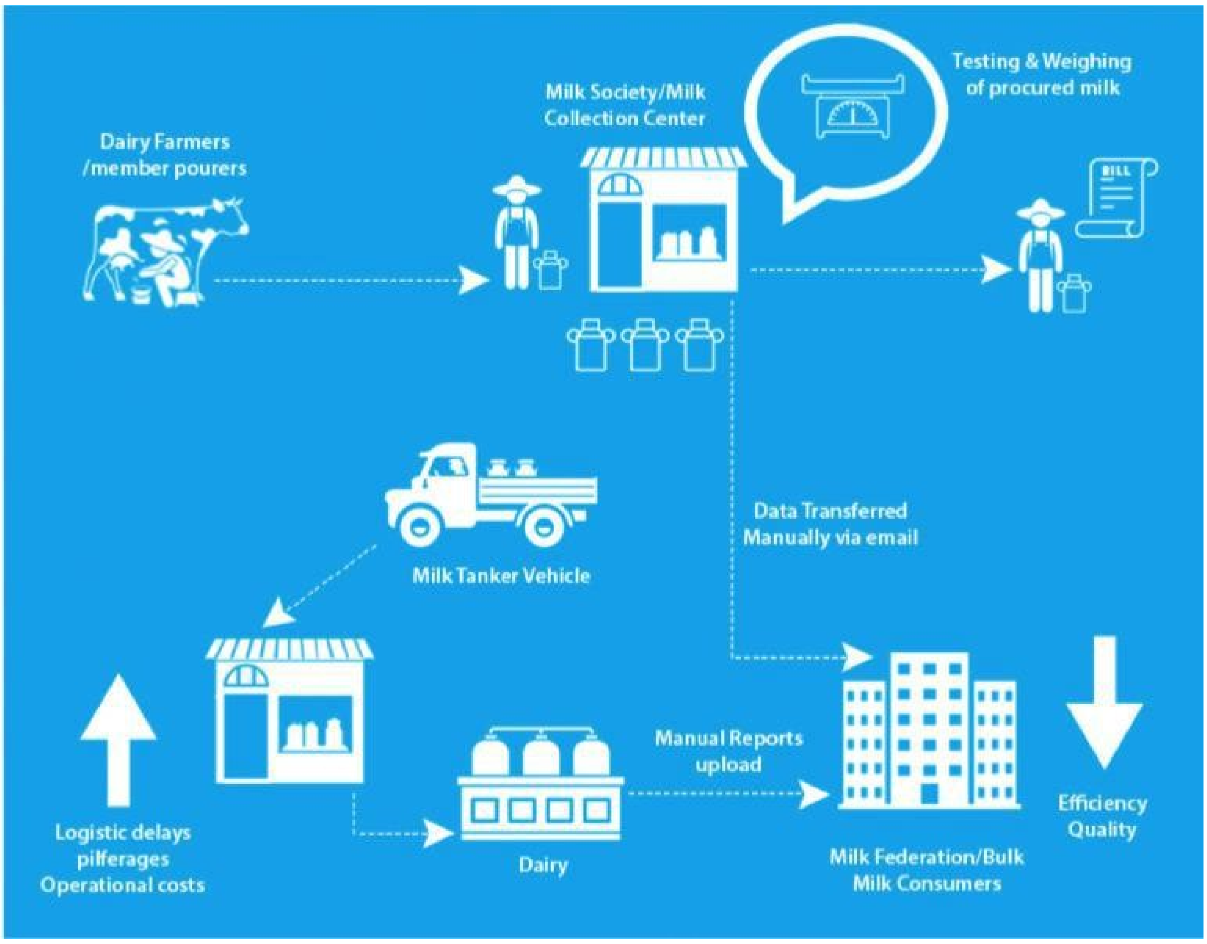
A PROJECT REPORT
FOR

DAIRY MANAGEMENT SYSTEM


Dairy Farmers /member pourers
Milk Society/Milk Collection Center
Testing & Weighing of procured milk
BILL
Milk Tanker Vehicle
Data Transferred Manually via email
Logistic delays pilferages Operational costs
Dairy
Manual Reports upload
Milk Federation/Bulk Milk Consumers
Efficiency Quality

Done By:
Nithin Kumar M R

Abstract:

Through this project we wish to unify the data of a Milk Cooperative and make their commercial processes easier and efficient by implementing a software using Oracle SQL Relational Database.

The existing technology in rural area is to manually keep track of all the details of the farmers in the cooperative and also the details of collection and transportation of milk.

This technology is obsolete and prone to errors and also has huge security drawbacks

The new proposed system using a database solves the existing challenges

1. Database enforces standards
2. Database provides security by restricting unauthorised unlawful access
3. Database removes redundancies
4. Integrity constraints associated with the databases ensure that the data entered is consistent

Existing System:

When we Analysis the Manage about this firm then we face that they working with manual. And we all know that the manual system has many disadvantages. Some are mentioned below…

– The manual system requires more time for processing.

– It requires more critical work.

– The manual system is more error prone.

– Difficult to maintain.

– Manual system is costly.

– Immediate response to the queries is difficult and time consuming.

– More men power needed.

Need for new system:

New system is required because of some advantage of new system are as below…

– The new system required less time for completion of any work.

– New system is decreasing the chances of error.

– New system should work smoothly and very fast.

– New system saving time and manpower.

– The system is user friendly and any one having computer knowledge can handle it easily.

– Suitability for computerized data entry. Maintaining
Dairy information,
Distributors information, Products Information.

Functional component:

- To facilitate easily maintenance of records of Products produceed each day
- Maintain Daily inventory reports
- Quick access of all records.
- Reduce manual work
- Prevent and reduce human error

Scope of diary management system:

This application is built such a way that it should that it makes the collection of milk from dairy farmers in rural districts efficient and the transport of milk to processing centre and subsequently to the distributors and the retailers in urban places efficiently.

There are database users at each level who are provided with different views of the database and given different level of authorisation and access.

The interface of the application is intuitive and easy to understand for naive users and people with only basic knowledge of internet

Data Requirements :-

Dairy Farmers:

1)F_ID is the primary key used to uniquely identify each dairy farmer in the cooperative

2) F_Name is the name of the dairy farmer

3)F_Adress is the address of the farmer's cattle farm

4)No_of cows is the number of cows owned by dairy farmer

5)F_phone is the contact number of each farmer

Constraint

1)One farmer can only produce one batch of milk in a day

2)The F_ID of a farmer is unique and is used to identify each farmer in the cooperative

3)F_phone of each farmer must be distinct

4)F_phone should be exactly 10 digits (Domain Constraint)

5)The minimum number of cows should be two

6)Referential Integrity Constraint should be valid

7)F_Id should be exactly 5 digits (Check constraint)

Milk Batch:

1)B_id is the unique ID assigned to each batch of milk produced by a farmer

2)F_ID is the unique ID of the farmer who produced that bath of milk

3)Density is used to assign the density value of the batch of milk and used as a measure of quality

4)Milk_Quantity is the quantity of milk produced in that batch

5)F_price is the price paid to the farmer for that batch of milk and it is derived from the quantity

Constraint

1)The density should be above a threshold value or the batch of milk is rejected because it fails to meet structural standards

2)The batch ID of each batch should be unique

3) Each batch should be associated with a unique farmer id

4)Referential Integrity Constraint should be valid

5) The quantity of milk in a batch must be above a minimum threshold value.

Retailers:

1)R_Id is unique

2)Distributor_Id is a unique ID associated with each retailer

3)Store_name is the name of the retail shop

4)Store_Address is the location of the retail shop

5)R_phone is the phone number of each retail shop

6)Type is the type of products bought from distributor

7)Quantity is the amount of each type of product bought from distributor

Constraints

1)R_Id should be unique (Primary Key constraint)

2)Distributor_ID should be unique (Referential Constraint)

3)Store_Phone number should be distinct

4)Store_phone number should be exactly 10 digits (Check Constraint)

5)Store_address must be within Vellore District (Check Constraint)

6) There should be at least one type of product that should be listed (Check Constraint)

7)The quantity of each type should be a minimum amount.
(Check Constraint)

8)If type of product is not mentioned then default value is null

Collection Centre:

1)Every milk collection centre will be uniquely identified by an ID number.
2)C_location gives the information regarding the location of the collection centre. This helps us to efficiently transport the milk products to the nearest location possible.
3)CTotal_capacity gives us the total capacity of the collection centre.
4)B_ID is the batch ID of the tuples in the collection centre.

Constraints

1. The milk being bought to a particular collection centre should not exceed it's storage limit.
2.ProcessC_ID must not be null and should must be unique.

Products:

1)Product_ID is used to Identify each product uniquely.
2)Product_Quantity allows us to maintain the products of similar types. i.e It gives us the quantity of the products of similar types.
3)Product_type is the type of product being produced. Example: cheese, milk, milk powders)
4)ProcessC_id is the refers to the Processing_centre where the product came from .
5)Date element is used to know the manufactured date of the product

Constraints

1)Product_id cannot be NULL and must be unique.

Processing centre:

1) ProcessC_Id maintains the identification number of all processing centre under the corporate milk producing company

2)C_Id stores the identification number of the particular collection centre which sending the batches of milk to the particular processing centre.

3) PTotal_capacity has the value of how much quantity of milk the particular processing centre can handle at a time.

Constraints

1) ProcessC_Id is unique (primary key constraint)

2) C_Id is distinct (foreign key constraint)

3) T_Id should have been made of strictly five digits (foreign key constraint)

4) PTotal_capacity should be at the maximum of 1 lakh litres

Distributors:

1) Distributors_Id has the value of the identification number of a particular distributor.

2) Product_Id has the value of the identification numbers of the range of products which is being sold by the the particular distributor.

Constraints

Distributors_Id is unique to each distributor (primary key constraint)

1) Product_Id is multi-valued and should strictly have 5 digits (foreign key constraint)

USERS OF DATABASE

COLLECTION CENTRE SUPERVISOR

ID : This is a unique ID number given to the supervisor

Name : This contains the name of the supervisor

Phone number : This number is used to contact the supervisor

The role of this supervisor is to maintain records of all the milk batches and farmers, deposited at his/her collection centre.

PROCESSING CENTRE SUPERVISOR

ID : This is a unique ID number given to the supervisor

Name : This contains the name of the supervisor

Phone number : This number is used to contact the supervisor

The role of this supervisor is to maintain the records of the milk coming in from collection centre and the end products being shipped to different distributors

DISTRIBUTION SUPERVISOR

ID : This is a unique ID number given to the supervisor

Name : This contains the name of the supervisor

Phone number : This number is used to contact the supervisor

The role of this supervisor is to maintain sales record of all the distributors as well as the retail stores who are currently in contract with the distributors

Functional Requirements:

1) Removal of Old Data

a) The Farmer should be deleted if he is no longer producing any Milk batches

b) The collection centre should be removed if it hasn't been functioning for a long time:

c) The Processing centre should be deleted if it is no longer producing any processed milk

d) The Processing supervisor should be removed if he/she is not in job anymore:

2) Modification Of existing data

a) The Farmer should be able to update his/her phone no and address as required:

b) The processing centre admin should be able to update the centre's total capacity required:

c) The Price of the milk batch can be increased due to supply demand:

d) A new supervisor is appointed to the processing centre replacing the existing one.

e)Updating the price of milk batch according to the Farmers.

3) Retrieval of Data

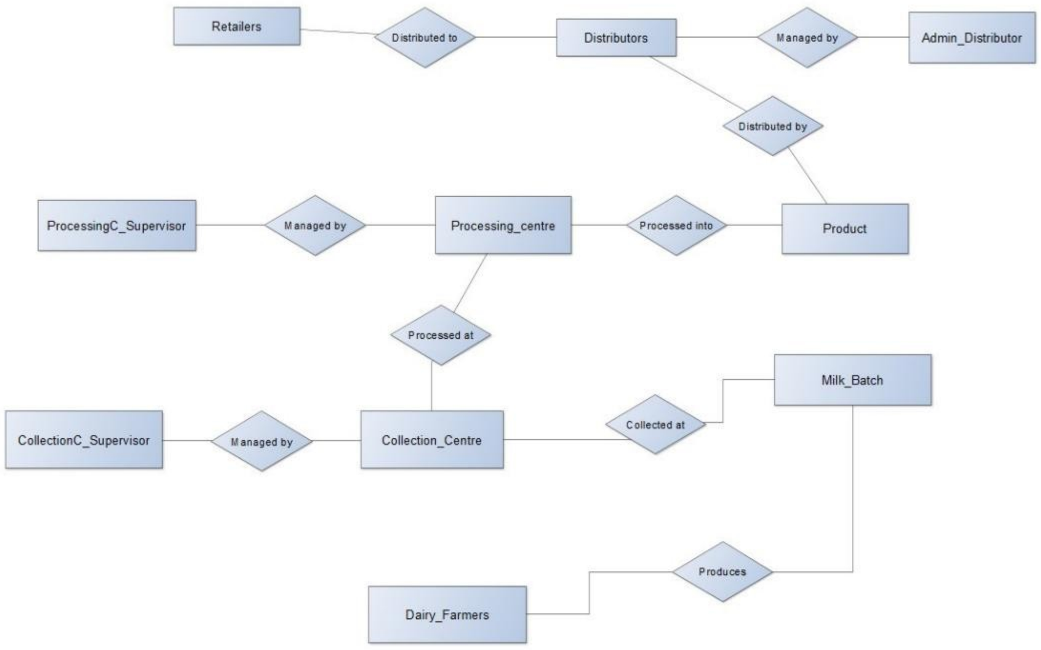
a)If retailer has problem with his problem he can trace back
to collection_centre

b) Selecting only those farmers who have more than 1 cow.

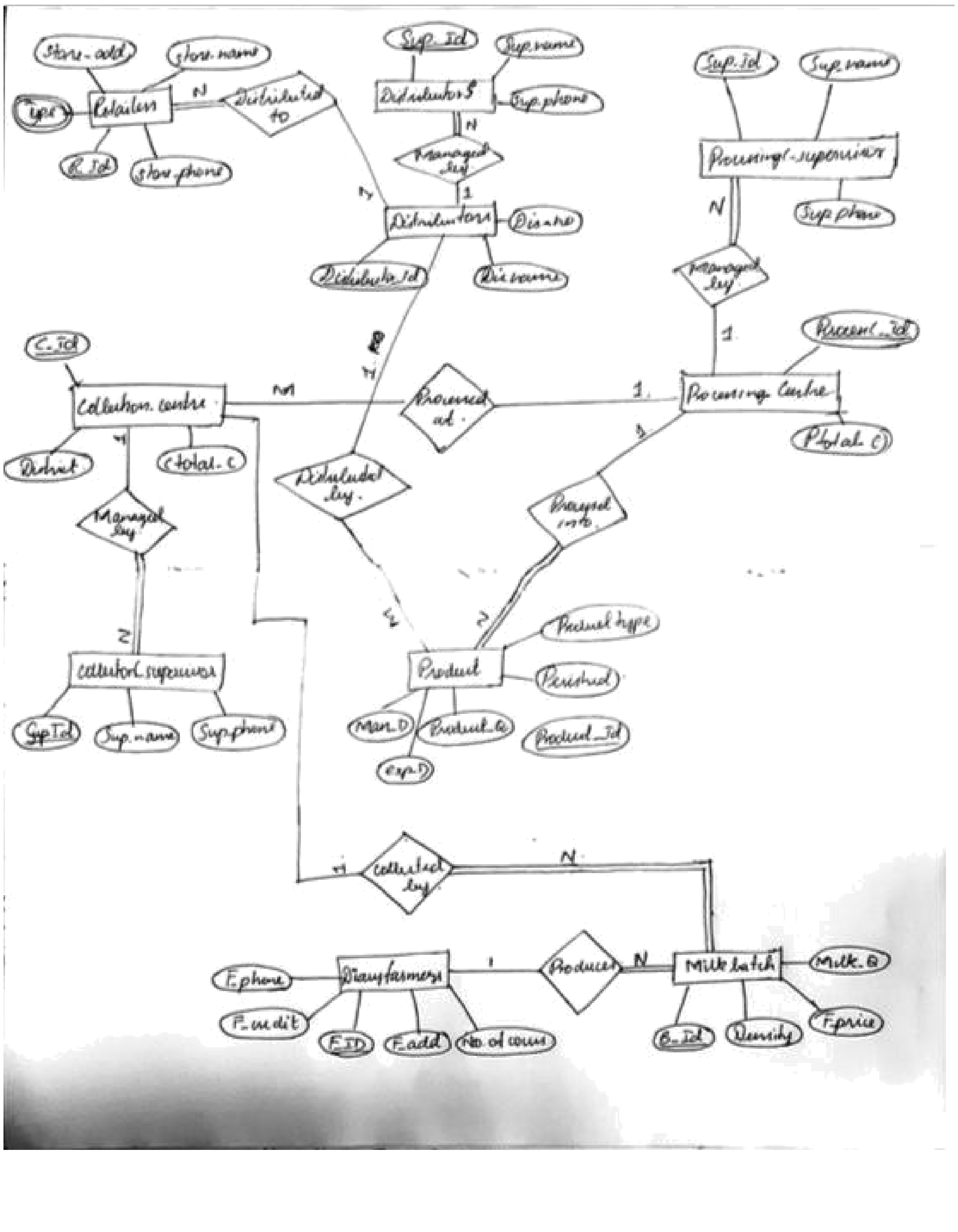
c) Selecting the Id and Density of milk batch with a condition:

d) Sum of Milk quantity produced by each farmer/

Entities and their Relational



The complete ER Diagram has been done in a chart paper .


Store-add
store.name
type
Retailers
R_Id
store.phone
N
Distributed to
M
Sup_Id
Sup.name
Distributor S
Sup.phone
N
Managed by
1
Distributors
Dis.no
Distributor_Id
Dis.name
Sup.Id
Sup.name
Processing-supervisor
Sup.phone
N
Managed by
1
Process_Id
Processing Centre
P.total. C
C_Id
Collection.centre
District
C.total. C
M
Processed at
1.
M
M
Distributed by.
1
Managed by
N
Processed into.
N
N
collection supervisor
Sup.Id
Sup.name
Sup.phone
Product
Product type
Price
Man_D
Product_Q
Product_Id
exp_D
M
collected by
N.
Dairyfarmers
1
Produces
N
Milk batch
Milk_Q
F.phone
F_credit
F_ID
F.add
No. of cows
B_Id
Density
F.price

RELATION DATABASE SCHEMA DIAGRAM

Dairy_farmers(F_ID PK, F_Name,
F_Address, No_of_cows,F_phone, F_credit);

Milk_batch(B_Id PK, **F_ID FK,C_Id FK**,
Density, Milk_Quantity,F_price)

Collection_centre(C_Id PK, **ProcessC_Id FK**,District,
CTotal_capacity)

Processing_centre(ProcessC_Id PK, PTotal_capacity)

Product (Product_ID PK, **ProcessC_ID FK, Distributors_Id F.K**,Product_type,Exp_Date, Manufacture_Day, Perished)

Distributors(Distributors_Id PK,Distributors name,Distributors no.
)

Retailers (R_Id PK, **Distributors_Id FK**,
Store_name,Store_Address,Store_phone,{Type})

CollectionC_supervisor(Supervisor_id, Supervisor_name,
Supervisor_phone,**C_ID FK**)

ProcessingC_supervisor(Supervisor_id, Supervisor_name,
Supervisor_phone,**ProcesssC_Id FK**)

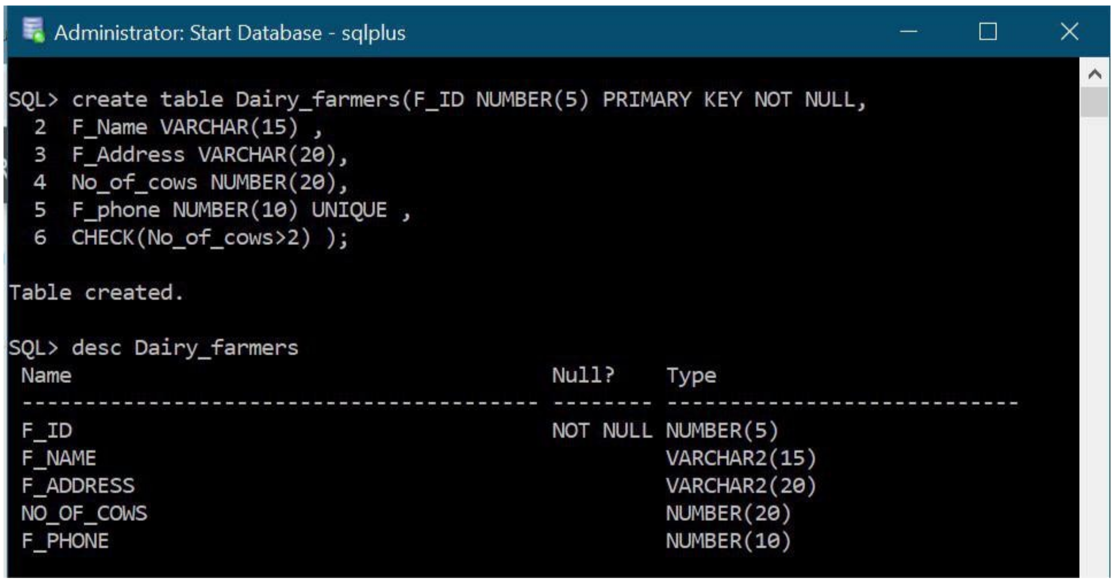
Distributor_supervisor(Supervisor_id, Supervisor_name,
Supervisor_phone,**Distributors_Id FK**)

Type(type P.K,**R_ID F.K**)

CREATION OF TABLES:

DAIRY FARMERS TABLE:

```
SQL> create table Dairy_farmers
(
  F_ID NUMBER(5) PRIMARY KEY NOT NULL,
     F_Name VARCHAR(15) ,
     F_Address VARCHAR(20),
 No_of_cows NUMBER(20),
     F_phone NUMBER(10) UNIQUE ,
     CHECK(No_of_cows>2)
);
```

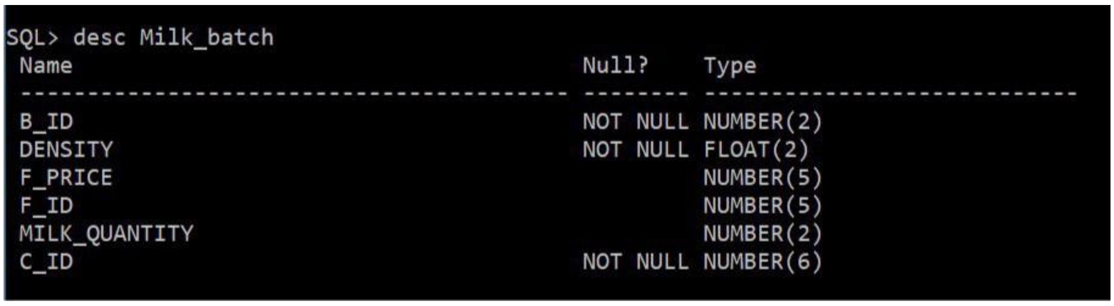


Administrator: Start Database - sqlplus

```
SQL> create table Dairy_farmers(F_ID NUMBER(5) PRIMARY KEY NOT NULL,
     F_Name VARCHAR(15) ,
     F_Address VARCHAR(20),
     No_of_cows NUMBER(20),
     F_phone NUMBER(10) UNIQUE ,
     CHECK(No_of_cows>2) );

Table created.

SQL> desc Dairy_farmers
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 F_ID                                      NOT NULL NUMBER(5)
 F_NAME                                             VARCHAR2(15)
 F_ADDRESS                                          VARCHAR2(20)
 NO_OF_COWS                                         NUMBER(20)
 F_PHONE                                            NUMBER(10)
```

MILK BATCH TABLE:

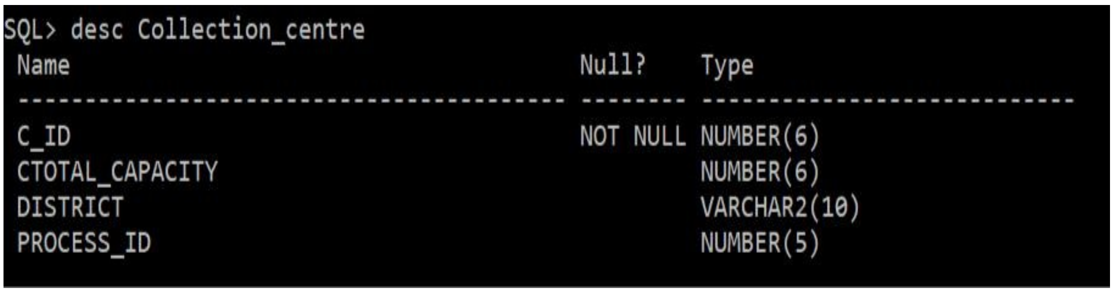
```
create table Milk_batch
(
 B_ID NUMBER(2) PRIMARY KEY NOT NULL,
    Density FLOAT(2) NOT NULL,
    F_Price NUMBER(5),
  F_ID NUMBER(5),
    Milk_Quantity NUMBER(2),
    FOREIGN KEY(F_ID) REFERENCES Dairy_farmers(F_ID),
    CHECK(Density>1.12 AND Milk_Quantity>1)
);
```



```
SQL> desc Milk_batch
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 B_ID                                      NOT NULL NUMBER(2)
 DENSITY                                   NOT NULL FLOAT(2)
 F_PRICE                                            NUMBER(5)
 F_ID                                               NUMBER(5)
 MILK_QUANTITY                                      NUMBER(2)
 C_ID                                      NOT NULL NUMBER(6)
```

COLLECTION CENTRE TABLE:

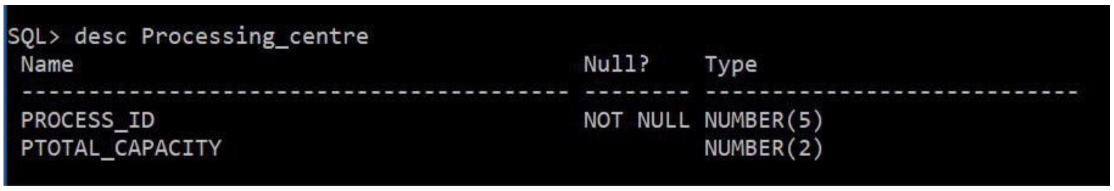
create table Collection_centre
(
C_ID NUMBER(5) PRIMARY KEY NOT NULL,
District VARCHAR(20),
CTotal_Capacity NUMBER(5),
Process_ID NUMBER(5),
FOREIGN KEY(ProcessC_ID)
REFERENCES
Dairy_farmers(ProcessC_ID));



```
SQL> desc Collection_centre
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 C_ID                                      NOT NULL NUMBER(6)
 CTOTAL_CAPACITY                                    NUMBER(6)
 DISTRICT                                           VARCHAR2(10)
 PROCESS_ID                                         NUMBER(5)
```

PROCESSING CENTER TABLE:

create table Processing_Centre
(
Process_ID number(5) PRIMARY
KEY, Ptotal_capacity number(2));

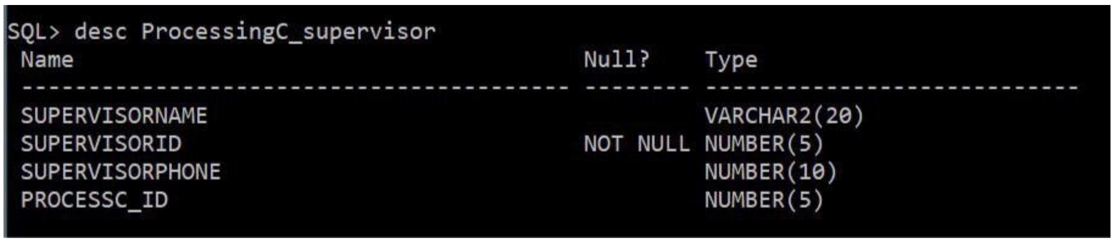


```
SQL> desc Processing_centre
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PROCESS_ID                                NOT NULL NUMBER(5)
 PTOTAL_CAPACITY                                    NUMBER(2)
```

PROCESSING CENTER SUPERVISOR TABLE:

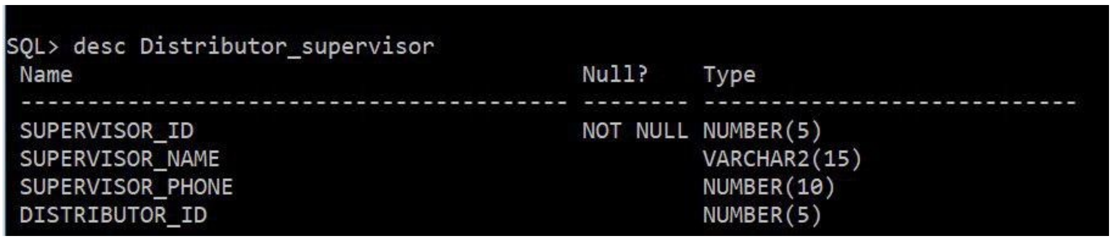
create table ProcessingC_Supervisor
(
SupervisorName varchar(20),
SupervisorID number(5) PRIMARY
KEY, SupervisorPhone number(10),

foreign key (Process_ID) references Processing_Centre(Process_ID)
);


```
SQL> desc ProcessingC_supervisor
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SUPERVISORNAME                                     VARCHAR2(20)
 SUPERVISORID                              NOT NULL NUMBER(5)
 SUPERVISORPHONE                                    NUMBER(10)
 PROCESSC_ID                                        NUMBER(5)
```

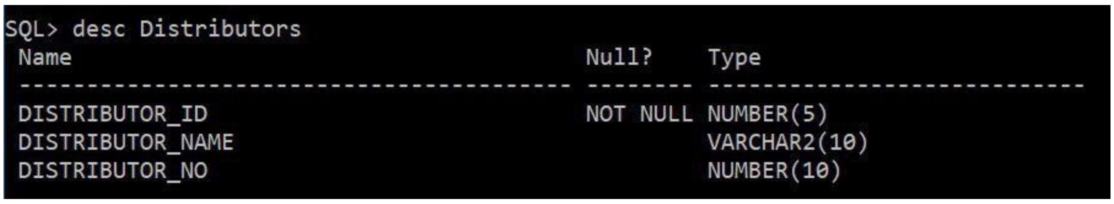
DISTRIBUTORS SUPERVISOR TABLE:

create table Distributor_supervisor
(
Supervisor_ID number(5) Primary key,
Supervisor_Name varchar(15),
Supervisor_phone number(10),
Distributor_ID number(5) foreign key (Distributor_ID) references Distributors(Distributor_ID)
);


```
SQL> desc Distributor_supervisor
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SUPERVISOR_ID                             NOT NULL NUMBER(5)
 SUPERVISOR_NAME                                    VARCHAR2(15)
 SUPERVISOR_PHONE                                   NUMBER(10)
 DISTRIBUTOR_ID                                     NUMBER(5)
```

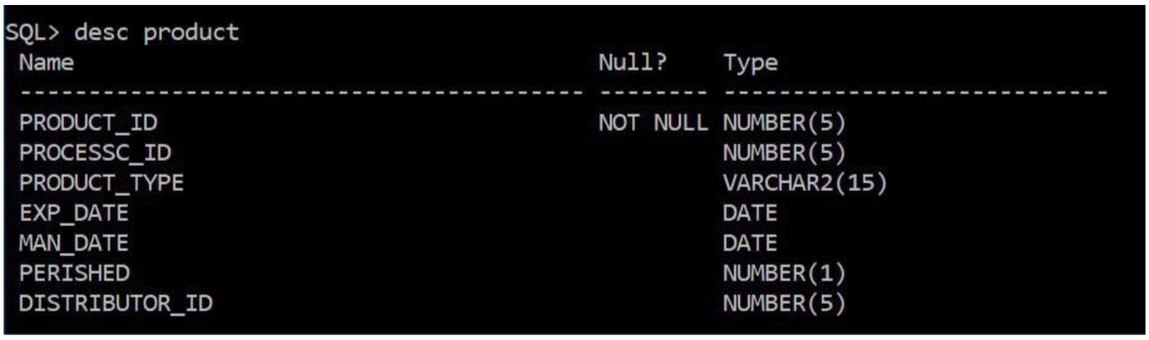
DISTRIBUTORS TABLE:

create table Distributors
(
Distributors_Id number(5) primary key not null,
Distributors_name varchar(10),
Distributors_no number(10)
);


```
SQL> desc Distributors
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 DISTRIBUTOR_ID                            NOT NULL NUMBER(5)
 DISTRIBUTOR_NAME                                   VARCHAR2(10)
 DISTRIBUTOR_NO                                     NUMBER(10)
```

PRODUCT TABLE:

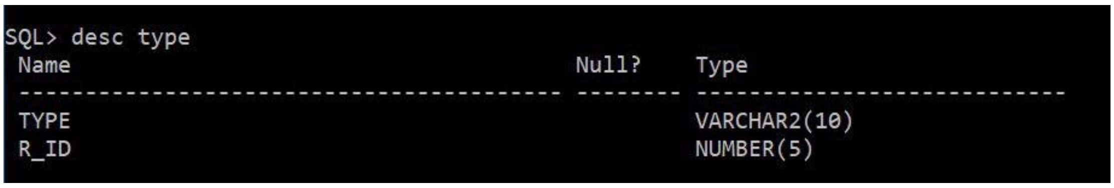
create table Product
(
Product_ID number(5) primary key,
ProcessC_ID number(5),
Product_type varchar(15),
Exp_Date date,
Man_Date date,
Perished number(1),
Distributor_ID number(5),
foreign key (Distributor_ID) references Distributors(Distributor_ID),
foreign key (ProcessC_ID) references Processing_centre(Process_ID));



```
SQL> desc product
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PRODUCT_ID                                NOT NULL NUMBER(5)
 PROCESSC_ID                                        NUMBER(5)
 PRODUCT_TYPE                                       VARCHAR2(15)
 EXP_DATE                                           DATE
 MAN_DATE                                           DATE
 PERISHED                                           NUMBER(1)
 DISTRIBUTOR_ID                                     NUMBER(5)
```

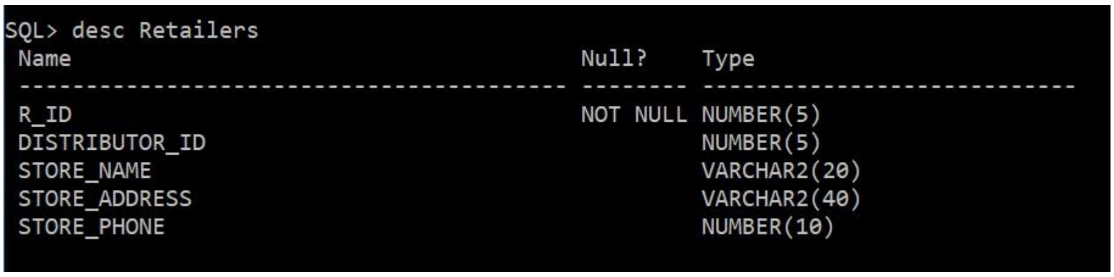
TYPE TABLE:

create table type
(
Type varchar(10) ,
R_ID number(5),
foreign key(R_ID) references Retailers(R_ID)
);



```
SQL> desc type
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TYPE                                               VARCHAR2(10)
 R_ID                                               NUMBER(5)
```

```
create table Retailers
(
R_ID number(5) PRIMARY KEY,
Distributor_ID number(5),
Store_Name varchar(20),
Store_Address varchar(40),
Store_Phone number(10),
foreign key (Distributor_ID) references Distributor(Disributor_ID)
);
```

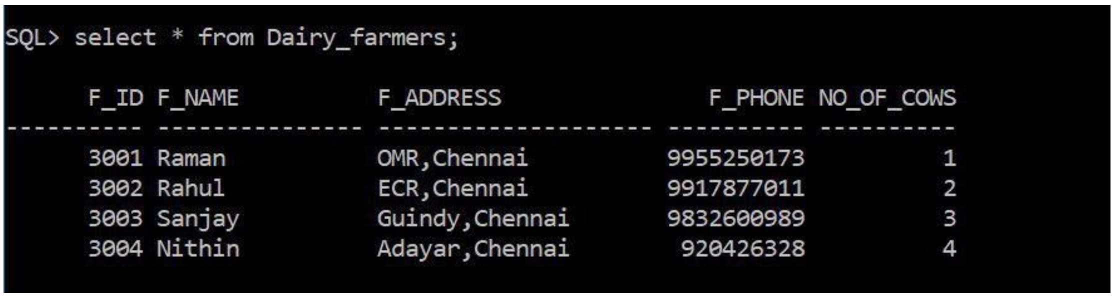


```
SQL> desc Retailers
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 R_ID                                      NOT NULL NUMBER(5)
 DISTRIBUTOR_ID                                     NUMBER(5)
 STORE_NAME                                         VARCHAR2(20)
 STORE_ADDRESS                                      VARCHAR2(40)
 STORE_PHONE                                        NUMBER(10)
```

INSERTION:

INSERTION INTO Dairy_farmers:

```
insert into Dairy_farmers
values(3001,'Raman','OMR,Chennai',1,9955250173)

insert into Dairy_farmers values
(3002,'Rahul','ECR,Chennai',2,9917877011)

insert into Dairy_farmers values
(3003,'Sanjay','Guindy,Chennai',3,9832600989)

insert into Dairy_farmers values
(3004,'Nithin','Adayar,Chennai',4,920426328)
```



```
SQL> select * from Dairy_farmers;

      F_ID F_NAME          F_ADDRESS               F_PHONE NO_OF_COWS
---------- --------------- -------------------- ---------- ----------
      3001 Raman           OMR,Chennai          9955250173          1
      3002 Rahul           ECR,Chennai          9917877011          2
      3003 Sanjay          Guindy,Chennai       9832600989          3
      3004 Nithin          Adayar,Chennai        920426328          4
```

INSERTION INTO Milk_batch:

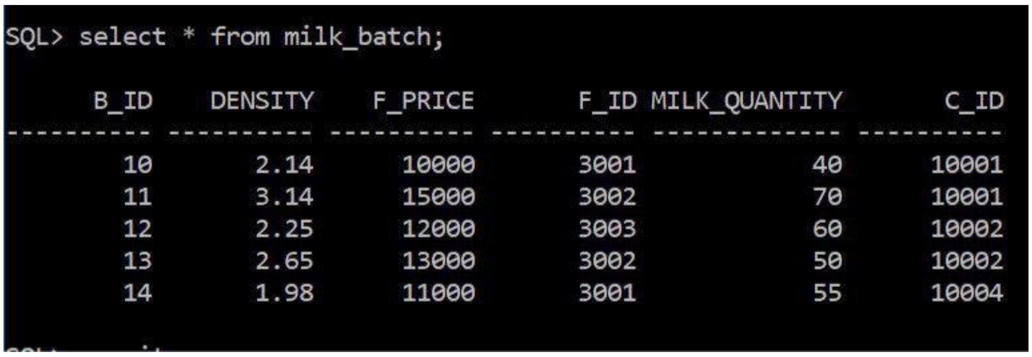
insert into Milk_batch values (10,2.14,10000,3001,40,10001);

insert into Milk_batch values (11,3.14,15000,3002,70,10001);

insert into Milk_batch values (12,2.25,12000,3003,60,10002);

insert into Milk_batch values (13,2.65,13000,3002,50,10002);

insert into Milk_batch values (14,1.98,11000,3001,55,10004);



```
SQL> select * from milk_batch;

      B_ID    DENSITY    F_PRICE       F_ID MILK_QUANTITY       C_ID
---------- ---------- ---------- ---------- ------------- ----------
        10       2.14      10000       3001            40      10001
        11       3.14      15000       3002            70      10001
        12       2.25      12000       3003            60      10002
        13       2.65      13000       3002            50      10002
        14       1.98      11000       3001            55      10004
```

INSERTION INTO Collection_centre:

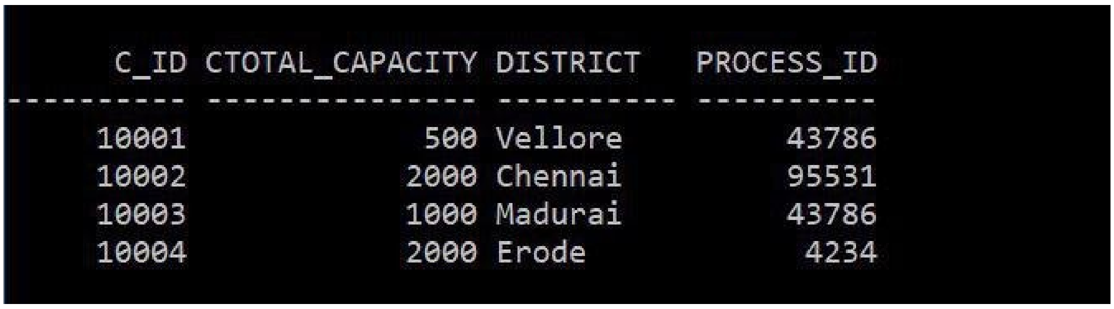
insert into Collection_centre values(10001,500, ’Vellore’,43786);

insert into Collection_centre values(10002,2000, ’Chennai’,95531);

insert into Collection_centre values(10003,1000, ’Madurai’,43786);

insert into Collection_centre

values(10004,2000, ’Erode’,04234);



```
      C_ID CTOTAL_CAPACITY DISTRICT   PROCESS_ID
---------- --------------- ---------- ----------
     10001             500 Vellore         43786
     10002            2000 Chennai         95531
     10003            1000 Madurai         43786
     10004            2000 Erode            4234
```

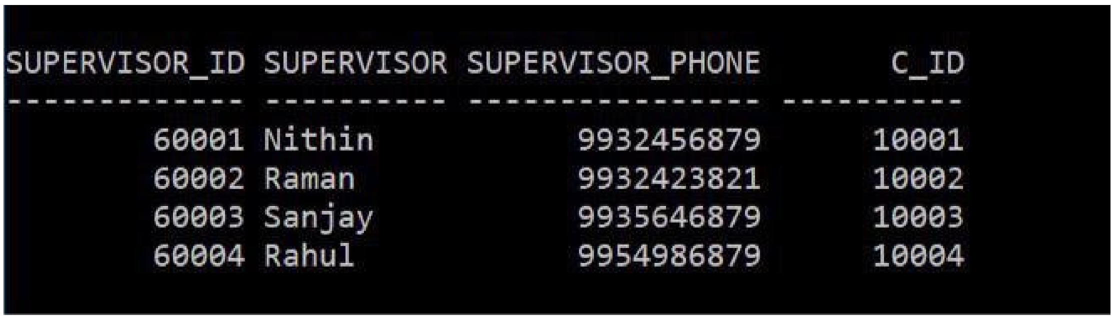
INSERTION INTO CollectionC_supervisor:

insert into CollectionC_supervisor values(60001,’Nithin’,9932456879,10001);

insert into CollectionC_supervisor values(60002,’Raman’,9932423821,10002);

insert into CollectionC_supervisor values(60003,’Sanjay’,9935646879,10003);

insert into CollectionC_supervisor values(60004,’Rahul’,9954986879,10004);



```
SUPERVISOR_ID SUPERVISOR SUPERVISOR_PHONE       C_ID
------------- ---------- ---------------- ----------
        60001 Nithin           9932456879      10001
        60002 Raman            9932423821      10002
        60003 Sanjay           9935646879      10003
        60004 Rahul            9954986879      10004
```

INSERTION INTO Distributors:

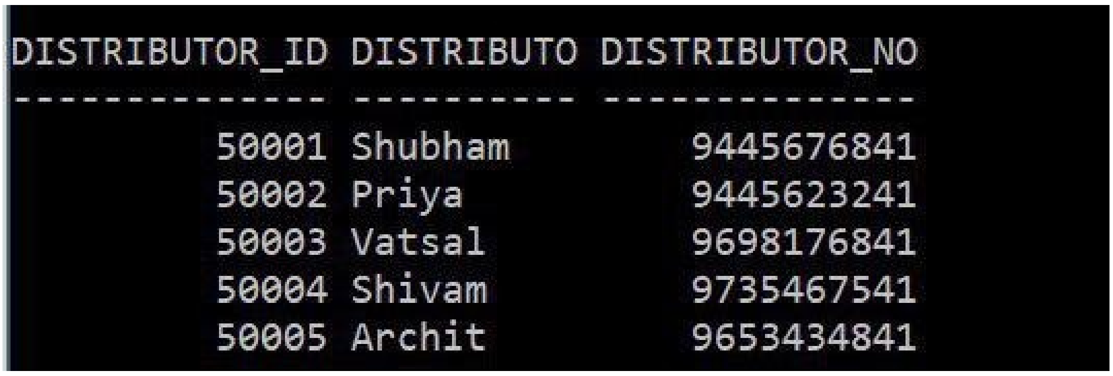
Insert into Distributors values(50001,’Shubham’,9445676841);

Insert into Distributors values(50002,’Priya’,9445623241);

Insert into Distributors values(50003,’Vatsal’,9698176841);

Insert into Distributors values(50004,’Shivam’,9735467541);

Insert into Distributors values(50005,’Archit’,9653434841);



```
DISTRIBUTOR_ID DISTRIBUTO DISTRIBUTOR_NO
-------------- ---------- --------------
         50001 Shubham        9445676841
         50002 Priya          9445623241
         50003 Vatsal         9698176841
         50004 Shivam         9735467541
         50005 Archit         9653434841
```

INSERTION INTO Products:

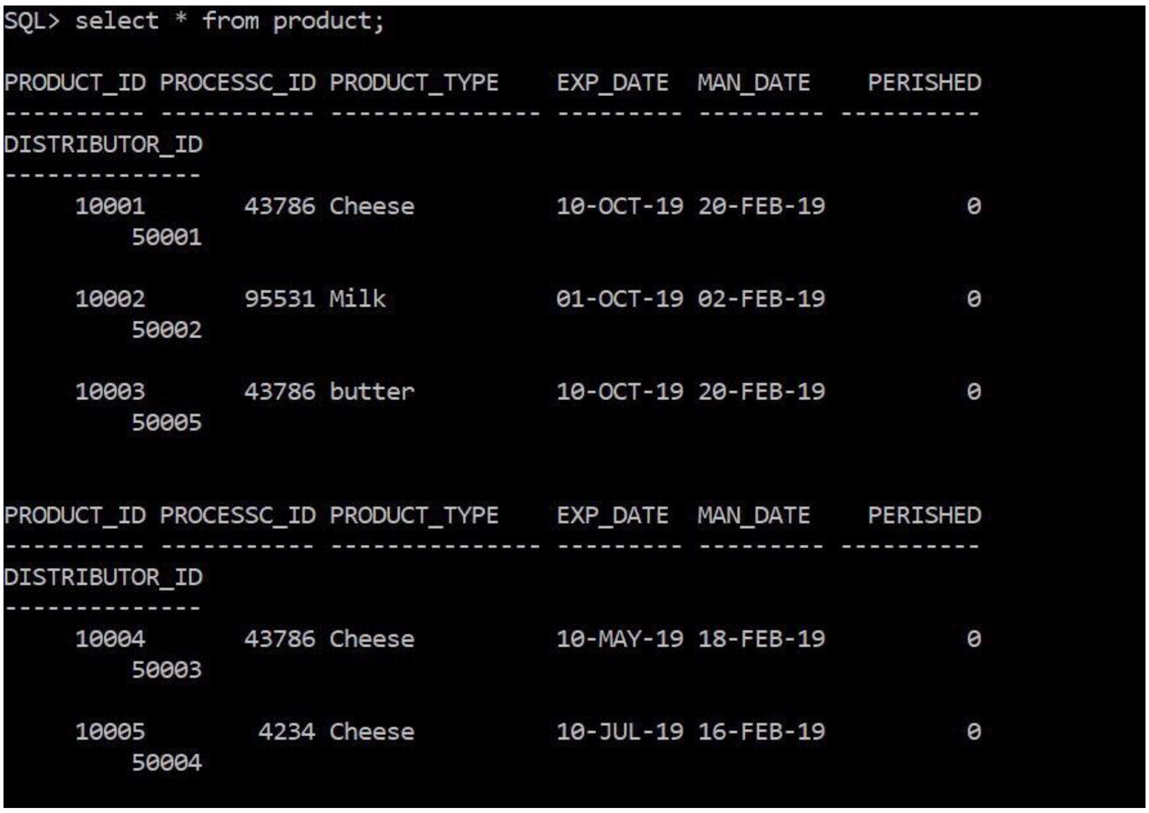
insert into Product values(10001,43786,'Cheese',to_date(’10-10-2019',’dd-mm-yyyy’),to_date(’20-2-2019’,’dd-mm-yyyy’),0,50001);

insert into Product values(10002,95531,'Milk', to_date (‘1-10-2019’,’dd-mm-yyyy’), to_date (’02-2-2019’,’dd-mm-yyyy’),0,50002);

insert into Product values(10003,43786,'butter',to_date(’10-10-2019’, ’dd-mm-yyyy’),to_date(’20-2-2019’, ’dd-mm-yyyy’),0,50005);

insert into Product values(10004,43786,'Cheese',to_date(’10-05-2019', ’dd-mm-yyyy’),to_date('18-2-2019', ’dd-mm-yyyy’),0,50003);

insert into Product values(10005,04234,'Cheese',to_date('10-07-2019', ’dd-mm-yyyy’),to_date('16-2-2019', ’dd-mm-yyyy’),0,50004);



```
SQL> select * from product;

PRODUCT_ID PROCESSC_ID PRODUCT_TYPE    EXP_DATE  MAN_DATE    PERISHED
---------- ----------- --------------- --------- --------- ----------
DISTRIBUTOR_ID
--------------
     10001       43786 Cheese          10-OCT-19 20-FEB-19          0
         50001

     10002       95531 Milk            01-OCT-19 02-FEB-19          0
         50002

     10003       43786 butter          10-OCT-19 20-FEB-19          0
         50005


PRODUCT_ID PROCESSC_ID PRODUCT_TYPE    EXP_DATE  MAN_DATE    PERISHED
---------- ----------- --------------- --------- --------- ----------
DISTRIBUTOR_ID
--------------
     10004       43786 Cheese          10-MAY-19 18-FEB-19          0
         50003

     10005        4234 Cheese          10-JUL-19 16-FEB-19          0
         50004
```

INSERTION INTO Distributor_supervisor:

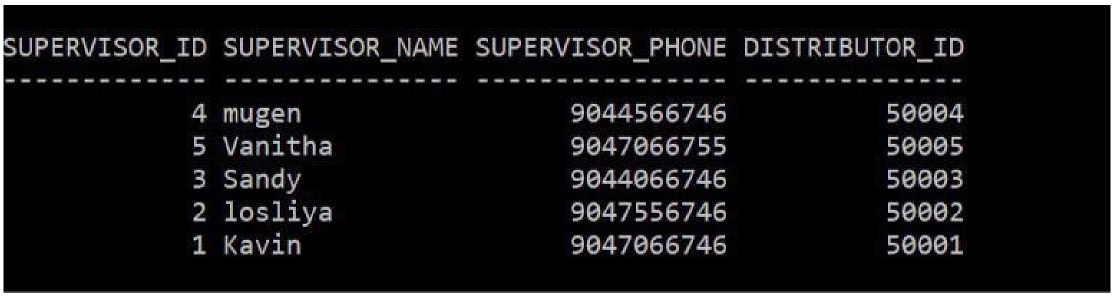
insert into Distributor_supervisor values(1,'Kavin',9047066746,50001);

insert into Distributor_supervisor values(2,'losliya',9047556746,50002);

insert into Distributor_supervisor values(3,'Sandy',9044066746,50003);

insert into Distributor_supervisor values(4,'mugen',9044566746,50004);

insert into Distributor_supervisor values(5,'Vanitha',9047066755,50005);



```
SUPERVISOR_ID SUPERVISOR_NAME SUPERVISOR_PHONE DISTRIBUTOR_ID
------------- --------------- ---------------- --------------
            4 mugen                 9044566746          50004
            5 Vanitha               9047066755          50005
            3 Sandy                 9044066746          50003
            2 losliya               9047556746          50002
            1 Kavin                 9047066746          50001
```

INSERTION INTO RETAILERS:

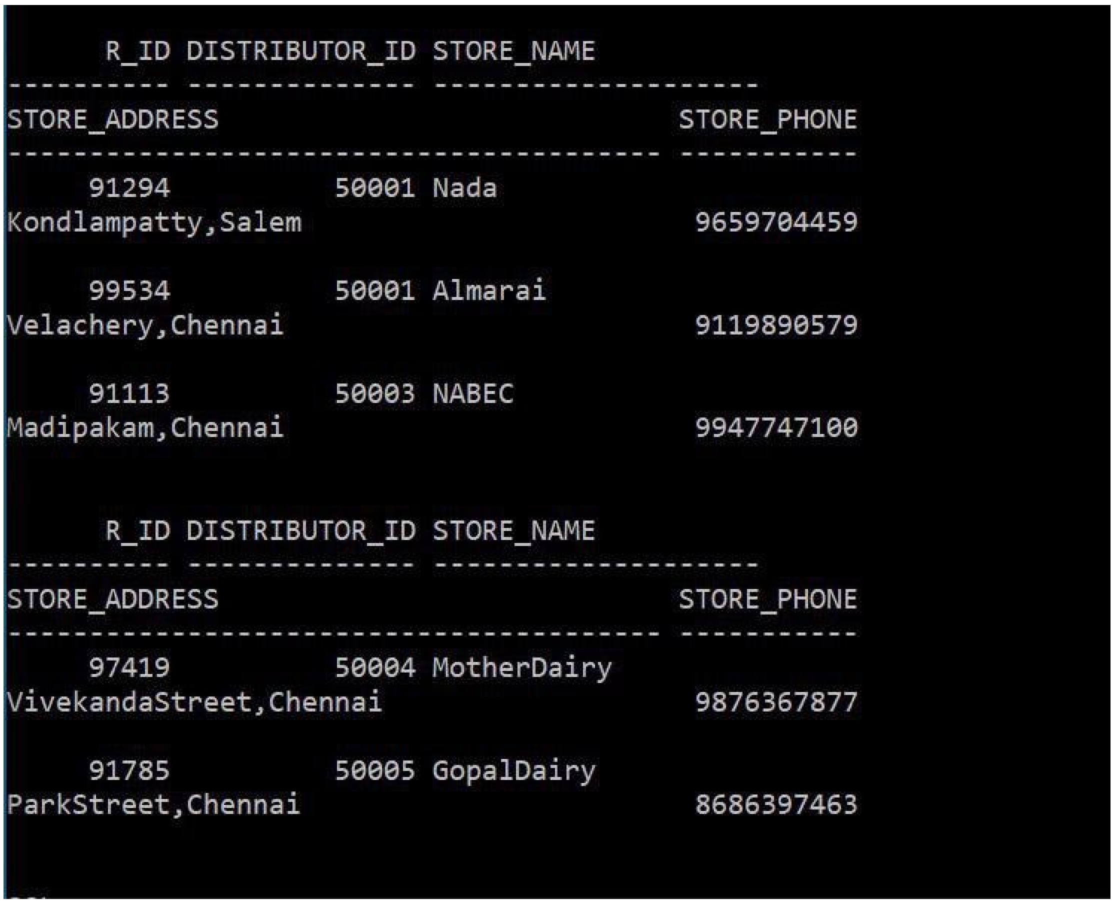
insert into RETAILERS values (91294 , 50001 ,'Nada', 'Kondlampatty,Salem', 9659704459);

insert into RETAILERS values(99534, 50001, 'Almarai', 'Velachery,Chennai', 9119890579);

insert into RETAILERS values (91113 ,50003,'NABEC', 'Madipakam,Chennai', 9947747100);

insert into RETAILERS values (97419 ,50004,'MotherDairy','VivekandaStreet,Chennai' ,9876367877);

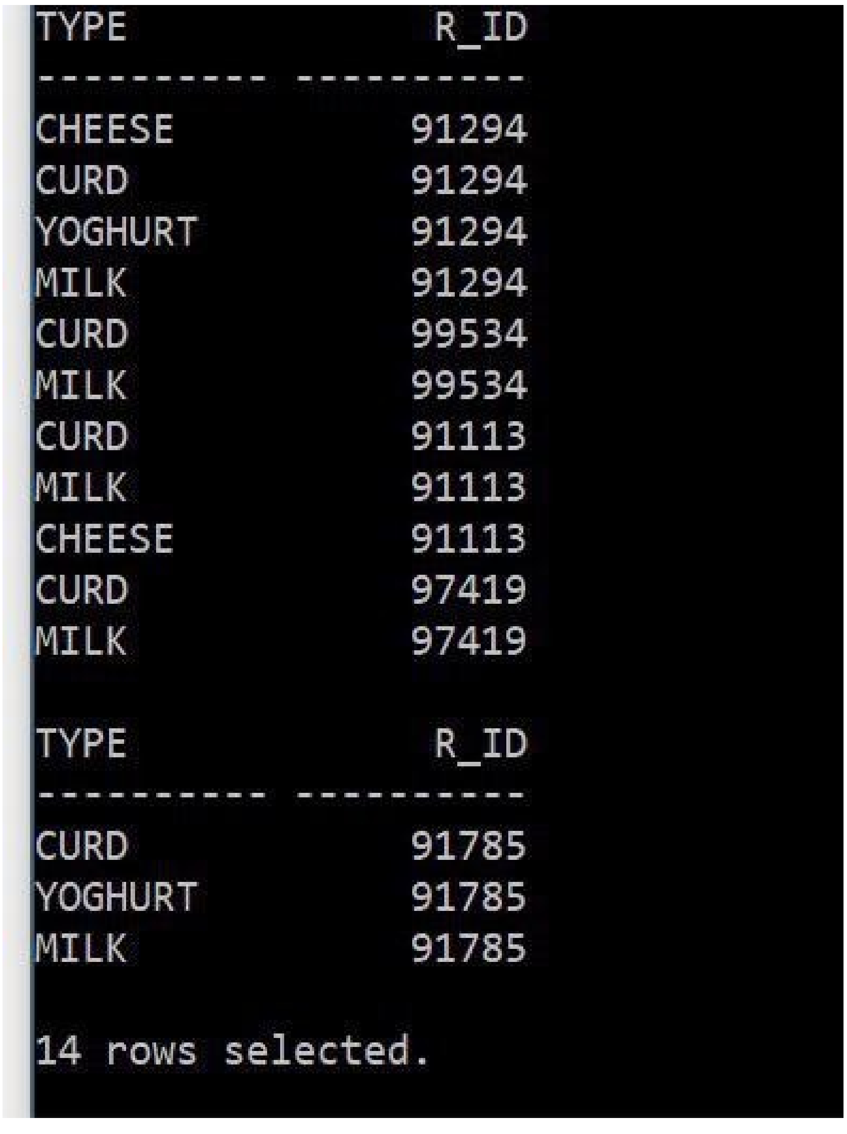
insert into RETAILERS values (91785,50005,'GopalDairy', 'ParkStreet','Chennai' 8686397463);



```
      R_ID DISTRIBUTOR_ID STORE_NAME
---------- -------------- --------------------
STORE_ADDRESS                            STORE_PHONE
---------------------------------------- -----------
     91294          50001 Nada
Kondlampatty,Salem                        9659704459

     99534          50001 Almarai
Velachery,Chennai                         9119890579

     91113          50003 NABEC
Madipakam,Chennai                         9947747100


      R_ID DISTRIBUTOR_ID STORE_NAME
---------- -------------- --------------------
STORE_ADDRESS                            STORE_PHONE
---------------------------------------- -----------
     97419          50004 MotherDairy
VivekandaStreet,Chennai                   9876367877

     91785          50005 GopalDairy
ParkStreet,Chennai                        8686397463
```

INSERTION INTO TYPE:

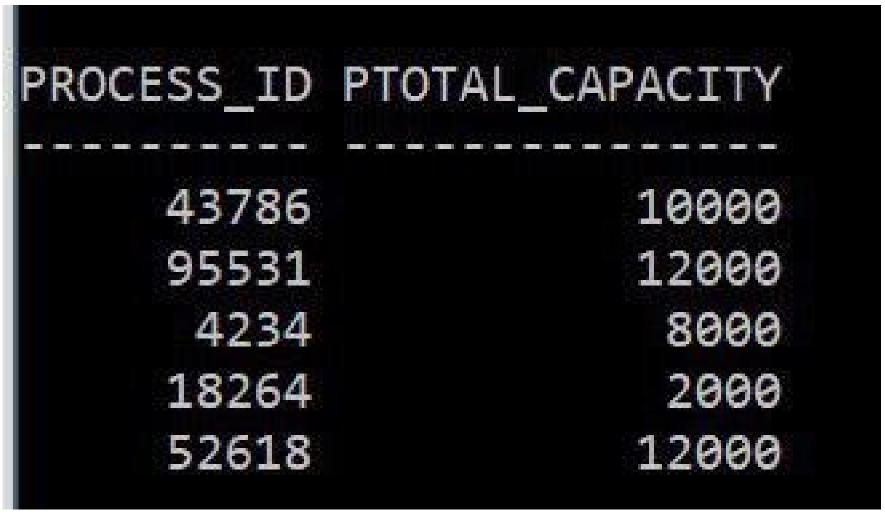
insert into TYPE values ('CHEESE', 91294);
insert into TYPE values ('CURD', 91294); insert
into TYPE values ('YOGHURT' ,91294); insert
into TYPE values ('MILK',91294); insert into
TYPE values ('CURD', 99534); insert into TYPE
values ('MILK', 99534); insert into TYPE values
('CURD' ,91113); insert into TYPE values
('MILK', 91113); insert into TYPE values
('CHEESE', 91113); insert into TYPE values
('CURD' ,97419); insert into TYPE values
('MILK', 97419); insert into TYPE values
('CURD' ,91785); insert into TYPE values
('YOGHURT',91785); insert into TYPE values
('MILK', 91785);



```
TYPE             R_ID
---------- ----------
CHEESE          91294
CURD            91294
YOGHURT         91294
MILK            91294
CURD            99534
MILK            99534
CURD            91113
MILK            91113
CHEESE          91113
CURD            97419
MILK            97419

TYPE             R_ID
---------- ----------
CURD            91785
YOGHURT         91785
MILK            91785

14 rows selected.
```

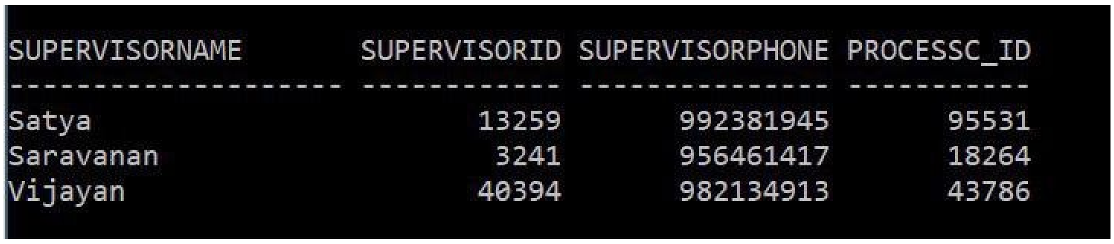
INSERTION PROCESSING CENTRE:

insert into PROCESSING_CENTRE values(43786, 10000);
insert into PROCESSING_CENTRE values (95531, 12000);
insert into PROCESSING_CENTRE values (04234, 8000);
insert into PROCESSING_CENTRE values (18264, 2000);
insert into PROCESSING_CENTRE values (52618, 12000);



```
PROCESS_ID PTOTAL_CAPACITY
---------- ---------------
     43786           10000
     95531           12000
      4234            8000
     18264            2000
     52618           12000
```

INSERTION PROCESSING CENTRE SUPERVISOR:
insert into PROCESSINGC_SUPERVISOR values (‘Vijayan’, 40394, 982134913,43786);
insert into PROCESSINGC_SUPERVISOR values ('Satya', 13259 , 992381945,95531);
insert into PROCESSINGC_SUPERVISOR values ('Saravanan', 03241, 956461417,18264);



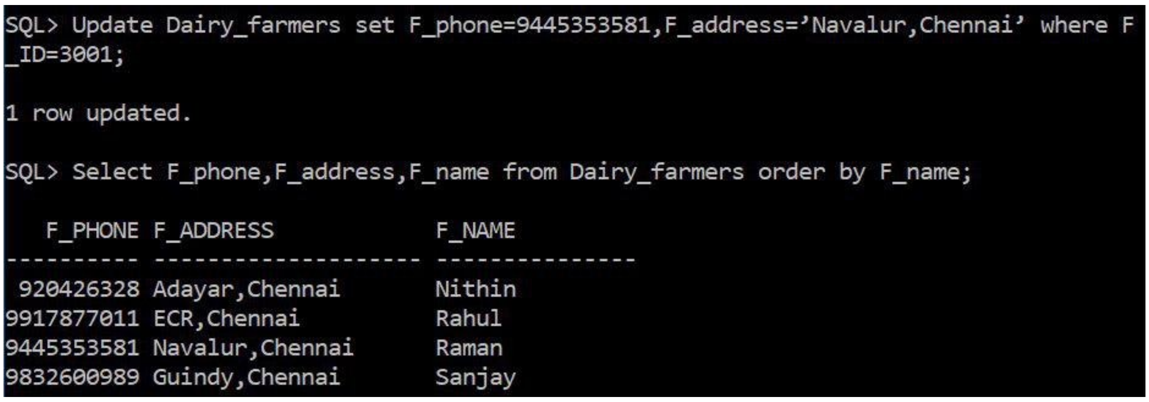
```
SUPERVISORNAME       SUPERVISORID SUPERVISORPHONE PROCESSC_ID
-------------------- ------------ --------------- -----------
Satya                       13259       992381945       95531
Saravanan                    3241       956461417       18264
Vijayan                     40394       982134913       43786
```

UPDATE STATEMENTS:

The Farmer should be able to update his/her phone no and address as required:

Update Dairy_farmers set F_phone=9445353581,F_address='Navalur,Chennai' where F_ID=3001;

Select F_phone,F_address,F_name from Dairy_farmers order by F_name;

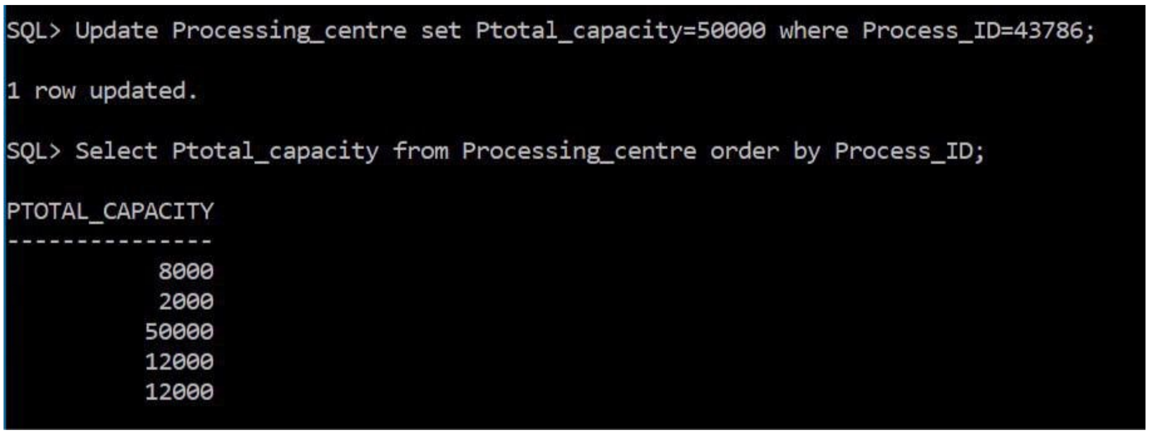


```
SQL> Update Dairy_farmers set F_phone=9445353581,F_address='Navalur,Chennai' where F
_ID=3001;

1 row updated.

SQL> Select F_phone,F_address,F_name from Dairy_farmers order by F_name;

   F_PHONE F_ADDRESS            F_NAME
---------- -------------------- ---------------
 920426328 Adayar,Chennai       Nithin
9917877011 ECR,Chennai          Rahul
9445353581 Navalur,Chennai      Raman
9832600989 Guindy,Chennai       Sanjay
```

The processing centre admin should be able to update the centre's total capacity required:

Update Processing_centre set Ptotal_capacity=50000 where Process_ID=43786;

Select Ptotal_capacity from Processing_centre order by Process_ID;

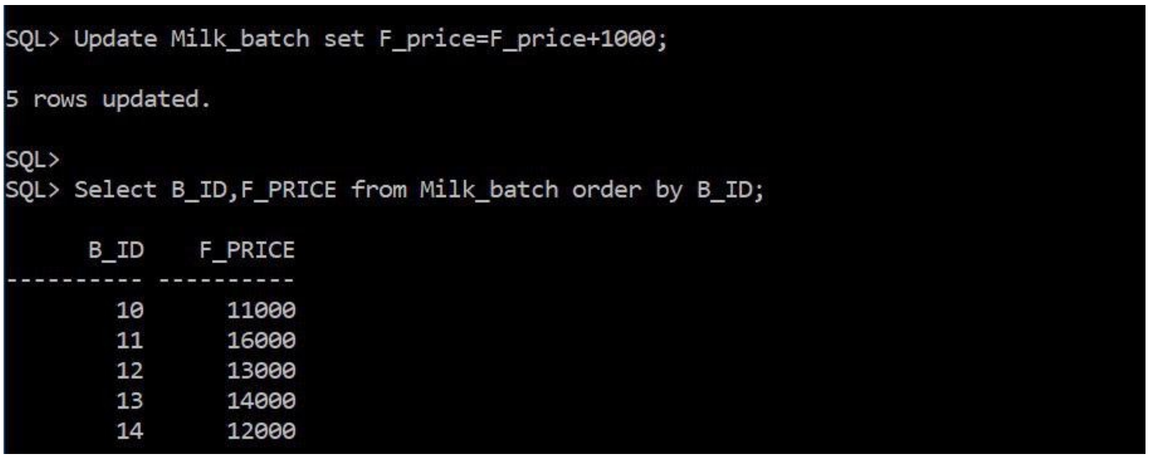


```
SQL> Update Processing_centre set Ptotal_capacity=50000 where Process_ID=43786;

1 row updated.

SQL> Select Ptotal_capacity from Processing_centre order by Process_ID;

PTOTAL_CAPACITY
---------------
           8000
           2000
          50000
          12000
          12000
```

The Price of the milk batch is increased due to supply demand:

Update Milk_batch set F_price=F_price+1000;

Select B_ID,F_PRICE from Milk_batch order by B_ID;

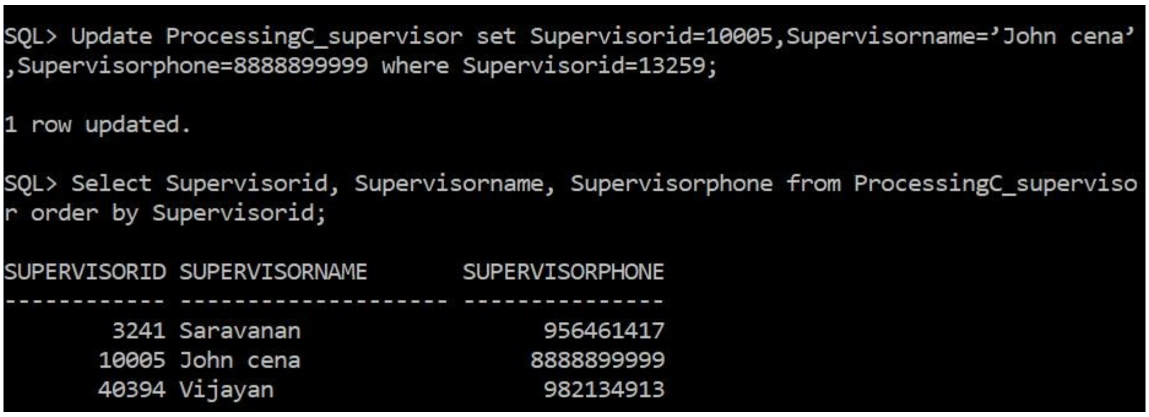


```
SQL> Update Milk_batch set F_price=F_price+1000;

5 rows updated.

SQL>
SQL> Select B_ID,F_PRICE from Milk_batch order by B_ID;

      B_ID    F_PRICE
---------- ----------
        10      11000
        11      16000
        12      13000
        13      14000
        14      12000
```

A new supervisor is appointed to the processing centre:

Update ProcessingC_supervisor set Supervisorid=10005,Supervisorname='John cena',Supervisorphone=8888899999 where Supervisorid=13259;

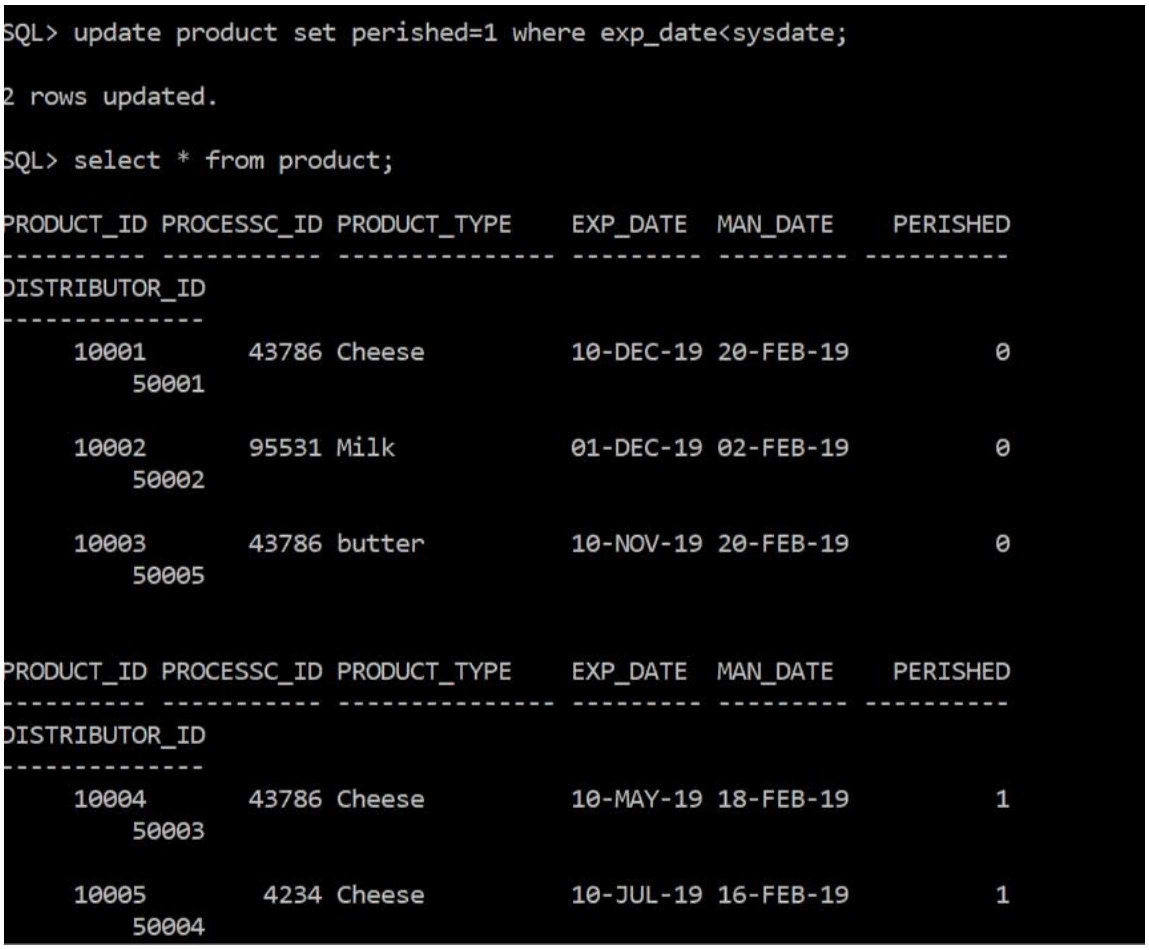
Select Supervisorid, Supervisorname, Supervisorphone from ProcessingC_supervisor order by Supervisorid;



```
SQL> Update ProcessingC_supervisor set Supervisorid=10005,Supervisorname='John cena'
,Supervisorphone=8888899999 where Supervisorid=13259;

1 row updated.

SQL> Select Supervisorid, Supervisorname, Supervisorphone from ProcessingC_superviso
r order by Supervisorid;

SUPERVISORID SUPERVISORNAME       SUPERVISORPHONE
------------ -------------------- ---------------
        3241 Saravanan                  956461417
       10005 John cena                 8888899999
       40394 Vijayan                    982134913
```

Update perished as 1 if exp date is greater than 61months:

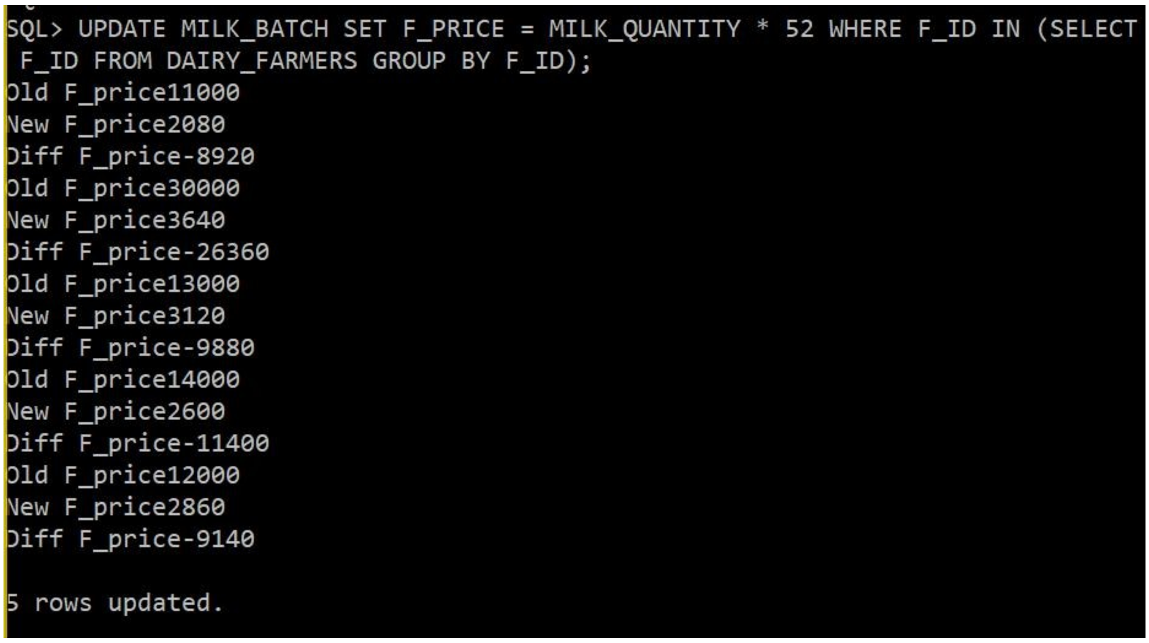
update product set perished=1 where exp_date>sysdate select*from product;



```
SQL> update product set perished=1 where exp_date<sysdate;

2 rows updated.

SQL> select * from product;

PRODUCT_ID PROCESSC_ID PRODUCT_TYPE    EXP_DATE  MAN_DATE    PERISHED
---------- ----------- -------------- --------- --------- ----------
DISTRIBUTOR_ID
--------------
     10001       43786 Cheese         10-DEC-19 20-FEB-19          0
         50001

     10002       95531 Milk           01-DEC-19 02-FEB-19          0
         50002

     10003       43786 butter         10-NOV-19 20-FEB-19          0
         50005


PRODUCT_ID PROCESSC_ID PRODUCT_TYPE    EXP_DATE  MAN_DATE    PERISHED
---------- ----------- -------------- --------- --------- ----------
DISTRIBUTOR_ID
--------------
     10004       43786 Cheese         10-MAY-19 18-FEB-19          1
         50003

     10005        4234 Cheese         10-JUL-19 16-FEB-19          1
         50004
```

Updating the New Milk Price

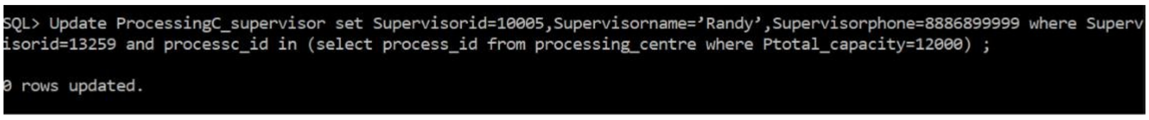
UPDATE MILK_BATCH SET F_PRICE = MILK_QUANTITY * 52 WHERE F_ID IN (SELECT F_ID FROM DAIRY_FARMERS GROUPBY F_ID);



```
SQL> UPDATE MILK_BATCH SET F_PRICE = MILK_QUANTITY * 52 WHERE F_ID IN (SELECT
 F_ID FROM DAIRY_FARMERS GROUP BY F_ID);
Old F_price11000
New F_price2080
Diff F_price-8920
Old F_price30000
New F_price3640
Diff F_price-26360
Old F_price13000
New F_price3120
Diff F_price-9880
Old F_price14000
New F_price2600
Diff F_price-11400
Old F_price12000
New F_price2860
Diff F_price-9140

5 rows updated.
```

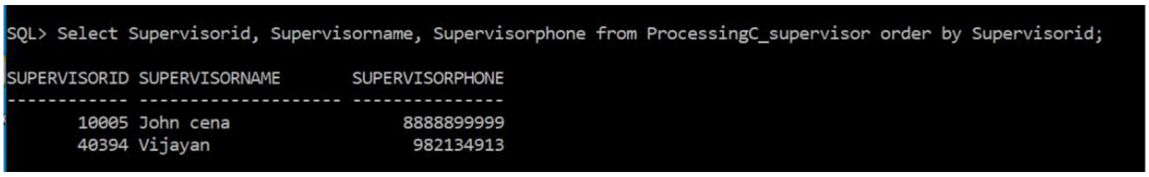
A new supervisor is appointed to a particular processing centre having a particular total capacity:

Update ProcessingC_supervisor set Supervisorid=10005,Supervisorname=’Randy’,Supervisorphone=8886899999 where Supervisorid=13259 and processc_id in (select process_id from processing_centre where Ptotal_capacity=12000) ;



```
SQL> Update ProcessingC_supervisor set Supervisorid=10005,Supervisorname=’Randy’,Supervisorphone=8886899999 where Superv
isorid=13259 and processc_id in (select process_id from processing_centre where Ptotal_capacity=12000) ;

0 rows updated.
```

Select Supervisorid, Supervisorname, Supervisorphone from ProcessingC_supervisor order by Supervisorid;

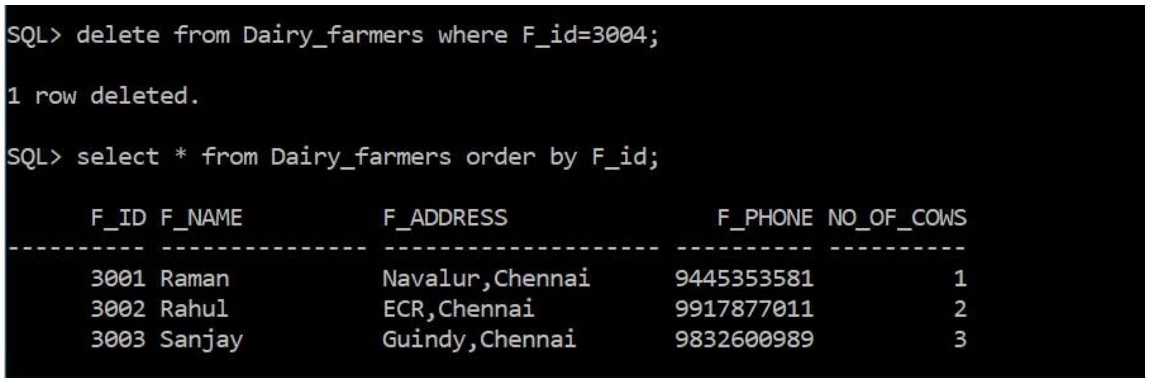


```
SQL> Select Supervisorid, Supervisorname, Supervisorphone from ProcessingC_supervisor order by Supervisorid;

SUPERVISORID SUPERVISORNAME       SUPERVISORPHONE
----------- ------------------- --------------
      10005 John cena                8888899999
      40394 Vijayan                   982134913
```

DELETE STATEMENTS:

The Farmer should be deleted if he is no longer producing any Milk batches

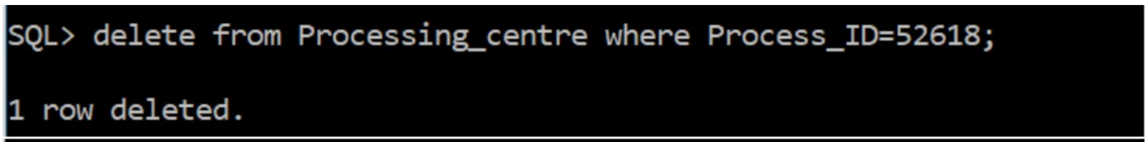
delete from Dairy_farmers where F_id=3004;
select * from Dairy_farmers order by F_id;



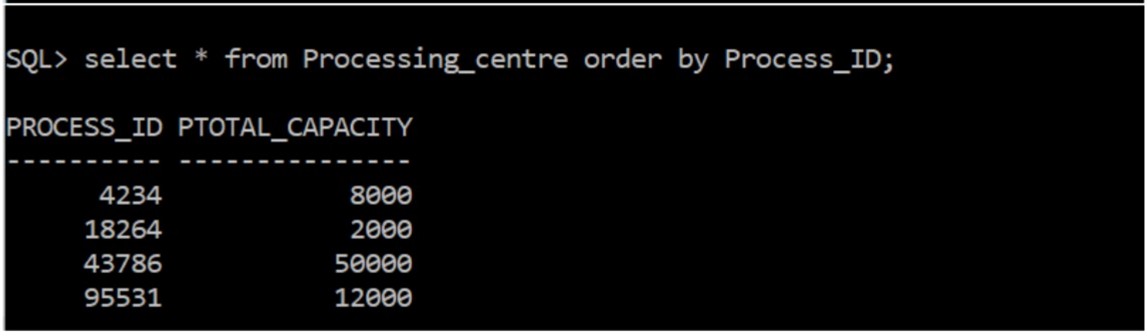
```
SQL> delete from Dairy_farmers where F_id=3004;

1 row deleted.

SQL> select * from Dairy_farmers order by F_id;

      F_ID F_NAME          F_ADDRESS               F_PHONE NO_OF_COWS
---------- --------------- -------------------- ---------- ----------
      3001 Raman           Navalur,Chennai      9445353581          1
      3002 Rahul           ECR,Chennai          9917877011          2
      3003 Sanjay          Guindy,Chennai       9832600989          3
```

The Processing centre should be removed if it hasn’t been functioning for a long time:

delete from ProcessingC_centre where ProcessC_ID=52618;
select * from ProcessingC_centre order by ProcessC_ID;



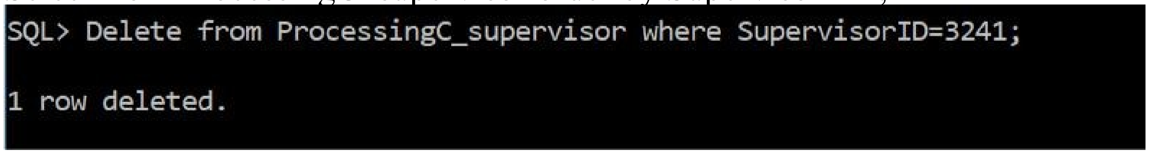
```
SQL> delete from Processing_centre where Process_ID=52618;

1 row deleted.
```



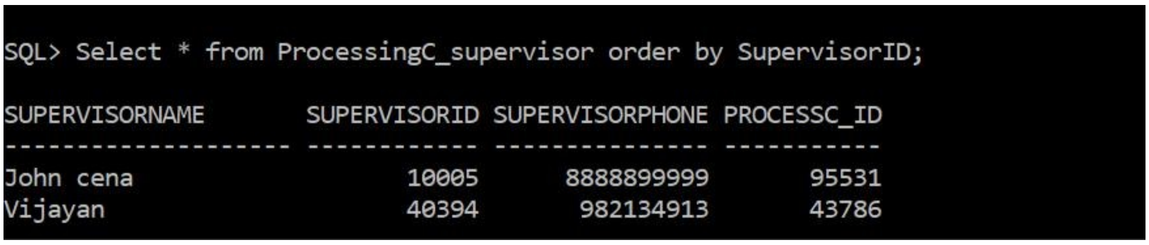
```
SQL> select * from Processing_centre order by Process_ID;

PROCESS_ID PTOTAL_CAPACITY
---------- ---------------
      4234            8000
     18264            2000
     43786           50000
     95531           12000
```

The Processing supervisor should be removed if he/she is not in job anymore:

Delete from ProcessingC_supervisor where Supervisor_ID=3241;
Select from ProcessingC_supervisor order by Supervisor_ID;



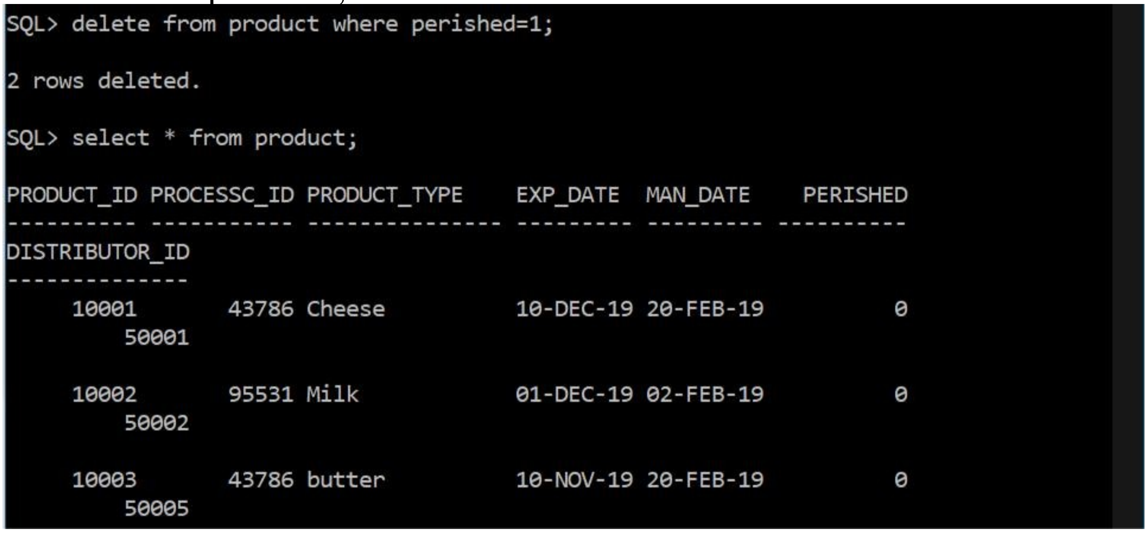
```
SQL> Delete from ProcessingC_supervisor where SupervisorID=3241;

1 row deleted.
```



```
SQL> Select * from ProcessingC_supervisor order by SupervisorID;

SUPERVISORNAME       SUPERVISORID SUPERVISORPHONE PROCESSC_ID
-------------------- ------------ --------------- -----------
John cena                   10005      8888899999       95531
Vijayan                     40394       982134913       43786
```

Delete if perished value is 1

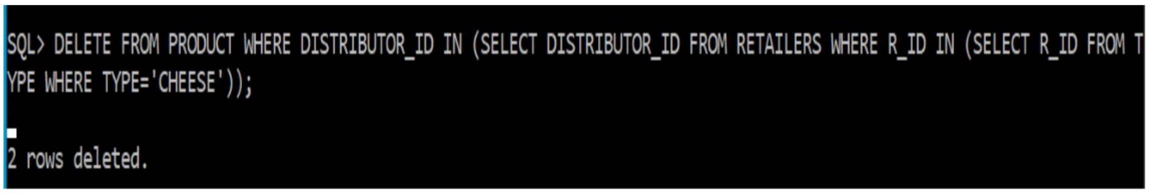
delete from product where perished=1;
select * from products;



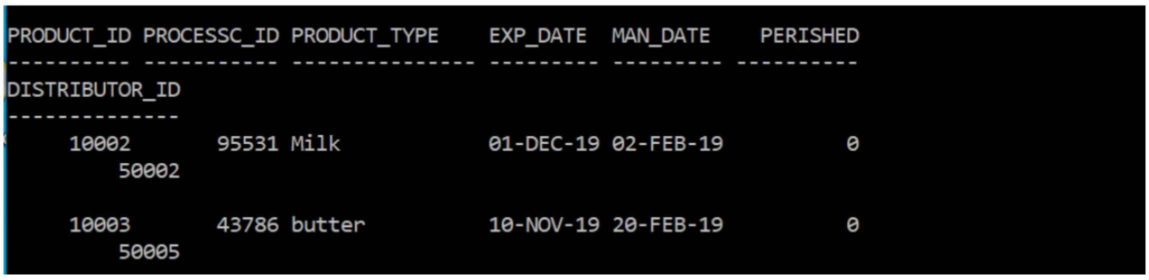
```
SQL> delete from product where perished=1;

2 rows deleted.

SQL> select * from product;

PRODUCT_ID PROCESSC_ID PRODUCT_TYPE    EXP_DATE  MAN_DATE    PERISHED
---------- ----------- --------------- --------- --------- ----------
DISTRIBUTOR_ID
--------------
     10001       43786 Cheese          10-DEC-19 20-FEB-19          0
         50001

     10002       95531 Milk            01-DEC-19 02-FEB-19          0
         50002

     10003       43786 butter          10-NOV-19 20-FEB-19          0
         50005
```

WHEN A PARTICULAR MILK TYPE IS EXPIRED WITH A RETAILER AND ALL THAT BELONGED TO IT HAS TO BE RECALLED:

DELETE FROM PRODUCT WHERE DISTRIBUTORS_ID IN (SELECT DISTRIBUTORS_ID FROM RETAILERS WHERE R_ID IN (SELECT R_ID FROM TYPE WHERE TYPE="CHEESE"));



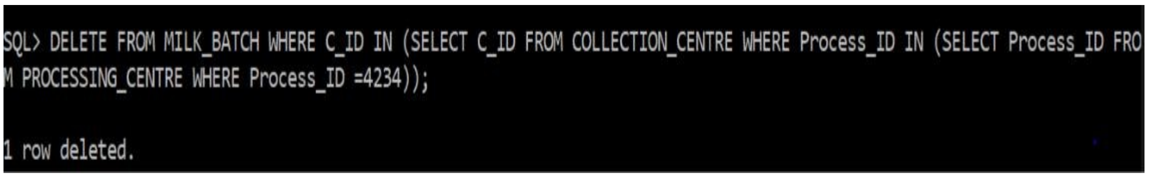
```
SQL> DELETE FROM PRODUCT WHERE DISTRIBUTOR_ID IN (SELECT DISTRIBUTOR_ID FROM RETAILERS WHERE R_ID IN (SELECT R_ID FROM T
YPE WHERE TYPE='CHEESE'));

2 rows deleted.
```



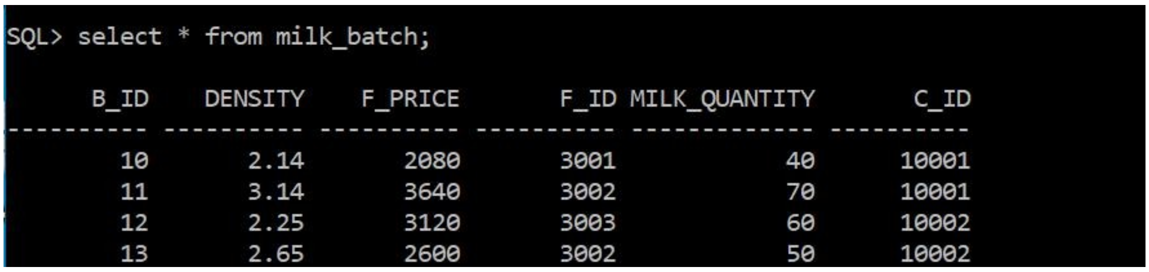
```
PRODUCT_ID PROCESSC_ID PRODUCT_TYPE    EXP_DATE  MAN_DATE    PERISHED
---------- ----------- -------------- --------- --------- ----------
DISTRIBUTOR_ID
--------------
     10002       95531 Milk           01-DEC-19 02-FEB-19          0
         50002

     10003       43786 butter         10-NOV-19 20-FEB-19          0
         50005
```

WHEN A MILK BATCH IS SPOILED AND IT IS NOTICED IN PROCESSING CENTER :

DELETE FROM MILK_BATCH WHERE C_ID IN (SELECT C_ID FROM COLLECTION_CENTRE WHERE PROCESS_ID IN (SELECT PROCESS_ID FROM PROCESSING_CENTRE WHERE PROCESS_ID =4234));



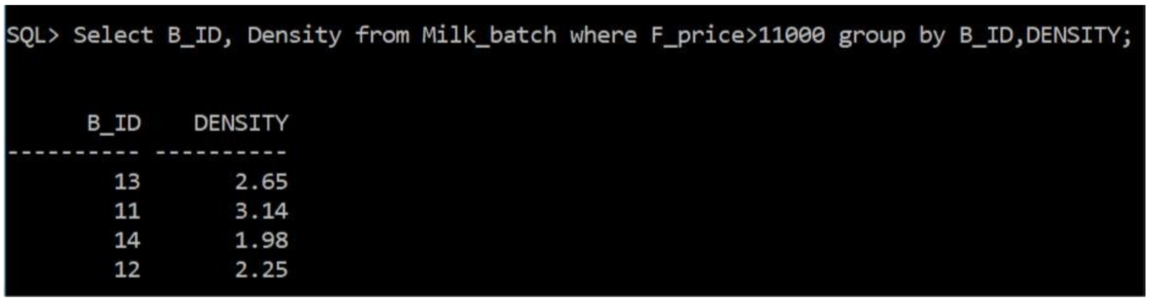
```
SQL> DELETE FROM MILK_BATCH WHERE C_ID IN (SELECT C_ID FROM COLLECTION_CENTRE WHERE Process_ID IN (SELECT Process_ID FRO
M PROCESSING_CENTRE WHERE Process_ID =4234));

1 row deleted.
```



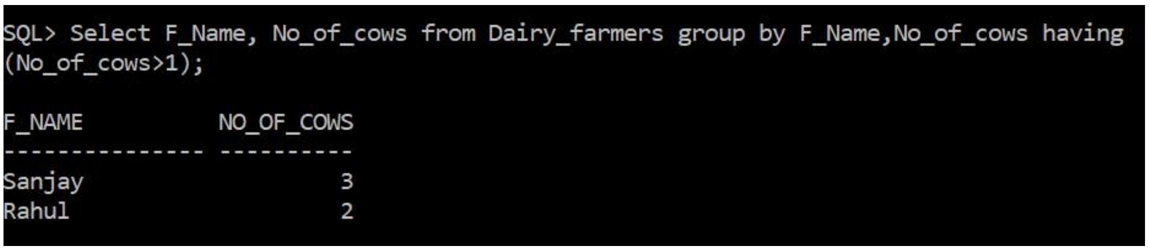
```
SQL> select * from milk_batch;

      B_ID    DENSITY    F_PRICE       F_ID MILK_QUANTITY       C_ID
---------- ---------- ---------- ---------- ------------- ----------
        10       2.14       2080       3001            40      10001
        11       3.14       3640       3002            70      10001
        12       2.25       3120       3003            60      10002
        13       2.65       2600       3002            50      10002
```

Select Statements:

Selecting the Id and Density of milk batch with a condition:
Select B_ID, Density from Milk_batch where F_price>11000 group by

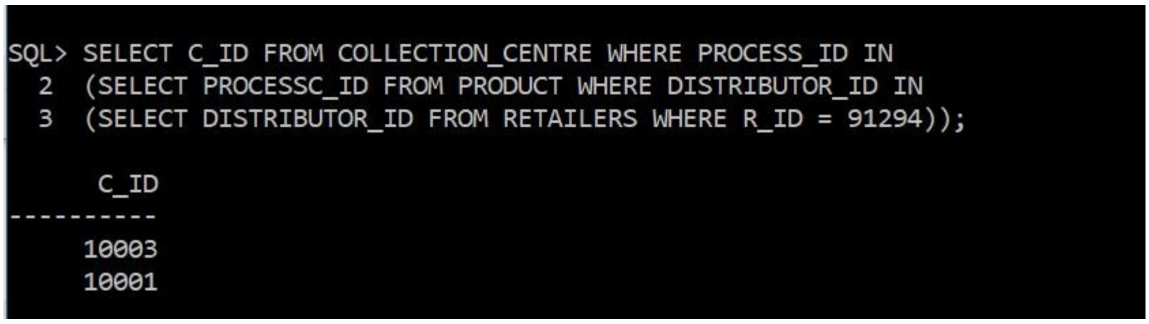

```
SQL> Select B_ID, Density from Milk_batch where F_price>11000 group by B_ID,DENSITY;

      B_ID    DENSITY
---------- ----------
        13       2.65
        11       3.14
        14       1.98
        12       2.25
```

Selecting only those farmers who have more than 1 cow. Select F_Name, No_of_cows from Dairy_farmers group by F_Name,No_of_cows having (No_of_cows>1);


```
SQL> Select F_Name, No_of_cows from Dairy_farmers group by F_Name,No_of_cows having
(No_of_cows>1);

F_NAME          NO_OF_COWS
--------------- ----------
Sanjay                   3
Rahul                    2
```

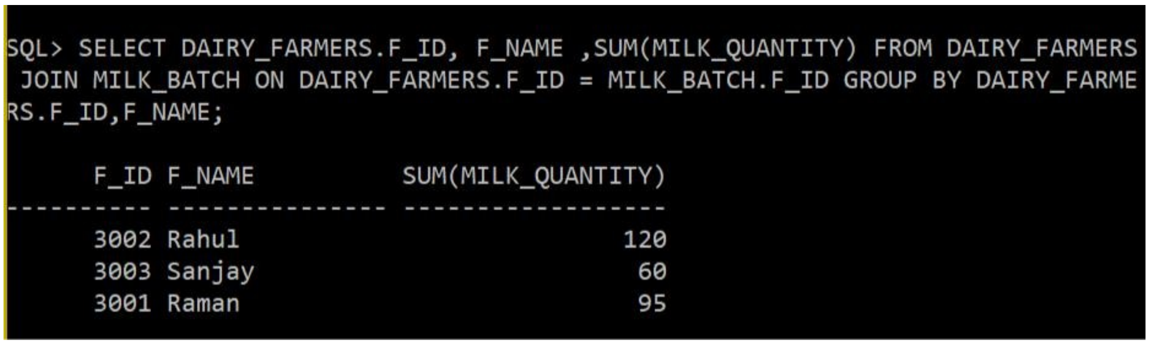
If retailer has problem with his problem he can trace back to collection_centre

SELECT DISTINCT C_ID FROM COLLECTION_CENTRE WHERE PROCESSC_ID IN (SELECT PROCESSC_ID FROM FROM DISTRIBUTORS WHERE DISTRIBUTORS_ID IN (SELECT DISTRIBUTORS_ID FROM RETAILERS WHERE R_ID = 'ID'));


```
SQL> SELECT C_ID FROM COLLECTION_CENTRE WHERE PROCESS_ID IN
  2  (SELECT PROCESSC_ID FROM PRODUCT WHERE DISTRIBUTOR_ID IN
  3  (SELECT DISTRIBUTOR_ID FROM RETAILERS WHERE R_ID = 91294));

      C_ID
----------
     10003
     10001
```

Calculating the sum of milk batch produced by the Dairy farmers

SELECT DAIRY_FARMERS.F_ID, F_NAME ,SUM(MILK_QUANTITY) FROM DAIRY_FARMERS JOIN MILK_BATCH ON DAIRY_FARMERS.F_ID = MILK_BATCH.F_ID GROUP BY DAIRY_FARMERS.F_ID,F_NAME;

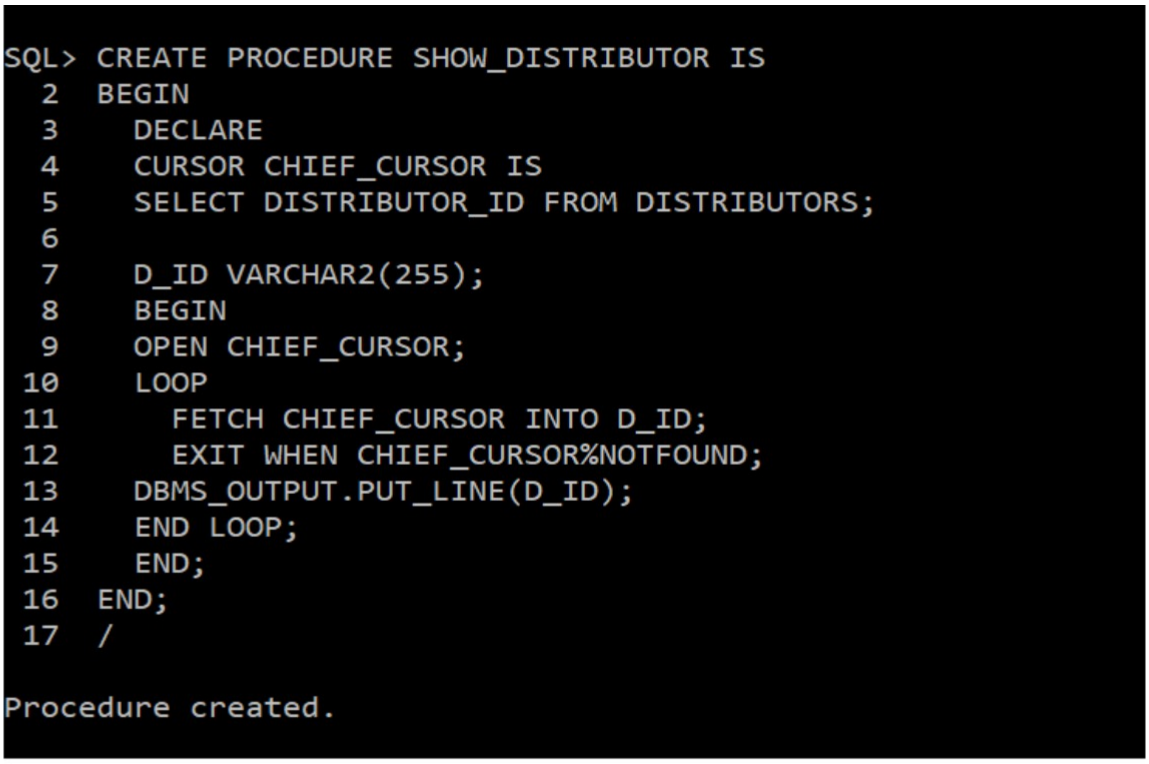


```
SQL> SELECT DAIRY_FARMERS.F_ID, F_NAME ,SUM(MILK_QUANTITY) FROM DAIRY_FARMERS
 JOIN MILK_BATCH ON DAIRY_FARMERS.F_ID = MILK_BATCH.F_ID GROUP BY DAIRY_FARME
RS.F_ID,F_NAME;

      F_ID F_NAME         SUM(MILK_QUANTITY)
---------- -------------- ------------------
      3002 Rahul                         120
      3003 Sanjay                         60
      3001 Raman                          95
```

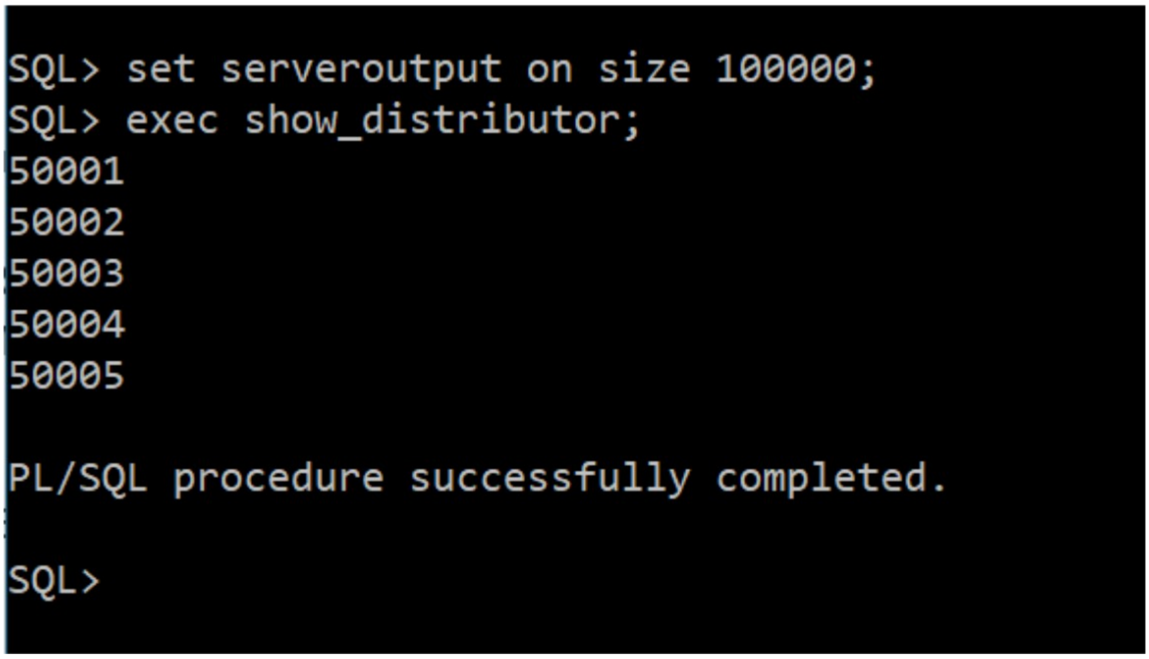
Procedure:

1. Selecting the Distributors id from the set of distributors:

```
CREATE PROCEDURE SHOW_DISTRIBUTOR
IS BEGIN
DECLARE
CURSOR CHIEF_CURSOR IS
SELECT DISTRIBUTOR_ID FROM DISTRIBUTORS;
D_ID VARCHAR2(255);
BEGIN OPEN CHIEF_CURSOR; LOOP FETCH
CHIEF_CURSOR INTO D_ID; EXIT WHEN
CHIEF_CURSOR%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(D_ID); END LOOP;

END;
END;
```



```
SQL> CREATE PROCEDURE SHOW_DISTRIBUTOR IS
     BEGIN
       DECLARE
       CURSOR CHIEF_CURSOR IS
       SELECT DISTRIBUTOR_ID FROM DISTRIBUTORS;

       D_ID VARCHAR2(255);
       BEGIN
       OPEN CHIEF_CURSOR;
       LOOP
         FETCH CHIEF_CURSOR INTO D_ID;
         EXIT WHEN CHIEF_CURSOR%NOTFOUND;
       DBMS_OUTPUT.PUT_LINE(D_ID);
       END LOOP;
       END;
     END;
     /

Procedure created.
```



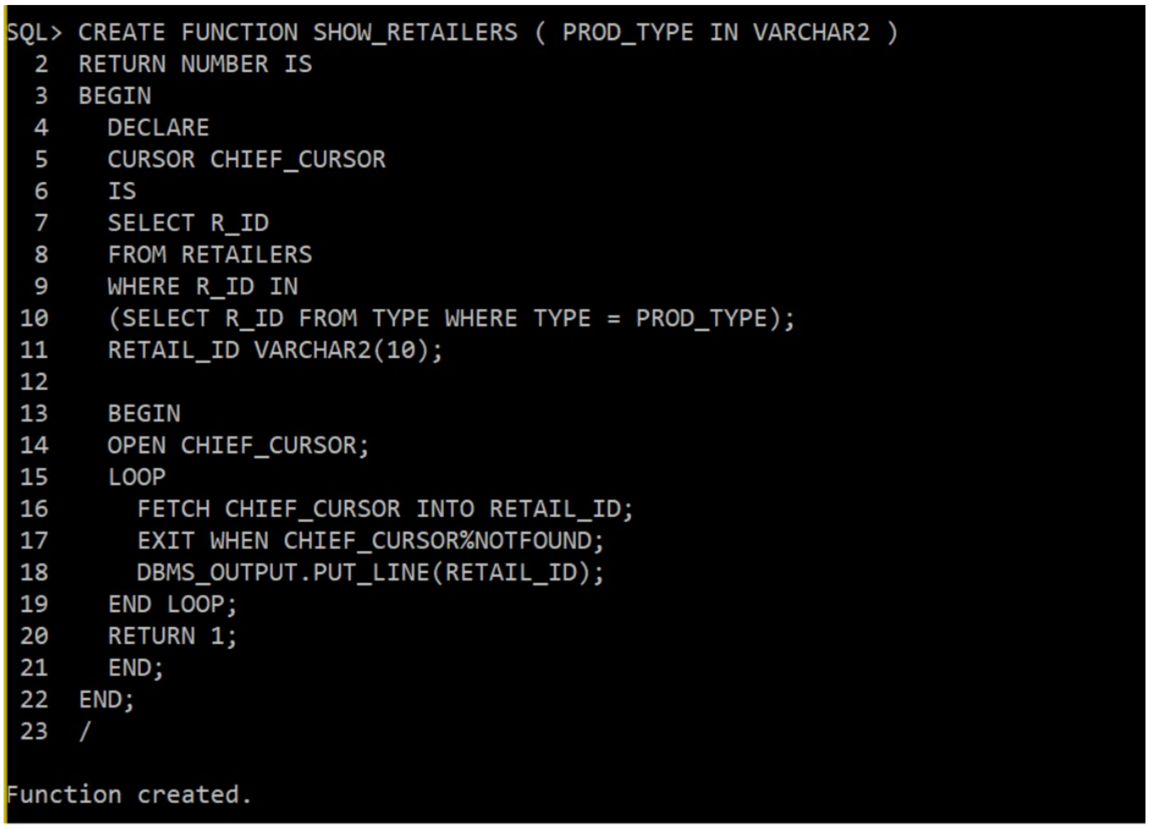
```
SQL> set serveroutput on size 100000;
SQL> exec show_distributor;
50001
50002
50003
50004
50005

PL/SQL procedure successfully completed.

SQL>
```

FUNCTION:

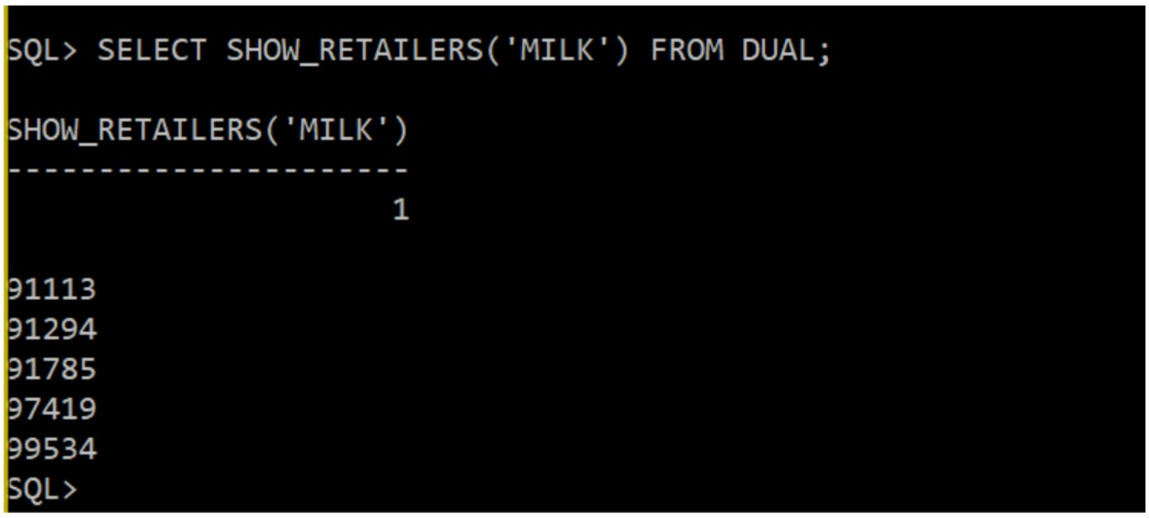
Given a Product type we can find the all the retailers who can sell them:

CREATE FUNCTION SHOW_RETAILERS (PROD_TYPE IN VARCHAR2) RETURN NUMBER IS BEGIN DECLARE CURSOR CHIEF_CURSOR IS SELECT R_ID FROM RETAILERS WHERE R_ID IN (SELECT R_ID FROM

TYPE WHERE TYPE = PROD_TYPE); RETAIL_ID VARCHAR2(10);   BEGIN OPEN CHIEF_CURSOR; LOOP FETCH CHIEF_CURSOR INTO RETAIL_ID;   EXIT WHEN CHIEF_CURSOR%NOTFOUND; DBMS_OUTPUT.PUT_LINE(RETAIL_ID); END LOOP; RETURN 1; END; END;



```
SQL> CREATE FUNCTION SHOW_RETAILERS ( PROD_TYPE IN VARCHAR2 )
RETURN NUMBER IS
BEGIN
  DECLARE
  CURSOR CHIEF_CURSOR
  IS
  SELECT R_ID
  FROM RETAILERS
  WHERE R_ID IN
  (SELECT R_ID FROM TYPE WHERE TYPE = PROD_TYPE);
  RETAIL_ID VARCHAR2(10);

  BEGIN
  OPEN CHIEF_CURSOR;
  LOOP
    FETCH CHIEF_CURSOR INTO RETAIL_ID;
    EXIT WHEN CHIEF_CURSOR%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE(RETAIL_ID);
  END LOOP;
  RETURN 1;
  END;
END;
/

Function created.
```



```
SQL> SELECT SHOW_RETAILERS('MILK') FROM DUAL;

SHOW_RETAILERS('MILK')
----------------------
                     1

91113
91294
91785
97419
99534
SQL>
```

BUSSINESS RULES

To account for different types of policies and decisions, business rules can be modelled in multiple ways. Two common types of business rules are formula rules and decision table rules.

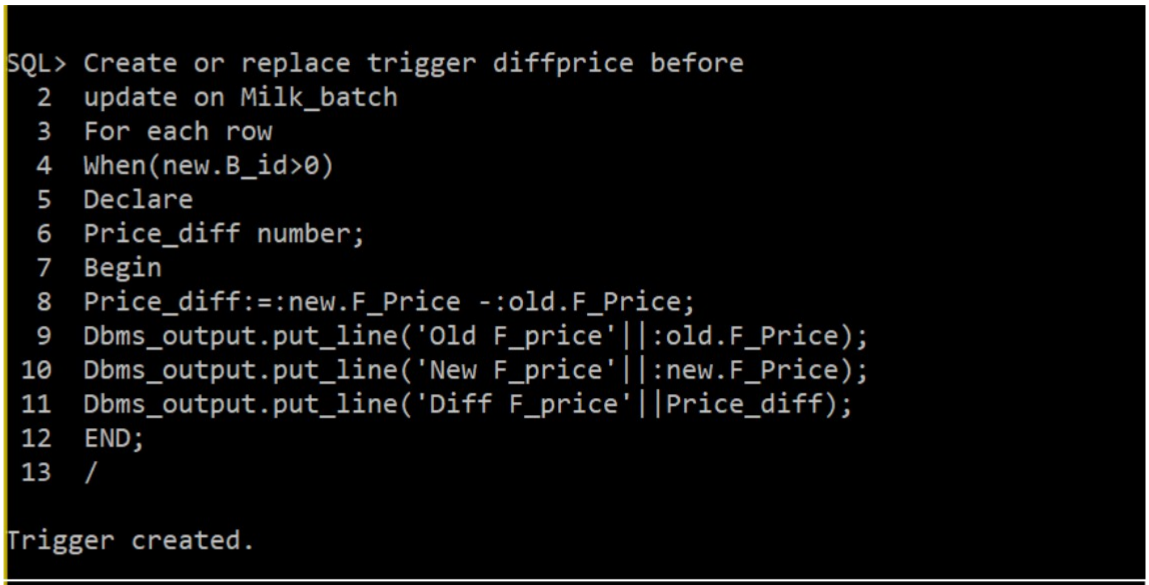
A **formula rule** allows employees to maintain calculations in a no-code format, similar to creating formulas in Microsoft Excel. Once a formula is defined, it can be reused as appropriate in multiple process designs. If the formula needs to be updated, only the formula itself needs to be changed without requiring an end user to manipulate code or individually adjust each applicable process. Many standard formulas are already built into the software, such as determining an average, sum, date, and maximum, among many others.

A **decision table rule** is a powerful feature that lets non-developers represent related conditional decisions or "if-then" logic in a concise manner as a spreadsheet style-table. Decision tables use columns as the conditions, while rows specify the appropriate outcomes. Approvals, application acceptance criteria, and loan eligibility checks are all general examples where decision tables can be applied and owned by the domain experts themselves. In traditional approaches, these decisions can be hard-coded directly as part of process designs, leading to complex implementations that require developers to make manual updates as they arise.

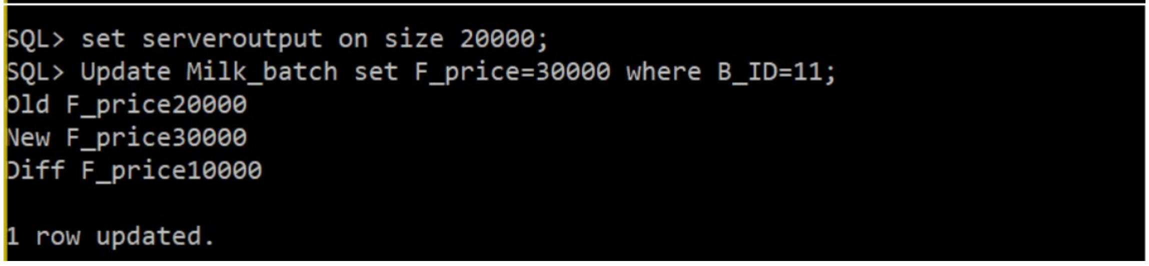
1) Generating Difference in price on the Milk batch.

Using Trigger:

```
Set serveroutput on;

Create or replace trigger diffprice before
Insert or update or delete on Milk_batch
For each row
When(new.B_id>0)
Declare
Price_diff number;
Begin
Price_diff:=:new.F_price -:old.salary;
Dbms_output.put_line(‘Old F_price’||:old.F_price);
Dbms_output.put_line(‘New F_price’||:new.F_price);
Dbms_output.put_line(‘Diff F_price’||Price_diff);
END;
```



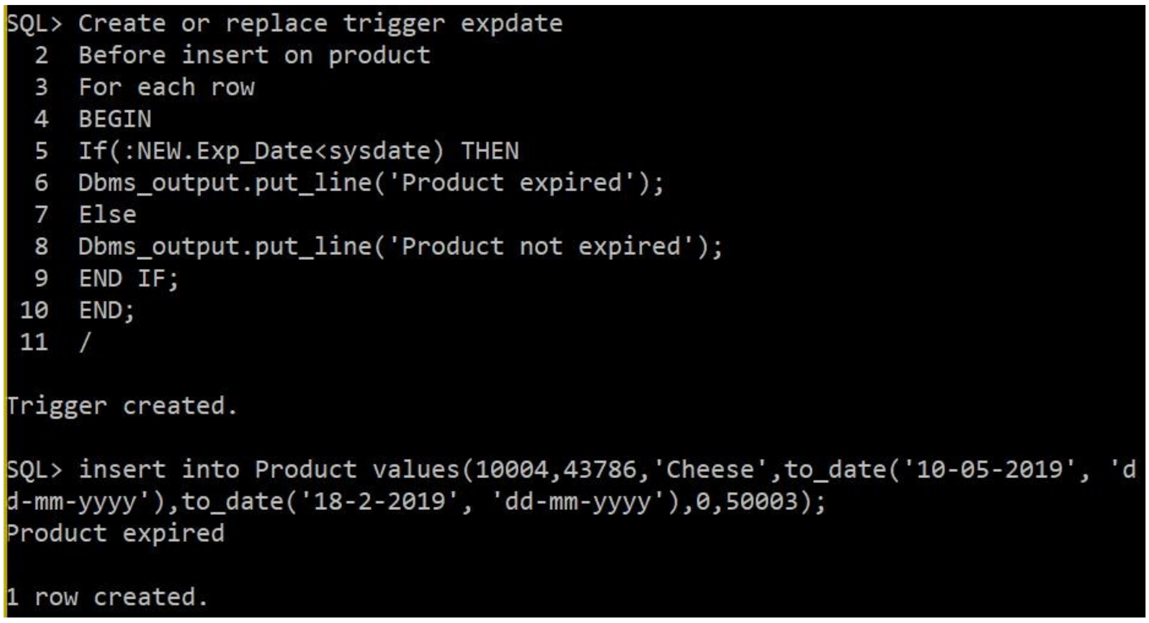
```
SQL> Create or replace trigger diffprice before
update on Milk_batch
For each row
When(new.B_id>0)
Declare
Price_diff number;
Begin
Price_diff:=:new.F_Price -:old.F_Price;
Dbms_output.put_line('Old F_price'||:old.F_Price);
Dbms_output.put_line('New F_price'||:new.F_Price);
Dbms_output.put_line('Diff F_price'||Price_diff);
END;
/

Trigger created.
```



```
SQL> set serveroutput on size 20000;
SQL> Update Milk_batch set F_price=30000 where B_ID=11;
Old F_price20000
New F_price30000
Diff F_price10000

1 row updated.
```

2)Checking when the product is expired and printing

```
Create or replace trigger expdate
Before insert on product
For each row
Declare
If(:NEW.Exp_Date<sysdate) then
Dbms_output.put_line('Product
expired'); Else
Dbms_output.put_line('Product not expired');
END If
END;
```



```
SQL> Create or replace trigger expdate
  2  Before insert on product
  3  For each row
  4  BEGIN
  5  If(:NEW.Exp_Date<sysdate) THEN
  6  Dbms_output.put_line('Product expired');
  7  Else
  8  Dbms_output.put_line('Product not expired');
  9  END IF;
 10  END;
 11  /

Trigger created.

SQL> insert into Product values(10004,43786,'Cheese',to_date('10-05-2019', 'd
d-mm-yyyy'),to_date('18-2-2019', 'dd-mm-yyyy'),0,50003);
Product expired

1 row created.
```