DATABASE MANAGEMENT SYSTEMS

(10211CS207)

TASK:12

VEHICLE SERVICE ,MAINTAINENCE AND TRACKER

Team Details:

Team members:

R.VENKATA KARTHIK	VTU29180
N.VENKATA YOGANADH	VTU29183
M.HEMA HARSHITHA	VTU29208
L.VUDHVI SAI	VTU29280
N.VISHNU VARDHAN	VTU29355
SK.VAHEED BASHA	VTU29379

AIM:

To develop a microproject on Vehicle Service Maintainence and Tracker System.

1.ER Diagram:

AIM:

To implement a conceptual design through FTR for Vehicle Service Maintainence and Tracker System.

PROCEDURE:

Identifying entites:

1. VEHICLE

2.OWNER

3. SERVICE_RECORD

4. MAINTENANCE

Identifying attributes:

VEHICLE-Owner_ID,Name,Email,Phone_Number,Address,City,State,Registration_Date

OWNER: Owner_ID, First_Name, Last_Name, Email, Phone_Number,

SERVICE_RECORD: Service_ID, Vehicle_ID, Service_Date, Service_Type, Mileage_At_Service, Next_Service_Date, Next_Service_Mileage, Total_Cost, Mechanic_ID, Status

MAINTENANCE: Maintenance_ID, Vehicle_ID, Maintenance_Type, Scheduled_Date, Completion_Date, Service_Provider, Cost, Status

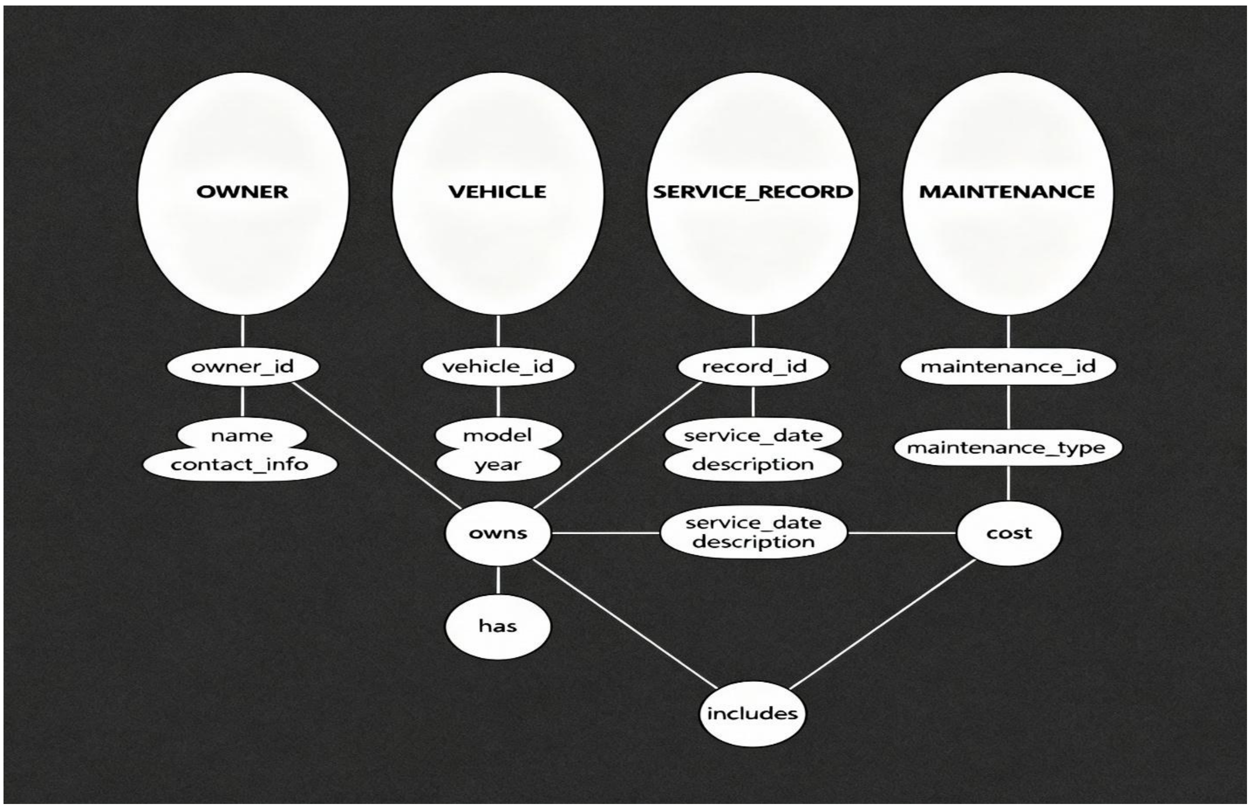
Identifying relationships:

- OWNER to VEHICLE
- VEHICLE to SERVICE_RECORD
- VEHICLE to MAINTENANCE
- SERVICE_RECORD to MAINTENANCE

Identifying the cardinality:

- OWNER (1) — (M) VEHICLE
- VEHICLE (1) — (M) SERVICE_RECORD
- VEHICLE (1) — (M) MAINTENANCE
- SERVICE_RECORD (1) — (0..M) MAINTENANCE

ER DIAGRAM:



Result: Implementation of conceptual design through FTR for Vehicle Service Maintainence and Tracker System is successfully executed.

02. SQL Queries & Relational operations:

Aim: To execute relational operationms ,sql,aggregate,join &nested queries for Vehicle Service Maintainence and Tracker System.

DDL commands:

1.create :

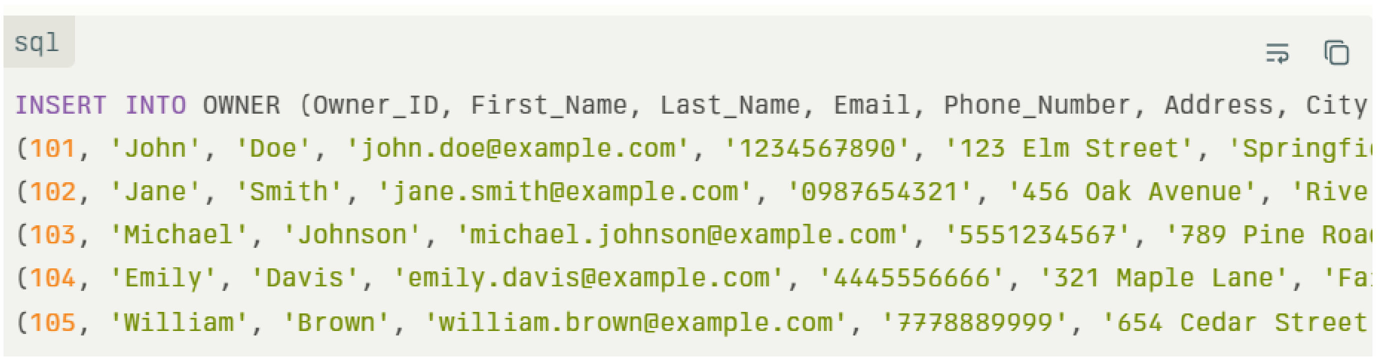
```sql
CREATE TABLE OWNER (
    Owner_ID INT PRIMARY KEY,
    First_Name VARCHAR(50),
    Last_Name VARCHAR(50),
    Email VARCHAR(100),
    Phone_Number VARCHAR(15),
    Address VARCHAR(255),
    City VARCHAR(50),
    State VARCHAR(50),
    Registration_Date DATE
);
```

2.alter:

```sql
ALTER TABLE OWNER
ADD Middle_Name VARCHAR(50);
```

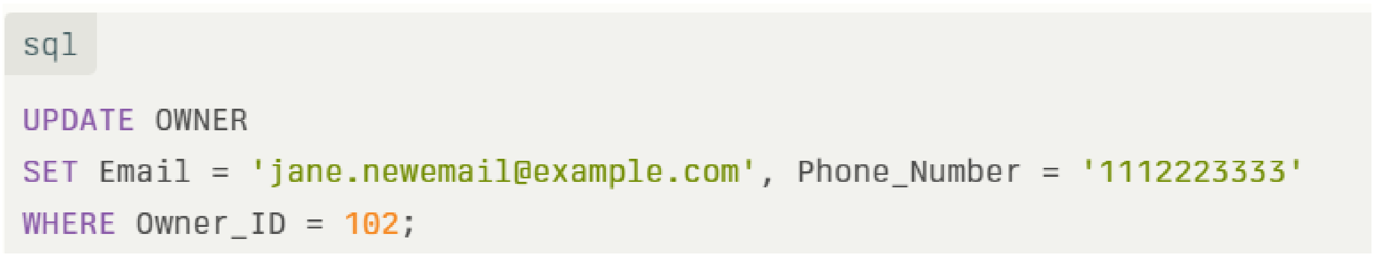
DML commands:

1.insert:



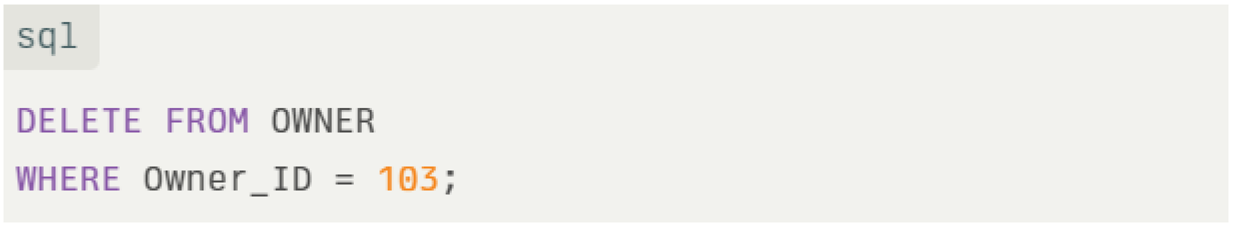
```sql
INSERT INTO OWNER (Owner_ID, First_Name, Last_Name, Email, Phone_Number, Address, City
(101, 'John', 'Doe', 'john.doe@example.com', '1234567890', '123 Elm Street', 'Springfi
(102, 'Jane', 'Smith', 'jane.smith@example.com', '0987654321', '456 Oak Avenue', 'Rive
(103, 'Michael', 'Johnson', 'michael.johnson@example.com', '5551234567', '789 Pine Roa
(104, 'Emily', 'Davis', 'emily.davis@example.com', '4445556666', '321 Maple Lane', 'Fa
(105, 'William', 'Brown', 'william.brown@example.com', '7778889999', '654 Cedar Street
```

2.update:



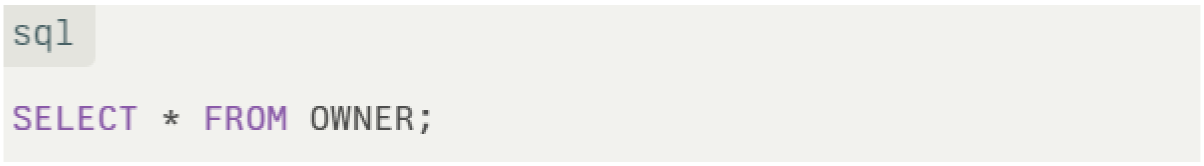
```sql
UPDATE OWNER
SET Email = 'jane.newemail@example.com', Phone_Number = '1112223333'
WHERE Owner_ID = 102;
```

3.delete:

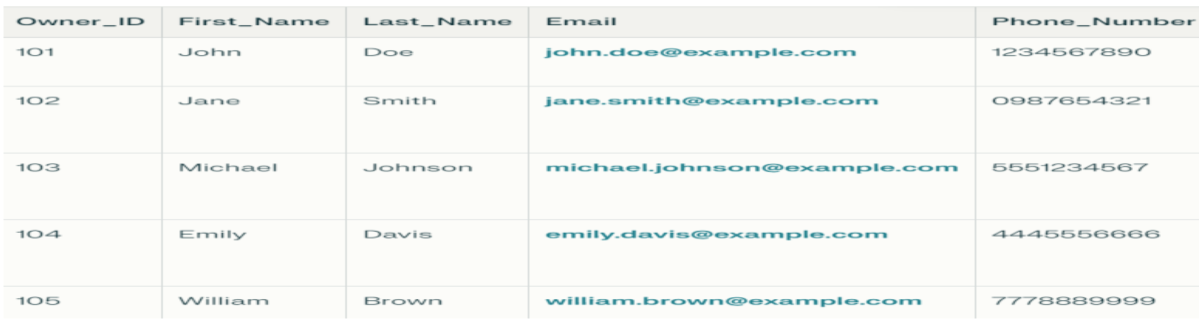


```sql
DELETE FROM OWNER
WHERE Owner_ID = 103;
```

DQL commands:

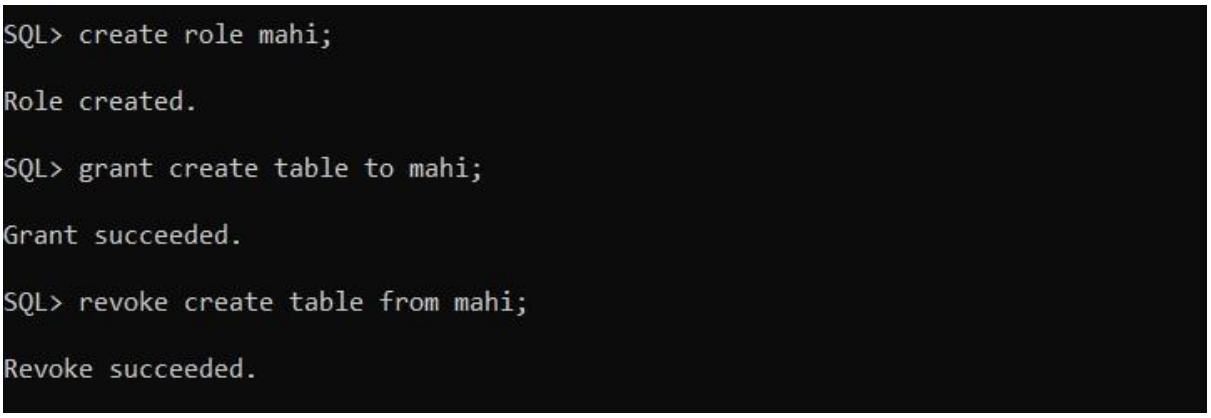


```sql
SELECT * FROM OWNER;
```



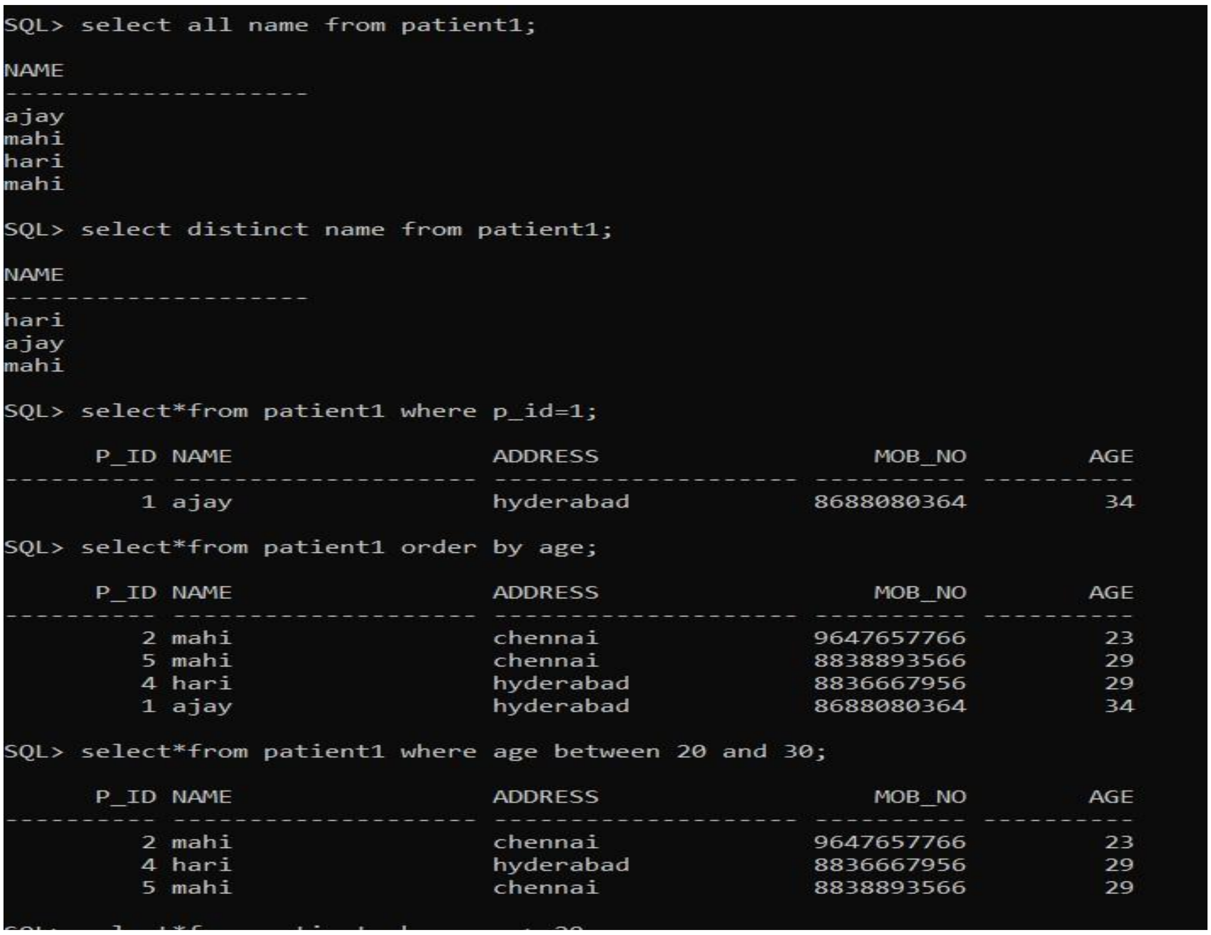
Owner_ID	First_Name	Last_Name	Email	Phone_Number
101	John	Doe	john.doe@example.com	1234567890
102	Jane	Smith	jane.smith@example.com	0987654321
103	Michael	Johnson	michael.johnson@example.com	5551234567
104	Emily	Davis	emily.davis@example.com	4445556666
105	William	Brown	william.brown@example.com	7778889999

DCL Commands:



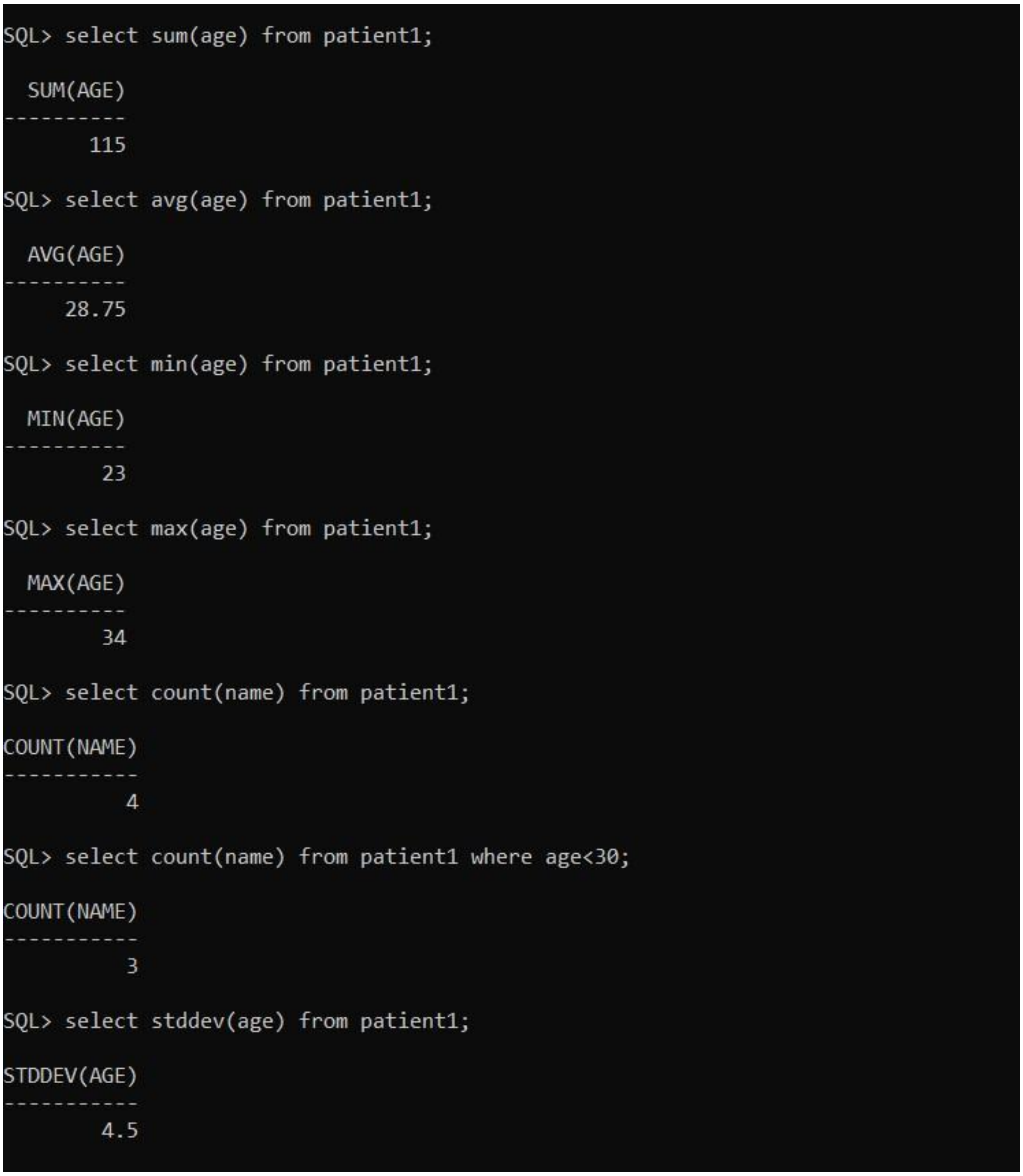
```
SQL> create role mahi;

Role created.

SQL> grant create table to mahi;

Grant succeeded.

SQL> revoke create table from mahi;

Revoke succeeded.
```

DML single row functions and operators:



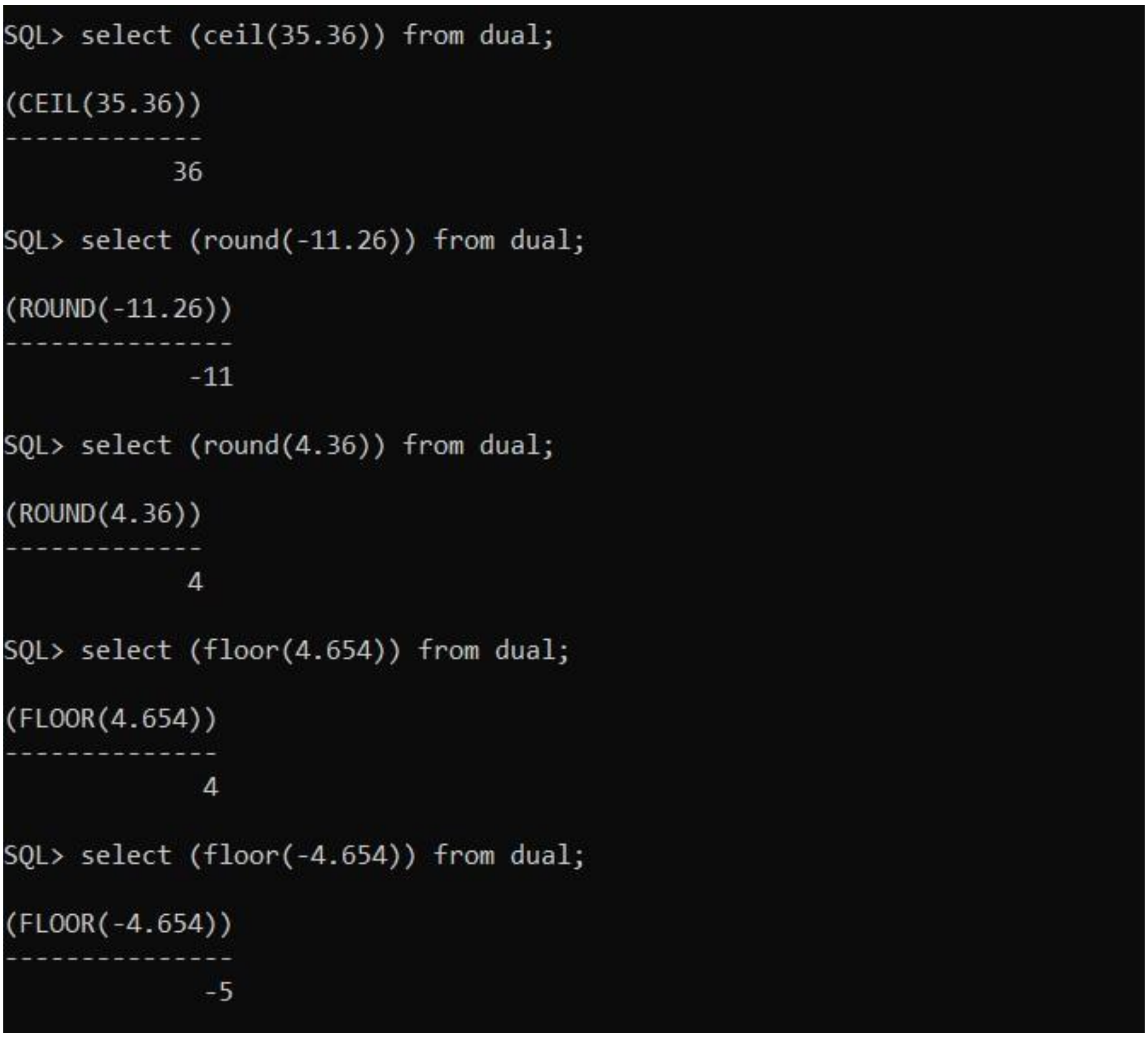
```
SQL> select all name from patient1;

NAME
--------------------
ajay
mahi
hari
mahi

SQL> select distinct name from patient1;

NAME
--------------------
hari
ajay
mahi

SQL> select*from patient1 where p_id=1;

      P_ID NAME                 ADDRESS                  MOB_NO        AGE
---------- -------------------- -------------------- ---------- ----------
         1 ajay                 hyderabad            8688080364         34

SQL> select*from patient1 order by age;

      P_ID NAME                 ADDRESS                  MOB_NO        AGE
---------- -------------------- -------------------- ---------- ----------
         2 mahi                 chennai              9647657766         23
         5 mahi                 chennai              8838893566         29
         4 hari                 hyderabad            8836667956         29
         1 ajay                 hyderabad            8688080364         34

SQL> select*from patient1 where age between 20 and 30;

      P_ID NAME                 ADDRESS                  MOB_NO        AGE
---------- -------------------- -------------------- ---------- ----------
         2 mahi                 chennai              9647657766         23
         4 hari                 hyderabad            8836667956         29
         5 mahi                 chennai              8838893566         29
```

Aggregate functions:



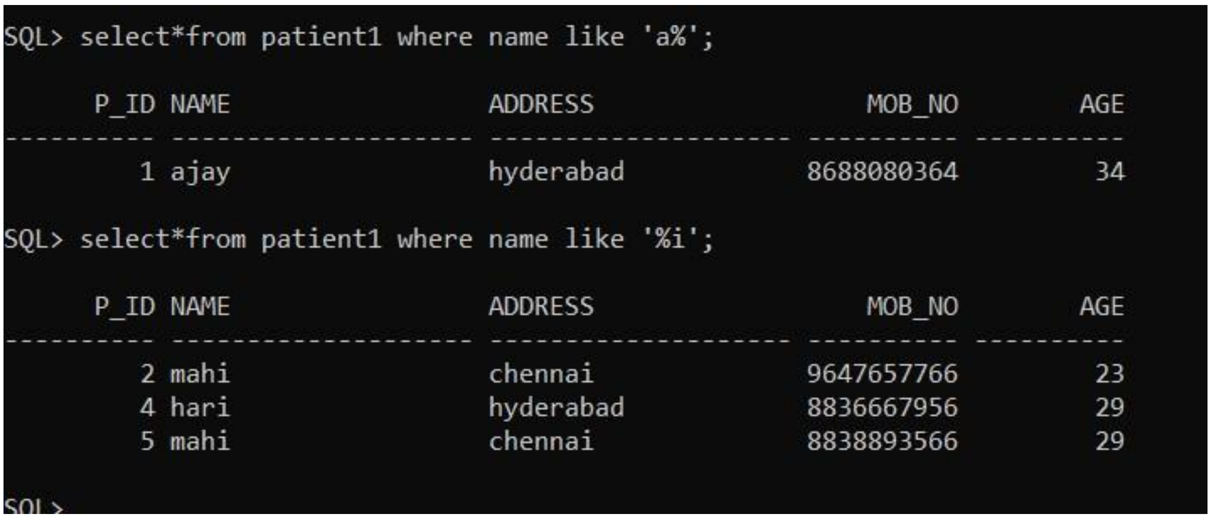
```
SQL> select sum(age) from patient1;

  SUM(AGE)
----------
       115

SQL> select avg(age) from patient1;

  AVG(AGE)
----------
     28.75

SQL> select min(age) from patient1;

  MIN(AGE)
----------
        23

SQL> select max(age) from patient1;

  MAX(AGE)
----------
        34

SQL> select count(name) from patient1;

COUNT(NAME)
-----------
          4

SQL> select count(name) from patient1 where age<30;

COUNT(NAME)
-----------
          3

SQL> select stddev(age) from patient1;

STDDEV(AGE)
-----------
        4.5
```

Mathematical functions:

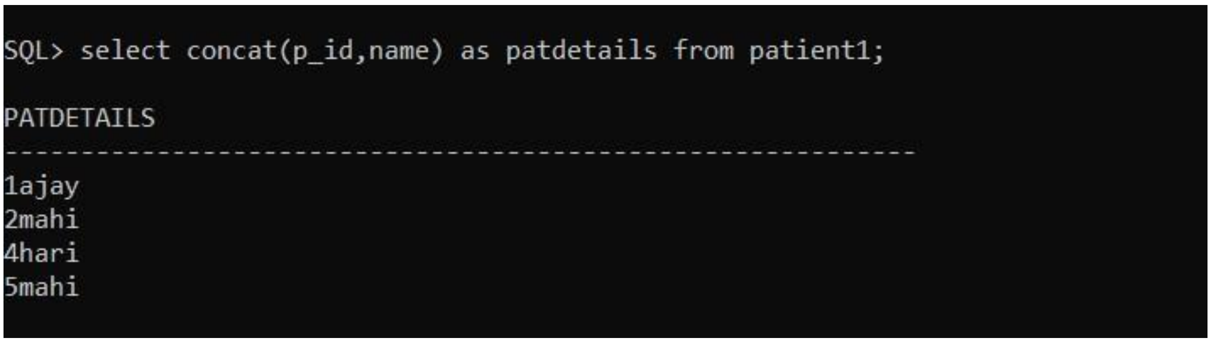


```
SQL> select (ceil(35.36)) from dual;

(CEIL(35.36))
-------------
           36

SQL> select (round(-11.26)) from dual;

(ROUND(-11.26))
---------------
            -11

SQL> select (round(4.36)) from dual;

(ROUND(4.36))
-------------
            4

SQL> select (floor(4.654)) from dual;

(FLOOR(4.654))
--------------
             4

SQL> select (floor(-4.654)) from dual;

(FLOOR(-4.654))
---------------
             -5
```

String functions:

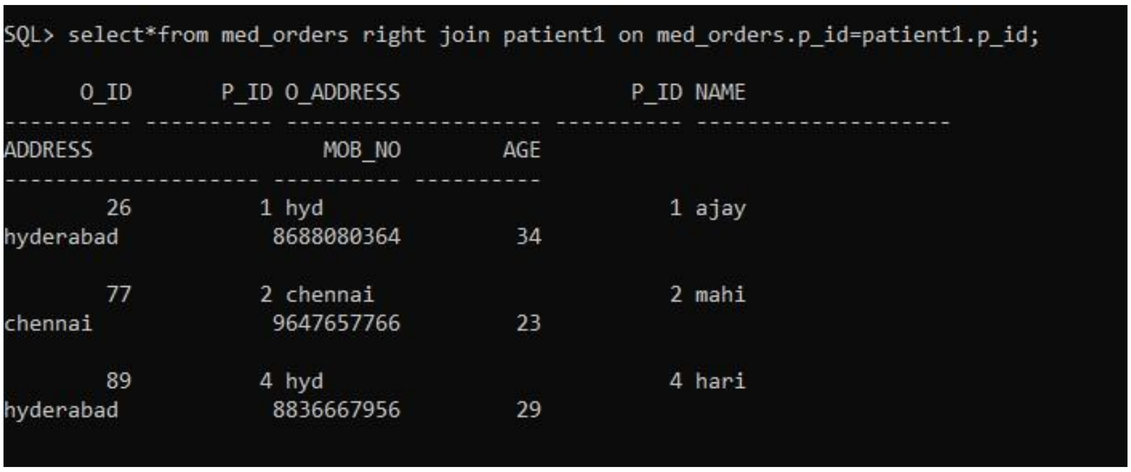


```
SQL> select*from patient1 where name like 'a%';

      P_ID NAME                 ADDRESS                  MOB_NO        AGE
---------- -------------------- -------------------- ---------- ----------
         1 ajay                 hyderabad            8688080364         34

SQL> select*from patient1 where name like '%i';

      P_ID NAME                 ADDRESS                  MOB_NO        AGE
---------- -------------------- -------------------- ---------- ----------
         2 mahi                 chennai              9647657766         23
         4 hari                 hyderabad            8836667956         29
         5 mahi                 chennai              8838893566         29

SQL>
```

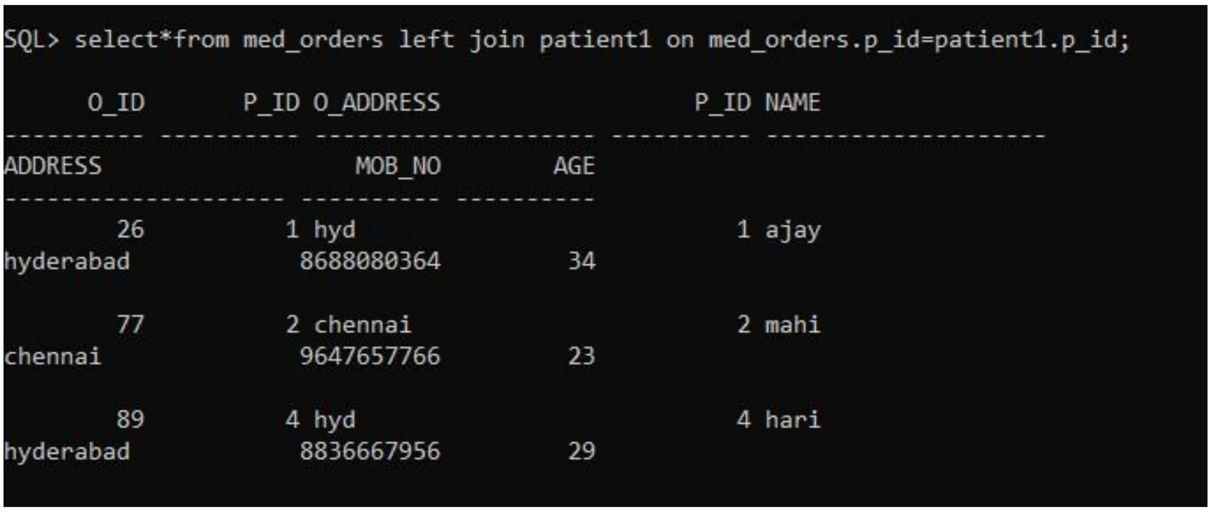

```
SQL> select concat(p_id,name) as patdetails from patient1;

PATDETAILS
------------------------------------------------------------
1ajay
2mahi
4hari
5mahi
```

SQL Joins:

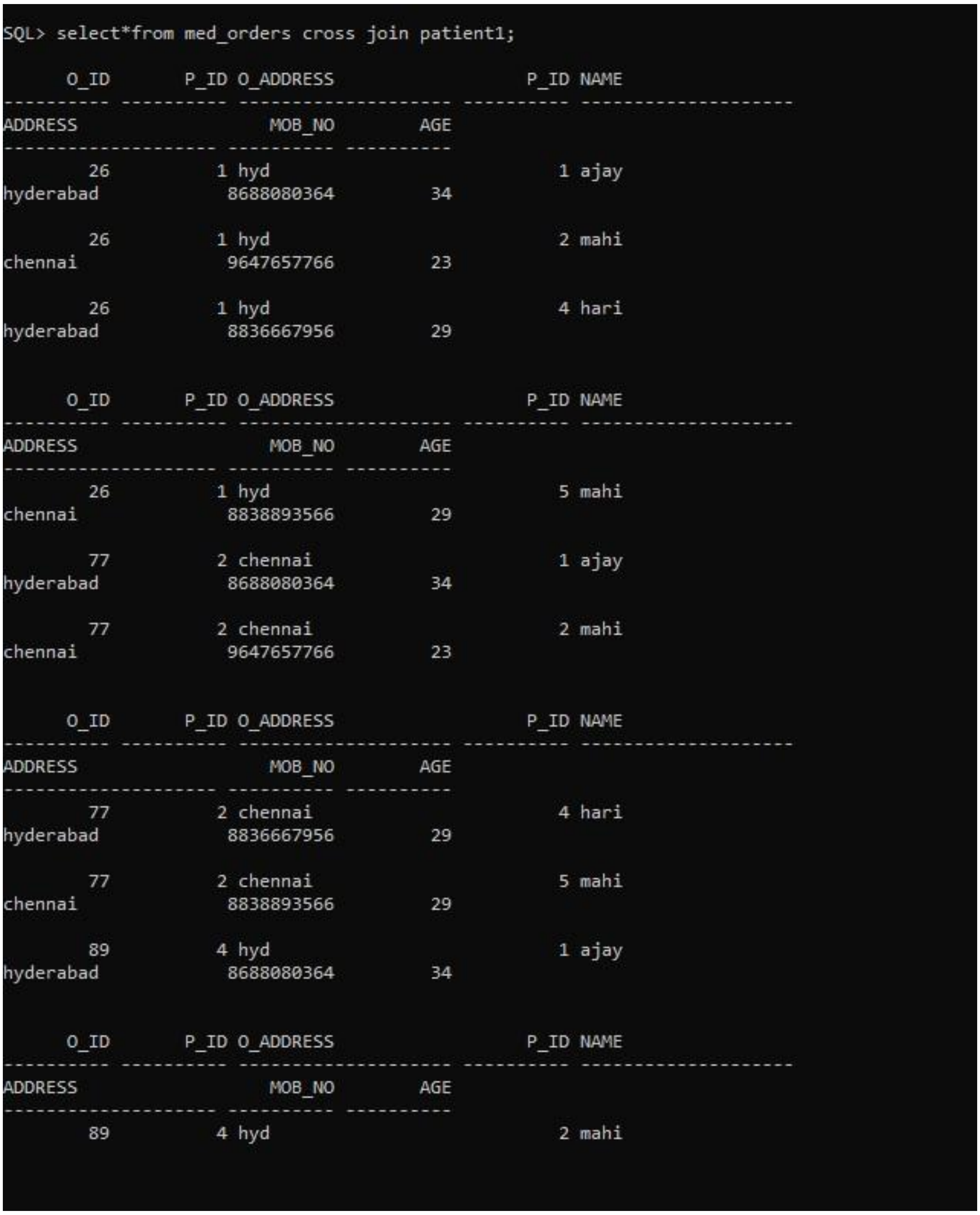
1.right join:


```
SQL> select*from med_orders right join patient1 on med_orders.p_id=patient1.p_id;

      O_ID       P_ID O_ADDRESS                  P_ID NAME
---------- ---------- -------------------- ---------- --------------------
ADDRESS                  MOB_NO        AGE
-------------------- ---------- ----------
        26          1 hyd                           1 ajay
hyderabad             8688080364         34

        77          2 chennai                       2 mahi
chennai               9647657766         23

        89          4 hyd                           4 hari
hyderabad             8836667956         29
```

2.left join:

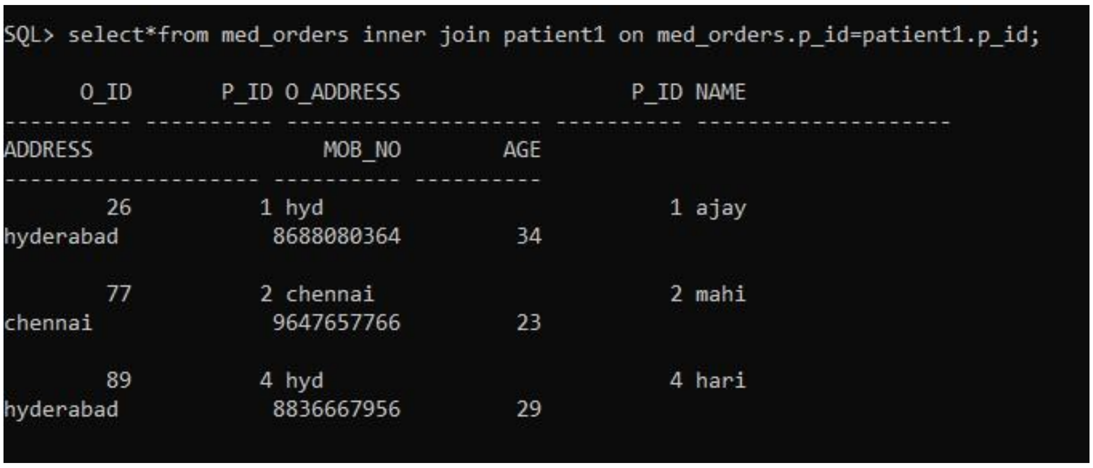

```
SQL> select*from med_orders left join patient1 on med_orders.p_id=patient1.p_id;

      O_ID       P_ID O_ADDRESS                  P_ID NAME
---------- ---------- -------------------- ---------- --------------------
ADDRESS                  MOB_NO        AGE
-------------------- ---------- ----------
        26          1 hyd                           1 ajay
hyderabad             8688080364         34

        77          2 chennai                       2 mahi
chennai               9647657766         23

        89          4 hyd                           4 hari
hyderabad             8836667956         29
```

3.cross join:



```
SQL> select*from med_orders cross join patient1;

      O_ID       P_ID O_ADDRESS                  P_ID NAME
---------- ---------- -------------------- ---------- --------------------
ADDRESS                  MOB_NO        AGE
-------------------- ---------- ----------
        26          1 hyd                           1 ajay
hyderabad             8688080364         34

        26          1 hyd                           2 mahi
chennai               9647657766         23

        26          1 hyd                           4 hari
hyderabad             8836667956         29


      O_ID       P_ID O_ADDRESS                  P_ID NAME
---------- ---------- -------------------- ---------- --------------------
ADDRESS                  MOB_NO        AGE
-------------------- ---------- ----------
        26          1 hyd                           5 mahi
chennai               8838893566         29

        77          2 chennai                       1 ajay
hyderabad             8688080364         34

        77          2 chennai                       2 mahi
chennai               9647657766         23


      O_ID       P_ID O_ADDRESS                  P_ID NAME
---------- ---------- -------------------- ---------- --------------------
ADDRESS                  MOB_NO        AGE
-------------------- ---------- ----------
        77          2 chennai                       4 hari
hyderabad             8836667956         29

        77          2 chennai                       5 mahi
chennai               8838893566         29

        89          4 hyd                           1 ajay
hyderabad             8688080364         34


      O_ID       P_ID O_ADDRESS                  P_ID NAME
---------- ---------- -------------------- ---------- --------------------
ADDRESS                  MOB_NO        AGE
-------------------- ---------- ----------
        89          4 hyd                           2 mahi
```

4.inner join:

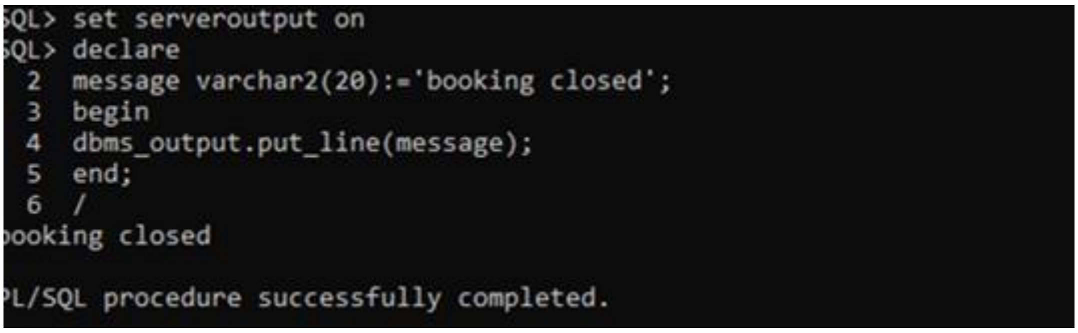


```
SQL> select*from med_orders inner join patient1 on med_orders.p_id=patient1.p_id;

      O_ID       P_ID O_ADDRESS                  P_ID NAME
---------- ---------- -------------------- ---------- --------------------
ADDRESS                  MOB_NO        AGE
-------------------- ---------- ----------
        26          1 hyd                           1 ajay
hyderabad             8688080364         34

        77          2 chennai                       2 mahi
chennai               9647657766         23

        89          4 hyd                           4 hari
hyderabad             8836667956         29
```

PL/SQL

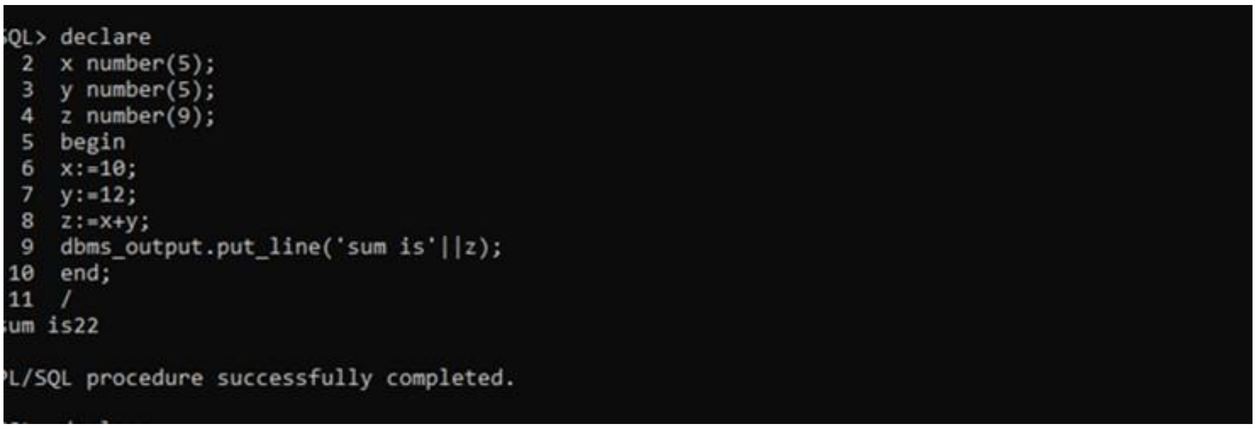
Procedure:

Sample program for printing a sentence:



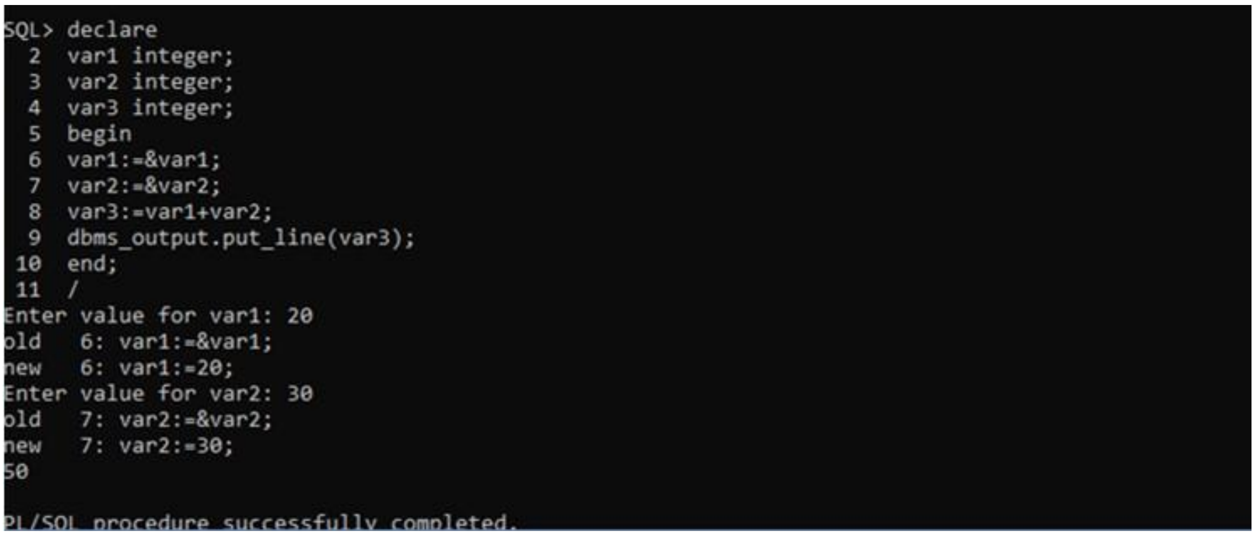
```
SQL> set serveroutput on
SQL> declare
  2  message varchar2(20):='booking closed';
  3  begin
  4  dbms_output.put_line(message);
  5  end;
  6  /
booking closed

PL/SQL procedure successfully completed.
```

Static input:



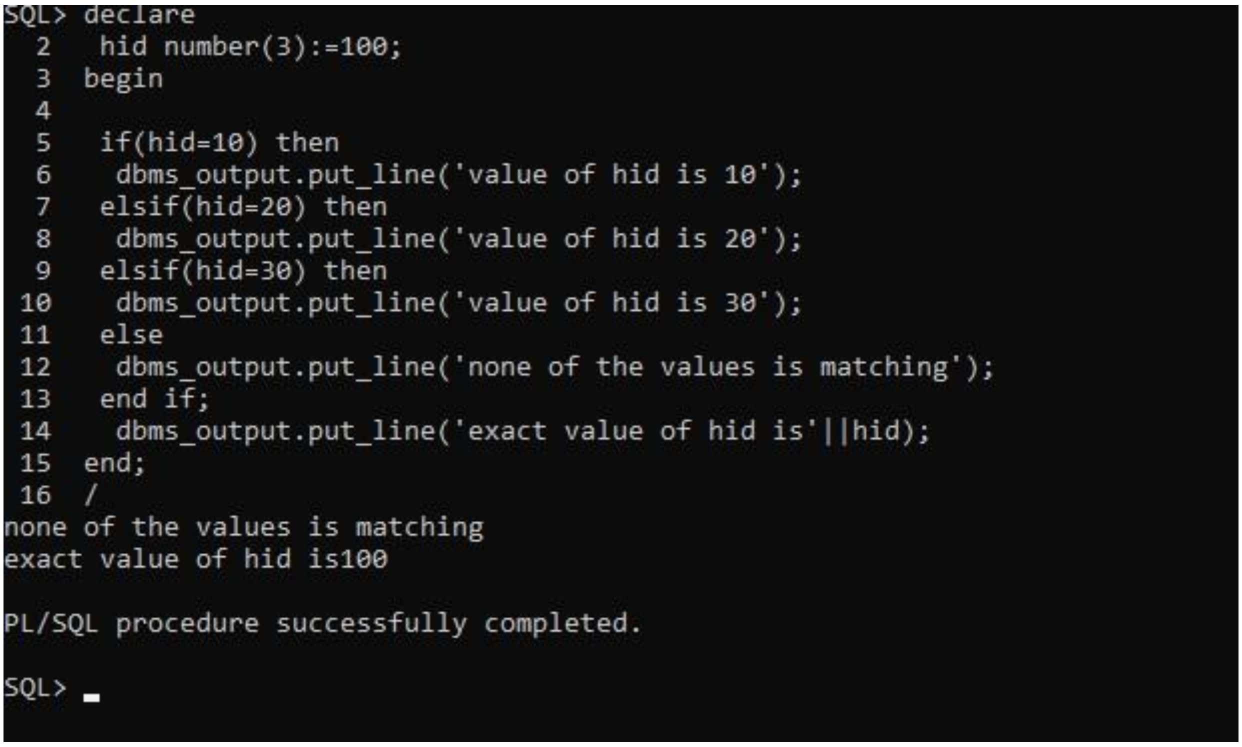
```
SQL> declare
  2  x number(5);
  3  y number(5);
  4  z number(9);
  5  begin
  6  x:=10;
  7  y:=12;
  8  z:=x+y;
  9  dbms_output.put_line('sum is'||z);
 10  end;
 11  /
sum is22

PL/SQL procedure successfully completed.
```

Dynamic input:

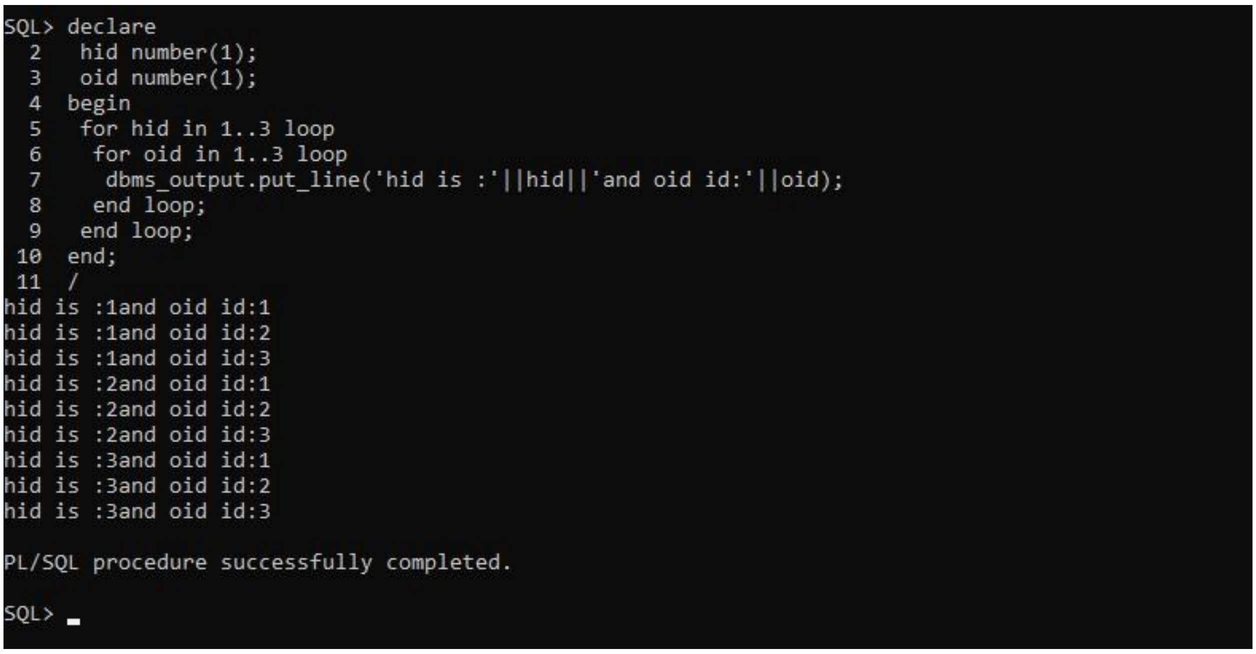


```
SQL> declare
  2  var1 integer;
  3  var2 integer;
  4  var3 integer;
  5  begin
  6  var1:=&var1;
  7  var2:=&var2;
  8  var3:=var1+var2;
  9  dbms_output.put_line(var3);
 10  end;
 11  /
Enter value for var1: 20
old   6: var1:=&var1;
new   6: var1:=20;
Enter value for var2: 30
old   7: var2:=&var2;
new   7: var2:=30;
50

PL/SQL procedure successfully completed.
```

Sample program for loops:

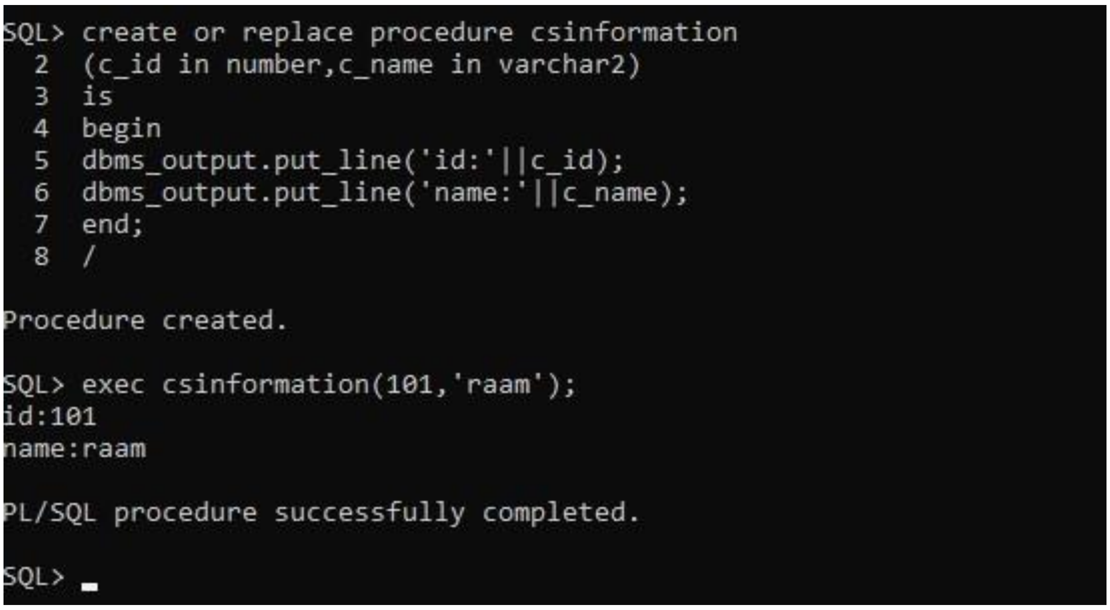


```
SQL> declare
  2   hid number(3):=100;
  3  begin
  4
  5   if(hid=10) then
  6    dbms_output.put_line('value of hid is 10');
  7   elsif(hid=20) then
  8    dbms_output.put_line('value of hid is 20');
  9   elsif(hid=30) then
 10    dbms_output.put_line('value of hid is 30');
 11   else
 12    dbms_output.put_line('none of the values is matching');
 13   end if;
 14    dbms_output.put_line('exact value of hid is'||hid);
 15  end;
 16  /
none of the values is matching
exact value of hid is100

PL/SQL procedure successfully completed.

SQL>
```



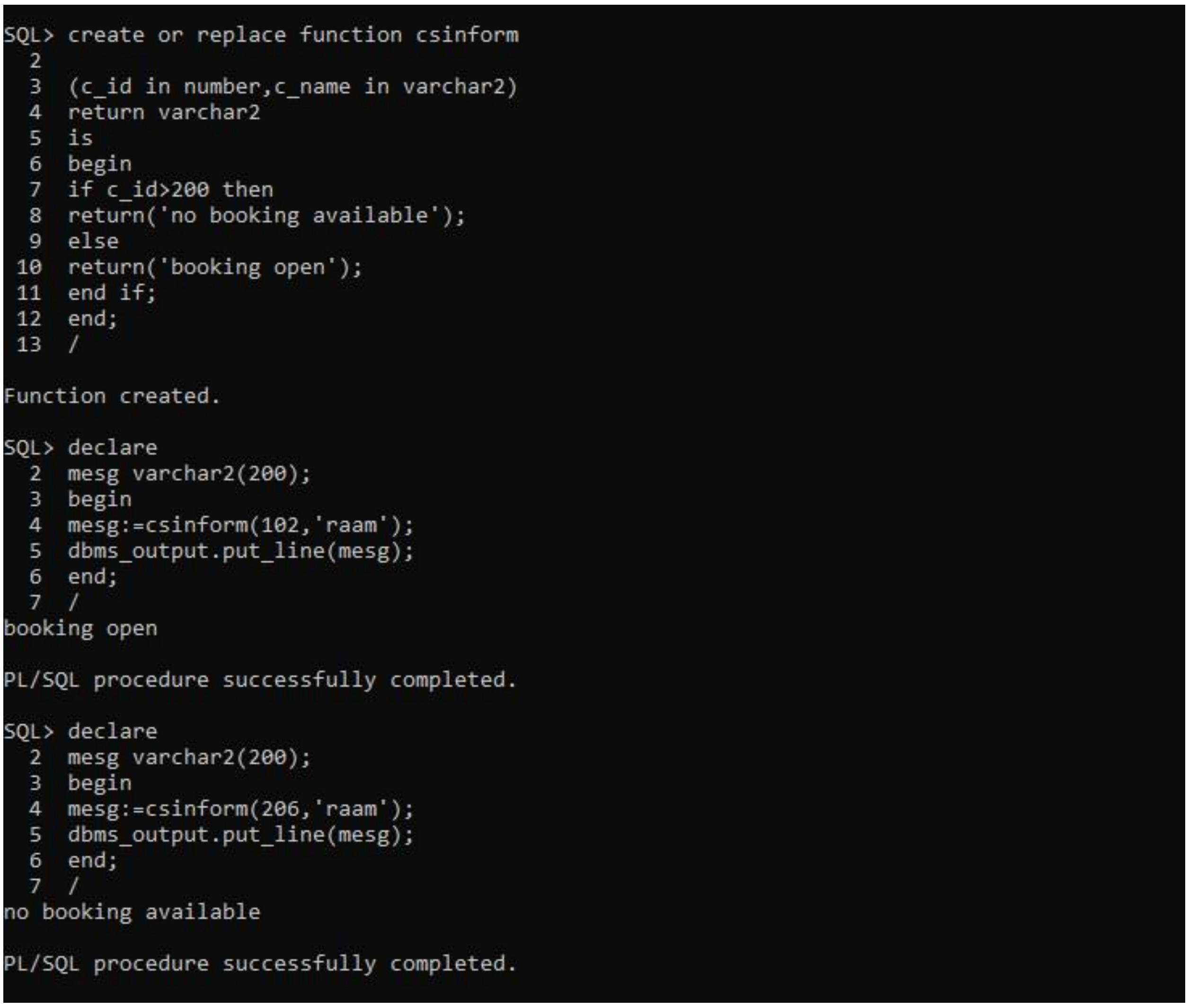
```
SQL> declare
   hid number(1);
   oid number(1);
  begin
   for hid in 1..3 loop
    for oid in 1..3 loop
     dbms_output.put_line('hid is :'||hid||'and oid id:'||oid);
    end loop;
   end loop;
  end;
  /
hid is :1and oid id:1
hid is :1and oid id:2
hid is :1and oid id:3
hid is :2and oid id:1
hid is :2and oid id:2
hid is :2and oid id:3
hid is :3and oid id:1
hid is :3and oid id:2
hid is :3and oid id:3

PL/SQL procedure successfully completed.

SQL>
```

Sample program for only procedure:



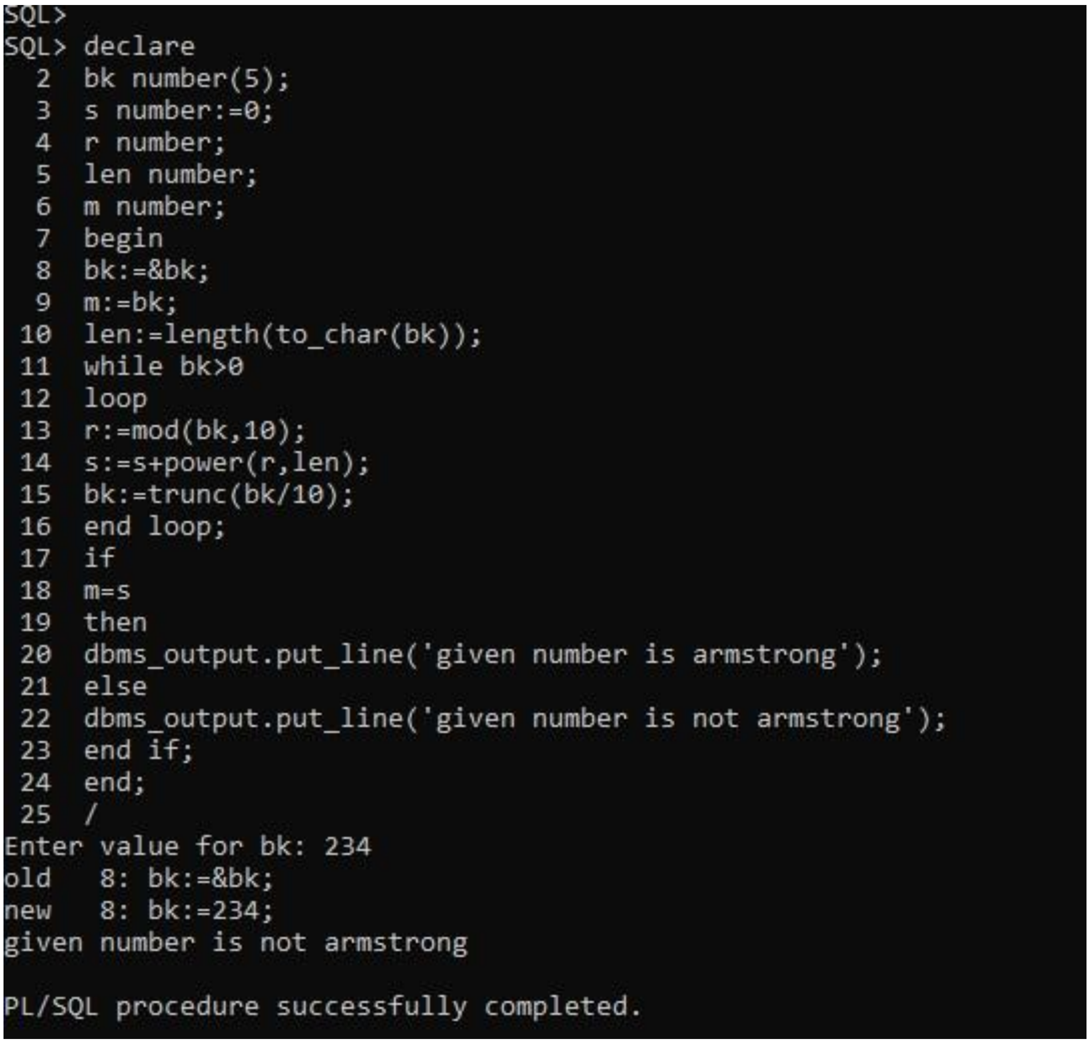
```
SQL> create or replace procedure csinformation
  (c_id in number,c_name in varchar2)
  is
  begin
  dbms_output.put_line('id:'||c_id);
  dbms_output.put_line('name:'||c_name);
  end;
  /

Procedure created.

SQL> exec csinformation(101,'raam');
id:101
name:raam

PL/SQL procedure successfully completed.

SQL>
```

Sample program for only function:

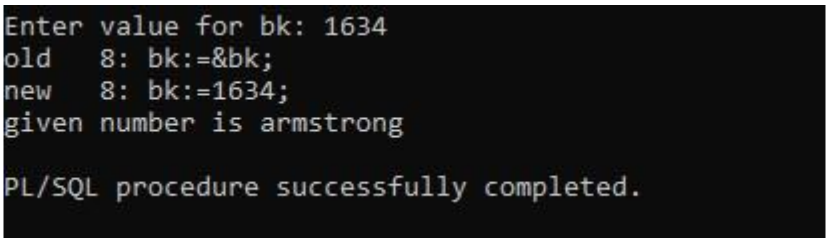


```
SQL> create or replace function csinform

(c_id in number,c_name in varchar2)
return varchar2
is
begin
if c_id>200 then
return('no booking available');
else
return('booking open');
end if;
end;
/

Function created.

SQL> declare
mesg varchar2(200);
begin
mesg:=csinform(102,'raam');
dbms_output.put_line(mesg);
end;
/
booking open

PL/SQL procedure successfully completed.

SQL> declare
mesg varchar2(200);
begin
mesg:=csinform(206,'raam');
dbms_output.put_line(mesg);
end;
/
no booking available

PL/SQL procedure successfully completed.
```

to check the given customer booking number is Armstrong



```
SQL>
SQL> declare
  2  bk number(5);
  3  s number:=0;
  4  r number;
  5  len number;
  6  m number;
  7  begin
  8  bk:=&bk;
  9  m:=bk;
 10  len:=length(to_char(bk));
 11  while bk>0
 12  loop
 13  r:=mod(bk,10);
 14  s:=s+power(r,len);
 15  bk:=trunc(bk/10);
 16  end loop;
 17  if
 18  m=s
 19  then
 20  dbms_output.put_line('given number is armstrong');
 21  else
 22  dbms_output.put_line('given number is not armstrong');
 23  end if;
 24  end;
 25  /
Enter value for bk: 234
old   8: bk:=&bk;
new   8: bk:=234;
given number is not armstrong

PL/SQL procedure successfully completed.
```

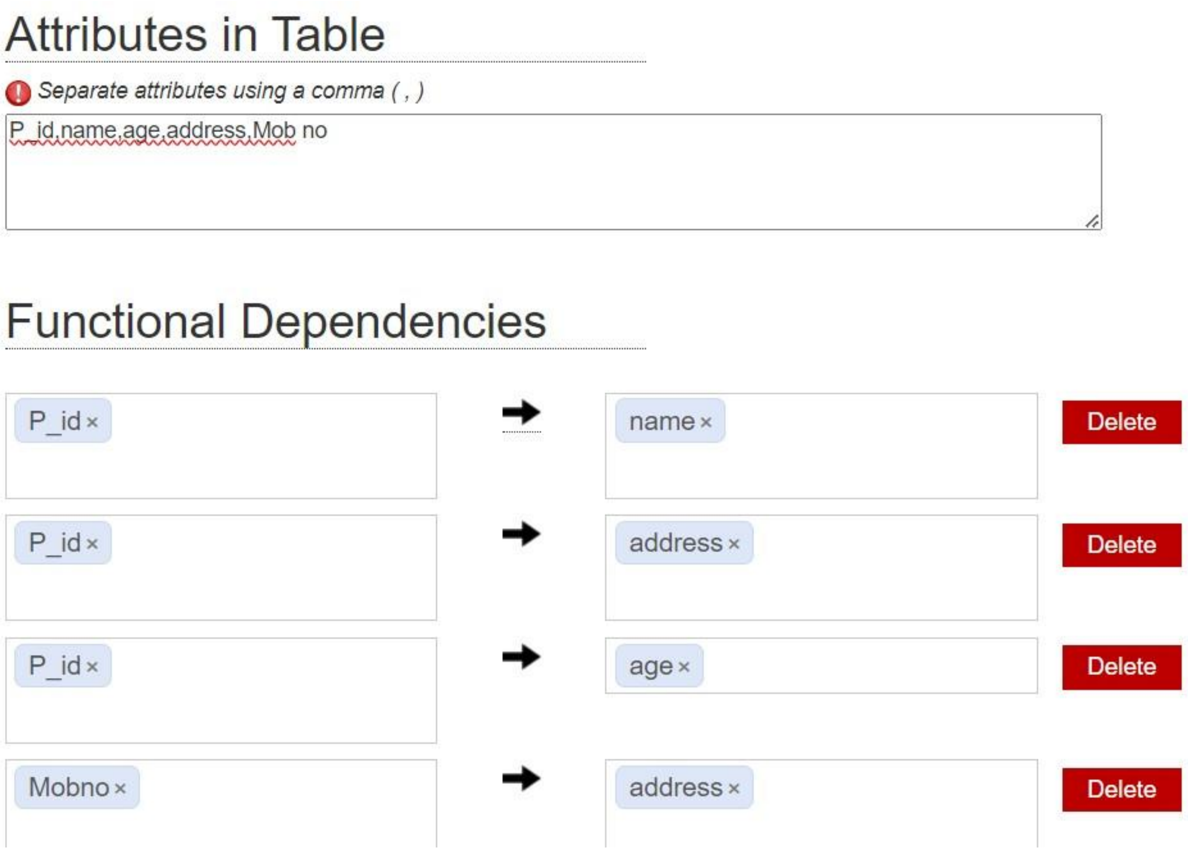


```
Enter value for bk: 1634
old   8: bk:=&bk;
new   8: bk:=1634;
given number is armstrong

PL/SQL procedure successfully completed.
```

Result: thus the execution of the relational operations,sql aggregates,joint&nested queries are successfully executed.

3.Normalisation:

Aim: to execute the normalisation forms for 1NF,2NF 3NF BCNF using Griffith tool **Attributes and functional dependencies:**



Attributes in Table

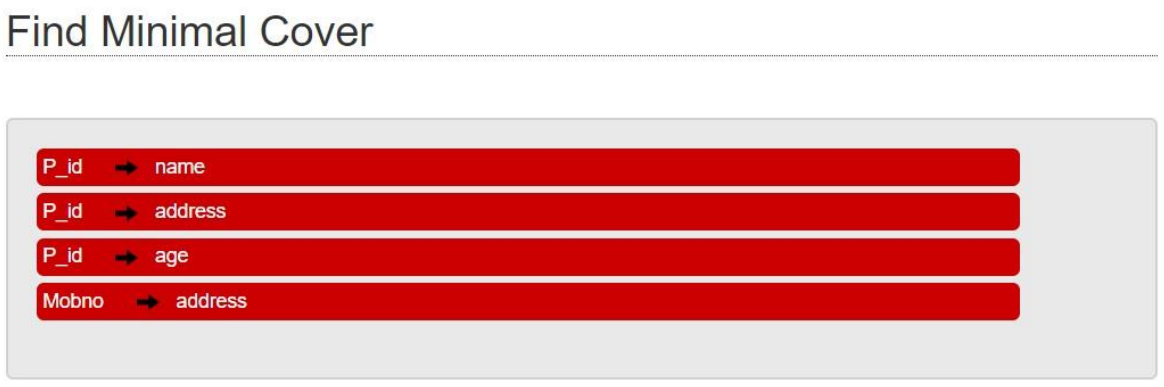
Separate attributes using a comma (,)

P_id,name,age,address,Mob no

Functional Dependencies

P_id ×	→	name ×	Delete
P_id ×	→	address ×	Delete
P_id ×	→	age ×	Delete
Mobno ×	→	address ×	Delete

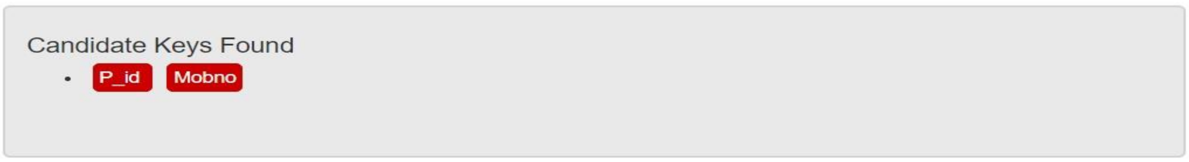
Minimal cover:



Find Minimal Cover

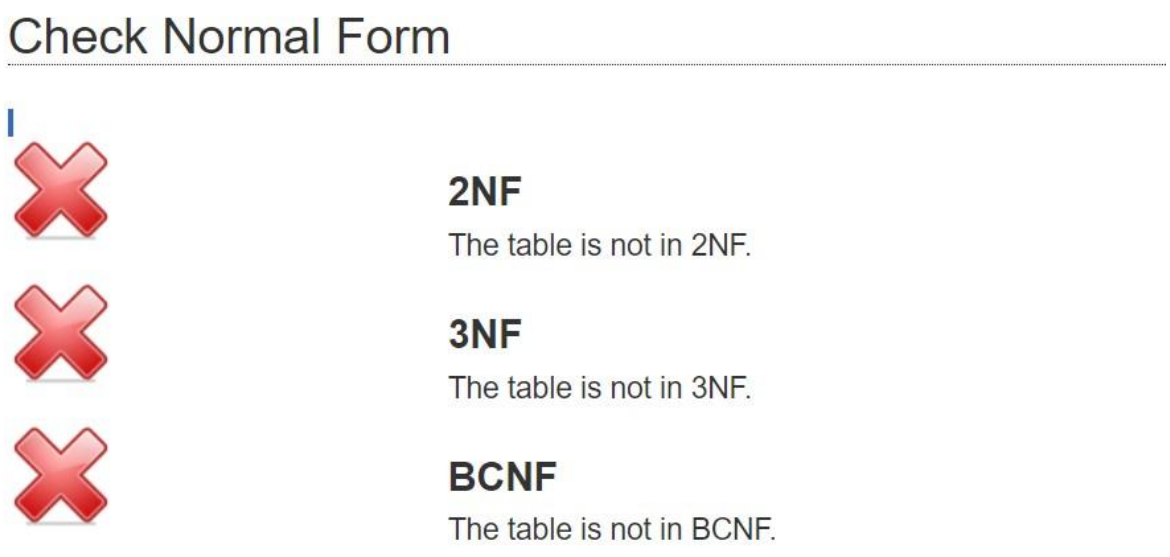
```
P_id   → name
P_id   → address
P_id   → age
Mobno  → address
```

Candidate keys:


Candidate Keys Found

- P_id Mobno

Normal form:

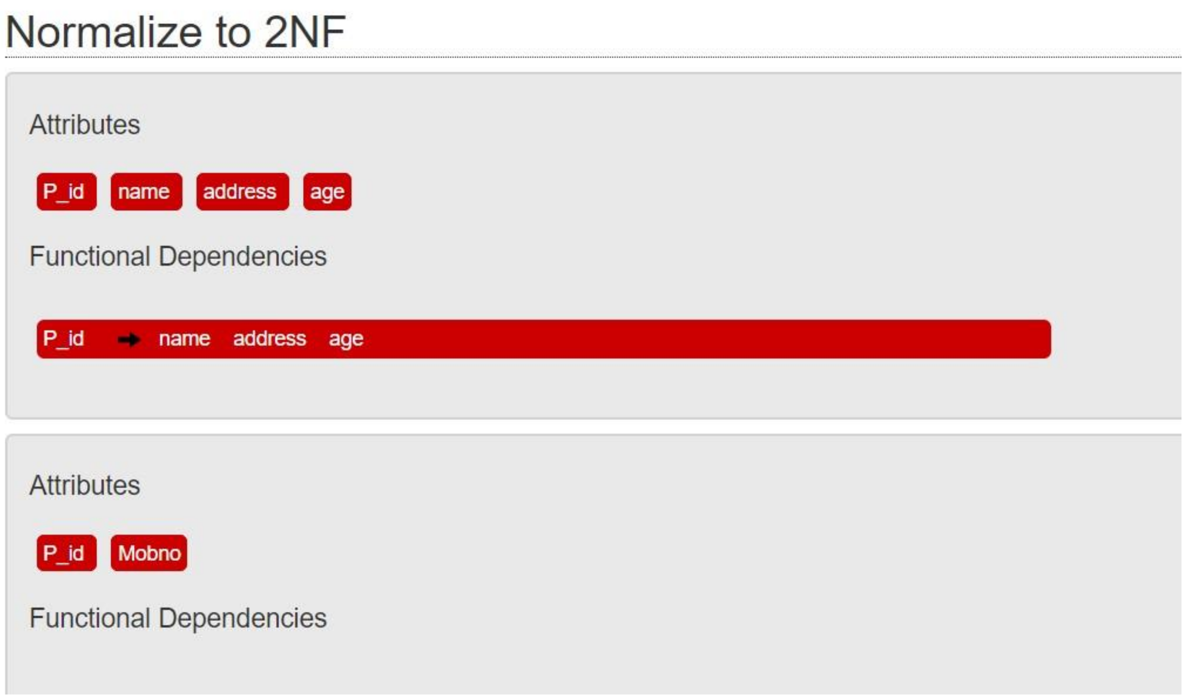

Check Normal Form

✗ **2NF**
The table is not in 2NF.

✗ **3NF**
The table is not in 3NF.

✗ **BCNF**
The table is not in BCNF.

2NF:


Normalize to 2NF

Attributes

P_id name address age

Functional Dependencies

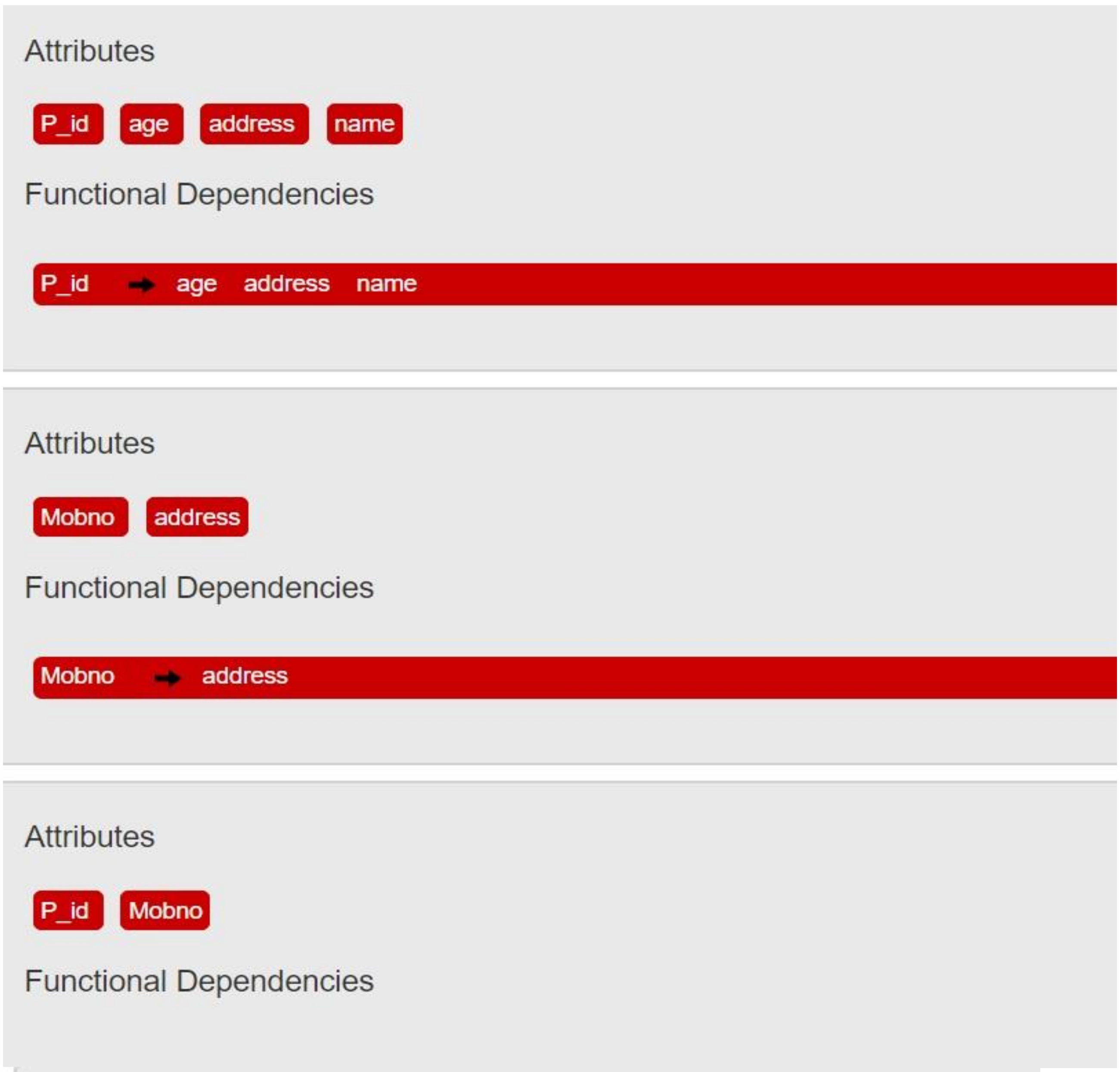
P_id → name address age

Attributes

P_id Mobno

Functional Dependencies

3NF:



Attributes

P_id age address name

Functional Dependencies

P_id → age address name

Attributes

Mobno address

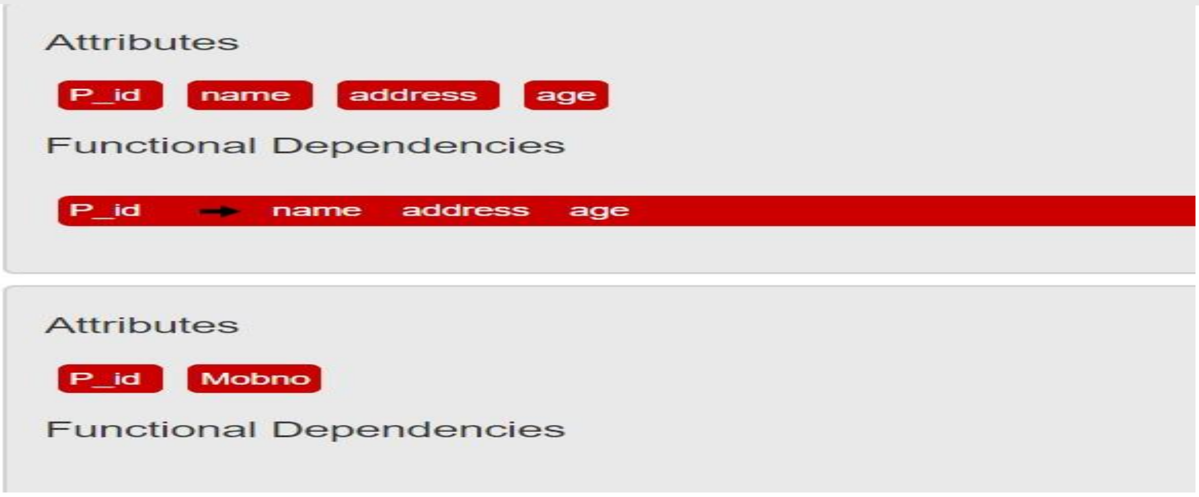
Functional Dependencies

Mobno → address

Attributes

P_id Mobno

Functional Dependencies



Attributes

P_id name address age

Functional Dependencies

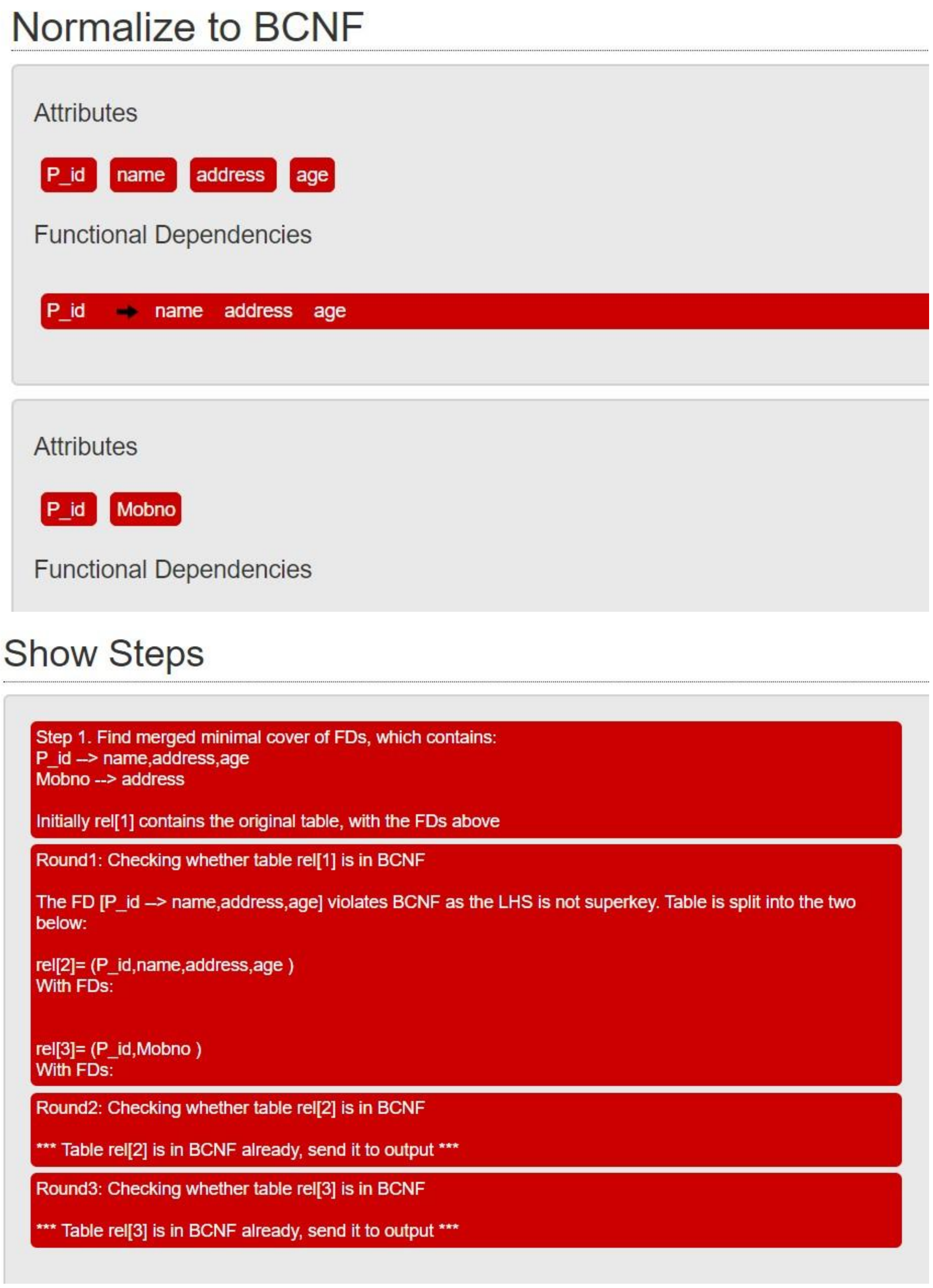
P_id → name address age

Attributes

P_id Mobno

Functional Dependencies

BCNF:



Normalize to BCNF

Attributes

P_id name address age

Functional Dependencies

P_id ➜ name address age

Attributes

P_id Mobno

Functional Dependencies

Show Steps

```
Step 1. Find merged minimal cover of FDs, which contains:
P_id --> name,address,age
Mobno --> address

Initially rel[1] contains the original table, with the FDs above
```

```
Round1: Checking whether table rel[1] is in BCNF

The FD [P_id --> name,address,age] violates BCNF as the LHS is not superkey. Table is split into the two
below:

rel[2]= (P_id,name,address,age )
With FDs:


rel[3]= (P_id,Mobno )
With FDs:
```

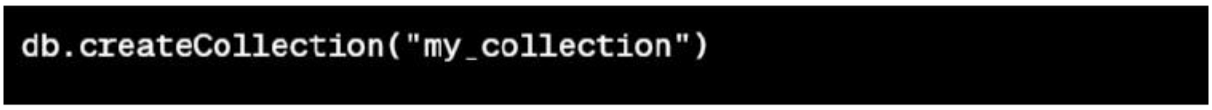
```
Round2: Checking whether table rel[2] is in BCNF

*** Table rel[2] is in BCNF already, send it to output ***
```

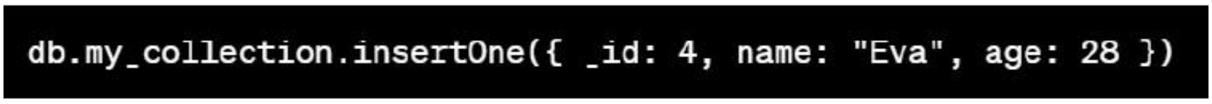
```
Round3: Checking whether table rel[3] is in BCNF

*** Table rel[3] is in BCNF already, send it to output ***
```

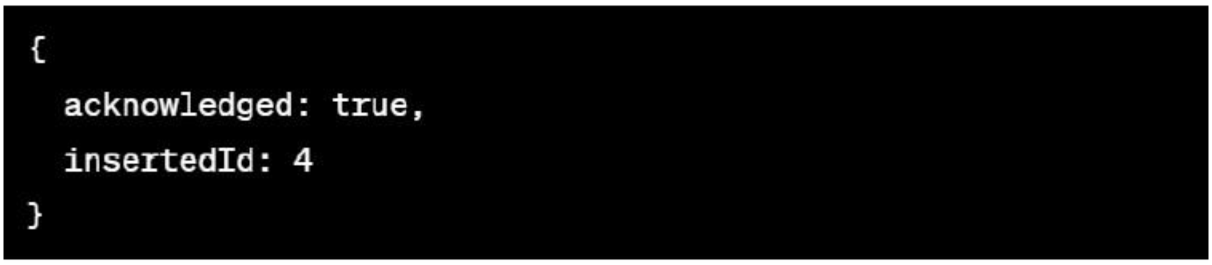
Result: Thus the normalisation to 1NF,2NF,3NF,BCNF is completed successfully

4.mangoDB

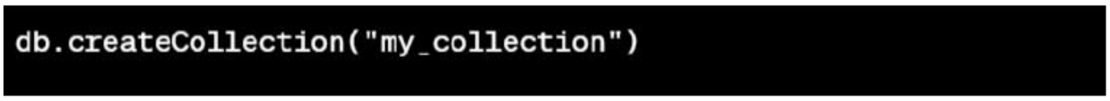
Aim: To implement the document database by using mangosh **Create:**

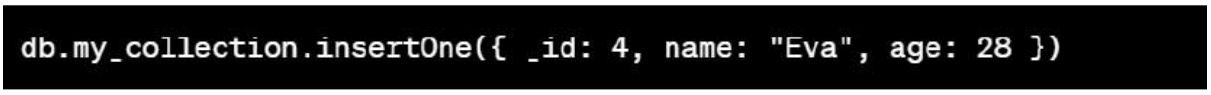

```
db.createCollection("my_collection")
```

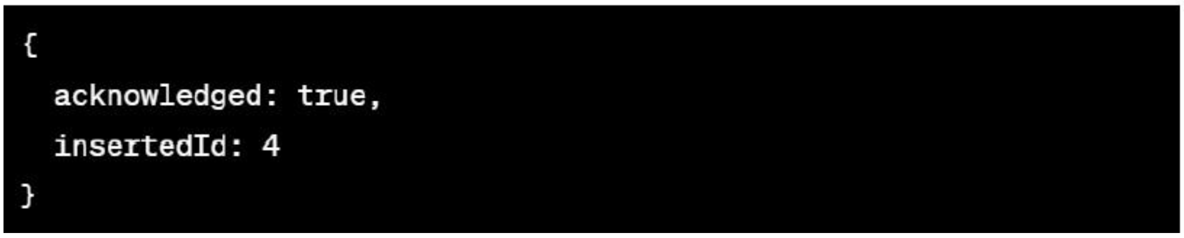

```
db.my_collection.insertOne({ _id: 4, name: "Eva", age: 28 })
```


```
{
  acknowledged: true,
  insertedId: 4
}
```

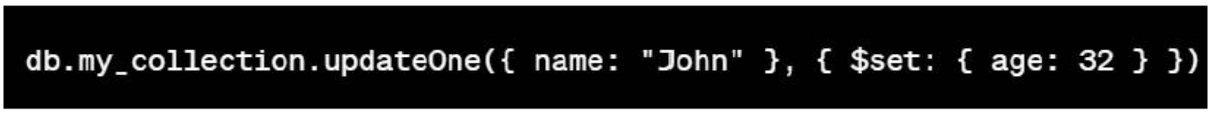
Read:

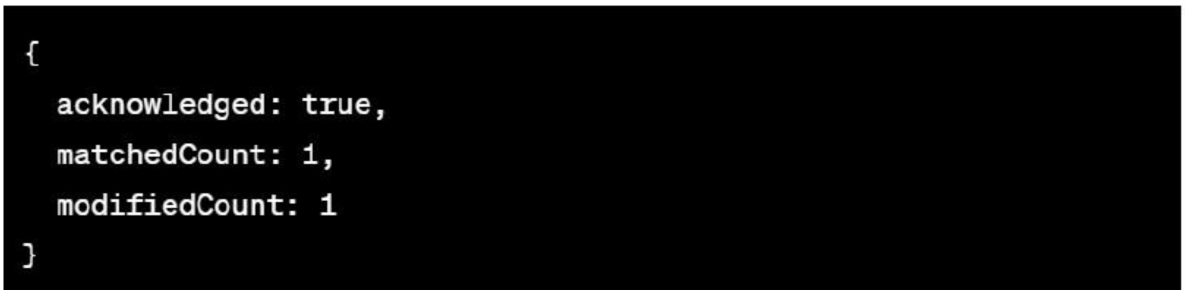

```
db.createCollection("my_collection")
```


```
db.my_collection.insertOne({ _id: 4, name: "Eva", age: 28 })
```

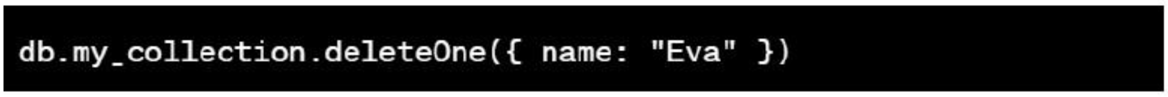

```
{
  acknowledged: true,
  insertedId: 4
}
```

Update:

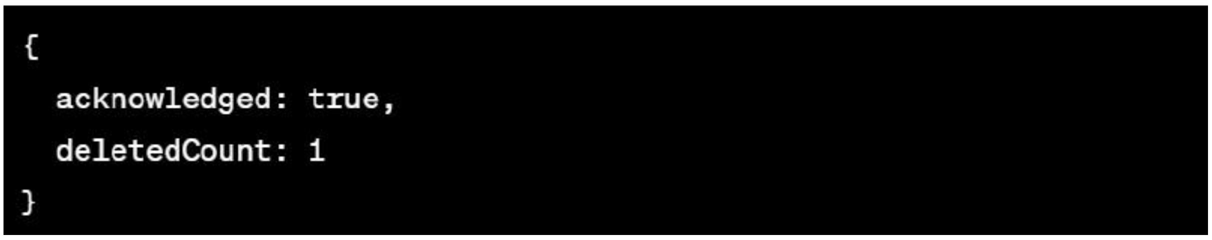

```
db.my_collection.updateOne({ name: "John" }, { $set: { age: 32 } })
```


```
{
  acknowledged: true,
  matchedCount: 1,
  modifiedCount: 1
}
```

Delete:



```
db.my_collection.deleteOne({ name: "Eva" })
```



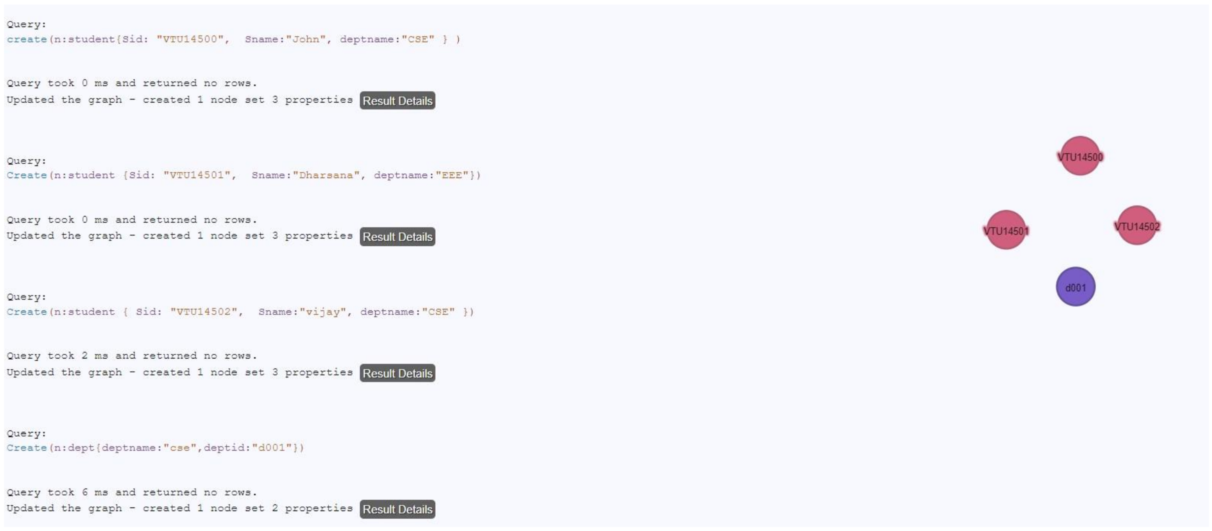
```
{
  acknowledged: true,
  deletedCount: 1
}
```

Result:

Thus CRUD using NPM design on MongoDB designing document database and performing CRUD operations like creating, inserting, querying, finding, removing operations are performed

5.graph database

Create:



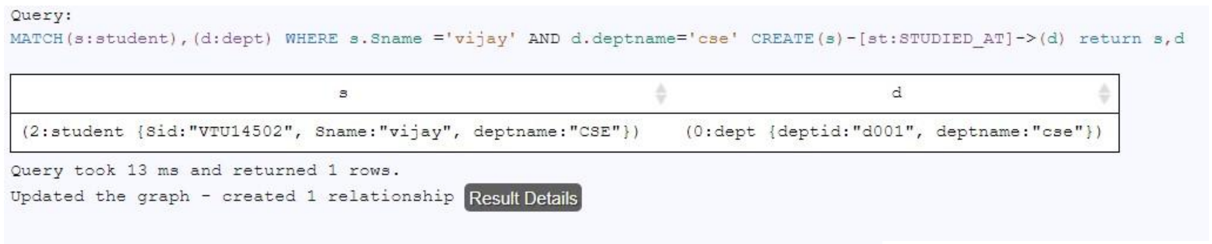
```
Query:
create(n:student{Sid: "VTU14500",  Sname:"John", deptname:"CSE" } )

Query took 0 ms and returned no rows.
Updated the graph - created 1 node set 3 properties
```

```
Query:
Create(n:student {Sid: "VTU14501",  Sname:"Dharsana", deptname:"EEE"})

Query took 0 ms and returned no rows.
Updated the graph - created 1 node set 3 properties
```

```
Query:
Create(n:student { Sid: "VTU14502",  Sname:"vijay", deptname:"CSE" })

Query took 2 ms and returned no rows.
Updated the graph - created 1 node set 3 properties
```

```
Query:
Create(n:dept{deptname:"cse",deptid:"d001"})

Query took 6 ms and returned no rows.
Updated the graph - created 1 node set 2 properties
```

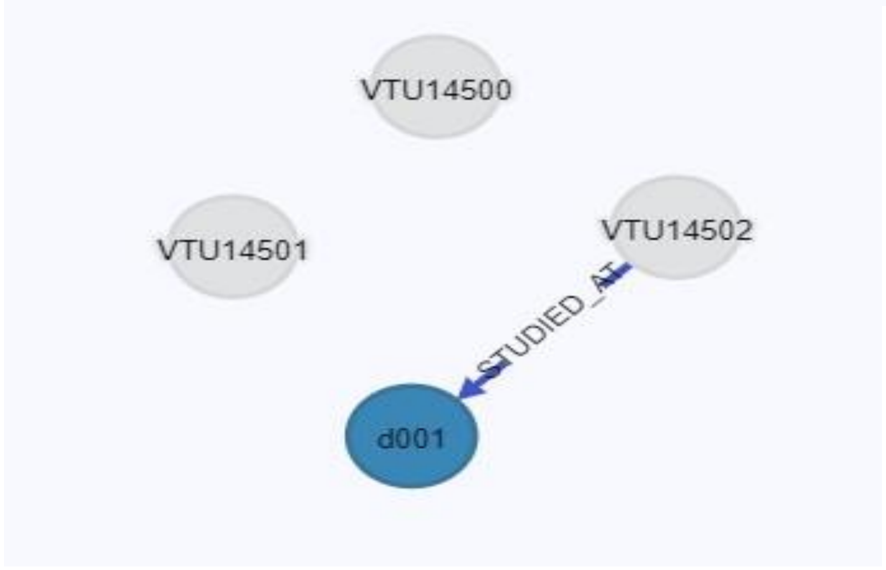
Relate:



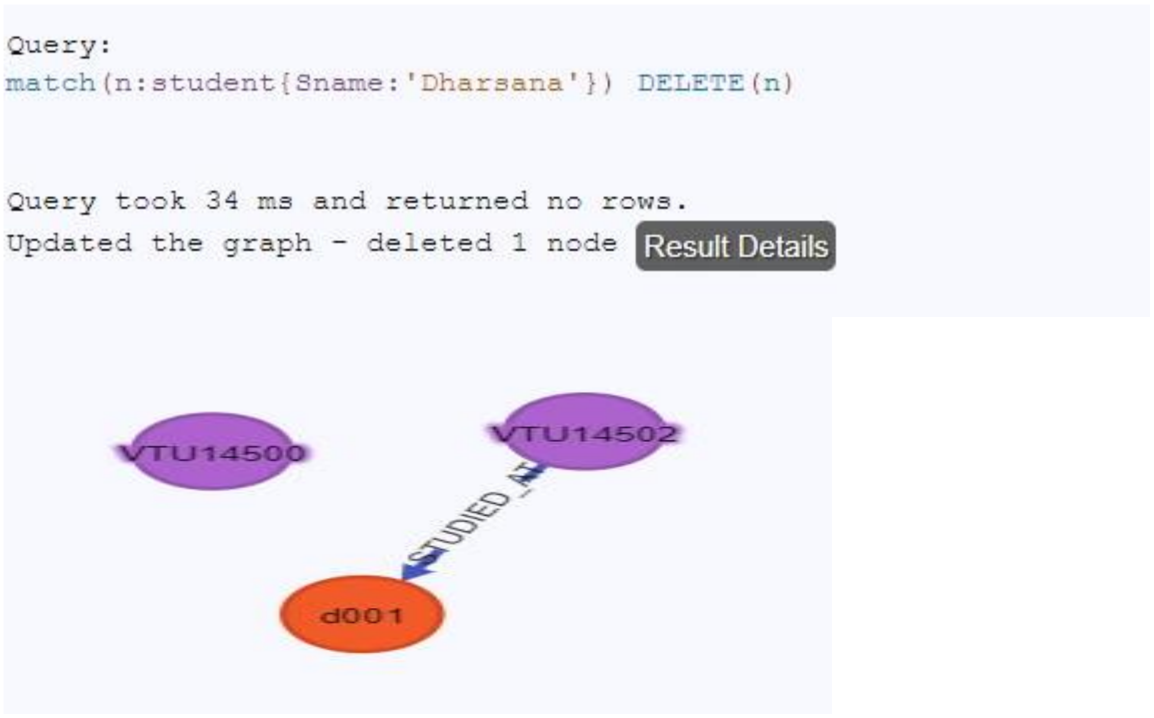
```
Query:
MATCH(s:student),(d:dept) WHERE s.Sname ='vijay' AND d.deptname='cse' CREATE(s)-[st:STUDIED_AT]->(d) return s,d
```

s	d
(2:student {Sid:"VTU14502", Sname:"vijay", deptname:"CSE"})	(0:dept {deptid:"d001", deptname:"cse"})

```
Query took 13 ms and returned 1 rows.
Updated the graph - created 1 relationship
```



Delete:



```
Query:
match(n:student{Sname:'Dharsana'}) DELETE(n)


Query took 34 ms and returned no rows.
Updated the graph - deleted 1 node
```

Result:

Thus the implementation of CRUD operations in graph spaces is completed successfully.

Result: thus micro project for Vehicle Service,Maintainence and Tracking System was developed and implemented successfully.