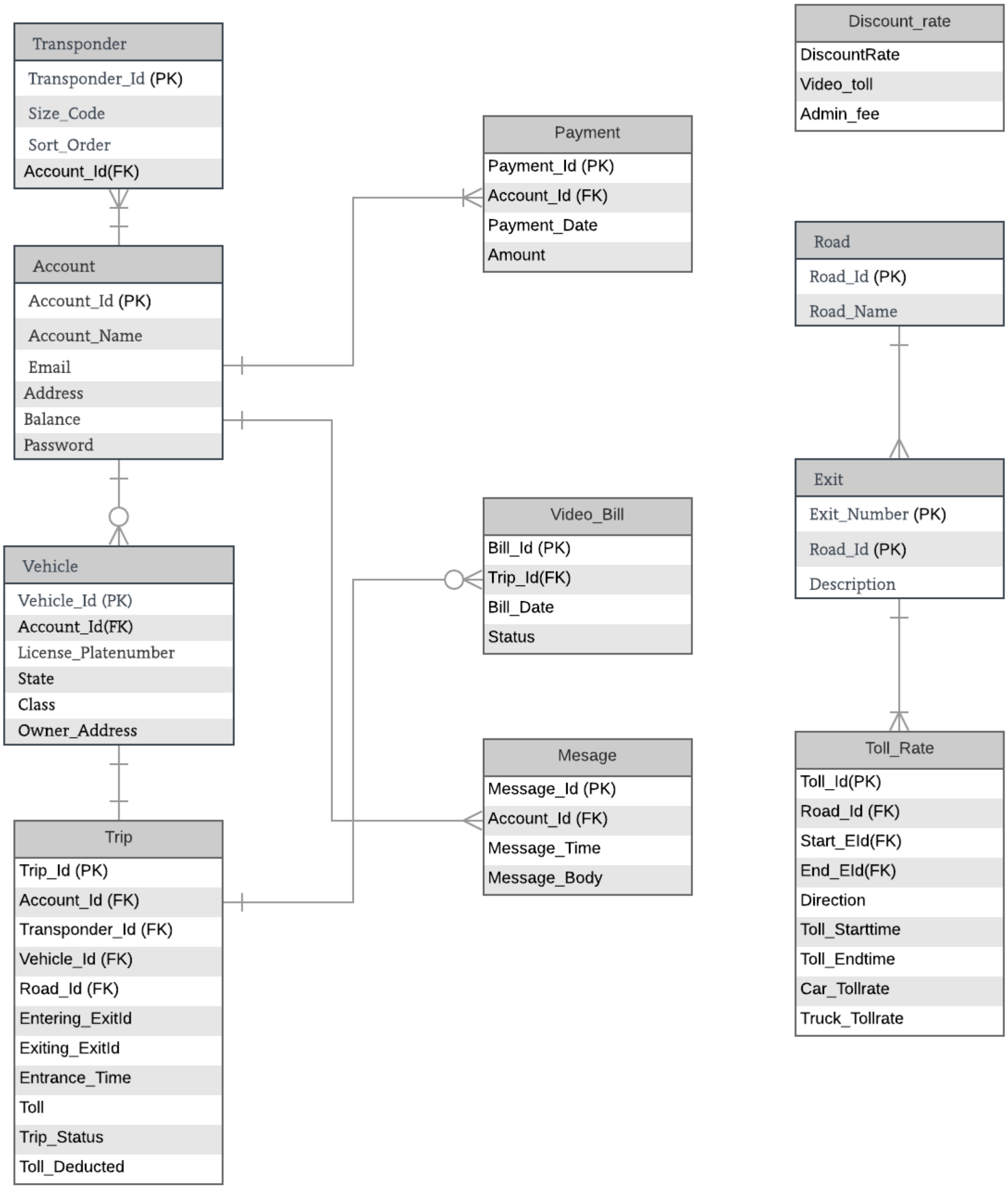
Toll Management Systems

ADVANCED DATABASE PROJECT

ER Diagram


Transponder
Transponder_Id (PK)
Size_Code
Sort_Order
Account_Id(FK)
Account
Account_Id (PK)
Account_Name
Email
Address
Balance
Password
Vehicle
Vehicle_Id (PK)
Account_Id(FK)
License_Platenumber
State
Class
Owner_Address
Trip
Trip_Id (PK)
Account_Id (FK)
Transponder_Id (FK)
Vehicle_Id (FK)
Road_Id (FK)
Entering_ExitId
Exiting_ExitId
Entrance_Time
Toll
Trip_Status
Toll_Deducted
Payment
Payment_Id (PK)
Account_Id (FK)
Payment_Date
Amount
Video_Bill
Bill_Id (PK)
Trip_Id(FK)
Bill_Date
Status
Mesage
Message_Id (PK)
Account_Id (FK)
Message_Time
Message_Body
Discount_rate
DiscountRate
Video_toll
Admin_fee
Road
Road_Id (PK)
Road_Name
Exit
Exit_Number (PK)
Road_Id (PK)
Description
Toll_Rate
Toll_Id(PK)
Road_Id (FK)
Start_EId(FK)
End_EId(FK)
Direction
Toll_Starttime
Toll_Endtime
Car_Tollrate
Truck_Tollrate

-- Drop Tables

```
drop table toll_rate cascade constraints;
drop table exit_road cascade constraints;
drop table discount_rate cascade constraints;
drop table message cascade constraints;
drop table video_bill cascade constraints;
drop table payment cascade constraints;
drop table trip cascade constraints;
drop table road cascade constraints;
drop table vehicle cascade constraints;
drop table account cascade constraints;
drop table transponder cascade constraints;
```

-- Delete Sequence

```
drop sequence account_id_updation;
drop sequence trip_id_updation;
drop sequence veh_seq;
drop sequence Transponder_id_seq;
drop sequence Bill_Id;
drop sequence message_id_seq;
drop sequence payment_id_seq;
```

-- Create Sequence

```
create sequence account_id_updation start with 6 increment by 1 order;
create sequence trip_id_updation start with 20 increment by 1 order;
create sequence Veh_seq start with  7;
create  sequence Transponder_id_seq start with 8;
create sequence Bill_Id start with 3006;
create  sequence message_id_seq start with 4006;
create  sequence payment_id_seq start with 2010;
```

-- Create Table

```
set serveroutput on;
-- Drop Tables
drop table toll_rate cascade constraints;
drop table exit_road cascade constraints;
drop table discount_rate cascade constraints;
drop table message cascade constraints;
drop table video_bill cascade constraints;
drop table payment cascade constraints;
drop table trip cascade constraints;
drop table road cascade constraints;
drop table vehicle cascade constraints;
drop table account cascade constraints;
drop table transponder cascade constraints;

-- Delete Sequence
drop sequence account_id_updation;
drop sequence trip_id_updation;
drop sequence veh_seq;
drop sequence Transponder_id_seq;
drop sequence Bill_Id;
drop sequence message_id_seq;
drop sequence payment_id_seq;

-- Create Sequence
create sequence account_id_updation start with 6 increment by 1 order;
create sequence trip_id_updation start with 20 increment by 1 order;
create sequence Veh_seq start with  7;
create  sequence Transponder_id_seq start with 8;
create sequence Bill_Id start with 3006;
create  sequence message_id_seq start with 4006;
create  sequence payment_id_seq start with 2010;


/
-- Create Table

-- Account Table --
create table Account(
Account_id int not null Primary Key,
Account_Name varchar(30) Not Null,
Email varchar(30) Not Null,
Address varchar(100),
Balance int,
password_ varchar(20) Not Null);


-- Transponder Table --
create table Transponder(
Transponder_id int not null Primary Key,
Size_Code int Not Null,
Sort_Order int,
Account_id int,
Foreign key(Account_id) references Account(Account_id));


-- Vehicle --
create table Vehicle(
Vehicle_Id int not null Primary Key,
Account_Id int,
License_Platenumber varchar(10) Not Null,
Vehicle_State varchar(5) Not Null,
Vehicle_class int,
Owner_Address varchar(30),
Foreign Key(Account_Id) references Account(Account_Id));



-- Road --
create table road(
Road_Id varchar(5) not null Primary Key,
Road_Name varchar(30) Not Null);
```

```
-- Exit --
create table exit_road(
Exit_Number int not null Primary Key,
Road_Id varchar(5) Not Null,
Description varchar(30),
Foreign Key(Road_Id) references Road(Road_Id));

-- Toll_Rate --
create table Toll_Rate(
Toll_Id int not null Primary Key,
Road_Id varchar(5) Not Null,
Start_EId int Not Null,
End_EId int Not Null,
Direction varchar(10),
Toll_Starttime date not null,
Toll_Endtime date not null,
Car_Tollrate int not null,
Truck_Tollrate int not null,
Foreign Key(Road_Id) references Road(Road_Id),
Foreign Key(Start_EId) references exit_road(exit_number),
Foreign Key(End_EId) references exit_road(exit_number));



-- Trip --
create table Trip(
Trip_Id int not null Primary Key,
Account_Id int,
Transponder_Id int,
Vehicle_Id int,
Road_Id varchar(5) Not null,
Entering_ExitId int Not Null,
Exiting_ExitId int Not Null,
Entrance_Time Timestamp Not Null,
trip_Status int Not Null,
Toll_Deducted int Not Null,
Toll float Not Null,
Trip_direction varchar(10),
Foreign Key(Account_Id) references Account(Account_Id),
Foreign Key(Transponder_Id) references Transponder(Transponder_Id),
Foreign Key(Vehicle_Id) references Vehicle(Vehicle_Id),
Foreign Key(Road_Id) references Road(Road_Id));


-- Payment --
create table payment(
Payment_id int Not Null Primary Key,
Account_Id int Not Null,
Payment_Date date not null,
Amount float,
Foreign Key(Account_Id) references Account(Account_Id));


-- Video Bill --
create table video_bill(
Bill_Id int not null Primary Key,
Trip_Id int Not Null,
Bill_Date date Not Null,
Status int Not Null,
Foreign Key(Trip_Id) references Trip(Trip_Id));


-- Message --
create table Message(
Message_Id int not null Primary Key,
Account_Id int,
Message_Time DATE,
Message_Body varchar(100) Not Null,
Foreign Key(Account_Id) references Account(Account_Id));


-- Discount_rate --
create table Discount_rate(
DiscountRate float Not Null,
Video_toll int Not Null,
Admin_fee int Not Null);




  insert into account values(1,'abc','abc@gmail.com','1001 Courtney', 5,'abc@1' );
  insert into account values(2,'def','def@gmail.com','1002 Courtney', 10,'def@2' );
  insert into account values(3,'ghi','ghi@gmail.com','1003 Courtney', 15,'ghi@3' );
  insert into account values(4,'jkl','jkl@gmail.com','1004 Courtney', 8,'jkl@4' );
  insert into account values(5,'mno','mno@gmail.com','1005 Courtney', 10,'mno@5' );

  insert into transponder values(1,101,201,1);
  insert into transponder values(2,102,202,2);
  insert into transponder values(3,103,203,3);
  insert into transponder values(4,104,204,4);
  insert into transponder values(5,105,205,5);
  insert into transponder values(7,105,205,null);


  insert into vehicle values(1,1,'z123','MD','1','1001 Courtney' );
  insert into vehicle values(2,2,'y123','VA','2','1002 Courtney' );
  insert into vehicle values(3,3,'x123','DC','1','1003 Courtney' );
  insert into vehicle values(4,null,'w123','CA','2','1004 Courtney' );
  insert into vehicle values(5,5,'v123','MA','1','1005 Courtney' );
  insert into vehicle values(6,null,'z1234','MA','1','1005 Courtney' );


  insert into road values('I95', 'Baltimore Highway' );
  insert into road values('I96', 'DC Highway' );
  insert into road values('I97', 'New York Road' );
  insert into road values('I98', 'Philadelphia Road' );
  insert into road values('I495', 'National Highway' );


  insert into exit_road values(143,'I95','Baltimore Juncture Exit 143');
  insert into exit_road values(149,'I95','Baltimore Juncture Exit 149');
  insert into exit_road values(152,'I95','Baltimore Juncture Exit 152');
  insert into exit_road values(155,'I96','DC Juncture Exit 155');
  insert into exit_road values(157,'I96','DC Juncture Exit 157');
  insert into exit_road values(159,'I96','DC Juncture Exit 159');
  insert into exit_road values(165,'I97','New York Juncture Exit 165');
  insert into exit_road values(168,'I97','New York Juncture Exit 168');
  insert into exit_road values(170,'I97','New York Juncture Exit 170');
  insert into exit_road values(135,'I98','Virgina Jucture Exit 135');
  insert into exit_road values(130,'I98','Virgina Jucture Exit 130');
  insert into exit_road values(125,'I98','Virgina Jucture Exit 125');
  insert into exit_road values(120,'I495','National Highway Thru 120');
  insert into exit_road values(110,'I495','National Highway Thru 110');
  insert into exit_road values(100,'I495','National Highway Thru 100');
```

```
 insert into toll_rate values(1,'I95',143,149,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',1,6);
insert into toll_rate values(2,'I95',149,152,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',3,2);
insert into toll_rate values(3,'I95',149,143,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',5,1);
insert into toll_rate values(4,'I95',152,149,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',8,3);
insert into toll_rate values(5,'I96',155,157,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',2,7);
insert into toll_rate values(6,'I96',157,159,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',3,1);
insert into toll_rate values(7,'I96',157,155,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',5,2);
insert into toll_rate values(8,'I96',159,157,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',6,4);
insert into toll_rate values(9,'I97',165,168,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',4,5);
insert into toll_rate values(10,'I97',168,170,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',6,6);
insert into toll_rate values(11,'I97',168,165,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',7,8);
insert into toll_rate values(12,'I97',170,168,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',3,6);
insert into toll_rate values(13,'I98',125,130,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',8,4);
insert into toll_rate values(14,'I98',130,135,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',6,2);
insert into toll_rate values(15,'I98',130,125,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',9,3);
insert into toll_rate values(16,'I98',135,130,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',6,1);
insert into toll_rate values(17,'I495',100,110,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',3,8);
insert into toll_rate values(18,'I495',110,120,'North',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',9,9);
insert into toll_rate values(19,'I495',110,100,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',2,10);
insert into toll_rate values(20,'I495',120,110,'South',timestamp '2018-05-01 10:00:00',timestamp '2018-12-31 12:00:00',7,1);
```

```
insert into trip values(1,1,1,1,'I95',143,149,TO_DATE('2018/08/10 11:00:00','yyyy/mm/dd hh:mi:ss'),1,1,1,'North');
insert into trip values(2,1,1,1,'I95',149,152,TO_DATE('2018/08/10 11:05:00','yyyy/mm/dd hh:mi:ss'),1,1,3,'North');
insert into trip values(3,1,1,1,'I96',155,157,TO_DATE('2018/08/10 11:15:00','yyyy/mm/dd hh:mi:ss'),1,0,2,'North');
insert into trip values(6,2,2,2,'I96',157,159,TO_DATE('2018-08-10 11:10:00','yyyy/mm/dd hh:mi:ss'),1,0,3,'North');
insert into trip values(4,1,1,1,'I96',157,159,TO_DATE('2018-08-10 11:20:00','yyyy/mm/dd hh:mi:ss'),1,0,3,'North');
insert into trip values(5,null,null,5,'I495',120,110,TO_DATE('2018-08-10 11:15:00','yyyy/mm/dd hh:mi:ss'),3,0,12,'South');
insert into trip values(8,4,2,4,'I495',120,110,TO_DATE('2018-08-10 11:25:00','yyyy/mm/dd hh:mi:ss'),1,1,7,'South');
insert into trip values(7,3,3,3,'I97',168,165,TO_DATE('2018-08-10 11:10:00','yyyy/mm/dd hh:mi:ss'),1,1,7,'South');
insert into trip values(9,null,null,5,'I98',130,135,TO_DATE('2018-11-28 12:15:00','yyyy/mm/dd hh:mi:ss'),3,0,11,'North');
insert into trip values(10,null,null,6,'I98',130,135,TO_DATE('2018-11-28 12:25:00','yyyy/mm/dd hh:mi:ss'),3,0,11,'North');
insert into trip values(11,null,null,5,'I97',170,168,TO_DATE('2018-10-25 01:15:00','yyyy/mm/dd hh:mi:ss'),3,0,8,'South');
insert into trip values(12,null,null,5,'I495',110,120,TO_DATE('2018-09-10 11:15:00','yyyy/mm/dd hh:mi:ss'),3,0,14,'North');
```

```
 insert into trip values(13,null,7,6,'I495',110,120,TO_DATE('2018-09-10 11:15:00','yyyy/mm/dd hh:mi:ss'),2,0,9,'North');
 insert into trip values(14,null,7,2,'I495',110,120,TO_DATE('2018-09-10 11:15:00','yyyy/mm/dd hh:mi:ss'),2,0,9,'North');
 insert into trip values(15,2,2,2,'I495',110,120,TO_DATE('2018-09-10 11:15:00','yyyy/mm/dd hh:mi:ss'),1,0,9,'North');
 insert into trip values(16,1,1,1,'I495',110,120,TO_DATE('2018-09-10 11:15:00','yyyy/mm/dd hh:mi:ss'),1,0,9,'North');


 insert into payment values(2001,1,date '2018-08-11',1);
 insert into payment values(2002,1,date '2018-08-12',3);
 insert into payment values(2003,4,date '2018-08-12',7);
 insert into payment values(2004,3,date '2018-08-12',7);




 insert into video_bill values(3001,5,date '2018-08-10',2);
 insert into video_bill values(3002,9,date '2018-11-28',2);
 insert into video_bill values(3003,10,date '2018-11-28',1);
 insert into video_bill values(3004,11,date '2018-10-25',2);
 insert into video_bill values(3005,12,date '2018-09-10',2);

 insert into message values(4001,1,date '2018-10-01','You have no message');
 insert into message values(4002,2,date '2018-10-02','You have no message');
 insert into message values(4003,3,date '2018-10-03','You have no message');
 insert into message values(4004,4,date '2018-10-04','You have no message');
 insert into message values(4005,5,date '2018-10-05','You have no message');



 insert into discount_rate values(0.5,1,5);
```

--Feature Code

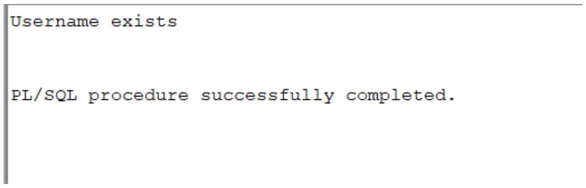
--Feature 1

```
set serveroutput on;
create or replace procedure feature_1(username varchar,e_email varchar,p_password varchar, a_address varchar, b_balance int)
 is
 check_username account.account_name%type;
 check_email account.email%type;
 cursor c1 is select account_name from account where account_name=username;
 Cursor c2 is select email from account where email=e_email;
 begin
 open c1;
 open c2;
 loop
 fetch c1 into check_username;
 fetch c2 into check_email;
 exit when c1%notfound and c2%notfound;
 end loop;
 if check_username like username then
 dbms_output.put_line('Username exists');
 elsif check_email like e_email then
 dbms_output.put_line('Email exists');
 else
```

```
  insert into account(account_id,account_name,email,password_,address, balance) values (account_id_updation.nextval,username,
e_email, p_password,a_address,b_balance);
dbms_output.put_line('Hello '||username||'!'||' Account created.');
  end if;
  close c1;
  close c2;
  end;
```

--Output

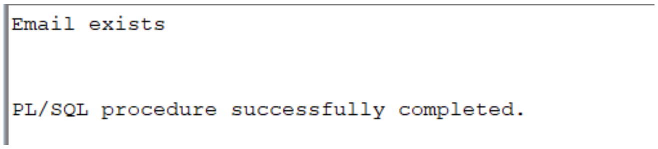
--username exists



```
Username exists


PL/SQL procedure successfully completed.
```

```
exec feature_1('abc','abc@gmail.com','abc@1','1001 Courtney',5);
```

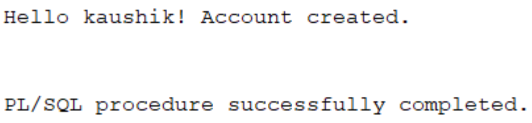
--Email exists



```
Email exists


PL/SQL procedure successfully completed.
```

```
exec feature_1('abc1','abc@gmail.com','abc@1','1001 Courtney',5);
```

--Account created



```
Hello kaushik! Account created.


PL/SQL procedure successfully completed.
```

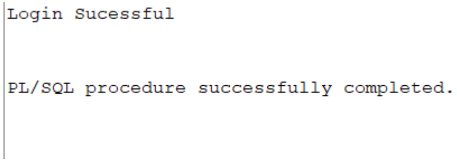
```
exec feature_1('kaushik','kau@gmail.com','kau@1','1010 Howland',5);
```

--Feature 2

```
set serveroutput on;
create or replace procedure feature_2 (e_email in varchar, p_password in varchar) is
cursor c1 is select email, password_
from account where email=e_email and password_=p_password;
check_email account.email%type;
check_password account.password_%type;
begin
open c1;
loop
fetch c1 into check_email, check_password;
exit when c1%notfound;
end loop;
if c1%rowcount=1 then
dbms_output.put_line('Login Sucessful');
else
dbms_output.put_line('Login Unsucessful');
end if;
close c1;
end;
```

--Output

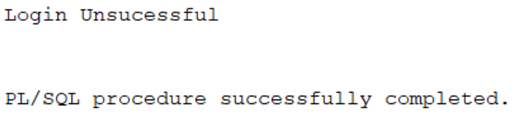
--Login Successful



```
Login Sucessful


PL/SQL procedure successfully completed.
```

```
exec feature_2(‘abc@gmail.com’,’abc@1’;
```

-- Login Unsuccessful



```
Login Unsucessful


PL/SQL procedure successfully completed.
```

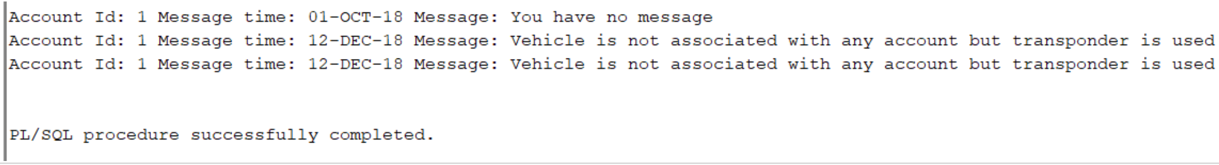
```
exec feature_2(‘abc@gmail.com’,’abc@11’;
```

--Feature 3

```
set serveroutput on;
create or replace procedure feature_3(a_account_id int, m_message_time date)
as
a_id message.account_id%type;
m_time message.message_time%type;
m_body message.message_body%type;
check_account account.account_id%type;
cursor c1 is select account_id, message_time, message_body into a_id, m_time, m_body from message where
account_id=a_account_id and message_time>=m_message_time;
begin
select count(*) into check_account from account where account.account_id=a_account_id;
if check_account=0 then
dbms_output.put_line('Invalid Account');
else
open c1;
loop
fetch c1 into a_id, m_time, m_body;
if c1%rowcount=0 then
dbms_output.put_line('No Mesasges to Display');
end if;
exit when c1%notfound;
dbms_output.put_line('Account Id: '||a_id||' Message time: '||m_time||' Message: '||m_body);
end loop;
close c1;
end if;
end;
```

--Output

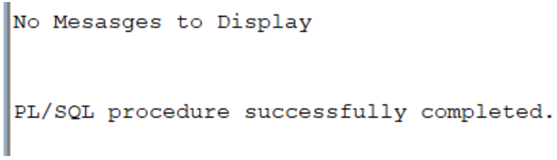
--Message Found



```
Account Id: 1 Message time: 01-OCT-18 Message: You have no message
Account Id: 1 Message time: 12-DEC-18 Message: Vehicle is not associated with any account but transponder is used
Account Id: 1 Message time: 12-DEC-18 Message: Vehicle is not associated with any account but transponder is used


PL/SQL procedure successfully completed.
```

```
exec feature_3(1,date'2018-08-09');
```

--Message Not Found



```
No Mesasges to Display


PL/SQL procedure successfully completed.
```

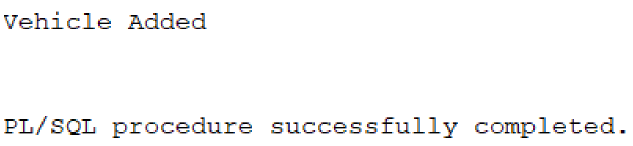
```
exec feature_3(2,date'2018-10-09');
```

--Feature 4
--Addition of Vehicle

```
set serveroutput on;
create or replace procedure feature_4_Add(Acc_Id in int,license in varchar,V_state in varchar,address in varchar,V_class in int)
AS
car_count account.account_id%type;
t_transponder_id transponder.transponder_id%type;
acc_count int;
begin
   select count(*) into car_count from vehicle v
   where  v.account_id=Acc_Id and v.license_platenumber=license and vehicle_state=v.vehicle_state;
   select count(*) into acc_count from account where account_id=acc_id;
   if car_count = 0 and acc_count=1 then

      select transponder_id into t_transponder_id from transponder where account_id=acc_id;
      insert into vehicle values(Veh_seq.nextval,Acc_Id,license,V_state,V_class,address);
      insert into message values(Message_id_seq.nextval,Acc_Id,sysdate,'Vehicle has been added');
      dbms_output.put_line('Vehicle Added');
   elsif car_count=0 and acc_count=0 then
       insert into vehicle values(Veh_seq.nextval,null,license,V_state,V_class,address);
       insert into message values(Message_id_seq.nextval,null,sysdate,'Vehicle has been added');
      dbms_output.put_line('Vehicle Added');
   else
      dbms_output.put_line('The vehicle exists');
      update vehicle
      set vehicle_class=V_class, owner_address=address
      where Account_id=Acc_Id;
   end if;

   Exception
   When no_data_found then
   dbms_output.put_line('No data found');
end;
```

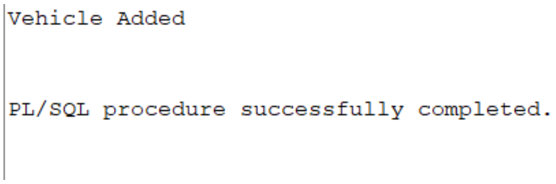
--Output

```
--Execute Addition function
exec feature_4_Add (2,'k123','MD','1010 Howland',1);
--Addition of Vehicle
```



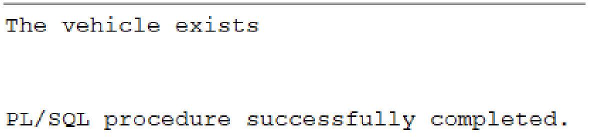
```
Vehicle Added


PL/SQL procedure successfully completed.
```

--Addition of Vehicle when account is null

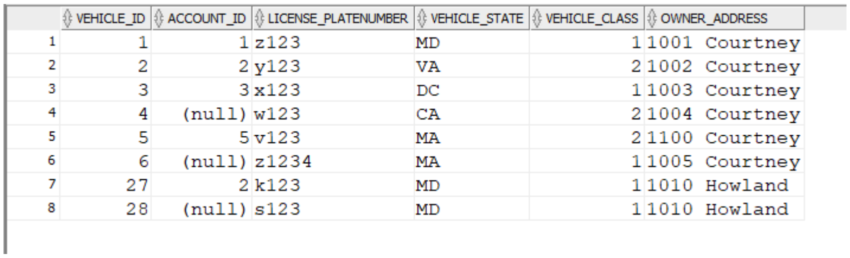


```
Vehicle Added


PL/SQL procedure successfully completed.
```

--Vehicle Exit but update values



```
The vehicle exists


PL/SQL procedure successfully completed.
```



	VEHICLE_ID	ACCOUNT_ID	LICENSE_PLATENUMBER	VEHICLE_STATE	VEHICLE_CLASS	OWNER_ADDRESS
1	1	1	z123	MD	1	1001 Courtney
2	2	2	y123	VA	2	1002 Courtney
3	3	3	x123	DC	1	1003 Courtney
4	4	(null)	w123	CA	2	1004 Courtney
5	5	5	v123	MA	2	1100 Courtney
6	6	(null)	z1234	MA	1	1005 Courtney
7	27	2	k123	MD	1	1010 Howland
8	28	(null)	s123	MD	1	1010 Howland

select * from vehicle;

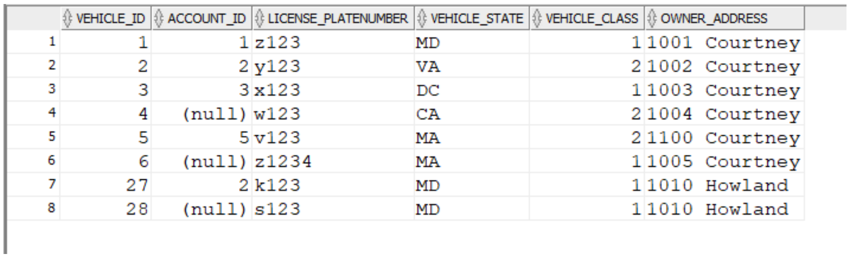
--Deletion of Vehicle

```
Set serveroutput on;
create or replace procedure feature_4_delete(Acc_Id in int,license in varchar,V_state in varchar)
as
A_id int;
O_license int;
O_state int;
V_id int;
T_id int;
begin
   -- check if valid account id
   select count(*) into A_id from vehicle v
   where v.account_id=Acc_Id;
   if A_id = 0 then
      dbms_output.put_line('Account id is invalid');
   else
      -- check if license plate matches
      select count(*) into O_license from vehicle v where v.License_Platenumber=license;
      if O_license=0 then
        dbms_output.put_line('License does not match');
      else
      select count(*) into O_state from vehicle v where v.Vehicle_State=V_state;
         if O_state=0 then
            dbms_output.put_line('State does not match');
         else
             select v.Account_Id into A_id from vehicle v
```

```
                where v.Account_Id=Acc_Id and v.License_Platenumber=license and v.Vehicle_state=V_state;

                select v.vehicle_id into V_id from vehicle v
                where v.account_id=Acc_id and v.license_platenumber=license and v.Vehicle_state=V_state;
                if A_id <> 0 and V_id <> 0 then
                    -- Delete references

                  delete from video_bill vb
                      where vb.Trip_Id in (select t.Trip_Id  from trip t
                                    where Account_id=A_id and vehicle_id=V_id);
                  delete from trip
                      where Account_id=A_id and vehicle_id=V_id;

                   -- delet from vehicle
                   delete from vehicle
                       where Account_id=A_id and License_Platenumber=license;
                   insert into message values(Message_id_seq.nextval,Acc_Id,sysdate,'Vehicle has been deleted');

               end if;
           end if;
       end if;
   end if;
Exception
  when no_data_found
  then
   dbms_output.put_line('No data found');
end;
```

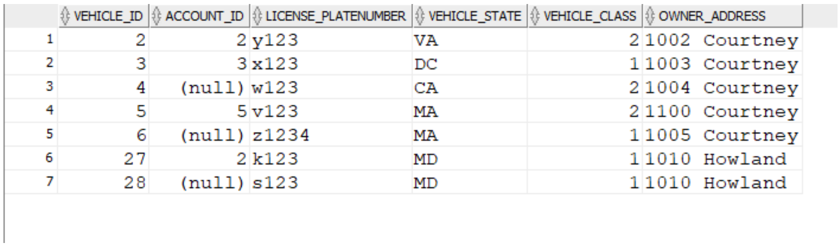
--Output

Before Deletion



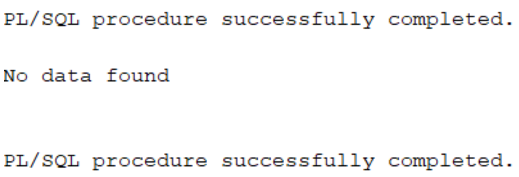
	VEHICLE_ID	ACCOUNT_ID	LICENSE_PLATENUMBER	VEHICLE_STATE	VEHICLE_CLASS	OWNER_ADDRESS
1	1	1	z123	MD	1	1001 Courtney
2	2	2	y123	VA	2	1002 Courtney
3	3	3	x123	DC	1	1003 Courtney
4	4	(null)	w123	CA	2	1004 Courtney
5	5	5	v123	MA	2	1100 Courtney
6	6	(null)	z1234	MA	1	1005 Courtney
7	27	2	k123	MD	1	1010 Howland
8	28	(null)	s123	MD	1	1010 Howland

After Deletion



	VEHICLE_ID	ACCOUNT_ID	LICENSE_PLATENUMBER	VEHICLE_STATE	VEHICLE_CLASS	OWNER_ADDRESS
1	2	2	y123	VA	2	1002 Courtney
2	3	3	x123	DC	1	1003 Courtney
3	4	(null)	w123	CA	2	1004 Courtney
4	5	5	v123	MA	2	1100 Courtney
5	6	(null)	z1234	MA	1	1005 Courtney
6	27	2	k123	MD	1	1010 Howland
7	28	(null)	s123	MD	1	1010 Howland

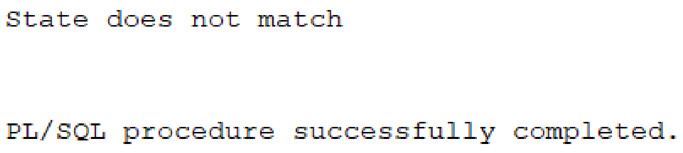
```
exec feature_4_delete(1,'z123','MD');

--Invalid Account id
```



```
PL/SQL procedure successfully completed.

No data found


PL/SQL procedure successfully completed.
```

exec feature_4_delete (2,'z1234','MD');

--Invalid State



```
State does not match


PL/SQL procedure successfully completed.
```

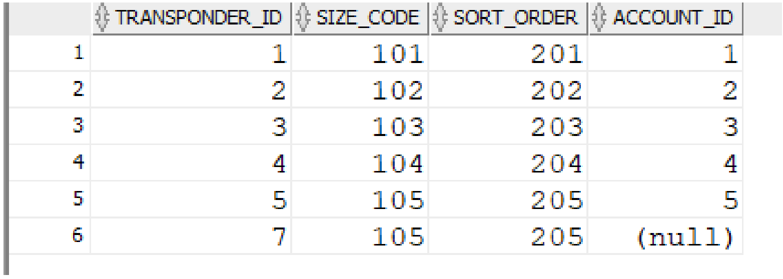
--Feature 5

-- Add Transponder

```
set serveroutput on;
create or replace procedure feature_5_add(Acc_id in int,T_id in int)
as
   check_Tid number;
   check_Aid number;
begin
select count(*) into check_Tid from Transponder t where t.transponder_id = T_id and t.Account_id = Acc_id;
select count(*) into check_Aid from Account where Account.Account_id=Acc_id;
   if  check_Tid = 0 then
       if check_Aid = 1 then
           insert into Transponder values(T_id,106,206,Acc_id);
           insert into Message values(Message_id_seq.nextval,Acc_id,sysdate,'Transponder has been added');
       else
           dbms_output.put_line('Invalid Account');
       end if;
  else
       dbms_output.put_line('Account_id already exists');
  end if;
Exception
   When no_data_found then
   dbms_output.put_line('No data found');
end;
```

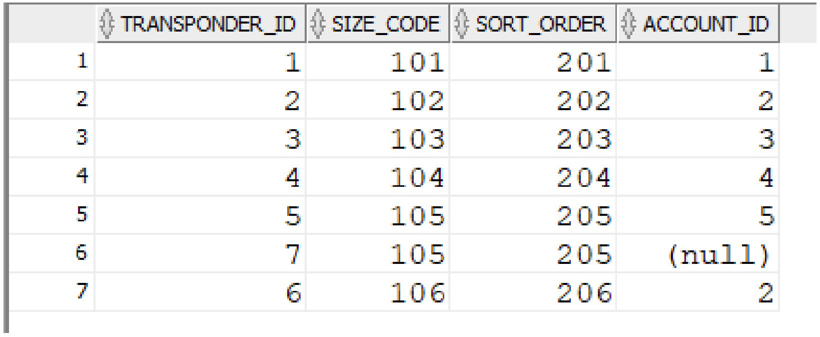
--Output

--Before Adding Transponder



	TRANSPONDER_ID	SIZE_CODE	SORT_ORDER	ACCOUNT_ID
1	1	101	201	1
2	2	102	202	2
3	3	103	203	3
4	4	104	204	4
5	5	105	205	5
6	7	105	205	(null)

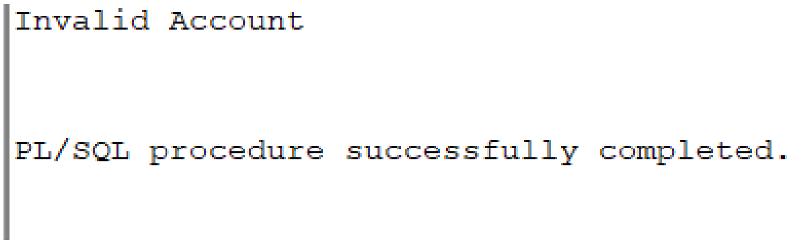
--After Adding Transponder



	TRANSPONDER_ID	SIZE_CODE	SORT_ORDER	ACCOUNT_ID
1	1	101	201	1
2	2	102	202	2
3	3	103	203	3
4	4	104	204	4
5	5	105	205	5
6	7	105	205	(null)
7	6	106	206	2

```
exec feature_5_add (2,6);
```

--Invalid Account



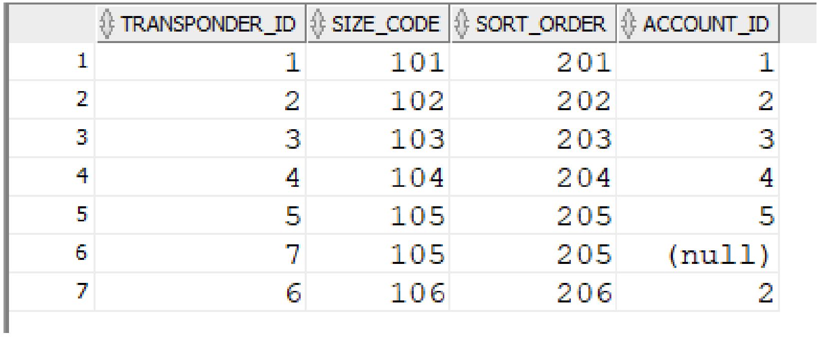
```
Invalid Account


PL/SQL procedure successfully completed.
```

--Deletion of Transponder

```
Set serveroutput on;
create or replace procedure Delete_transponder(Acc_id in int,Transpond_id in int)
as
A_id int;
T_id int;
begin
    -- check if valid account id
    select count(*) into A_id from account a
    where a.account_id=Acc_Id;
    if A_id = 0 then
        dbms_output.put_line('Account id is invalid');
    else
        -- check if license plate matches
        select count(*) into T_id from Transponder t where t.transponder_id = Transpond_id and t.Account_id = Acc_id;
        if T_id=0 then
          dbms_output.put_line('Transponder does not match any existing transponder in that account.');
        else
        if A_id <> 0 and T_id <> 0 then
                    -- Delete references


                     delete from video_bill vb
                        where vb.Trip_Id in (select t.Trip_Id  from trip t
                                           where Account_id=A_id and transponder_id=Transpond_id);

                     delete from trip
                       where Account_id=A_id and Transponder_Id=Transpond_id;

                     delete from transponder
                        where Account_id=A_id and transponder_id=transpond_id;
                    insert into message values(Message_id_seq.nextval,Acc_Id,sysdate,'Transponder has been deleted');

                end if;
            end if;
        end if;

Exception
  when no_data_found
  then
    dbms_output.put_line('No data found');
end;
```

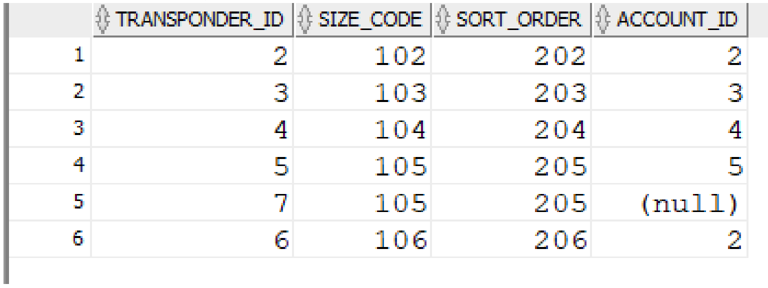
--Output

--Before deleting transponder



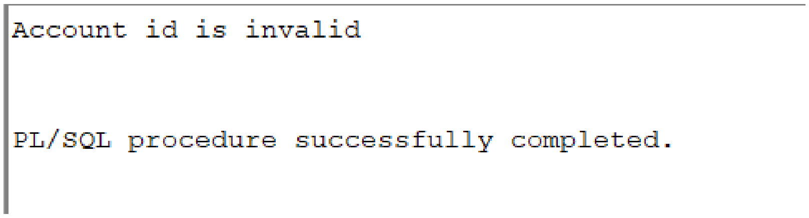
	TRANSPONDER_ID	SIZE_CODE	SORT_ORDER	ACCOUNT_ID
1	1	101	201	1
2	2	102	202	2
3	3	103	203	3
4	4	104	204	4
5	5	105	205	5
6	7	105	205	(null)
7	6	106	206	2

--After deleting transponder



	TRANSPONDER_ID	SIZE_CODE	SORT_ORDER	ACCOUNT_ID
1	2	102	202	2
2	3	103	203	3
3	4	104	204	4
4	5	105	205	5
5	7	105	205	(null)
6	6	106	206	2

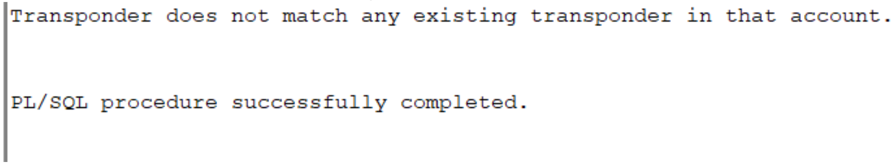
--Invalid Account



```
Account id is invalid


PL/SQL procedure successfully completed.
```

exec delete_transponder(8,1);

--Transponder does not exist in any account



```
Transponder does not match any existing transponder in that account.


PL/SQL procedure successfully completed.
```

exec delete_transponder(2,1);

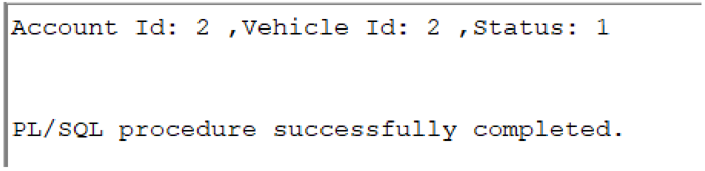
--Feature 6

```
create or replace function feature_6_function(trans_id int, licence_p
varchar, state_s varchar) return int
  is
  t_transponder_id transponder.transponder_id%type;
  l_license vehicle.License_Platenumber%type;
  s_state vehicle.Vehicle_State%type;
  a_account_id account.account_id%type;
  v_vehicle_id vehicle.vehicle_id%type;
  vehicle_check int;
  all_ok int;
  begin


    select count(*) into a_account_id from account,vehicle
    where account.account_id=vehicle.account_id and
vehicle.license_platenumber=licence_p and
vehicle.vehicle_state=state_s;

    select account_id into vehicle_check from vehicle  where
vehicle.license_platenumber=licence_p and
vehicle.vehicle_state=state_s;
    select count(*) into all_ok from transponder
    where account_id=vehicle_check and transponder_id=trans_id;

    select count(*) into t_transponder_id from transponder where
transponder.transponder_id=trans_id;

    select count(*) into l_license from vehicle where
vehicle.license_platenumber=licence_p and
vehicle.vehicle_state=state_s;
    select count(*) into  s_state from vehicle where
vehicle.license_platenumber=licence_p and
vehicle.vehicle_state=state_s;


      if all_ok=1 then
        return 1;
      end if;

    if (trans_id is null or t_transponder_id=0) and a_account_id=1 and
(l_license=1 and s_state=1)  then

        return 2;
      end if;

      if (trans_id is null or t_transponder_id=0) and a_account_id=0
and (l_license=1 and s_state=1)  then
        return 3;
     else return 0;
       end if;
end;
```

```
create or replace procedure feature_6_procedure(trans_id int, licence_p varchar, state_s varchar) is
 t_transponder_id transponder.transponder_id%type;
 l_license vehicle.License_Platenumber%type;
 s_state vehicle.Vehicle_State%type;
 a_account_id account.account_id%type;
 v_vehicle_id vehicle.vehicle_id%type;
 return_val varchar(10);
 begin
 return_val:=feature_6_function(trans_id, licence_p, state_s);
 if return_val=1 then
 select vehicle_id,account.account_id into v_vehicle_id,a_account_id from vehicle, account where
account.account_id=vehicle.account_id and License_Platenumber=licence_p and vehicle_state=state_s;
 dbms_output.put_line('Account Id: '||a_account_id||' ,Vehicle Id: '||v_vehicle_id||' ,Status: 1');
 elsif return_val=2 then
 select vehicle_id,account.account_id into v_vehicle_id,a_account_id from vehicle, account where
account.account_id=vehicle.account_id and License_Platenumber=licence_p and vehicle_state=state_s;
 dbms_output.put_line('Account Id: '||a_account_id||' ,Vehicle Id: '||v_vehicle_id||' ,Status: 2');
 elsif return_val=0 then
 insert into message values(message_id_seq.nextval, a_account_id, sysdate, 'Vehicle is not associated with any account but
transponder is used');
 dbms_output.put_line('Vehicle is not associated with any account but transponder is used');
 else
 select vehicle_id into v_vehicle_id from vehicle where License_Platenumber=licence_p and vehicle_state=state_s;
 dbms_output.put_line('Vehicle Id: '||v_vehicle_id||' ,Status: 3');
 end if;
 end;
```

--Output

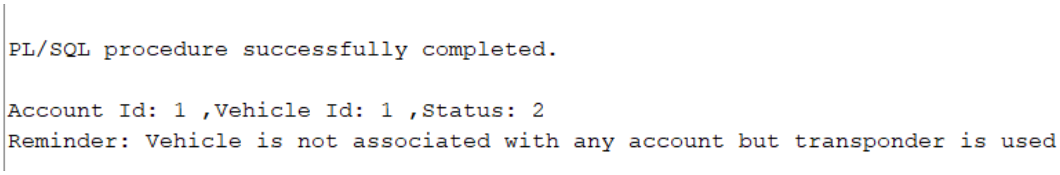
-- All parameters exist



```
Account Id: 2 ,Vehicle Id: 2 ,Status: 1


PL/SQL procedure successfully completed.
```

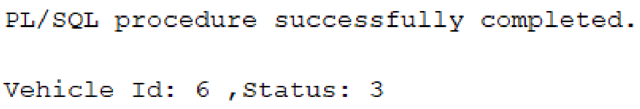
exec feature_6_procedure(2, 'y123','VA');

--Account exists but transponder is null / not valid



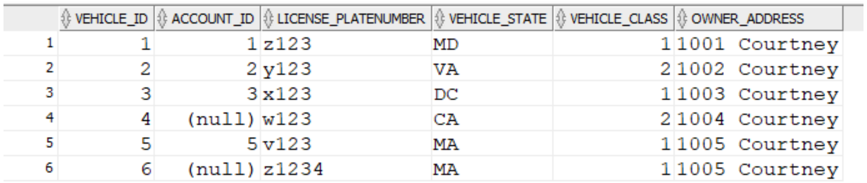
```
PL/SQL procedure successfully completed.

Account Id: 1 ,Vehicle Id: 1 ,Status: 2
Reminder: Vehicle is not associated with any account but transponder is used
```

--Account does not exist



```
PL/SQL procedure successfully completed.

Vehicle Id: 6 ,Status: 3
```

Vehicle table



	VEHICLE_ID	ACCOUNT_ID	LICENSE_PLATENUMBER	VEHICLE_STATE	VEHICLE_CLASS	OWNER_ADDRESS
1	1	1	z123	MD	1	1001 Courtney
2	2	2	y123	VA	2	1002 Courtney
3	3	3	x123	DC	1	1003 Courtney
4	4	(null)	w123	CA	2	1004 Courtney
5	5	5	v123	MA	1	1005 Courtney
6	6	(null)	z1234	MA	1	1005 Courtney

--Feature 7

```
create or replace procedure feature_7(trans_id int, licence_p varchar, state_s varchar, road_id_inp varchar, enter_exit_id int,
exit_exit_id int, enter_time date) is
a_account_id account.account_id%type;
v_vehicle_id vehicle.vehicle_id%type;
t_toll_status trip.trip_status%type;
vehicle_discount float;
v_vehicle_class vehicle.vehicle_class%type;
computed_toll float;
admin_fee_val float;
t_transponder_id transponder.transponder_id%type;
return_val int;
s_start_eid int;
e_end_id int;
d_direction varchar(10);

Cursor c1 is select start_eid, end_eid, direction from toll_rate
        where road_id=road_id_inp
        and start_eid>=enter_exit_id and end_eid<=exit_exit_id and start_eid<end_eid;
Cursor c2 is select start_eid, end_eid, direction from toll_rate
        where road_id=road_id_inp
        and start_eid<=enter_exit_id and end_eid>=exit_exit_id and start_eid>end_eid;
begin
return_val := feature_6_function(trans_id, licence_p, state_s);
select vehicle_class into v_vehicle_class from vehicle where license_platenumber=licence_p and vehicle_state=state_s;
select vehicle_id into v_vehicle_id from vehicle where license_platenumber=licence_p and vehicle_state=state_s;
  if(return_val=1 or return_val=2) then
  select discountrate into vehicle_discount from discount_rate;
    if return_val=1 then
      if(v_vehicle_class=1) then

        select account_id into a_account_id from transponder where transponder_id=trans_id;

        if(enter_exit_id<exit_exit_id) then

          open c1;
          loop
            fetch c1 into s_start_eid, e_end_id, d_direction;
            exit when c1%notfound;
            computed_toll:=0;
            select sum(car_tollrate) into computed_toll from toll_rate
            where road_id=road_id_inp
            and start_eid>=s_start_eid and end_eid<=e_end_id and start_eid<end_eid group by road_id;
```

```
                        computed_toll:=computed_toll*vehicle_discount;

                        insert into trip values(trip_id_updation.nextval, a_account_id,
trans_id,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                     end loop;
                     close c1;

                else
                    open c2;
                    loop
                        fetch c2 into s_start_eid, e_end_id, d_direction;
                         exit when c2%notfound;
                         computed_toll:=0;
                        select sum(car_tollrate) into computed_toll from toll_rate
                        where road_id=road_id_inp
                        and start_eid<=s_start_eid and end_eid>=e_end_id and start_eid>end_eid group by road_id;

                        computed_toll:=computed_toll*vehicle_discount;

                        insert into trip values(trip_id_updation.nextval, a_account_id,
trans_id,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                    end loop;
                    close c2;

                end if;
            else
                if(enter_exit_id<exit_exit_id) then
                    open c1;
                    loop
                        fetch c1 into s_start_eid, e_end_id, d_direction;
                        exit when c1%notfound;
                        computed_toll:=0;
                        select sum(truck_tollrate) into computed_toll from toll_rate
                        where road_id=road_id_inp
                        and start_eid>=s_start_eid and end_eid<=e_end_id and start_eid<end_eid group by road_id;

                        computed_toll:=computed_toll*vehicle_discount;

                        insert into trip values(trip_id_updation.nextval, a_account_id,
trans_id,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                 end loop;
                    close c1;
                else
                    open c2;
                    loop
                        fetch c2 into s_start_eid, e_end_id, d_direction;
                        exit when c2%notfound;
                        computed_toll:=0;
                        select sum(truck_tollrate) into computed_toll from toll_rate
                        where road_id=road_id_inp
                        and start_eid<=s_start_eid and end_eid>=e_end_id and start_eid>end_eid group by road_id;
                        computed_toll:=computed_toll*vehicle_discount;
                        insert into trip values(trip_id_updation.nextval, a_account_id,
trans_id,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                    end loop;
                    close c2;

                end if;
        end if;
    else

                if(v_vehicle_class=1) then
                select vehicle_id into v_vehicle_id from vehicle where license_platenumber=licence_p and vehicle_state=state_s;

                 select account_id into a_account_id from vehicle where vehicle_id=v_vehicle_id;
                select discountrate into vehicle_discount from discount_rate;
                if(enter_exit_id<exit_exit_id) then

                     open c1;
                     loop
                        fetch c1 into s_start_eid, e_end_id, d_direction;
                        exit when c1%notfound;
                        computed_toll:=0;
                        select sum(car_tollrate) into computed_toll from toll_rate
                        where road_id=road_id_inp
                        and start_eid>=s_start_eid and end_eid<=e_end_id and start_eid<end_eid group by road_id;

                        computed_toll:=computed_toll*vehicle_discount;

                        insert into trip values(trip_id_updation.nextval, a_account_id,
null,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);
                     end loop;

                     close c1;

                else
                    open c2;
                    loop
                        fetch c2 into s_start_eid, e_end_id, d_direction;
                        exit when c2%notfound;
                        computed_toll:=0;
                        select sum(car_tollrate) into computed_toll from toll_rate
                        where road_id=road_id_inp
                        and start_eid<=s_start_eid and end_eid>=e_end_id and start_eid>end_eid group by road_id;

                        computed_toll:=computed_toll*vehicle_discount;

                        insert into trip values(trip_id_updation.nextval, a_account_id,
null,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                end loop;

                    close c2;

                end if;
```

```
            else
                if(enter_exit_id<exit_exit_id) then
                    open c1;
                    loop
                        fetch c1 into s_start_eid, e_end_id, d_direction;
                        exit when c1%notfound;
                        computed_toll:=0;
                        select sum(truck_tollrate) into computed_toll from toll_rate
                        where road_id=road_id_inp
                        and start_eid>=s_start_eid and end_eid<=e_end_id and start_eid<end_eid group by road_id;

                        computed_toll:=computed_toll*vehicle_discount;

                        insert into trip values(trip_id_updation.nextval, a_account_id,
null,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                    end loop;
                    close c1;
                else
                    open c2;
                    loop
                        fetch c2 into s_start_eid, e_end_id, d_direction;

                         exit when c2%notfound;
                         computed_toll:=0;
                        select sum(truck_tollrate) into computed_toll from toll_rate
                        where road_id=road_id_inp
                        and start_eid<=s_start_eid and end_eid>=e_end_id and start_eid>end_eid group by road_id;

                        computed_toll:=computed_toll*vehicle_discount;
                        insert into trip values(trip_id_updation.nextval, a_account_id,
null,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                    end loop;
                    close c2;

                end if;

         end if;
        end if;

    else
         select video_toll,admin_fee into vehicle_discount,admin_fee_val from discount_rate;
        if(enter_exit_id<exit_exit_id) then
        open c1;
        loop
        fetch c1 into s_start_eid, e_end_id, d_direction;
        exit when c1%notfound;
        computed_toll:=0;
             select car_tollrate into computed_toll from toll_rate
                where road_id=road_id_inp
                and start_eid>=s_start_eid and end_eid<=e_end_id and start_eid<end_eid;

            computed_toll:=computed_toll*vehicle_discount+admin_fee_val;
            insert into trip values(trip_id_updation.nextval, null,
null,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                end loop;
                close c1;

        else
        open c2;
         loop
        fetch c2 into s_start_eid, e_end_id, d_direction;

         exit when c2%notfound;
         computed_toll:=0;
        select car_tollrate into computed_toll from toll_rate
                where road_id=road_id_inp
                and start_eid<=s_start_eid and end_eid>=e_end_id and start_eid>end_eid;


            computed_toll:=computed_toll*vehicle_discount+admin_fee_val;


                    insert into trip values(trip_id_updation.nextval, null,
null,v_vehicle_id,road_id_inp,s_start_eid,e_end_id,enter_time,return_val,0,computed_toll,d_direction);

                end loop;
                close c2;
        end if;
    end if;
end;
```

--Output

```
exec feature_7(1,'z123','MD','I95',143,152,sysdate);

Toll 143 to 149 = 1  …… After Discount 50% ….   0.5
Toll 149 to 152 = 3 ……. After Discount 50% …… 1.5

exec feature_7(7,'z123','MD','I95',143,152,sysdate);

Toll 143 to 149 = 1  …… After Discount 50% ….   0.5
Toll 149 to 152 = 3 ……. After Discount 50% …… 1.5

exec feature_7(6,'z1234','MA','I495',120,100,sysdate);

Toll 120 to 110 = 7 …… 100% video toll + admin fee ….   12
Toll 110 to 100 = 2 ……. 100% video toll + admin fee …… 7
```

--Feature 8

```
create or replace function compute_video_toll(t_trip_id int,
entering_id int, exiting_id int)
```

```
return float is
e_entering int;
e_exiting int;
computed_toll float;
admin_fee_val float;
vehicle_discount float;
e_entrance_time trip.entrance_time%type;
tripid int;
vehi int;
acc_id int;
v_vehicle_class int;
Begin
select entrance_time into e_entrance_time from trip where trip_id=t_trip_id;
select account_id into acc_id from trip where trip_id=t_trip_id;
select vehicle_class into v_vehicle_class from vehicle where account_id=acc_id;
if(entering_id<exiting_id) then
            select entering_exitid,exiting_exitid,entrance_time into
e_entering,e_exiting,e_entrance_time from trip where
trip_id=t_trip_id;
            select car_tollrate into computed_toll from toll_rate
where start_eid=e_entering and end_eid=e_exiting and
toll_starttime<=e_entrance_time and toll_endtime>=e_entrance_time;
            select video_toll,admin_fee into
vehicle_discount,admin_fee_val from discount_rate;
            computed_toll:=(computed_toll*vehicle_discount)+admin_fee_val;
        else
         select entering_exitid,exiting_exitid,entrance_time into
e_entering,e_exiting,e_entrance_time from trip where
trip_id=t_trip_id;
            select car_tollrate into computed_toll from toll_rate
where start_eid=e_exiting and end_eid=e_entering and
toll_starttime<=e_entrance_time and toll_endtime>=e_entrance_time;
            computed_toll:=(computed_toll*vehicle_discount)+admin_fee_val;
        end if;
        return computed_toll;
end;


set serveroutput on;
create or replace procedure feature_8 (tripID int)
As
TollDeducted trip.toll_deducted%type;
tripStatus trip.trip_status%type;
Account_balance Account.balance%type;
TollApplied trip.toll%type;
AccountId account.account_id%type;
toll_compute float;
e_entering int;
e_exiting int;
Begin
    select t.toll_deducted into TollDeducted from trip t
    where t.trip_id=tripId;
    select entering_exitid, exiting_exitid into e_entering, e_exiting
    from trip where trip_id=tripID;
    if TollDeducted = 1 then
    dbms_output.put_line('The toll has been deducted');
    else
        select t.trip_status into tripStatus from trip t
        where t.trip_id=tripId;
        if tripStatus in (1,2) then

            select a.balance,t.toll,a.account_id into Account_balance,TollApplied,AccountId from account a,trip t
            where a.account_id=t.account_id and t.trip_id=6;
                if Account_balance >= TollApplied then
                    update Account
                    set balance = balance-tollapplied
                    where Account_id=AccountId;
                    update trip
                    set toll_deducted=1
                    where trip_id=tripid;
                else
                    toll_compute := compute_video_toll(tripid,e_entering,e_exiting);
                    update trip
                    set toll=toll_compute,trip_status=3
                    where trip_id=tripId;
                    insert into video_bill
                    values(Bill_Id.nextval,tripID,sysdate,1);
                    insert into message
                    values(message_id_seq.nextval,AccountId,sysdate,'Please replenish your account');
                    dbms_output.put_line('Please replenish your account');
                end if;
        ELSIF tripStatus = 3 then
             insert into video_bill
             values(Bill_Id.nextval,tripID,sysdate,1);
             dbms_output.put_line('Video Bill Generated');
       end if;
      end if;
end;
```

--Output

```
exec  feature_8(6); --Normal Case

exec  feature_8(16); --Less Balance

exec  feature_8(5); --Video Bill Generated
```

--Feature 9

```
Set serveroutput on;
create or replace procedure feature_9(acc_id in int,Amt in int)
is
check_aid integer;
check_payment integer;
begin
--check if valid account id--
 select count(*) into check_aid from Account where Account_id=acc_id;
 if check_aid = 0 then
    dbms_output.put_line('The account cannot be found');
 else
--insert into payment table and message table--
    insert into payment
    values(payment_id_seq.nextval, Acc_id,sysdate,Amt);
    insert into message
    values(message_id_seq.nextval,Acc_id,sysdate,'payment recieved');
    dbms_output.put_line('Payment Received');
 end if;
end;
```

--Output

```
exec feature_9(1,5); --Normal Case

exec feature_9(10,5); --Account Not Found
```

--Feature 10

```
set serveroutput on;
create or replace procedure feature_10(Vbill_id varchar,New_status int)
IS
v_status int;
check_bill int;
begin
--Check for valid bill id
    select count(*) into check_bill from video_bill where bill_id=Vbill_id;
    if check_bill=0 then
       dbms_output.put_line('Invalid Bill id');
    else
    select vb.status into v_status from video_bill vb, trip t
    where vb.trip_id=t.trip_id and vb.bill_id=Vbill_id;
    if v_status=1 then
         update video_bill
         set status=New_status
         where bill_id=Vbill_id;
         dbms_output.put_line('The status has been changed');
    else
         dbms_output.put_line('No video bill to update');
  end if;
 end if;
end;
```

--Output

```
exec feature_10(3003,2);  --Normal Case

exec feature_10(3010,2); --Invalid Bill

exec feature_10(3002,2); --Bill Status cannot be updated
```

--Feature 11

```
set serveroutput on;
create or replace procedure feature_11 (aa_id in integer,mindate in date, maxdate in date ) is
cursor c1 is select trip.Transponder_id,License_Platenumber, Road_id, Entering_ExitId,Exiting_ExitId, trip.Entrance_Time, toll
from trip, vehicle,transponder where transponder.account_id=trip.account_id and vehicle.account_id=trip.account_id and
trip.account_id=aa_id and entrance_time>=mindate and maxdate>=entrance_time;
cursor c2 is select payment_id, payment_date,amount from payment where account_id=aa_id and payment_date>=mindate and
maxdate>=payment_date;
trans_ponder_id transponder.transponder_id%type;
license_pno vehicle.license_platenumber%type;
r_oad_name road.road_name%type;
en_id trip.Entering_ExitId%type;
ex_id trip.Exiting_ExitId%type;
en_time trip.Entrance_Time%type;
toll_d trip.toll_deducted%type;
payment_d payment.payment_date%type;
amountpaid payment.amount%type;
pay_id int;
a_id integer;
begin
--Check if valid Account Id
select count(*) into a_id from account where account_id=aa_id;
if a_id=0 then
dbms_output.put_line('Entered account id is not valid');
else
open c1;
loop
fetch c1 into trans_ponder_id,license_pno,r_oad_name,en_id,ex_id,en_time,amountpaid;
exit when c1%notfound;
dbms_output.put_line('Transponder id: '|| trans_ponder_id|| ' , license platenumber: '|| license_pno || ' , Road Name: '||r_oad_name|| ' ,
Entering Exit Id: ' ||en_id||' , Exit Exit id: ' ||ex_id|| ' , Entering Time: ' ||en_time||' , Amount Paid: ' ||amountpaid);
end loop;
close c1;
DBMS_OUTPUT.NEW_LINE;
open c2;
loop
fetch c2 into pay_id,payment_d,amountpaid;
exit when c2%notfound;
dbms_output.put_line('Payment id: ' || pay_id || '  payment date: ' || payment_d || '  Amount: ' || amountpaid);
end loop;
close c2;
end if;
end;
```

--Output

```
exec feature_11(1, date '2018-05-09', date '2018-08-15'); --Normal Case

exec feature_11(10, date '2018-05-09', date '2018-08-15'); --Account Invalid
```

--Feature 12

```
set serveroutput on;
create or replace procedure feature_12(acc_id int, last_date date) is
```

```
total_toll float;
total_payment float;
check_acc int;
check_trip int;
begin

select count(*) into check_acc from account where account_id=acc_id;
----Check if valid Account or trip exists
if check_acc=0 then
dbms_output.put_line('Account not found');
elsif check_trip=0 then
dbms_output.put_line('Trip not found');
else
-- Call Feature 11
feature_11(acc_id,last_date,last_date + interval '30' DAY);
DBMS_OUTPUT.NEW_LINE;

--Get total Toll for the account id & date range mentioned
select sum(toll) into total_toll from trip where account_id=acc_id and entrance_time>last_date and entrance_time<=last_date+interval '30' DAY;
dbms_output.put_line('Total toll: ' || total_toll);
DBMS_OUTPUT.NEW_LINE;

--Calculate Total Payment for the account id & date range mentioned
 select sum(amount) into total_payment from payment where account_id=acc_id and payment_date>last_date and payment_date<=last_date+interval '30' DAY;
if total_payment is null then total_payment:=0; end if;
dbms_output.put_line('Total Payment: ' || total_payment);
end if;
 end;
```

--Output

```
exec feature_12(1,date '2018-08-09'); --Normal Case

exec feature_12(10,date '2018-08-09'); -- Invalid Account
```

--Feature 13

```
set serveroutput on;
create or replace procedure feature_13(date_range1 date, date_range2 date)
as
--Get Road name and sum in the given date range
cursor c1 is select r.road_name as roadName, sum(t.toll)
from road r, trip t
where
t.entrance_time <= date_range2 and t.entrance_time >=date_range1 and t.road_id=r.road_id
group by r.road_name
order by sum(t.toll) desc;
x int;
R_name varchar(50);
total_toll_deducted int;
begin

---Get count of trips
```

```
    select count(*) into x from trip
    where entrance_time<= date_range2 and entrance_time>=date_range1;
    Open c1;
    loop
        fetch c1 into R_name,total_toll_deducted;
        exit when c1%Notfound;
        dbms_output.put_line('Total Toll for Road Name : ' || R_name ||' Total toll : ' || total_toll_deducted );
    end loop;
    close c1;
    dbms_output.put_line(' Total Number of trips: '|| x );
end;
```

--Output

```
exec feature_13(date '2018-08-09', date '2018-08-11');
```

--Feature 14

```
set serveroutput on;
create or replace procedure feature_14(date_range1 date, date_range2 date,roadId varchar)
as
cursor c1 is  select entering_exitid,count(*) from trip
    where exiting_exitid > entering_exitid
    and trunc(entrance_time) > date_range1
    and trunc(entrance_time) < date_range2
    and road_id = roadId
    group by exiting_exitid, entering_exitid
    order by count(*) desc;
-- This is in descending order
cursor c2 is  select entering_exitid,count(*) from trip
    where exiting_exitid < entering_exitid
    and trunc(entrance_time) > date_range1
    and trunc(entrance_time) < date_range2
    and road_id = roadId
    group by exiting_exitid, entering_exitid
    order by count(*) desc;
x int;
R_name varchar(50);
Entering_id int;
exiting_id int;
exiting_exitid int;
total_trips int;
direction varchar(20);
RoadIdcount int;
begin
--Get road count matching the given road id
    select count(*) into RoadIdcount from trip
    where  road_id=roadId;
--Check for invalid road Id
if RoadIdcount = 0 then
    dbms_output.put_line('Invalid Road Id');
else
 open c1;
```

```
    loop
        fetch c1 into Entering_id, total_trips;
        exit when c1%notfound;
        dbms_output.put_line('Exit Id : ' || Entering_id || ', Direction : '|| ', Increasing Exit Id '||', Total Trips ' ||total_trips);
     end loop;
  close c1;
open c2;
    loop
         fetch c2 into exiting_id, total_trips;
         exit when c2%notfound;
         dbms_output.put_line('Exit Id : ' || Exiting_Id || ', Direction : '|| ' Decresing Exit Id '||', Total Trips ' ||total_trips);
    end loop;
Close c2;
end if;
end;
```

--Output

```
exec feature_14(date'2018-08-09',date'2018-12-20','I495');
```

--Feature 15

```
create or replace procedure Feature_15(start_date in date,end_date
in date,k in VARCHAR) AS
Cursor c1 is select * from
(select account.Account_id,sum(toll),count(trip_id),trip.vehicle_id from
account,trip
where trip.Account_id = account.Account_id
and trunc(entrance_time) > start_date
and trunc(entrance_time) < end_date
group by account.Account_id, trip.vehicle_id order by sum(toll) desc)
where rownum <= k;

Cursor c2 is select * from
(select trip.vehicle_id, sum(toll),count(toll)
from trip,vehicle
where trip.vehicle_id=vehicle.vehicle_id and
trunc(entrance_time) > start_date
and trunc(entrance_time) < end_date
and trip.trip_status = 3
group by trip.vehicle_id
order by count(trip.trip_id) desc)
where rownum <= k;


account_id account.Account_id%type;
total_toll number;
count_trips number;
veh_ID vehicle.vehicle_id%type;
total_toll1 number;
count_trips1 number;
```

```
vehicle1 number;
vehicle2 number;

BEGIN
open c1;
dbms_output.put_line('Toll Bill');
   Loop
      fetch c1 into account_id,total_toll,count_trips,vehicle1;
      exit when c1%NOTFOUND;

      dbms_output.put_line('Account ID: '||account_id||' '||'Toll: '||total_toll||' Number of Trips: '||count_trips || 'Vehicle ID: '||vehicle1);
   end loop;
close c1;
open c2;
dbms_output.put_line('--------------');
dbms_output.put_line('Video Toll Bill');
   Loop
      fetch c2 into veh_ID,total_toll,count_trips;
      exit when c2%NOTFOUND;
      dbms_output.put_line('Vehicle ID: '||veh_ID||' '||'Total Toll: '||total_toll||' Number of Trips: '||count_trips);
   end loop;
close c2;
Exception
   when no_data_found then
      dbms_output.put_line( 'No data found');
END;
```

--Output

```
exec Feature_15(date '2018-08-09',date '2018-11-30’,3);
```

--Feature 16

```
create or replace procedure feature_16 (mindate in date, maxdate in date ) as
cursor c1 is select  t1.vehicle_id, t2.vehicle_id, t1.road_id, t1.entering_exitid, t1.exiting_exitid, t1.entrance_time ,t2.entrance_time
from trip t1, trip t2
where t1.trip_id <> t2.trip_id
and t1.road_id=t2.road_id
and t1.entering_exitid=t2.entering_exitid
and t1.exiting_exitid=t2.exiting_exitid
and (t1.entrance_time<=t2.entrance_time+interval '30' minute and t1.entrance_time >= t2.entrance_time - interval '30' minute)
and t1.vehicle_id>t2.vehicle_id
and trunc(t1.entrance_time) >= mindate and trunc(t1.entrance_time) <= maxdate;

v_ehicle_id1 trip.vehicle_id%type;
v_ehicle_id2 trip.vehicle_id%type;
r_oad_id trip.road_id%type;
e_ntering_exitid trip.entering_exitid%type;
e_xiting_exitid trip.exiting_exitid%type;
e_ntrance_time1 trip.entrance_time%TYPE;
e_ntrance_time2 trip.entrance_time%TYPE;
begin

  open c1;
```

```
    loop
         fetch c1 into v_ehicle_id1,v_ehicle_id2,r_oad_id,e_ntering_exitid,e_xiting_exitid,e_ntrance_time1,e_ntrance_time2;

         exit when c1%notfound;

         dbms_output.put_line(' Vehicle id1 : '|| v_ehicle_id1|| ' , Vehicle id2 : '|| v_ehicle_id2|| ', Road id: '|| r_oad_id|| ', Entering Exit Id: 
'|| e_ntering_exitid||'
         ,Exiting Exit Id: ' || e_xiting_exitid || ', Entrance Time 1: '|| e_ntrance_time1 ||', Entrance Time 2: '|| e_ntrance_time2 );
      end loop;
    close c1;

    EXCEPTION when no_data_found
    then
        dbms_output.put_line(' No pairs of cars travelled during the range of time mentioned ');
end;
```

--Output

```
exec feature_16( date '2018-09-09', date '2018-09-11');
```