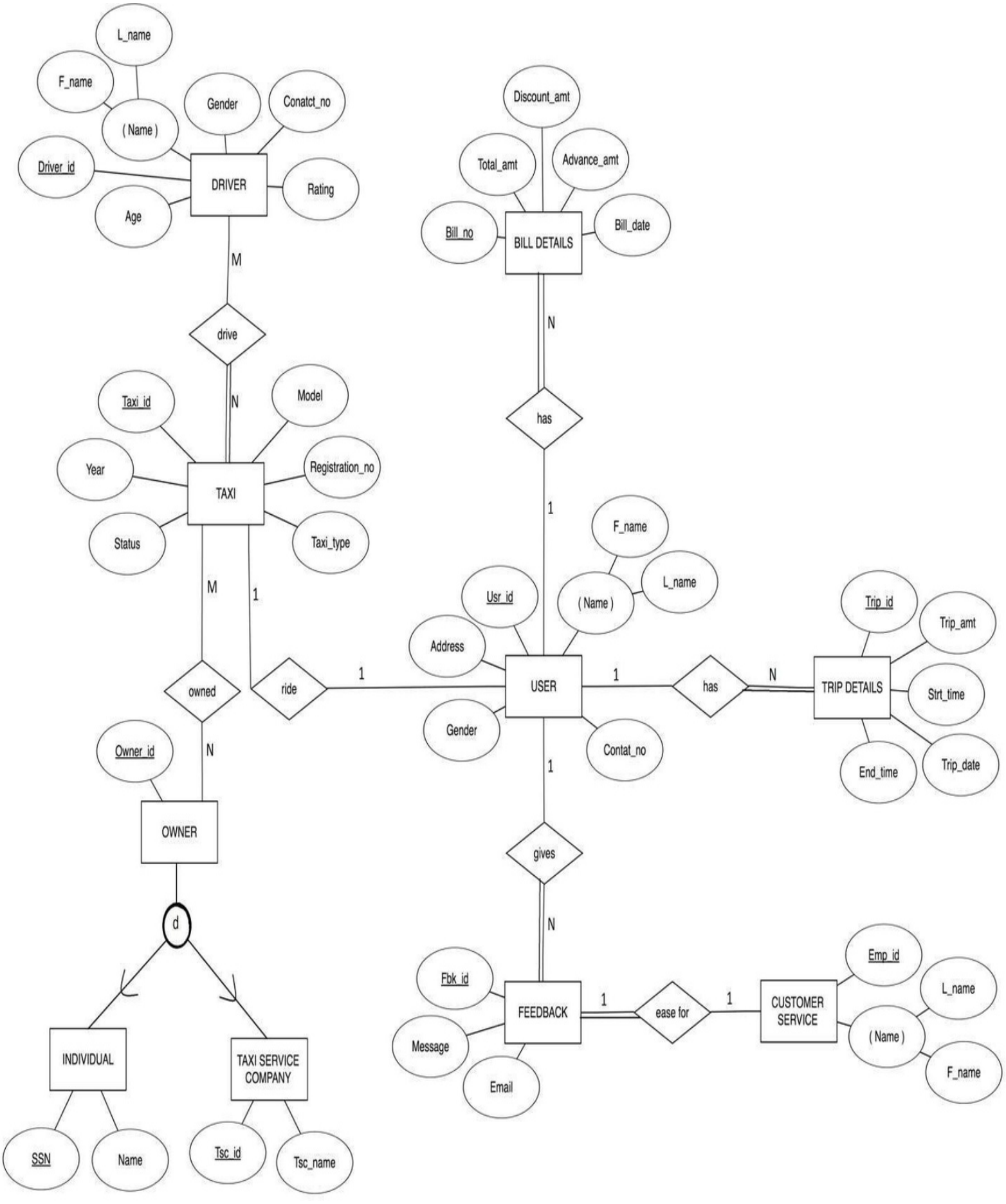
TAXI MANAGEMENT SYSTEM

INDEX

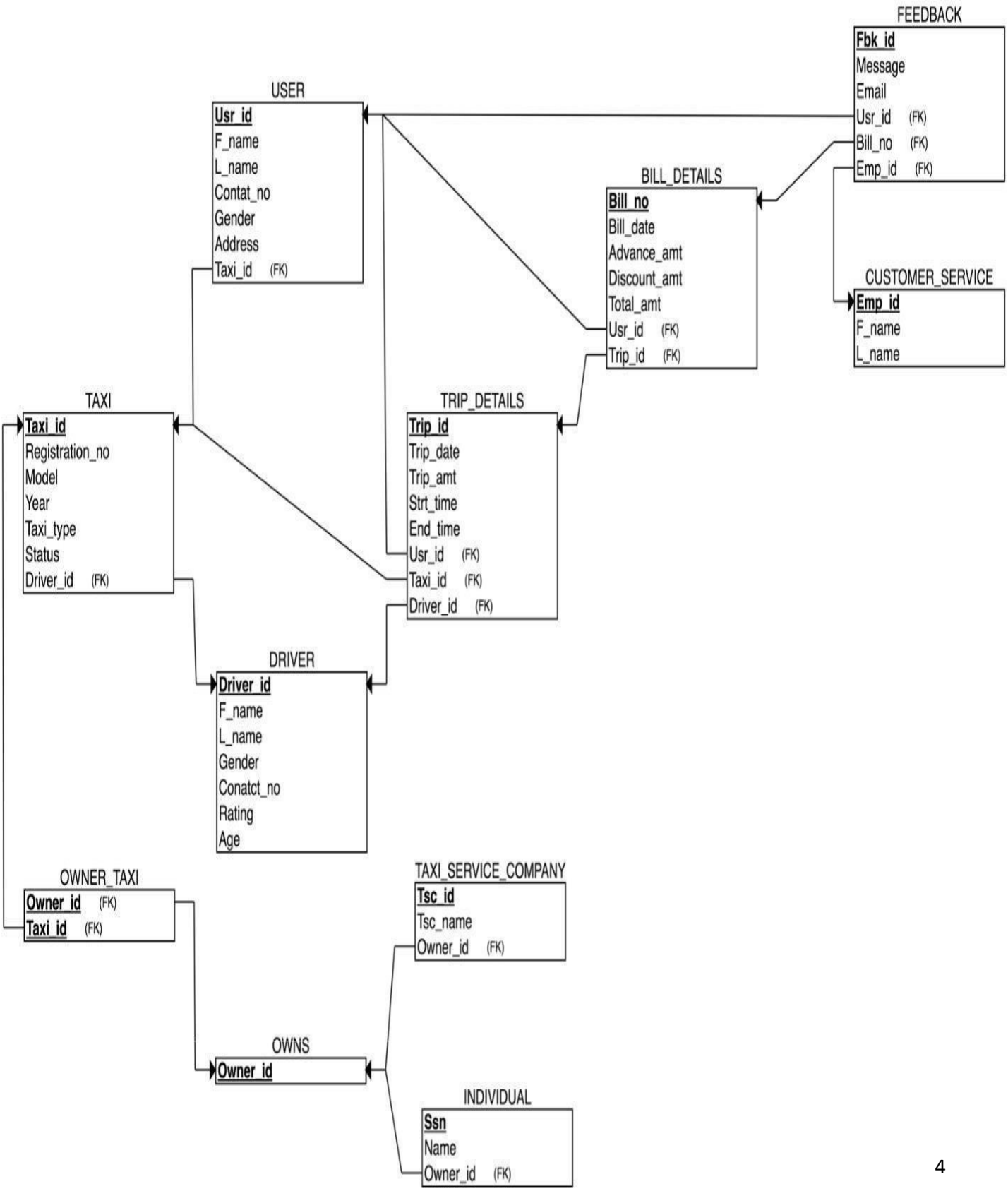
PROBLEM STATEMENT

Many individuals reserve taxis throughout the day. Booking taxis, managing trip details, managing passenger details, managing driver details, managing bill details, and maintaining customer service and feedback are only some of the activities involved in the day-to-day operations of taxis. This project attempts to demonstrate how databases can help handle data linked to these responsibilities.

ER DIAGRAM


L_name
F_name
(Name)
Gender
Conatct_no
Driver_id
DRIVER
Rating
Age
M
drive
N
Taxi_id
Model
Year
TAXI
Registration_no
Status
Taxi_type
M
1
owned
ride
1
Owner_id
N
OWNER
d
INDIVIDUAL
TAXI SERVICE COMPANY
SSN
Name
Tsc_id
Tsc_name
Discount_amt
Total_amt
Advance_amt
Bill_no
BILL DETAILS
Bill_date
N
has
1
F_name
L_name
(Name)
Usr_id
Address
USER
Gender
Contat_no
1
1
has
N
Trip_id
Trip_amt
TRIP DETAILS
Strt_time
End_time
Trip_date
1
gives
N
Fbk_id
FEEDBACK
Message
Email
1
ease for
1
Emp_id
CUSTOMER SERVICE
L_name
(Name)
F_name

ER TO TABLES


FEEDBACK
Fbk_id
Message
Email
Usr_id (FK)
Bill_no (FK)
Emp_id (FK)
USER
Usr_id
F_name
L_name
Contat_no
Gender
Address
Taxi_id (FK)
BILL_DETAILS
Bill_no
Bill_date
Advance_amt
Discount_amt
Total_amt
Usr_id (FK)
Trip_id (FK)
CUSTOMER_SERVICE
Emp_id
F_name
L_name
TAXI
Taxi_id
Registration_no
Model
Year
Taxi_type
Status
Driver_id (FK)
TRIP_DETAILS
Trip_id
Trip_date
Trip_amt
Strt_time
End_time
Usr_id (FK)
Taxi_id (FK)
Driver_id (FK)
DRIVER
Driver_id
F_name
L_name
Gender
Conatct_no
Rating
Age
OWNER_TAXI
Owner_id (FK)
Taxi_id (FK)
TAXI_SERVICE_COMPANY
Tsc_id
Tsc_name
Owner_id (FK)
OWNS
Owner_id
INDIVIDUAL
Ssn
Name
Owner_id (FK)

TAXI

Taxi_id	Registration_no	Taxi_Model	Taxi_Year	Taxi_type	Status	Driver_id

- Primary Key: Taxi_id
- Foreign Keys: Driver_id

USER_TBL

Usr_id	F_name	L_name	Contat_no	Gender	Address	Taxi_id

- Primary Key: Usr_id
- Foreign Keys: Taxi_id

DRIVER

Driver_id	F_name	L_name	Gender	Conatct_no	Rating	Age

- Primary Key: Driver_id
- Foreign Keys: NA

TRIP_DETAILS

Trip_id	Trip_date	Trip_amt	Driver_id	Usr_id	Taxi_id	Strt_time	End_time

- Primary Key: Trip_id
- Foreign Keys: Taxi_id, Usr_id, Driver_id

BILL_DETAILS

Bill_no	Bill_date	Advance_amt	Discount_amt	Total_amt	Usr_id	Trip_id

- Primary Key: Bill_no
- Foreign Keys: Usr_id, Trip_id

CUSTOMER_SERVICE

Emp_id	F_name	L_name

- Primary Key: Emp_id
- Foreign Keys: NA

FEEDBACK

Fbk_id	Message	Email	Emp_id	Usr_id	Trip_id

- Primary Key: Fbk_id
- Foreign Keys: Usr_id, Emp_id, Trip_id

OWNER_TAXI

Owner_id	Taxi_id

- Primary Key: Owner_id, Taxi_id
- Foreign Keys: Owner_id, Taxi_id

OWNS

Owner_id	No_Cars

- Primary Key: Owner_id
- Foreign Keys: NA

INDIVIDUAL

Ssn	Name	Owner_id

- Primary Key: Ssn
- Foreign Keys: Owner_id

TAXI_SERVICE_COMPANY

Tsc_id	Tsc_name	Owner_id

- Primary Key: Tsc_id
- Foreign Keys: Owner_id

NORMALIZATION

TAXI
{Taxi_id Registration_no, Taxi_Model, Taxi_Year, Taxi_type, Status}

USER
{Usr_id F_name, L_name, Contat_no, Gender, Address, Taxi_id}
DRIVER
{Driver_id F_name, L_name, Gender, Conatct_no, Rating, Age}
TRIP_DETAILS
{Trip_id Trip_date, Trip_amt, Driver_id, Usr_id, Taxi_id, Strt_time, End_time}
BILL_DETAILS
{Bill_no Bill_date, Advance_amt, Discount_amt, Total_amt, Usr_id, Trip_id}

CUSTOMER_SERVICE
{Emp_id F_name, L_name}
FEEDBACK
{Fbk_id Message, Email, Emp_id, Usr_id, Trip_id}

OWNER_TAXI
{Owner_id Taxi_id}
OWNS
{Owner_id No_Cars}
INDIVIDUAL
{Ssn Name, Owner_id}
TAXI_SERVICE_COMPANY
{Tsc_id Tsc_name, Owner_id}

SQL/PLSQL CODE

SQL STATEMENTS FOR TABLE CREATION

```
CREATE TABLE TAXI (
Taxi_id integer NOT NULL,
Registration_no VARCHAR(20),
Taxi_Model VARCHAR(20),
Taxi_Year DATE,
Taxi_type VARCHAR(20),
Status VARCHAR(20),
Driver_id integer,
PRIMARY KEY (Taxi_id),
UNIQUE (Registration_no)
);
CREATE TABLE USER_TBL (
Usr_id integer NOT NULL,
F_name VARCHAR(20),
L_name VARCHAR(20),
Contat_no integer,
Gender VARCHAR(10),
Address VARCHAR(50),
Taxi_id integer,
PRIMARY KEY (Usr_id)
);
CREATE TABLE DRIVER (
Driver_id integer NOT NULL,
F_name VARCHAR(10),
L_name VARCHAR(20),
Gender VARCHAR(10),
Conatct_no VARCHAR(20),
Rating integer,
Age integer,
PRIMARY KEY (Driver_id)
);
CREATE TABLE TRIP_DETAILS (
Trip_id integer NOT NULL,
Trip_date DATE,
Trip_amt decimal(10,2),
Driver_id integer,
Usr_id integer,
Taxi_id integer,
Strt_time TIMESTAMP,
End_time TIMESTAMP,
PRIMARY KEY (Trip_id)
```

```
);
CREATE TABLE BILL_DETAILS (
Bill_no integer NOT NULL,
Bill_date DATE,
Advance_amt decimal(10,2),
Discount_amt decimal(10,2),
Total_amt decimal(10,2),
Usr_id integer,
Trip_id integer,
PRIMARY KEY (Bill_no)
);
CREATE TABLE CUSTOMER_SERVICE (
Emp_id integer NOT NULL,
F_name VARCHAR(20),
L_name VARCHAR(20),
PRIMARY KEY (Emp_id)
);
CREATE TABLE FEEDBACK (
Fbk_id integer NOT NULL,
Message VARCHAR(140),
Email VARCHAR(50),
Emp_id integer,
Usr_id integer,
Trip_id integer,
PRIMARY KEY (Fbk_id)
);
CREATE TABLE OWNS (
Owner_id integer NOT NULL,
No_Cars integer,
PRIMARY KEY (Owner_id)
);
CREATE TABLE OWNER_TAXI (
Owner_id integer NOT NULL,
Taxi_id integer,
PRIMARY KEY (Owner_id, Taxi_id)
);
CREATE TABLE INDIVIDUAL (
Ssn integer NOT NULL,
Name VARCHAR(20),
Owner_id integer,
PRIMARY KEY (Ssn)
);
CREATE TABLE TAXI_SERVICE_COMPANY (
Tsc_id integer NOT NULL,
Tsc_name VARCHAR(20),
Owner_id integer,
PRIMARY KEY (Tsc_id)
```

);

SQL STATEMENTS FOR FOREIGN KEY CREATION

```
ALTER TABLE TAXI ADD CONSTRAINT fketadr FOREIGN KEY (Driver_id)
REFERENCES DRIVER(Driver_id) ON DELETE CASCADE;
ALTER TABLE USER_TBL ADD CONSTRAINT fkusta FOREIGN KEY (Taxi_id)
REFERENCES TAXI(Taxi_id) ON DELETE CASCADE;
ALTER TABLE TRIP_DETAILS ADD CONSTRAINT fktddr FOREIGN KEY
(Driver_id) REFERENCES DRIVER(Driver_id) ON DELETE CASCADE;
ALTER TABLE TRIP_DETAILS ADD CONSTRAINT fktdusr FOREIGN KEY (Usr_id)
REFERENCES USER_TBL(Usr_id) ON DELETE CASCADE;
ALTER TABLE TRIP_DETAILS ADD CONSTRAINT fktdtax FOREIGN KEY
(Taxi_id) REFERENCES TAXI(Taxi_id) ON DELETE CASCADE;
ALTER TABLE BILL_DETAILS ADD CONSTRAINT fkbdtd FOREIGN KEY (Trip_id)
REFERENCES TRIP_DETAILS(Trip_id) ON DELETE CASCADE;
ALTER TABLE BILL_DETAILS ADD CONSTRAINT fkbdusr FOREIGN KEY (Usr_id)
REFERENCES USER_TBL(Usr_id) ON DELETE CASCADE;
ALTER TABLE FEEDBACK ADD CONSTRAINT fkfbemp FOREIGN KEY (Emp_id)
REFERENCES CUSTOMER_SERVICE(Emp_id) ON DELETE CASCADE;
ALTER TABLE FEEDBACK ADD CONSTRAINT fkfbtd FOREIGN KEY (Trip_id)
REFERENCES TRIP_DETAILS(Trip_id) ON DELETE CASCADE;
ALTER TABLE FEEDBACK ADD CONSTRAINT fkfbusr FOREIGN KEY (Usr_id)
REFERENCES USER_TBL(Usr_id) ON DELETE CASCADE;
  ALTER TABLE OWNER_TAXI ADD CONSTRAINT fkeowtax FOREIGN KEY (Taxi_id)
REFERENCES TAXI(Taxi_id) ON DELETE CASCADE;
ALTER TABLE OWNER_TAXI ADD CONSTRAINT fkeowowns FOREIGN KEY
(Owner_id) REFERENCES OWNS(Owner_id) ON DELETE CASCADE;
ALTER TABLE INDIVIDUAL ADD CONSTRAINT fkeinowns FOREIGN KEY
(Owner_id) REFERENCES OWNS(Owner_id) ON DELETE CASCADE;
  ALTER TABLE TAXI_SERVICE_COMPANY ADD CONSTRAINT fketscowns FOREIGN
KEY (Owner_id) REFERENCES OWNS(Owner_id) ON DELETE CASCADE;
```

SQL STATEMENTS FOR INSERT COMMANDS

```
  INSERT  INTO  DRIVER  VALUES(1,'Abhi','Gowda','Male','4693805870',5,25);
  insert into DRIVER Values(2,'abhi','mane','Male','5967125945',4,30);
  insert into DRIVER Values(3,'Abhi','Aguero','Male','7566694250',5,35);
  INSERT INTO CUSTOMER_SERVICE VALUES(1,'abhi','gowda');
  INSERT INTO OWNS VALUES(1,1);
  INSERT INTO OWNS VALUES(2,1);
```

```
INSERT INTO TAXI VALUES(1,'KA-15R-
3367','BENZE300',to_date('01/01/2017','mm/dd/yyyy'),'SUV','Available',1)
INSERT INTO USER_TBL VALUES(1,'USER1','LNAME','123456','Male','MCCAllum','1');
INSERT INTO TRIP_DETAILS
VALUES(1,to_date('01/01/2017','mm/dd/yyyy'),123,1,1,1,TO_TIMESTAMP('2017-01-01
06:14:00', 'YYYY-MM-DD HH24:MI:SS'),TO_TIMESTAMP('2017-01-01 08:14:00', 'YYYY-MM-
DD HH24:MI:SS'));
INSERT INTO BILL_DETAILS
VALUES(1,to_date('01/01/2017','mm/dd/yyyy'),1000.10,20.11,null,1,1);
INSERT INTO FEEDBACK VALUES(1,'not so good','abhi@gmail.com',1,1,1);
INSERT INTO OWNER_TAXI Values (1,1);
INSERT INTO INDIVIDUAL VALUES(123,'abhi owner ind',1);
INSERT INTO TAXI_SERVICE_COMPANY VALUES (1,'abhi taxi comp',2);
```

PL/SQL – PROCEDURES

Procedure Code block for Book_Taxi

```
----------------------------------------
-- Procedure Creation
-- this procedure creates a use_table entry and creates the trip and bill_detail for the trip
-- input parameters : Name , Address, Contact, Taxi_Model, Gender, Advance
----------------------------------------
```

```
CREATE OR REPLACE PROCEDURE BOOK_TAXI
  (Name IN VARCHAR2,v_Address IN VARCHAR2,v_Contact IN VARCHAR2,Taxi_Model IN
VARCHAR2,v_Gender IN VARCHAR2,Advance IN decimal)
IS
BEGIN
DECLARE
v_usr_id INT :=-1;
v_Trip_id INT :=-1;
v_Bill_no INT :=-1;
v_Taxi_id INT :=-1;
v_Driver_id INT :=1;
BEGIN
select MAX(Usr_id)+1 into v_usr_id from USER_TBL ;
```

```
select MAX(Trip_id)+1 into v_Trip_id from TRIP_DETAILS ;
select MAX(Bill_no)+1 into v_Bill_no from BILL_DETAILS ;
select taxi_id,Driver_id into v_Taxi_id,v_Driver_id from TAXI
where Status = 'Available' and Taxi_Model = Taxi_Model;
   insert into USER_TBL values(v_usr_id, SUBSTR (Name, 1,INSTR(Name,' ',1)),SUBSTR (Name,
INSTR(Name,' ',1)+1,LENGTH(Name)),v_Contact,v_Gender,v_Address,v_Taxi_id);
insert into TRIP_DETAILS
values(v_Trip_id,sysdate,50,v_Driver_id,v_usr_id,v_Taxi_id,sysdate,null);
insert into BILL_DETAILS
values(v_Bill_no,null,Advance,null,null,v_usr_id,v_Trip_id);
END;
END;

begin
BOOK_TAXI ('Prateek Rustagi','dwarka','167147869','BENZE300','Male','200.10');
dbms_output.put_line('Taxi booked');
END;
select * from USER_TBL where L_NAME=' Rustagi';
```

Procedure Code block for TRIP_END

```
-- Procedure Creation

-- this procedure will calculate the final amount for the trip and update the
amount attributes in trip and bill details
-- input parameters : trip_id, discount


CREATE OR REPLACE PROCEDURE TRIP_END(v_trip IN INT , v_discount
IN Decimal )
AS
BEGIN
DECLARE
v_total_time INT := -1;
v_bill_no INT :=-1;
BEGIN
select extract(day from (sysdate - Strt_time))*24 + extract(hour
from (sysdate - Strt_time)) into v_total_time from TRIP_DETAILS
where Trip_id = v_trip;
update TRIP_DETAILS set End_time = sysdate where Trip_id =
Trip_id ;
update BILL_DETAILS set Bill_date = sysdate , Discount_amt =
v_discount ,Total_amt = (v_total_time * 15) - v_discount where
```

```
Trip_id = v_trip ;
END ;
END ;


begin
TRIP_END (3,0.00);
END;
dbms_output.put_line('Taxi reached destination');
select * from BILL_DETAILS where Usr_id=3;

select * from TRIP_DETAILS;
```

Procedure Code block for Update_Driver_Rating

```
-- Trigger Creation

-- this trigger is called when inserted(After) to the feedback table

-- the trigger will decrease the driver rating by 1 if user feed back is bad for a driver


CREATE OR REPLACE TRIGGER UPDATE_DRIVER_RATING
AFTER INSERT ON FEEDBACK
FOR EACH ROW
WHEN (NEW.Message like '%Bad Driver%' )
DECLARE
v_driver_id INT;
BEGIN
select driver_id into v_driver_id from TRIP_DETAILS where
trip_id = :NEW.Trip_id;
update DRIVER set Rating = Rating -1 where driver_id =
v_driver_id;
END;
```

Procedure Code block for Add_no_of_cars

-- Trigger Creation

-- this trigger is called before the INSERT OR UPDATE ON OWNS table

-- the trigger will calculate the number of cars owned by the owner and updates the no_of_cars columns in the OWNS table

---

```
CREATE OR REPLACE TRIGGER ADD_NO_OF_CARS
BEFORE INSERT OR UPDATE ON OWNS
FOR EACH ROW
DECLARE
v_no_of_cars INT;
BEGIN
select count(Taxi_id) into v_no_of_cars from OWNER_TAXI where
Owner_id = :NEW.Owner_id group by Owner_id;
:NEW.No_Cars := v_no_of_cars;
END;

CREATE OR REPLACE PROCEDURE Driver_Check(driv_id IN INT)
AS
begin
DECLARE
Reg_no INT;
BEGIN
  select Registration_no into Reg_no from TAXI where driver_id=driv_id;
EXCEPTION
WHEN no_data_found THEN
dbms_output.put_line('No such driver');
END ;
END ;

select * from TAXI;

begin
Driver_Check (15);
END;

create or replace FUNCTION model (Driv_id in number)
RETURN varchar IS
model_name varchar(100);
BEGIN
  select Taxi_Model into model_name from TAXI where driver_id=driv_id;
```

```
RETURN(model_name);
END;

declare
varr varchar(100);
begin
varr := model(1);
dbms_output.put_line(varr);
END;

      CREATE OR REPLACE PROCEDURE Many_Row (First_name IN varchar)
AS
begin
DECLARE
driv_id int;
BEGIN
       select driver_id into driv_id from Driver where F_name= First_name;
EXCEPTION
WHEN too_many_rows THEN
dbms_output.put_line('error trying to select too many rows');
END ;
END ;

select * from Driver

begin
Many_Row ('Abhi');
END;




    CREATE or REPLACE PROCEDURE delete_table (First_name IN varchar2)
AS
begin
declare
last_name varchar(100);
begin
  select L_name into last_name from Driver where F_NAME = First_name;
EXCEPTION
WHEN NO_DATA_FOUND THEN
null;
end;
IF SQL%NOTFOUND THEN
DBMS_OUTPUT.PUT_LINE('RECORD NOT DELETED');
ELSE
DBMS_OUTPUT.PUT_LINE('RECORD DELETED');
END IF;
```

```
   Delete from DRIVER where F_name=First_name;
END;

select * from DRIVER
begin
delete_table('david');
end;
```

SCREENSHOTS

SQL Worksheet

Clear Find Actions Save Run

```
INSERT INTO TAXI_SERVICE_COMPANY VALUES (1,'abhi taxi comp',2);

CREATE OR REPLACE PROCEDURE BOOK_TAXI
(Name IN VARCHAR2,v_Address IN VARCHAR2,v_Contact IN VARCHAR2,Taxi_Model IN VARCHAR2,v_Gender IN VARCHAR2,Advance IN decimal)
IS
BEGIN
DECLARE
v_usr_id INT :=-1;
v_Trip_id INT :=-1;
v_Bill_no INT :=-1;
v_Taxi_id INT :=-1;
v_Driver_id INT :=1;
BEGIN
select MAX(Usr_id)+1 into v_usr_id from USER_TBL ;
select MAX(Trip_id)+1 into v_Trip_id from TRIP_DETAILS ;
select MAX(Bill_no)+1 into v_Bill_no from BILL_DETAILS ;
select taxi_id,Driver_id into v_Taxi_id,v_Driver_id from TAXI
where Status = 'Available' and Taxi_Model = Taxi_Model;
insert into USER_TBL values(v_usr_id, SUBSTR (Name, 1,INSTR(Name,' ',1)),SUBSTR (Name, INSTR(Name,' ',1)+1,LENGTH(Name)),v_Contact,v_Gender,v_Address,v_Taxi_id);
insert into TRIP_DETAILS values(v_Trip_id,sysdate,50,v_Driver_id,v_usr_id,v_Taxi_id,sysdate,null);
insert into BILL_DETAILS
values(v_Bill_no,null,Advance,null,null,v_usr_id,v_Trip_id);
END;
END;
```

Procedure created.

```
begin
BOOK_TAXI ('Prateek  Rustagi','dwarka','167147869','BENZE300','Male','200.10');
dbms_output.put_line('Taxi booked');
END;
```

```
Statement processed.
Taxi booked
```

```
select * from USER_TBL where L_NAME=' Rustagi'
```

USR_ID	F_NAME	L_NAME	CONTAT_NO	GENDER	ADDRESS	TAXI_ID
2	Prateek	Rustagi	167147869	Male	dwarka	1

Download CSV

SQL Worksheet

```
CREATE OR REPLACE PROCEDURE TRIP_END(v_trip IN INT , v_discount
IN Decimal )
AS
BEGIN
DECLARE
v_total_time INT := -1;
v_bill_no INT :=-1;
BEGIN
select extract(day from (sysdate - Strt_time))*24 + extract(hour
from (sysdate - Strt_time)) into v_total_time from TRIP_DETAILS
where Trip_id = v_trip;
update TRIP_DETAILS set End_time = sysdate where Trip_id =
Trip_id ;
update BILL_DETAILS set Bill_date = sysdate , Discount_amt =
v_discount ,Total_amt = (v_total_time * 15) - v_discount where
Trip_id = v_trip ;
END ;
END ;
```

Procedure created.

SQL Worksheet

Clear

```
189 select * from TRIP_DETAILS;
190
191 CREATE OR REPLACE TRIGGER UPDATE_DRIVER_RATING
192 AFTER INSERT ON FEEDBACK
193 FOR EACH ROW
194 WHEN (NEW.Message like '%Bad Driver%' )
195 DECLARE
196 v_driver_id INT;
197 BEGIN
198 select driver_id into v_driver_id from TRIP_DETAILS where
199 trip_id = :NEW.Trip_id;
200 update DRIVER set Rating = Rating -1 where driver_id =
201 v_driver_id;
202 END;
203
204 CREATE OR REPLACE TRIGGER ADD_NO_OF_CARS
205 BEFORE INSERT OR UPDATE ON OWNS
206 FOR EACH ROW
207 DECLARE
208
```

TRIP_ID	TRIP_DATE	TRIP_AMT	DRIVER_ID	USR_ID	TAXI_ID	STRT_TIME	END_TIME
2	16-MAY-22	50	1	2	1	16-MAY-22 07.33.21.000000 PM	-
1	01-JAN-17	123	1	1	1	01-JAN-17 06.14.00.000000 AM	01-JAN-17 08.14.00.000000 AM

Download CSV

2 rows selected.

SQL Worksheet

```
189 select * from TRIP_DETAILS;
190
191 CREATE OR REPLACE TRIGGER UPDATE_DRIVER_RATING
192 AFTER INSERT ON FEEDBACK
193 FOR EACH ROW
194 WHEN (NEW.Message like '%Bad Driver%' )
195 DECLARE
196 v_driver_id INT;
197 BEGIN
198 select driver_id into v_driver_id from TRIP_DETAILS where
199 trip_id = :NEW.Trip_id;
200 update DRIVER set Rating = Rating -1 where driver_id =
201 v_driver_id;
202 END;
203
204 CREATE OR REPLACE TRIGGER ADD_NO_OF_CARS
205 BEFORE INSERT OR UPDATE ON OWNS
206 FOR EACH ROW
207 DECLARE
208
```

Trigger created.

```
225 END ;
226 END ;
227
228 select * from TAXI;
229
230 begin
231 Driver_Check (15);
232 END;
233
234 create or replace FUNCTION model (Driv_id in number)
235 RETURN varchar IS
236 model_name varchar(100);
237 BEGIN
238 select Taxi_Model into model_name from TAXI where driver_id=driv_id;
239    RETURN(model_name);
240 END;
241
242 declare
243 varr varchar(100);
244 begin
245
```

TAXI_ID	REGISTRATION_NO	TAXI_MODEL	TAXI_YEAR	TAXI_TYPE	STATUS	DRIVER_ID
1	KA-15R-3367	BENZE300	01-JAN-17	SUV	Available	1

Download CSV

Statement processed.
No such driver

```
241
242 declare
243 varr varchar(100);
244 begin
245  varr := model(1);
246  dbms_output.put_line(varr);
247 END;
248
249 CREATE OR REPLACE PROCEDURE Many_Row (First_name IN varchar)
250 AS
251 begin
252 DECLARE
253 driv_id int;
254 BEGIN
255 select driver_id into driv_id from Driver where F_name= First_name;
256 EXCEPTION
257    WHEN too_many_rows THEN
258       dbms_output.put_line('error trying to select too many rows');
259 END ;
260
```

```
Statement processed.
BENZE300
```

```
251 begin
252 DECLARE
253 driv_id int;
254 BEGIN
255 select driver_id into driv_id from Driver where F_name= First_name;
256 EXCEPTION
257    WHEN too_many_rows THEN
258       dbms_output.put_line('error trying to select too many rows');
259 END ;
260 END ;
261
262 select * from Driver
263
264 begin
265 Many_Row ('Abhi');
266 END;
267
268
269
270 CREATE or REPLACE PROCEDURE delete_table (First_name IN varchar2)
271
```

DRIVER_ID	F_NAME	L_NAME	GENDER	CONATCT_NO	RATING	AGE
1	Abhi	Gowda	Male	4693805870	5	25
2	abhi	mane	Male	5967125945	4	30
3	Abhi	Aguero	Male	7566694250	5	35

Download CSV
3 rows selected.

```
263
264 begin
265 Many_Row ('Abhi');
266 END;
267
268
269
270 CREATE or REPLACE PROCEDURE delete_table (First_name IN varchar2)
271 AS
272 begin
273 declare
274 last_name varchar(100);
275 begin
276 select L_name into last_name from Driver where F_NAME = First_name;
277 EXCEPTION
278                WHEN NO_DATA_FOUND THEN
279                    null;
280 end;
281
```

```
Statement processed.
error trying to select too many rows
```