**DATABASE DESIGN PROJECT REPORT CAR RENTAL SYSTEM**

**TEAM MEMBERS:**

Parth Khandelwal (RA1911003010891)

Yohenba Kshetrimayum (RA1911003010904)

**INDEX**

**1. Introduction 3**

**2. Entities 4**

**3. Relations 5**

**4.Assumptions 6**

**5. ER/EER Diagram 7**

**6. Functional Dependencies 8**

**7. Final relational schema 10**

**8. SQL Statements 11**

**9. PL/SQL Statements 20**

**10. Conclusion 27**

**1. INTRODUCTION**

We have chosen to produce a Car Rental system. In our system, Customer can rent a car based on make and a model. Our system provides customer to have different pick-up and drop off locations and will impose late fee if the rental car is returned beyond the return date and time. The Customers can purchase car rental insurance which is optional and can use upto one discount coupon to their final bill. Customers who have membership will be by default given a 10% discount in their final bill. We will see detailed description below.

1.1 **REQUIREMENTS**

a) Car rental agency should have collection of cars.

b) Each car should belong to a particular Car Category and each car will belong to a particular location.

c) Customer, based on his location and car category preferences, rents a car. d) Based on his location and car category preferences, list of cars available to rent will be shown along with available date and time (from and to).

e) Customer will select a car from the suggestions and should be able to reserve it for rent. f) When a customer reserves a car, he/she should be able to optionally purchase a Car Insurance Plan and should be able to apply at most one discount code.

g) If a customer is also a member of the car rental agency and has a membership ID then he/she will be given a default 10% discount in additional to the discount code applied. Therefore the total discount percentage will be 10 plus the discount percentage given by the discount code applied.

h) Billing is generated when a car is returned.

i) Customer can return the car before the due date, on the due date or he/she can return it late also.

j) If a customer returns a car after the due date, additional late fee is calculated and added to the bill.

k) A default 8.25% tax is applied on the amount which also includes the late fee and this tax is added to obtain the total amount to which the discount will be applied and a final amount is obtained.

l) Once the car is returned it becomes available for the booking.

m) A booking can be cancelled until 5 days before the actual pick up.

3

n) Company may have several discount plans like weekend discount, corporate discount etc.

o) Car price will be calculated based on the selected make and model.

**2. ENTITIES**

**a) Customer:**

Customer will be the one who is using car rental system for reserving a car. He can be a member of the system or a non-member of the system. Member of the system will have membership id. Customer entity will store details like customer driving license number, email, address, name, and phone number.

**b) Car:**

Car entity will have list of cars available in the system. Each car will be associated with a car category and car will have attributes like make, model, mileage and registration number. Car will also have separate flag to check the availability of the car.

**c) Car Category:**

Every car has a car category. Price is calculated based on the car category. Car category will have attributes like no of person, no of luggage’s, name, and cost per day and late fee per hour.

**d) Location**

Location entity here denotes the pickup and drop off location of the car. Customer can pick up the car from the particular location and can have same or different drop off location. Location will have attributes like Location id, name and address.

**e) Booking**

Each car reservation will be monitored in the entity called booking. Booking will have attributes like booking id, from date and time of booking and due return date and time and actual return date and time of the booking, and booking status. This booking amount might also include rental insurance and discount code.

4

**f) Billing**

When a customer returns a car, a bill will be generated on the particular booking. Billing have attributes like Bill ID, bill date, bill status, total late fee, tax amount, and total amount.

**g) Discount**

Customer can apply discount code while the bill is generated. Each discount code has different discount percentage. Discount will have attributes like discount code, name, expiry date and discount percentage.

**h) Car Rental Insurance**

Customer may already have car rental insurance or can buy one while booking the car. Car rental insurance will have attributes like insurance code, coverage type, name and cost per day.

**3. RELATIONS**

**a) Car to Car Category:**

Every car is associated with a car category. Once customer selects a car, the cost per day is obtained from the car category that the selected car belongs to. The relation name is ‘Belongs to’.

**b) Car to Location:**

Customer will be picking up or dropping the car in a particular location. Customer can pick up or drop-off the car at the particular location. So, cars will be present at a location. The relation name is ‘Current location’

**c) Booking to Billing:**

Once customer returns a car bill will be generated for each booking. There can be case like booking is cancelled in that case no bill will be associated with the booking. The relation name is ‘Gives’.

5

**d) Booking to Discount:**

Customer may apply a discount code when he/she books a car. This discount will be applied to the total amount after tax and late fee while the bill is generated. Based on the discount code total amount will be reduced by some percentage. The relation name is ‘Has’.

**e) Booking to Car Rental Insurance:**

Customer can select rental insurance while booking a car so that rental insurance will cover damages based on the coverage type. The relation name is ‘Includes’.

**f) Booking to Location:**

Customer can pick a car for rent from a particular location. The relation name is ‘Pick up location’.

**g) Booking to Location:**

Customer can drop off rental car in a particular location. The relation name is ‘Drop off location’.

**h) Customer to Car to Booking:**

Customer will select car for rent. So the customer will be related to the both car and the booking. The relation between these 3 entities is a ternary relation and the relation name is ‘Rents’.

**4. ASSUMPTIONS**

a) Each booking is associated with only one car reservation at a time.

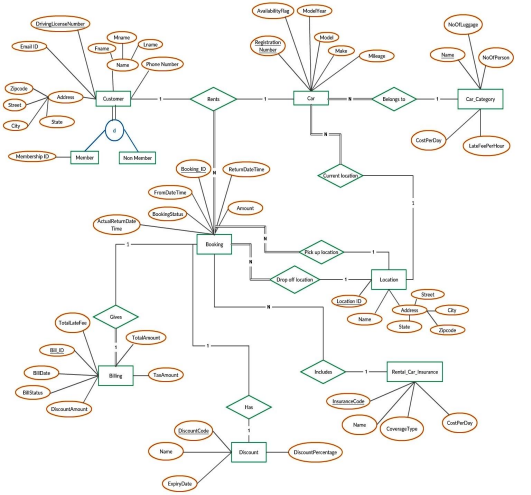
b) Car available in the system should be present at some location.

c) Billing may or may not have discount code applied.

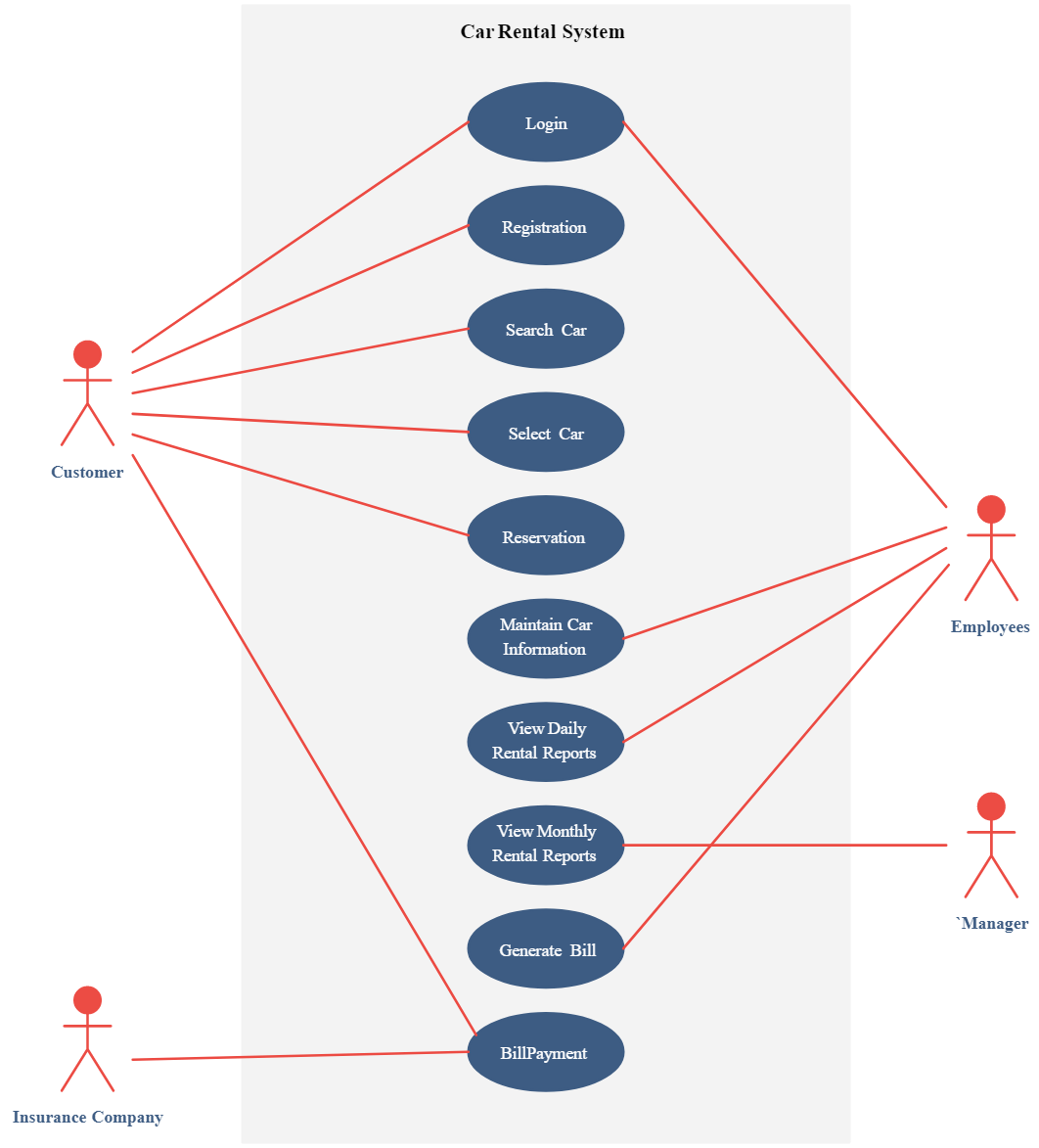
d) Not all Booking is associated with billing because of the cancelled bookings. e) Booking may or may not have rental insurance because customer may have his own insurance.

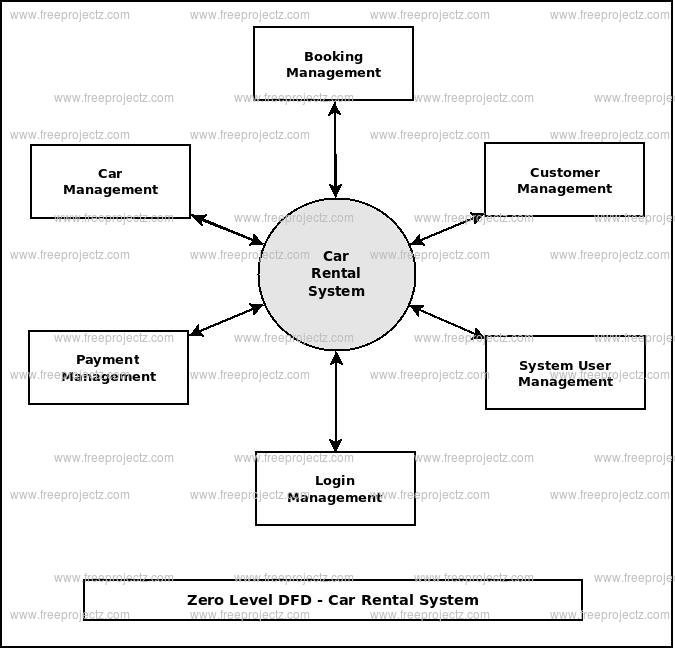
6

**5. ER/EER DIAGRAM**

****7

**Use case diagram-**

****

****

**6. FUNCTIONAL DEPENDENCIES**

**a) Customer\_Details Relation:**

∙ DL\_number -> Fname, Mname, Lname, Phone\_number, Email\_id, Street, City, State, Zipcode, Membership\_id, Membership\_type

∙ Zipcode -> State,City

**b) Car Relation:**

∙ Registration\_number -> Model, Make, Model\_year, Car\_category\_name, Loc\_id, Mileage, Availability\_flag

∙ Model -> Make

**c) Car\_Category Relation:**

∙ Category\_name -> No\_of\_luggage, No\_of\_person, Cost\_per\_day,

Late\_fee\_per\_hour

**d) Location \_Details Relation:**

∙ Location\_id -> Name,Street,City,State,Zipcode

∙ Zipcode -> State,City

**e) Booking\_Details Relation:**

∙ Booking\_id -> From\_dt\_time, Ret\_dt\_time, Amount, Booking\_status, Pickup\_loc, Drop\_loc, Reg\_num, DL\_num, Ins\_code, Act\_ret\_dt\_time,

Discount\_code

**f) Billing\_Details Relation:**

∙ Bill\_id -> Bill\_date, Bill\_status, Discount\_amt, Total\_amt, Tax\_amt, Booking\_id, Total\_late\_fee

**g) Discount\_Details Relation:**

∙ Discount\_code -> Discount\_name,Expiry\_date,Discount\_percentage ∙ Discount\_name -> Discount\_code ,Expiry\_date,Discount\_percentage

8

**h) Rental\_Car\_Insurance Relation:**

∙ Insurance\_code -> Insurance\_name,Coverage\_type,Cost\_per\_day ∙ Insurance\_name -> Insurance\_code ,Coverage\_type,Cost\_per\_day

**6.1 Functional dependencies that violated normalization rules:** The Following transitive dependencies exist in the relational schema.

∙ **Customer\_Details Relation**

o DL\_number -> Zipcode

o Zipcode -> State, City

∙ **Car Relation**

o Registration\_number -> Model\_name

o Model\_name -> Make

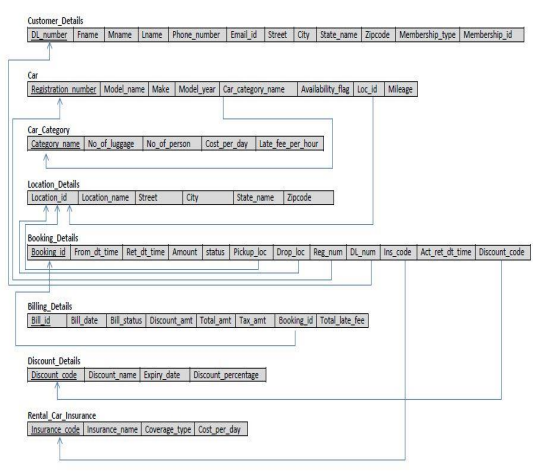
∙ **Location\_Details Relation**

o Location\_id -> Zipcode

o Zipcode -> State, City

**FINAL RELATIONAL SCHEMA**

For convenience, we have chosen to represent our final relational schema in 2NF normalization. We have de-normalized from 3NF to 2NF.

10

**8. SQL STATEMENTS**

**8.1 Create table Statements**

a) **Customer\_Details**

CREATE TABLE CUSTOMER\_DETAILS

( DL\_NUMBER CHAR(8) NOT NULL,

FNAME VARCHAR(25) NOT NULL,

MNAME VARCHAR(15),

LNAME VARCHAR(25) NOT NULL,

PHONE\_NUMBER NUMBER(10) NOT NULL,

EMAIL\_ID VARCHAR(30) NOT NULL,

STREET VARCHAR(30) NOT NULL,

CITY VARCHAR(20) NOT NULL,

STATE\_NAME VARCHAR(20) NOT NULL,

ZIPCODE NUMBER(5) NOT NULL,

MEMBERSHIP\_TYPE CHAR(1) DEFAULT 'N' NOT NULL, MEMBERSHIP\_ID CHAR(5),

CONSTRAINT CUSTOMERPK

PRIMARY KEY (DL\_NUMBER)

);

**b) Car\_Category**

CREATE TABLE CAR\_CATEGORY

( CATEGORY\_NAME VARCHAR(25) NOT NULL,

NO\_OF\_LUGGAGE INTEGER NOT NULL,

NO\_OF\_PERSON INTEGER NOT NULL,

COST\_PER\_DAY NUMBER(5,2) NOT NULL,

LATE\_FEE\_PER\_HOUR NUMBER(5,2) NOT NULL, CONSTRAINT CARCATEGORYPK

PRIMARY KEY (CATEGORY\_NAME)

);

**c) Location\_Details**

CREATE TABLE LOCATION\_DETAILS

( LOCATION\_ID CHAR(4) NOT NULL,

LOCATION\_NAME VARCHAR(50) NOT NULL,

11

STREET VARCHAR(30) NOT NULL,

CITY VARCHAR(20) NOT NULL,

STATE\_NAME VARCHAR(20) NOT NULL,

ZIPCODE NUMBER(5) NOT NULL,

CONSTRAINT LOCATIONPK

PRIMARY KEY (LOCATION\_ID)

);

**d) Car**

CREATE TABLE CAR

( REGISTRATION\_NUMBER CHAR(7) NOT NULL,

MODEL\_NAME VARCHAR(25) NOT NULL,

MAKE VARCHAR(25) NOT NULL,

MODEL\_YEAR NUMBER(4) NOT NULL,

MILEAGE INTEGER NOT NULL,

CAR\_CATEGORY\_NAME VARCHAR(25) NOT NULL,

LOC\_ID CHAR(4) NOT NULL,

AVAILABILITY\_FLAG CHAR(1) NOT NULL,

CONSTRAINT CARPK

PRIMARY KEY (REGISTRATION\_NUMBER),

CONSTRAINT CARFK1

FOREIGN KEY (CAR\_CATEGORY\_NAME) REFERENCES

CAR\_CATEGORY(CATEGORY\_NAME),

CONSTRAINT CARFK2

FOREIGN KEY (LOC\_ID) REFERENCES LOCATION\_DETAILS(LOCATION\_ID) );

**e) Discount\_Details**

CREATE TABLE DISCOUNT\_DETAILS

( DISCOUNT\_CODE CHAR(4) NOT NULL,

DISCOUNT\_NAME VARCHAR(25) NOT NULL,

EXPIRY\_DATE DATE NOT NULL,

DISCOUNT\_PERCENTAGE NUMBER(4,2) NOT NULL,

CONSTRAINT DISCOUNTPK

PRIMARY KEY (DISCOUNT\_CODE),

CONSTRAINT DISCOUNTSK

UNIQUE (DISCOUNT\_NAME)

);

12

**f) Rental\_Car\_Insurance**

CREATE TABLE RENTAL\_CAR\_INSURANCE

( INSURANCE\_CODE CHAR(4) NOT NULL,

INSURANCE\_NAME VARCHAR(50) NOT NULL,

COVERAGE\_TYPE VARCHAR(200) NOT NULL,

COST\_PER\_DAY NUMBER(4,2) NOT NULL,

CONSTRAINT INSURANCEPK

PRIMARY KEY (INSURANCE\_CODE),

CONSTRAINT INSURANCESK

UNIQUE (INSURANCE\_NAME)

);

**g) Booking\_Details**

CREATE TABLE BOOKING\_DETAILS

( BOOKING\_ID CHAR(5) NOT NULL,

FROM\_DT\_TIME TIMESTAMP NOT NULL,

RET\_DT\_TIME TIMESTAMP NOT NULL,

AMOUNT NUMBER(10,2) NOT NULL,

BOOKING\_STATUS CHAR(1) NOT NULL,

PICKUP\_LOC CHAR(4) NOT NULL,

DROP\_LOC CHAR(4) NOT NULL,

REG\_NUM CHAR(7) NOT NULL,

DL\_NUM CHAR(8) NOT NULL,

INS\_CODE CHAR(4),

ACT\_RET\_DT\_TIME TIMESTAMP,

DISCOUNT\_CODE CHAR(4),

CONSTRAINT BOOKINGPK

PRIMARY KEY (BOOKING\_ID),

CONSTRAINT BOOKINGFK1

FOREIGN KEY (PICKUP\_LOC) REFERENCES LOCATION\_DETAILS(LOCATION\_ID), CONSTRAINT BOOKINGFK2

FOREIGN KEY (DROP\_LOC) REFERENCES LOCATION\_DETAILS(LOCATION\_ID), CONSTRAINT BOOKINGFK3

FOREIGN KEY (REG\_NUM) REFERENCES CAR(REGISTRATION\_NUMBER), CONSTRAINT BOOKINGFK4

FOREIGN KEY (DL\_NUM) REFERENCES CUSTOMER\_DETAILS(DL\_NUMBER), CONSTRAINT BOOKINGFK5

FOREIGN KEY (INS\_CODE) REFERENCES RENTAL\_CAR\_INSURANCE(INSURANCE\_CODE),

13

CONSTRAINT BOOKINGFK6

FOREIGN KEY (DISCOUNT\_CODE) REFERENCES DISCOUNT\_DETAILS(DISCOUNT\_CODE) );

**h) Billing\_Details**

CREATE TABLE BILLING\_DETAILS

( BILL\_ID CHAR(6) NOT NULL,

BILL\_DATE DATE NOT NULL,

BILL\_STATUS CHAR(1) NOT NULL,

DISCOUNT\_AMOUNT NUMBER(10,2) NOT NULL,

TOTAL\_AMOUNT NUMBER(10,2) NOT NULL,

TAX\_AMOUNT NUMBER(10,2) NOT NULL,

BOOKING\_ID CHAR(5) NOT NULL,

TOTAL\_LATE\_FEE NUMBER(10,2) NOT NULL,

CONSTRAINT BILLINGPK

PRIMARY KEY (BILL\_ID),

CONSTRAINT BILLINGFK1

FOREIGN KEY (BOOKING\_ID) REFERENCES BOOKING\_DETAILS(BOOKING\_ID) );

**8.2 Insert SQL Statements**

INSERT INTO CUSTOMER\_DETAILS VALUES('F1234554', 'NAVEEN',

NULL,'RAJ','4696004267', 'naveen@gmail.com','700 CAMPBELL RD',

'RICHARDSON','TEXAS',75080,'M','M1001');

INSERT INTO CUSTOMER\_DETAILS VALUES('F9764521', 'NIVEDITHA',

NULL,'VARADHA CHANDRASEKARAN','4696478596', 'nivi07@gmail.com', '800 RENNER RD','RICHARDSON','TEXAS',75080,'M','M1002');

INSERT INTO CUSTOMER\_DETAILS VALUES('F2345611', 'SURESH',

'KUMAR','GOPALAKRISHNAN','8189187546', 'suresh2234@gmail.com',

'6547 CANOGA AVE','CANOGA PARK','CALIFORNIA',91303,'N',NULL);

INSERT INTO CUSTOMER\_DETAILS VALUES('R8763578', 'MARK',

NULL,'HUFF','7345678902', 'markhuff@gmail.com','1445 ROSS AVE',

'DALLAS','TEXAS',75202,'N',NULL);

INSERT INTO CUSTOMER\_DETAILS VALUES('I3478953', 'MARK',

'S','TOWNSEND','9872563478', 'markstown@gmail.com','7825 MCCALLUM BLVD', 'DALLAS','TEXAS',75252,'M','M1003');

14

INSERT INTO CUSTOMER\_DETAILS VALUES('E7521097', 'MITA',

NULL,'RANA','9098123429', 'mitarana@gmail.com','367 MEANDERING WAY', 'HOUSTON','TEXAS',76245,'N',NULL);

INSERT INTO CUSTOMER\_DETAILS VALUES('T0981237', 'DANISH',

NULL,'HASSAN','6712890345', 'danishhasan@gmail.com','888 PRESTON ROAD', 'DULLES','VIRGINIA',92367,'M','M1004');

INSERT INTO CUSTOMER\_DETAILS VALUES('F0091266', 'MIKE',

NULL,'BOYEAR','7892340918', 'mikeboy@gmail.com','1007 DALLAS PARKWAY', 'DALLAS','TEXAS',72212,'N',NULL);

INSERT INTO CUSTOMER\_DETAILS VALUES('P1234567', 'CHRIS',

NULL,'ALEXANDER','9902489', 'chrisalex@gmail.com','2256 WALL STREET', 'NEWARK','NEW JERSEY',65289,'M','M1005');

INSERT INTO CUSTOMER\_DETAILS VALUES('V5690245',

'VELA','R','REYNALDO','9908762514', 'reyvela@gmail.com','0099 ALMA ROAD', 'DULLES','VIRGINIA',97325,'N',NULL);

INSERT INTO CAR\_CATEGORY VALUES('ECONOMY',2,5,30,0.9);

INSERT INTO CAR\_CATEGORY VALUES('COMPACT',3,5,32,0.96);

INSERT INTO CAR\_CATEGORY VALUES('MID SIZE',3,5,35,1.05);

INSERT INTO CAR\_CATEGORY VALUES('STANDARD',3,5,38,1.14);

INSERT INTO CAR\_CATEGORY VALUES('FULL SIZE',4,5,40,1.2);

INSERT INTO CAR\_CATEGORY VALUES('LUXURY CAR',5,5,75,2.25);

INSERT INTO CAR\_CATEGORY VALUES('MID SIZE SUV',2,5,36,1.08); INSERT INTO CAR\_CATEGORY VALUES('STANDARD SUV',3,5,40,1.2); INSERT INTO CAR\_CATEGORY VALUES('FULL SIZE SUV',2,8,60,1.8);

INSERT INTO CAR\_CATEGORY VALUES('MINI VAN',5,7,70,2.1);

INSERT INTO LOCATION\_DETAILS VALUES('L101','DALLAS LOVE FIELD AIRPORT', 'Herb Kelleher Way','Dallas','Texas',75235);

INSERT INTO LOCATION\_DETAILS VALUES('L102','LOS ANGELES INTL AIRPORT', 'World Way','Los Angeles','California',90045);

INSERT INTO LOCATION\_DETAILS VALUES('L103','DALLAS/ FORT WORTH INTL AIRPORT', 'International Pkwy','DFW Airport','Texas',75261);

INSERT INTO LOCATION\_DETAILS VALUES('L104','WEST HOUSTON AIRPORT', 'Groschke Rd','Houston','Texas',77094);

INSERT INTO LOCATION\_DETAILS VALUES('L105','WASHINGTON DULLES INTL AIRPORT',

15

'Saarinen Cir','Dulles','Virginia',20166);

INSERT INTO LOCATION\_DETAILS VALUES('L106','NEWARK LIBERTY INTL AIRPORT', 'Brewster Rd','Newark','New Jersey',07114);

INSERT INTO LOCATION\_DETAILS VALUES('L107','SALT LAKE CITY INTL AIRPORT', 'N Terminal Dr','Salt Lake City','Utah',84122);

INSERT INTO CAR VALUES('ABX1234','CIVIC','HONDA',

2014,10000,'ECONOMY','L101','A');

INSERT INTO CAR VALUES('SDF4567','FIESTA','FORD',

2015,15000,'ECONOMY','L102','N');

INSERT INTO CAR VALUES('WER3245','ACCENT','HYUNDAI',

2014,12356,'ECONOMY','L103','A');

INSERT INTO CAR VALUES('GLZ2376','COROLLA','TOYOTA',

2016,5000,'ECONOMY','L104','A');

INSERT INTO CAR VALUES('HJK1234','CIVIC','HONDA',

2015,20145,'ECONOMY','L102','N');

INSERT INTO CAR VALUES('GLS7625','FOCUS','FORD',

2014,12000,'COMPACT','L107','A');

INSERT INTO CAR VALUES('FKD8202','GOLF','VOLKSWAGAN',

2016,9000,'COMPACT','L106','A');

INSERT INTO CAR VALUES('HNX1890','PRIUS','TOYOTA',

2015,15690,'COMPACT','L105','N');

INSERT INTO CAR VALUES('KJS1983','PRIUS','TOYOTA',

2014,20900,'COMPACT','L104','A');

INSERT INTO CAR VALUES('SDL9356','FOCUS','FORD',

2016,10009,'COMPACT','L103','A');

INSERT INTO CAR VALUES('OTY7293','CRUZE','CHEVROLET',

2016,17800,'MID SIZE','L102','A');

INSERT INTO CAR VALUES('QWE4562','LEGACY','SUBARU',

2012,13420,'MID SIZE','L101','A');

INSERT INTO CAR VALUES('CXZ2356','AVENGER','DODGE',

2015,5000,'MID SIZE','L102','A');

INSERT INTO CAR VALUES('ASD9090','ACCORD','HONDA',

2016,200,'MID SIZE','L103','A');

INSERT INTO CAR VALUES('UYT3981','LEGACY','SUBARU',

2013,16750,'MID SIZE','L104','A');

INSERT INTO CAR VALUES('TRE9726','200','CHRYSTLER',

16

2012,14320,'STANDARD','L105','A');

INSERT INTO CAR VALUES('HGF5628','TAURUS','FORD', 2013,15540,'STANDARD','L106','A');

INSERT INTO CAR VALUES('LKJ7253','200','CHRYSTLER', 2014,16300,'STANDARD','L107','A');

INSERT INTO CAR VALUES('VBN6283','TAURUS','FORD', 2015,17500,'STANDARD','L101','A');

INSERT INTO CAR VALUES('POI7281','200','CHRYSTLER', 2016,18830,'STANDARD','L102','N');

INSERT INTO CAR VALUES('MNB8654','FALCON','FORD', 2012,10900,'FULL SIZE','L103','A');

INSERT INTO CAR VALUES('UHV9786','IMPALA','CHEVROLET', 2013,11500,'FULL SIZE','L104','A');

INSERT INTO CAR VALUES('EFB5427','WAYFARER','FORD', 2014,14350,'FULL SIZE','L105','A');

INSERT INTO CAR VALUES('PLM9873','IMPALA','CHEVROLET', 2015,18900,'FULL SIZE','L106','A');

INSERT INTO CAR VALUES('WDV2458','FALCON','FORD', 2016,5600,'FULL SIZE','L107','A');

INSERT INTO CAR VALUES('QSC8709','MKZ','LINCOLN', 2012,18700,'LUXURY CAR','L101','A');

INSERT INTO CAR VALUES('TGB8961','GENESIS','HYUNDAI', 2013,17620,'LUXURY CAR','L102','A');

INSERT INTO CAR VALUES('MKU0172','TLX','ACURA', 2014,12345,'LUXURY CAR','L103','A');

INSERT INTO CAR VALUES('CFT1908','328I','BMW', 2015,10800,'LUXURY CAR','L104','A');

INSERT INTO CAR VALUES('WHM7619','AVALON','TOYOTA', 2016,7800,'LUXURY CAR','L105','A');

INSERT INTO CAR VALUES('WLZ8955','ESCAPE','FORD', 2012,19800,'MID SIZE SUV','L106','A');

INSERT INTO CAR VALUES('QIO7621','EQUINOX','CHEVROLET', 2013,17560,'MID SIZE SUV','L107','A');

INSERT INTO CAR VALUES('YSN1927','PATHFINDER','NISSAN', 2014,14390,'MID SIZE SUV','L101','A');

INSERT INTO CAR VALUES('EDM8610','GLA','MERCEDEZ BENZ', 2015,12900,'MID SIZE SUV','L102','A');

INSERT INTO CAR VALUES('AHK7325','RAV4','TOYOTA',

17

2016,3400,'MID SIZE SUV','L103','A');

INSERT INTO CAR VALUES('OHZ0976','EDGE','FORD', 2012,27890,'STANDARD SUV','L104','A');

INSERT INTO CAR VALUES('RKS9862','TAHOE','CHEVROLET', 2013,20390,'STANDARD SUV','L105','A');

INSERT INTO CAR VALUES('WIJ6190','EDGE','FORD',

2014,18700,'STANDARD SUV','L106','A');

INSERT INTO CAR VALUES('ZDT8612','TAHOE','CHEVROLET', 2015,14300,'STANDARD SUV','L107','A');

INSERT INTO CAR VALUES('LDJ7719','EDGE','FORD',

2016,5690,'STANDARD SUV','L101','A');

INSERT INTO CAR VALUES('UIA8709','EXPEDITION','FORD', 2012,19870,'FULL SIZE SUV','L102','A');

INSERT INTO CAR VALUES('WKJ7972','SEQUOIA','TOYOTA', 2013,14500,'FULL SIZE SUV','L103','A');

INSERT INTO CAR VALUES('JLS1097','SUBURBAN','CHEVROLET', 2014,13290,'FULL SIZE SUV','L104','A');

INSERT INTO CAR VALUES('UHJ6782','EXPEDITION','FORD', 2015,11750,'FULL SIZE SUV','L105','A');

INSERT INTO CAR VALUES('XBM6822','SUBURBAN','CHEVROLET', 2016,3400,'FULL SIZE SUV','L106','A');

INSERT INTO CAR VALUES('SHK7767','QUEST','NISSAN', 2012,23478,'MINI VAN','L107','A');

INSERT INTO CAR VALUES('JSL7920','ODYSSEY','HONDA', 2013,19320,'MINI VAN','L106','A');

INSERT INTO CAR VALUES('PAJ5289','GRAND CARAVAN','DODGE', 2014,23478,'MINI VAN','L105','A');

INSERT INTO CAR VALUES('TSJ6290','QUEST','NISSAN', 2015,13200,'MINI VAN','L104','A');

INSERT INTO CAR VALUES('MWO9296','ODYSSEY','HONDA', 2016,2300,'MINI VAN','L103','A');

INSERT INTO DISCOUNT\_DETAILS VALUES ('D678','IBM CORPORATE', to\_date('2018-01-25','YYYY-MM-DD'),25);

INSERT INTO DISCOUNT\_DETAILS VALUES ('D234','CTS CORPORATE', to\_date('2019-09-02','YYYY-MM-DD'),20);

INSERT INTO DISCOUNT\_DETAILS VALUES ('D756','HOLIDAY SPECIAL',

18

to\_date('2017-10-29','YYYY-MM-DD'),10);

INSERT INTO DISCOUNT\_DETAILS VALUES ('D109','WEEKLY RENTALS', to\_date('2020-11-09','YYYY-MM-DD'),25);

INSERT INTO DISCOUNT\_DETAILS VALUES ('D972','ONE WAY SPECIAL', to\_date('2016-12-15','YYYY-MM-DD'),20);

INSERT INTO DISCOUNT\_DETAILS VALUES ('D297','UPGRADE SPECIAL', to\_date('2018-02-18','YYYY-MM-DD'),20);

INSERT INTO RENTAL\_CAR\_INSURANCE VALUES('I201', 'COLLISION DAMAGE WAIVER', 'Covers theft and total damage to the rental car',15);

INSERT INTO RENTAL\_CAR\_INSURANCE VALUES('I202',

'SUPPLEMENTAL LIABILITY PROTECTION', 'Covers damage done to others',12); INSERT INTO RENTAL\_CAR\_INSURANCE VALUES('I203',

'PERSONAL ACCIDENT INSURANCE', 'Covers medical costs for driver and passengers',10); INSERT INTO RENTAL\_CAR\_INSURANCE VALUES('I204',

'PERSONAL EFFECTS COVERAGE', 'Covers theft of personal belongings',5);

INSERT INTO BOOKING\_DETAILS VALUES('B1001',TO\_TIMESTAMP('2016-01-20 10:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-01-25 10:00:00', 'YYYY-MM-DD HH24:MI:SS'),

150,'R','L101','L101','ABX1234','F1234554',NULL,

TO\_TIMESTAMP('2016-01-25 10:00:00', 'YYYY-MM-DD HH24:MI:SS'),'D756'); INSERT INTO BOOKING\_DETAILS VALUES('B1002',TO\_TIMESTAMP('2016-01-21 11:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-01-24 10:00:00', 'YYYY-MM-DD HH24:MI:SS'),

90,'C','L102','L102','SDF4567','T0981237',NULL,NULL,NULL);

INSERT INTO BOOKING\_DETAILS VALUES('B1003',TO\_TIMESTAMP('2016-02-10 13:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-01-15 13:00:00', 'YYYY-MM-DD HH24:MI:SS'),

450,'R','L101','L101','QSC8709','R8763578','I201',

TO\_TIMESTAMP('2016-01-15 13:00:00', 'YYYY-MM-DD HH24:MI:SS'),NULL); INSERT INTO BOOKING\_DETAILS VALUES('B1004',TO\_TIMESTAMP('2016-04-24 13:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-04-25 20:30:00', 'YYYY-MM-DD HH24:MI:SS'),

48,'R','L106','L106','WLZ8955','F0091266','I202',

TO\_TIMESTAMP('2016-04-23 20:30:00', 'YYYY-MM-DD HH24:MI:SS'),'D234');

19

INSERT INTO BOOKING\_DETAILS VALUES('B1005',TO\_TIMESTAMP('2016-04-18 09:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-04-25 09:00:00', 'YYYY-MM-DD HH24:MI:SS'),

266,'B','L102','L106','POI7281','P1234567',NULL,NULL,'D972');

INSERT INTO BOOKING\_DETAILS VALUES('B1006',TO\_TIMESTAMP('2016-04-21 17:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-04-25 17:00:00', 'YYYY-MM-DD HH24:MI:SS'),

168,'B','L105','L107','HNX1890','V5690245','I203',NULL,'D234');

INSERT INTO BOOKING\_DETAILS VALUES('B1007',TO\_TIMESTAMP('2016-04-16 08:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-04-25 17:00:00', 'YYYY-MM-DD HH24:MI:SS'),

405,'B','L102','L102','SDF4567','I3478953','I201',NULL,'D756');

INSERT INTO BOOKING\_DETAILS VALUES('B1008',TO\_TIMESTAMP('2016-04-11 08:00:00', 'YYYY-MM-DD HH24:MI:SS'),TO\_TIMESTAMP('2016-04-25 17:00:00', 'YYYY-MM-DD HH24:MI:SS'),

630,'B','L102','L102','HJK1234','T0981237','I201',NULL,'D756');

INSERT INTO BILLING\_DETAILS VALUES('BL1001',to\_date('2016-01-25','YYYY-MM-DD'), 'P',24.36 ,138.03,12.38 ,'B1001',0);

INSERT INTO BILLING\_DETAILS VALUES('BL1002',to\_date('2016-01-15','YYYY-MM-DD'), 'P',0 ,487.13 ,12.38 ,'B1003',0);

INSERT INTO BILLING\_DETAILS VALUES('BL1003',to\_date('2016-04-24','YYYY-MM-DD'), 'P',10.39 ,41.57 ,3.96 ,'B1004',0);

**9. PL/SQL STATEMEMTS**

**9.1 Stored Procedure 1: CALCULATE\_LATE\_FEE\_AND\_TAX**

Given the return date and time while booking, actual return date and time, registration number of the car and booking amount, this procedure calculates the total late fee using late fee per hour, return date and time and the actual return date and time of the rental car. Once the total late fee is obtained, it is added to the amount and the total tax is calculated.

20

-------------------------------------------------------------------------------------------

--Procedure Name: CALCULATE\_LATE\_FEE\_AND\_TAX

--This stored procedure calculates the total late fee and tax.

-------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE CALCULATE\_LATE\_FEE\_AND\_TAX

(actualReturnDateTime IN BOOKING\_DETAILS.ACT\_RET\_DT\_TIME%TYPE, ReturnDateTime IN BOOKING\_DETAILS.RET\_DT\_TIME%TYPE,

regNum IN BOOKING\_DETAILS.REG\_NUM%TYPE,

amount IN BOOKING\_DETAILS.AMOUNT%TYPE,

totalLateFee OUT BILLING\_DETAILS.TOTAL\_AMOUNT%TYPE,

totalTax OUT BILLING\_DETAILS.TAX\_AMOUNT%TYPE ) AS

--local declarations

lateFeePerHour CAR\_CATEGORY.LATE\_FEE\_PER\_HOUR%TYPE;

hourDifference DECIMAL(10,2);

BEGIN

SELECT LATE\_FEE\_PER\_HOUR INTO lateFeePerHour

FROM CAR\_CATEGORY CC INNER JOIN CAR C ON CC.CATEGORY\_NAME = C.CAR\_CATEGORY\_NAME WHERE C.REGISTRATION\_NUMBER = regNum;

IF actualReturnDateTime > ReturnDateTime THEN

hourDifference := (TO\_DATE (TO\_CHAR (actualReturnDateTime,

'dd/mm/yyyy hh24:mi:ss'), 'dd/mm/yyyy hh24:mi:ss')

- TO\_DATE (TO\_CHAR (ReturnDateTime, 'dd/mm/yyyy hh24:mi:ss') ,'dd/mm/yyyy hh24:mi:ss'))\*(24);

totalLateFee := hourDifference \* lateFeePerHour;

ELSE

totalLateFee := 0;

END IF;

totalTax := (amount + totalLateFee)\*0.0825;

END;

/

**9.2 Stored Procedure 2: CALCULATE\_DISCOUNT\_AMOUNT**

Given the driving license number, total amount and the discount code, this procedure calculates the discount amount based on the discount code used by the customer while booking the rental car. Additional 10% discount is given to the customers who have membership ID. The discount amount is calculated on the total amount obtained after adding the late fee and the tax amount.

∙ For Non-members:

21

Discount amount = total amount \* ((discountPercentage)/100)

∙ For members:

Discount amount = total amount \* ((discountPercentage+10)/100)

------------------------------------------------------------------------------------------- --Procedure Name: CALCULATE\_DISCOUNT\_AMOUNT

--This stored procedure calculates the discount amount for a booking. ------------------------------------------------------------------------------------------- CREATE OR REPLACE PROCEDURE CALCULATE\_DISCOUNT\_AMOUNT (dlNum IN CUSTOMER\_DETAILS.DL\_NUMBER%TYPE,

amount IN BILLING\_DETAILS.TOTAL\_AMOUNT%TYPE,

discountCode IN DISCOUNT\_DETAILS.DISCOUNT\_CODE%TYPE,

discountAmt OUT BILLING\_DETAILS.DISCOUNT\_AMOUNT%TYPE) AS --local declarations

memberType CUSTOMER\_DETAILS.MEMBERSHIP\_TYPE%TYPE;

discountPercentage DISCOUNT\_DETAILS.DISCOUNT\_PERCENTAGE%TYPE; BEGIN

SELECT MEMBERSHIP\_TYPE INTO memberType FROM CUSTOMER\_DETAILS WHERE DL\_NUMBER = dlNum;

IF NVL(discountCode,'NULL') <> 'NULL' THEN

SELECT DISCOUNT\_PERCENTAGE INTO discountPercentage FROM DISCOUNT\_DETAILS WHERE DISCOUNT\_CODE = discountCode; IF memberType = 'M' THEN

discountAmt := amount \* ((discountPercentage+10)/100); ELSE

discountAmt := amount \* (discountPercentage/100); END IF;

ELSE

IF memberType = 'M' THEN

discountAmt := amount \* 0.1;

ELSE

discountAmt := 0;

END IF;

END IF;

END;

/

22

**9.3 Stored Procedure 3: GENERATE\_REVENUE\_REPORT**

This procedure generates monthly revenue report based on the bill date, location and car categories. For every location, the no of cars in each car category along with the total revenue is calculated and a monthly report is generated. This stored procedure makes use of cursor for report generation.

-------------------------------------------------------------------------------------------

--Procedure Name: GENERATE\_REVENUE\_REPORT

--This stored procedure calculates and generates the monthly revenue report. -------------------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE GENERATE\_REVENUE\_REPORT AS

--local declarations

thisLocationID LOCATION\_DETAILS.LOCATION\_ID%TYPE;

currentLocationID LOCATION\_DETAILS.LOCATION\_ID%TYPE;

locationName LOCATION\_DETAILS.LOCATION\_NAME%TYPE;

thisCategoryName CAR\_CATEGORY.CATEGORY\_NAME%TYPE;

thisNoOfCars integer; thisRevenue DECIMAL(15,2);

--Cursor declaration

CURSOR CURSOR\_REPORT IS SELECT TABLE1.LOCATIONID, TABLE1.CATNAME , TABLE1.NOOFCARS,SUM(NVL((TABLE2.AMOUNT),0)) AS REVENUE

FROM (SELECT LC.LID AS LOCATIONID, LC.CNAME AS CATNAME ,

COUNT(C.REGISTRATION\_NUMBER) AS NOOFCARS FROM (SELECT

L.LOCATION\_ID AS LID, CC.CATEGORY\_NAME AS CNAME FROM

CAR\_CATEGORY CC CROSS JOIN LOCATION\_DETAILS L) LC LEFT OUTER JOIN CAR C ON LC.CNAME = C.CAR\_CATEGORY\_NAME AND LC.LID = C.LOC\_ID GROUP BY LC.LID, LC.CNAME ORDER BY LC.LID) TABLE1 LEFT OUTER JOIN (SELECT BC.PLOC AS PICKLOC,BC.CNAME AS CNAMES, SUM(BL.TOTAL\_AMOUNT) AS AMOUNT FROM (SELECT B.PICKUP\_LOC AS PLOC, C1.CAR\_CATEGORY\_NAME AS CNAME, B.BOOKING\_ID AS BID FROM BOOKING\_DETAILS B INNER JOIN CAR C1 ON B.REG\_NUM = C1.REGISTRATION\_NUMBER) BC INNER JOIN BILLING\_DETAILS BL ON BC.BID = BL.BOOKING\_ID WHERE

(to\_date (SYSDATE,'dd-MM-yyyy') - to\_date(BL.BILL\_DATE,'dd-MM-yyyy')) <=30 GROUP BY BC.PLOC,BC.CNAME ORDER BY BC.PLOC) TABLE2

ON TABLE1.LOCATIONID=TABLE2.PICKLOC AND TABLE1.CATNAME = TABLE2.CNAMES GROUP BY TABLE1.LOCATIONID, TABLE1.CATNAME, TABLE1.NOOFCARS ORDER BY TABLE1.LOCATIONID;

BEGIN

dbms\_output.put\_line(' ');

dbms\_output.put\_line('Revenue Report');

dbms\_output.put\_line('\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

23

dbms\_output.put\_line(' ');

OPEN CURSOR\_REPORT;

FETCH CURSOR\_REPORT INTO thisLocationID, thisCategoryName, thisNoOfCars, thisRevenue;

IF CURSOR\_REPORT%NOTFOUND THEN

dbms\_output.put\_line('No Report to be Generated');

ELSE

currentLocationID := thisLocationID;

<<LABEL\_NEXTLOC>>

SELECT LOCATION\_NAME INTO locationName from LOCATION\_DETAILS WHERE LOCATION\_ID = currentLocationID;

dbms\_output.put\_line('Location Name: '|| locationName); dbms\_output.put\_line(' ');

dbms\_output.put\_line('Car Category' || ' '||'Number of Cars' ||' '|| 'Revenue');

dbms\_output.put\_line('------------' || ' '||'--------------' ||' '|| '-------');

dbms\_output.put\_line(thisCategoryName ||

RPAD(' ', (16 - LENGTH(thisCategoryName)))||thisNoOfCars ||RPAD(' ', (18 - LENGTH(thisNoOfCars)))|| thisRevenue); LOOP

FETCH CURSOR\_REPORT INTO thisLocationID, thisCategoryName, thisNoOfCars, thisRevenue;

EXIT WHEN (CURSOR\_REPORT%NOTFOUND);

IF thisLocationID = currentLocationID THEN

dbms\_output.put\_line(thisCategoryName || RPAD(' ', (16 - LENGTH(thisCategoryName)))||thisNoOfCars ||RPAD(' ', (18 - LENGTH(thisNoOfCars)))|| thisRevenue); ELSE

currentLocationID := thisLocationID;

dbms\_output.put\_line(' ');

dbms\_output.put\_line('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');

dbms\_output.put\_line(' ');

GOTO LABEL\_NEXTLOC;

END IF;

END LOOP;

END IF;

END;

/

24

**9.4 Trigger 1: GENERATE\_BILLING**

This trigger inserts a tuple into the Billing\_Details table when the actual return date is updated and booking status is updated to ‘R’ in Booking\_Details table. It generates Bill when the rental car is returned. This is triggered whenever a row is updated in Booking\_Details table.

-------------------------------------------------------------------------------------------

--Trigger Name: GENERATE\_BILLING

--This trigger generates the bill and inserts a row in Billing\_Details table

-------------------------------------------------------------------------------------------

CREATE OR REPLACE TRIGGER GENERATE\_BILLING

AFTER UPDATE ON BOOKING\_DETAILS

FOR EACH ROW

WHEN (NVL(TO\_CHAR(NEW.ACT\_RET\_DT\_TIME),'NULL') <> 'NULL' AND

NEW.BOOKING\_STATUS ='R')

DECLARE

-- declaration section

lastBillId BILLING\_DETAILS.BILL\_ID%TYPE;

newBillId BILLING\_DETAILS.BILL\_ID%TYPE;

discountAmt BILLING\_DETAILS.DISCOUNT\_AMOUNT%TYPE;

totalLateFee BILLING\_DETAILS.TOTAL\_LATE\_FEE%TYPE;

totalTax BILLING\_DETAILS.TAX\_AMOUNT%TYPE;

totalAmountBeforeDiscount BILLING\_DETAILS.TOTAL\_AMOUNT%TYPE;

finalAmount BILLING\_DETAILS.TOTAL\_AMOUNT%TYPE;

BEGIN

SELECT BILL\_ID INTO lastBillId FROM ( SELECT BILL\_ID, ROWNUM AS

RN FROM BILLING\_DETAILS)

WHERE RN= (SELECT MAX(ROWNUM) FROM BILLING\_DETAILS);

newBillId := 'BL' || TO\_CHAR(TO\_NUMBER(SUBSTR(lastBillId,3))+1);

CALCULATE\_LATE\_FEE\_AND\_TAX(:NEW.ACT\_RET\_DT\_TIME, :NEW.RET\_DT\_TIME, :NEW.REG\_NUM,:NEW.AMOUNT, totalLateFee, totalTax);

totalAmountBeforeDiscount := :NEW.AMOUNT + totalLateFee + totalTax;

CALCULATE\_DISCOUNT\_AMOUNT(:NEW.DL\_NUM, totalAmountBeforeDiscount, :NEW.DISCOUNT\_CODE, discountAmt);

25

finalAmount := totalAmountBeforeDiscount - discountAmt;

--insert new bill into the billing\_details table

INSERT INTO BILLING\_DETAILS (BILL\_ID,BILL\_DATE,BILL\_STATUS,DISCOUNT\_AMOUNT, TOTAL\_AMOUNT,TAX\_AMOUNT,BOOKING\_ID,TOTAL\_LATE\_FEE)

VALUES (newBillId,to\_date(SYSDATE,'YYYY-MM-DD'),'P',

discountAmt,finalAmount,totalTax,:NEW.BOOKING\_ID,totalLateFee);

END;

/

**9.5 Trigger 2: UPDATE\_CAR\_DETAILS**

This trigger updates the availability flag, mileage and location of the car in the car table when the actual return date is updated or when a booking is cancelled. This is triggered whenever a row is updated in Booking\_Details table.

-------------------------------------------------------------------------------------------

--Trigger Name: UPDATE\_CAR\_DETAILS

--This trigger updates the availability flag, mileage and location in the car table --when the car is returned.

-------------------------------------------------------------------------------------------

CREATE OR REPLACE TRIGGER UPDATE\_CAR\_DETAILS

AFTER UPDATE ON BOOKING\_DETAILS

FOR EACH ROW

WHEN (NVL(TO\_CHAR(NEW.ACT\_RET\_DT\_TIME),'NULL') <> 'NULL' OR

NEW.BOOKING\_STATUS ='C')

DECLARE

BEGIN

IF :NEW.BOOKING\_STATUS ='C' THEN

UPDATE CAR SET AVAILABILITY\_FLAG = 'A' , LOC\_ID = :NEW.PICKUP\_LOC WHERE REGISTRATION\_NUMBER = :NEW.REG\_NUM;

ELSE

IF NVL(TO\_CHAR(:NEW.ACT\_RET\_DT\_TIME),'NULL') <> 'NULL' THEN UPDATE CAR SET AVAILABILITY\_FLAG = 'A' , LOC\_ID = :NEW.DROP\_LOC, MILEAGE = MILEAGE+GET\_MILEAGE WHERE REGISTRATION\_NUMBER = :NEW.REG\_NUM;

END IF;

END IF;

END;

/

26

**10.CONCLUSION**

During the course of this project, we learnt a lot of the work and best practices that go into creating a database, the rules to construct a good ER diagram, How to come up with relational schema mapping from the ER diagram, deriving the functional dependencies and how to normalize the relational schema. We learnt on how to design a system from Database perspective and how to efficiently store and manipulate data.

27