

**AMERICAN INTERNATIONAL UNIVERSITY BANGLADESH**  
**(AIUB)**

FACULTY OF SCIENCE & TECHNOLOGY



**Course Title**  
**INTRODUCTION TO DATABASE**

**Summer 2022-2023**  
**Section: G**

**TITLE**  
Car Shop Management System

**Supervised By**

MD Sajid Bin Faisal

**Submitted By: Group no:12**

Name	ID
MD. Nazmus Sadat Numan	22-48497-3

## **TABLE OF CONTENTS**

SL	Topic	page
01	Introduction	02
02	Case study	02
03	ER Diagram	03
04	Normalization	04 - 05
05	Finalization and final tables	06
06	Table creation	07 - 14
07	Value insertion	15 - 22
08	Query test	23 - 31
09	Conclusion	32

## **Introduction**

This initiative brings together elements, in the car shop to create a harmonious environment for managers, employees and customers. Built on Oracle Live, it empowers managers with insights based on data analysis. Employees benefit from workflows and tools that prioritize customer satisfaction. As for the customers themselves they can expect a buying experience, with support and flexible payment options. This project isn't just software, it's an economic engine, driving customer satisfaction, xemployee productivity, and informed decisions. Whether you're running the shop playing your part or simply enjoying the journey this system takes your car shop experience to a level.

## **Case Study**

In a Car shop management system, the Car shop has one Manager. The system is responsible for keeping the manager's name, ID, salary, address, and phone number. The manager manages the employees. There is only one manager responsible for managing multiple employees. The system also stores the employee's information, including their employee ID, name, job type, salary, address, and phone number. Employee assists customers. Multiple employees can assist multiple customers at the same time. Customer buys cars. Here, one customer can buy multiple cars, and one car can be bought by multiple customers. The system stores the customer's ID, name, email, phone number, address, and car information such as car ID, car name, car color, car brand, and car price. After buying cars, the customer makes payment using any payment method. Payment information such as payment date, payment ID, and payment type are stored by the system.

## ER-Diagram

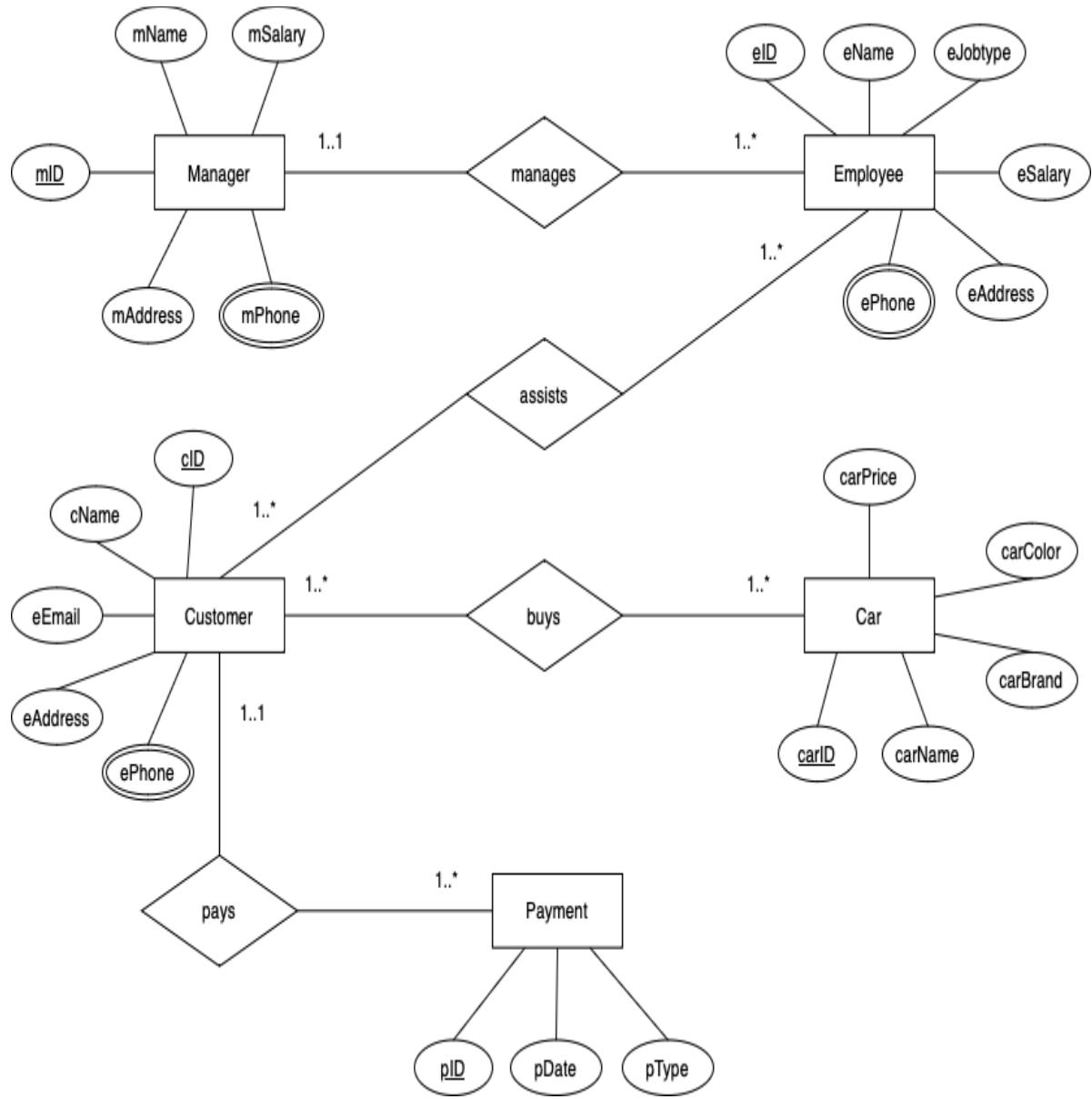


Fig3.1:ER-Diagram

## Normalization

### Manages Relation: (1..\*)

UNF: (m\_id, m\_name, m\_phone, m\_address, m\_salary, e\_id, e\_name, e\_phone, e\_address, e\_salary, e\_jobtype)

1NF: (m\_id, m\_name, m\_phone, m\_address, m\_salary, e\_id, e\_name, e\_phone, e\_address, e\_salary, e\_jobtype)

2NF:

1. m\_id (PK), m\_name, m\_phone, m\_address, m\_salary
2. e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype, m\_id (FK)

3NF:

1. m\_id (PK), m\_name, m\_phone, m\_address, m\_salary
2. e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype, m\_id (FK)

### Assists Relation: (\*..\*)

UNF: ( e\_id, e\_name, e\_phone, e\_address, e\_salary, e\_jobtype, c\_id, c\_name, c\_address, c\_phone, c\_email)

1NF: ( e\_id, e\_name, e\_phone, e\_address, e\_salary, e\_jobtype, c\_id, c\_name, c\_address, c\_phone, c\_email)

2NF:

1. e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype
2. c\_id (PK), c\_name, c\_address, c\_phone, c\_email
3. e\_id (PK), c\_id (FK)

3NF:

1. e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype
2. c\_id (PK), c\_name, c\_address, c\_phone, c\_email
3. e\_id (PK), c\_id (FK)

### **Buys Relation:** (\*..\*)

UNF: (c\_id, c\_name, c\_address, c\_phone, c\_email, car\_id, car\_name, car\_brand, car\_color, car\_price)

1NF: (c\_id, c\_name, c\_address, c\_phone, c\_email, car\_id, car\_name, car\_brand, car\_color, car\_price)

2NF:

1. c\_id (PK), c\_name, c\_address, c\_phone, c\_email
2. car\_id (PK), car\_name, car\_brand, car\_color, car\_price
3. c\_id (PK), car\_id (FK)

3NF:

1. c\_id (PK), c\_name, c\_address, c\_phone, c\_email
2. car\_id (PK), car\_name, car\_brand, car\_color, car\_price
3. c\_id (PK), car\_id (FK)

### **Pays Relation:** (1..\*)

UNF: (c\_id, c\_name, c\_address, c\_phone, c\_email, p\_id, p\_date, p\_type)

1NF: (c\_id, c\_name, c\_address, c\_phone, c\_email, p\_id, p\_date, p\_type)

2NF:

1. c\_id (PK), c\_name, c\_address, c\_phone, c\_email
2. p\_id (PK), p\_date, p\_type, c\_id(FK)

3NF:

1. c\_id (PK), c\_name, c\_address, c\_phone, c\_email
2. p\_id (PK), p\_date, p\_type, c\_id(FK)

## Finalization:

1. m\_id (PK), m\_name, m\_phone, m\_address, m\_salary
2. e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype, m\_id (FK)
3. e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype
4. c\_id (PK), c\_name, c\_address, c\_phone, c\_email
5. e\_id (PK), c\_id (FK)
6. ~~e\_id (PK), e\_name, e\_address, e\_phone, e\_email~~
7. car\_id (PK), car\_name, car\_brand, car\_color, car\_price
8. c\_id (PK), car\_id (FK)
9. ~~e\_id (PK), e\_name, e\_address, e\_phone, e\_email~~
10. p\_id (PK), p\_date, p\_type, c\_id (FK)

## Final Table

1. **Manager Table:** (m\_id (PK), m\_name, m\_phone, m\_address, m\_salary)
2. **Employee Table:** (e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype)
3. **Customer Table:** (c\_id (PK), c\_name, c\_address, c\_phone, c\_email)
4. **Car Table:** (car\_id (PK), car\_name, car\_brand, car\_color, car\_price)
5. **Manage Table:** (e\_id (PK), e\_name, e\_phone, e\_address, e\_salary, e\_jobtype, m\_id (FK))
6. **Assist Table:** (e\_id (PK), c\_id (FK))
7. **Buy Table:** (c\_id (PK), car\_id (FK))
8. **Pay Table:** (p\_id (PK), p\_date, p\_type, c\_id (FK))

## Table creation

### 1. Manager Table:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, nazmussadatnuman92@gmail.com, Dark mode switch.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- SQL Code:**

```
1 v create table manager(m_id number(10) primary key, m_name varchar2(30), m_salary number(10),
2      m_phone number(11), m_address varchar2(30));
3  describe manager;
```
- Output:** Table created.
- Table Definition:** TABLE MANAGER

Column	Null?	Type
M_ID	NOT NULL	NUMBER(10,0)
M_NAME	-	VARCHAR2(30)
M_SALARY	-	NUMBER(10,0)
M_PHONE	-	NUMBER(11,0)
M_ADDRESS	-	VARCHAR2(30)

- Buttons:** Download CSV
- Message:** 5 rows selected.

Fig6.1: creation and description of the manager table.

## 2. Employee Table:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, nazmussadatnuman92@gmail.com, Dark Mode toggle.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- SQL Editor:** SQL code to create and describe the employee table.

```
1 v create table employee(e_id number(10) primary key, e_name varchar2(30), e_salary number(10),
2      e_phone number(11), e_jobtype varchar2(30), e_address varchar2(30));
3  describe employee;
```
- Output:** Confirmation message "Table created." and a table description for the EMPLOYEE table.
- EMPLOYEE Table Description:**

Column	Null?	Type
E_ID	NOT NULL	NUMBER(10,0)
E_NAME	-	VARCHAR2(30)
E_SALARY	-	NUMBER(10,0)
E_PHONE	-	NUMBER(11,0)
E_JOBTYPE	-	VARCHAR2(30)
E_ADDRESS	-	VARCHAR2(30)
- Buttons:** Download CSV.
- Status:** 6 rows selected.

Fig6.2: creation and description of the employee table.

### 3. Customer Table:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, nazmussadatnuman92@gmail.com, Dark Mode toggle.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- SQL Editor:** The code entered is:

```
1 create table customer(c_id number(10) primary key, c_name varchar2(30), c_phone number(11),
2      c_email varchar2(30), c_address varchar2(30));
3 describe customer;
```
- Output:** "Table created."
- Table Definition:** TABLE CUSTOMER
- Table Data:**

Column	Null?	Type
C_ID	NOT NULL	NUMBER(10,0)
C_NAME	-	VARCHAR2(30)
C_PHONE	-	NUMBER(11,0)
C_EMAIL	-	VARCHAR2(30)
C_ADDRESS	-	VARCHAR2(30)
- Buttons:** Download CSV
- Message:** 5 rows selected.

Fig6.3: creation and description of the customer table.

#### 4. Car Table:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, nazmussadatnuman92@gmail.com, Dark Mode toggle.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- SQL Editor:** SQL code to create and describe the 'car' table.

```
1 v create table car (car_id number(10) primary key, car_name varchar2(30), car_brand varchar2(30),
2      car_color varchar2(15), car_price number(10));
3  describe car;
```
- Output:** Confirmation message "Table created." and a table description for the 'CAR' table.
- Table Description:** TABLE CAR
- Table Data:**

Column	Null?	Type
CAR_ID	NOT NULL	NUMBER(10,0)
CAR_NAME	-	VARCHAR2(30)
CAR_BRAND	-	VARCHAR2(30)
CAR_COLOR	-	VARCHAR2(15)
CAR_PRICE	-	NUMBER(10,0)
- Buttons:** Download CSV.
- Message:** 5 rows selected.

Fig6.4: creation and description of the table.

## 5. Pay Table:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, User: nazmussadatnuman92@gmail.com, Dark Mode toggle.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- SQL Editor:** Contains the following code:

```
1 v create table pay(p_id number(10) primary key, p_date varchar2(30), p_type varchar2(30),
2      c_id number(10) constraint payfk references customer(c_id));
3  describe pay;
```
- Output:** Shows the message "Table created."
- Table Definition:** A table named "TABLE PAY" with four columns:

Column	Null?	Type
P_ID	NOT NULL	NUMBER(10,0)
P_DATE	-	VARCHAR2(30)
P_TYPE	-	VARCHAR2(30)
C_ID	-	NUMBER(10,0)
- Buttons:** Download CSV.
- Status:** 4 rows selected.

Fig6.5: creation and description of the pay table.

## 6. Mange Table:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, nazmussadatnuman92@gmail.com, Logout.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- SQL Editor:** The code entered is:

```
1 ✓ create table manage(e_id number(10) primary key, e_name varchar2(30), e_salary number(10), e_phone number(11),
2 e_jobtype varchar2(30), e_address varchar2(30), m_id number(10) constraint managefk references manager(m_id));
3 describe manage;
```
- Output:** Table created.
- Table Definition:** TABLE MANAGE

Column	Null?	Type
E_ID	NOT NULL	NUMBER(10,0)
E_NAME	-	VARCHAR2(30)
E_SALARY	-	NUMBER(10,0)
E_PHONE	-	NUMBER(11,0)
E_JOBTYPE	-	VARCHAR2(30)
E_ADDRESS	-	VARCHAR2(30)
M_ID	-	NUMBER(10,0)

- Buttons:** Download CSV.
- Message:** 7 rows selected.

Fig6.6: creation and description of the manage table.

## 7. Assist Table:

The screenshot shows an SQL worksheet interface with the following details:

- Toolbar:** Includes icons for L... (Logout), Feedback, Help, and a user account (nazmussadatnuman92@gmail.com) with a dropdown menu.
- SQL Worksheet Header:** Shows "SQL Worksheet" and buttons for Clear, Find, Actions (with a dropdown arrow), Save, and Run.
- SQL Code:** The code entered is:

```
1 ✓ create table assist(e_id number(10) primary key,
2   c_id number(10) constraint assistfk references customer(c_id));
3 describe assist;
```
- Output:** The message "Table created." is displayed.
- Table Definition:** A table named "TABLE ASSIST" is shown with the following structure:

Column	Null?	Type
E_ID	NOT NULL	NUMBER(10,0)
C_ID	-	NUMBER(10,0)
- CSV Option:** A button labeled "Download CSV" is available.
- Status:** The message "2 rows selected." is displayed at the bottom.

Fig6.7: creation and description of the assist table.

## 8. Buy Table:

The screenshot shows a database interface with a dark theme. At the top, there's a navigation bar with icons for message, help, user, and settings. Below it is a toolbar with icons for refresh, search, actions, and export. The main area is titled "SQL Worksheet". In the code editor, the following SQL statements are shown:

```
1 ✓ create table buy (c_id number(10) primary key,  
2   car_id number(10) constraint buyfk references car(car_id));  
3 describe buy;
```

Below the code, the message "Table created." is displayed. A table titled "TABLE BUY" is shown with two rows of data:

Column	Null?	Type
C_ID	NOT NULL	NUMBER(10,0)
CAR_ID	-	NUMBER(10,0)

A "Download CSV" button is located below the table. At the bottom of the interface, the message "2 rows selected." is displayed.

Fig6.8: creation and description of the buy table.

## Value Insertion

### 1. Manager:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, User: nazmussadathuman92@gmail.com, Dark Mode switch.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- SQL Query:**

```
1 ✓ insert into manager (m_id,m_name,m_salary,m_phone,m_address)
2     values(1,'MD. Nazmus Sadat Numan',100000,01724972425,'Rajbari,Dhaka');
3 select * from manager;
```
- Result Table:** A table showing the inserted data.

M_ID	M_NAME	M_SALARY	M_PHONE	M_ADDRESS
1	MD. Nazmus Sadat Numan	100000	1724972425	Rajbari,Dhaka
- Buttons:** Download CSV.

Fig7.1: Input values for the manager table and all its data.

## 2.Employee:

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar:** Includes "Live SQL" button, "Feedback", "Help", user "nazmussadatnuman92@gmail.com", and a "Run" button.
- SQL Worksheet:** Displays the following SQL code:

```
1 insert into employee (e_id,e_name,e_salary,e_phone,e_jobtype,e_address) values(101,'Hasibul Islam Hasib',10000,0170001111,'Selesman','Basundhara R/A');
2 insert into employee (e_id,e_name,e_salary,e_phone,e_jobtype,e_address) values(102,'Adriyana Jannat',10000,01710001112,'Selesman','Gulshan');
3 insert into employee (e_id,e_name,e_salary,e_phone,e_jobtype,e_address) values(103,'Mohammad Doad',10000,0170000131,'Selesman','Motijheel');
4 insert into employee (e_id,e_name,e_salary,e_phone,e_jobtype,e_address) values(104,'Shahanoor Islam',10000,01700121311,'Selesman','Dhamondi');
5 insert into employee (e_id,e_name,e_salary,e_phone,e_jobtype,e_address) values(105,'Redwanul Haque',10000,01700110011,'Selesman','Mirpur');
```
- Table:** A grid showing the inserted data:

E_ID	E_NAME	E_SALARY	E_PHONE	E_JOBTYPE	E_ADDRESS
101	Hasibul Islam Hasib	10000	1700001111	Selesman	Basundhara R/A
102	Adriyana Jannat	10000	1710001112	Selesman	Gulshan
103	Mohammad Doad	10000	1700000131	Selesman	Motijheel
104	Shahanoor Islam	10000	1700121311	Selesman	Dhamondi
105	Redwanul Haque	10000	1700110011	Selesman	Mirpur
- Buttons:** "Download CSV" and "5 rows selected."

Fig7.2: Input values for the employee table and all its data.

### 3.Customer:

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar:** Includes icons for Feedback, Help, and a user account (nazmussadatnuman92@gmail.com). There are also buttons for Clear, Find, Actions (with a dropdown arrow), Save, and Run.
- SQL Worksheet Area:** Displays the following SQL code:

```
1 insert into customer (c_id,c_name,c_phone,c_email,c_address) values(201,'Sheikh Hasina Wazed',01700000000,'sheikhhasina@gmail.com','Gopalganj');
2 insert into customer (c_id,c_name,c_phone,c_email,c_address) values(202,'Khaleda Khanam Putul',01700000001,'khaledazia@gmail.com','Dinajpur');
3 insert into customer (c_id,c_name,c_phone,c_email,c_address) values(203,'Sayem Sobhan Anvir',01700000011,'sayemsobhananvir@gmail.com','Basundhara R/A');
4 insert into customer (c_id,c_name,c_phone,c_email,c_address) values(204,'Salman F Rahman',01700000111,'salmanfrahman@gmail.com','Dhanmondi');
5 insert into customer (c_id,c_name,c_phone,c_email,c_address) values(205,'Shakib Al Hasan',01700100111,'shakibalhasan@gmail.com','Magura');
6
7 select * from customer;
```
- Table View:** A grid showing the data inserted into the customer table. The columns are C\_ID, C\_NAME, C\_PHONE, C\_EMAIL, and C\_ADDRESS. The data is as follows:

C_ID	C_NAME	C_PHONE	C_EMAIL	C_ADDRESS
201	Sheikh Hasina Wazed	1700000000	sheikhhasina@gmail.com	Gopalganj
202	Khaleda Khanam Putul	1700000001	khaledazia@gmail.com	Dinajpur
203	Sayem Sobhan Anvir	1700000011	sayemsobhananvir@gmail.com	Basundhara R/A
204	Salman F Rahman	1700000111	salmanfrahman@gmail.com	Dhanmondi
205	Shakib Al Hasan	1700100111	shakibalhasan@gmail.com	Magura

Fig7.3: Input values for the customer table and all its data.

#### 4.Car:

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, nazmussadatnuman92@gmail.com, Run button.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- Code Area:**

```

1 insert into car (car_id,car_name,car_brand,car_color,car_price) values(301,'Lamborghini Huracán Evo','Lamborghini','Blu Asteria',17519880);
2 insert into car (car_id,car_name,car_brand,car_color,car_price) values(302,'Rolls-Royce Ghost','Rolls-Royce','Black Sapphire',15894760);
3 insert into car (car_id,car_name,car_brand,car_color,car_price) values(303,'Bentley Flying Spur','Bentley','Beluga',14582340);
4 insert into car (car_id,car_name,car_brand,car_color,car_price) values(304,'Mercedes-Maybach S-Class','Mercedes-Maybach','Obsidian Black',13877000);
5 insert into car (car_id,car_name,car_brand,car_color,car_price) values(305,'Range Rover SVAutobiography','Land Rover','Corris Grey',13757500);
6 insert into car (car_id,car_name,car_brand,car_color,car_price) values(306,'Ferrari Roma','Ferrari','Rosso Corsa',12873200);
7 insert into car (car_id,car_name,car_brand,car_color,car_price) values(307,'Porsche Panamera Turbo S','Porsche','Black',12342200);
8 insert into car (car_id,car_name,car_brand,car_color,car_price) values(308,'Aston Martin DB11 V8','Aston Martin','Quantum Silver',11811160);
9 insert into car (car_id,car_name,car_brand,car_color,car_price) values(309,'Lexus LS 500h','Lexus','Deep Blue Mica',11280000);
10 insert into car (car_id,car_name,car_brand,car_color,car_price) values(310,'Quattroporte GranTurismo','Maserati','Nero Ribelle',10848840);
11
12 select * from customer;

```

Fig7.4: Input values for the car table.

The screenshot shows a table with the following data:

CAR_ID	CAR_NAME	CAR_BRAND	CAR_COLOR	CAR_PRICE
301	Lamborghini Huracán Evo	Lamborghini	Blu Asteria	17519880
302	Rolls-Royce Ghost	Rolls-Royce	Black Sapphire	15894760
303	Bentley Flying Spur	Bentley	Beluga	14582340
304	Mercedes-Maybach S-Class	Mercedes-Maybach	Obsidian Black	13877000
305	Range Rover SVAutobiography	Land Rover	Corris Grey	13757500
306	Ferrari Roma	Ferrari	Rosso Corsa	12873200
307	Porsche Panamera Turbo S	Porsche	Black	12342200
308	Aston Martin DB11 V8	Aston Martin	Quantum Silver	11811160
309	Lexus LS 500h	Lexus	Deep Blue Mica	11280000
310	Quattroporte GranTurismo	Maserati	Nero Ribelle	10848840

Download CSV  
10 rows selected.

Fig7.5: All data of car table.

## 5.Pay:

The screenshot shows a "Live SQL" interface with the following details:

- Toolbar:** Includes "Feedback", "Help", user info "nazmussadatnuman92@gmail.com", and a light/dark mode switch.
- SQL Worksheet:** Labeled "SQL Worksheet" with buttons for "Clear", "Find", "Actions", "Save", and "Run".
- SQL Code:** The following SQL statements are listed:

```
1 insert into pay (p_id,p_date,p_type,c_id) values(401,'10-Jan-2023','visa card',201);
2 insert into pay (p_id,p_date,p_type,c_id) values(402,'20-Feb-2023','cash',202);
3 insert into pay (p_id,p_date,p_type,c_id) values(403,'30-Mar-2023','cash',203);
4 insert into pay (p_id,p_date,p_type,c_id) values(404,'11-Apr-2023','cash',204);
5 insert into pay (p_id,p_date,p_type,c_id) values(405,'30-Aug-2023','visa card',205);
6
7 select * from pay;
```
- Data Table:** A table titled "pay" is displayed with the following rows:

P_ID	P_DATE	P_TYPE	C_ID
401	10-Jan-2023	visa card	201
402	20-Feb-2023	cash	202
403	30-Mar-2023	cash	203
404	11-Apr-2023	cash	204
405	30-Aug-2023	visa card	205
- Buttons:** "Download CSV" button.
- Message:** "5 rows selected."

Fig7.6: Input values for the pay table and all its data.

## 6.Manage:

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar:** Includes "Live SQL" button, Feedback, Help, User (nazmussadatnuman92@gmail.com), and a light/dark mode switch.
- SQL Worksheet:** Displays the following SQL code:

```
1 v insert into manage (e_id,e_name,e_salary,e_phone,e_jobtype,e_address,m_id) values(101,'Hasibul Islam Hasib',  
2      10000,01700001111,'Selesman','Basundhara R/A',1);  
3 v insert into manage (e_id,e_name,e_salary,e_phone,e_jobtype,e_address,m_id) values(102,'Adriyana Jannat',  
4      10000,01710001112,'Selesman','Gulshan',1);  
5 v insert into manage (e_id,e_name,e_salary,e_phone,e_jobtype,e_address,m_id) values(103,'Mohammad Doad',  
6      10000,01700000131,'Selesman','Motijheel',1);  
7 v insert into manage (e_id,e_name,e_salary,e_phone,e_jobtype,e_address,m_id) values(104,'Shahanoor Islam',  
8      10000,01700121311,'Selesman','Dhanmondi',1);  
9 v insert into manage (e_id,e_name,e_salary,e_phone,e_jobtype,e_address,m_id) values(105,'Redwanul Haque',  
10     10000,01700110011,'Selesman','Mirpur',1);  
11  
12 select * from manage;
```
- Data Table:** A grid showing the data inserted into the manage table.

E_ID	E_NAME	E_SALARY	E_PHONE	E_JOBTYPE	E_ADDRESS	M_ID
101	Hasibul Islam Hasib	10000	1700001111	Selesman	Basundhara R/A	1
102	Adriyana Jannat	10000	1710001112	Selesman	Gulshan	1
103	Mohammad Doad	10000	1700000131	Selesman	Motijheel	1
104	Shahanoor Islam	10000	1700121311	Selesman	Dhanmondi	1
105	Redwanul Haque	10000	1700110011	Selesman	Mirpur	1
- Buttons:** "Download CSV" and "5 rows selected."

Fig7.7: Input values for the manage table and all its data.

## 7.Assist:

The screenshot shows a "Live SQL" interface with a "SQL Worksheet". The code entered is:

```
1 insert into assist (e_id,c_id) values(101,201);
2 insert into assist (e_id,c_id) values(102,202);
3 insert into assist (e_id,c_id) values(103,203);
4 insert into assist (e_id,c_id) values(104,204);
5 insert into assist (e_id,c_id) values(105,205);
6
7 select * from assist;
```

The results are displayed in a table:

E_ID	C_ID
101	201
102	202
103	203
104	204
105	205

A "Download CSV" button is present below the table. The message "5 rows selected." is shown at the bottom.

Fig7.8: Input values for the assist table and all its data.

### 8.Buy:

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar:** Includes icons for message, help, user, and settings.
- Section Header:** "SQL Worksheet".
- Actions:** Buttons for refresh, save, and run.
- SQL Code:**

```
1 insert into buy (c_id,car_id) values(201,305);
2 insert into buy (c_id,car_id) values(202,304);
3 insert into buy (c_id,car_id) values(203,306);
4 insert into buy (c_id,car_id) values(204,301);
5 insert into buy (c_id,car_id) values(205,310);
6
7 select * from buy;
```
- Table Output:** A grid showing the data inserted into the buy table.

C_ID	CAR_ID
201	305
202	304
203	306
204	301
205	310
- Download CSV:** A button to download the data as a CSV file.
- Message:** "5 rows selected."

Fig7.9: Input values for the buy table and all its data.

## Query Test

### Simple Query:

Showing the new salary with 5% increment of all employees with their NAME, SALARY, 5% INCREMENT.

```
SELECT e_name "NAME", e_salary "SALARY", e_salary*0.05 "5%  
INCREMENT", (e_salary*0.05)+e_salary "NEW SALARY" FROM employee;
```

The screenshot shows a SQL worksheet interface with the following details:

- Header: Live SQL, Feedback, Help, User: nazmussadatnuman92@gmail.com, Dark Mode switch.
- Toolbar: Clear, Find, Actions, Save, Run.
- SQL Worksheet area:
  - Query 1: SELECT e\_name "NAME", e\_salary "SALARY", e\_salary\*0.05 "5% INCREMENT", (e\_salary\*0.05)+e\_salary "NEW SALARY" FROM employee;
- Result Table:

NAME	SALARY	5% INCREMENT	NEW SALARY
Hasibul Islam Hasib	10000	500	10500
Adriyana Jannat	10000	500	10500
Mohammad Doad	10000	500	10500
Shahanoor Islam	10000	500	10500
Redwanul Haque	10000	500	10500
- Buttons: Download CSV.
- Message: 5 rows selected.

Fig8.1: Simple Query command and full table as a result

### Single row function:

Showing the id, name, job, salary of the employees by changing the salary for the people whose id is 202, 203, 204, 205 respectively 5%, 7%, 9% and 10% increment as “New salary”.

```
SELECT e_id AS "ID", e_name AS "Name", e_jobtype AS "JOB", e_salary AS  
"SALARY", DECODE( e_id, 102, (e_salary * 0.05)+e_salary, 103, (e_salary *  
0.07)+e_salary, 104, (e_salary * 0.09)+e_salary, 105, (e_salary * 0.1)+e_salary,  
e_salary ) AS "NEW SALARY" FROM employee;
```

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar:** Includes "Live SQL", "Feedback", "Help", user info "nazmussadatnuman92@gmail.com", and a theme switch.
- SQL Worksheet:** Contains the following SQL code:
 

```
1 v SELECT e_id AS "ID", e_name AS "Name", e_jobtype AS "JOB", e_salary AS "SALARY", DECODE(e_id,
2           102, (e_salary * 0.05)+e_salary,
3           103, (e_salary * 0.07)+e_salary,
4           104, (e_salary * 0.09)+e_salary,
5           105, (e_salary * 0.1)+e_salary,
6           e_salary
7     ) AS "NEW SALARY"
8   FROM employee;
```
- Result Table:** A grid showing the output of the query. The columns are ID, Name, JOB, SALARY, and NEW SALARY. The data is as follows:
 

ID	Name	JOB	SALARY	NEW SALARY
101	Hasibul Islam Hasib	Selesman	10000	10000
102	Adriyana Jannat	Selesman	10000	10500
103	Mohammad Doad	Selesman	10000	10700
104	Shahanoor Islam	Selesman	10000	10900
105	Redwanul Haque	Selesman	10000	11000
- Buttons:** "Download CSV" and "5 rows selected."

Fig8.2: Single row function command and full table as a result.

### Multiple row function:

Calculating the average salary, sum of total salary, and count of employees for the manager from the manage table.

```
SELECT m_id, AVG(e_salary) AS avg_salary, SUM(e_salary) AS SUM_Salary,
COUNT(*) AS employee_count FROM manage GROUP BY m_id;
```

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar:** Includes "Live SQL", "Feedback", "Help", user info "nazmussadatnuman92@gmail.com", and a theme switch.
- SQL Worksheet:** Contains the following SQL code:
 

```
1 v SELECT m_id, AVG(e_salary) AS avg_salary, SUM(e_salary) AS SUM_Salary, COUNT(*) AS employee_count
2   FROM manage GROUP BY m_id;
```
- Result Table:** A grid showing the output of the query. The columns are M\_ID, AVG\_SALARY, SUM\_SALARY, and EMPLOYEE\_COUNT. The data is as follows:
 

M_ID	AVG_SALARY	SUM_SALARY	EMPLOYEE_COUNT
1	10000	50000	5

Fig8.3: Multiple row function command and full table as a result.

### Single row subquery:

Showing the name, ID, and brand of the car that has a price less than the average.

```
SELECT car_id, car_name, car_brand FROM car WHERE car_price < (SELECT AVG(car_price) FROM car);
```

The screenshot shows a SQL worksheet interface with the following details:

- Header: Live SQL, Feedback, Help, nazmussadatnuman92@gmail.com, Dark Mode icon.
- Toolbar: SQL Worksheet, Clear, Find, Actions, Save, Run.
- Query: 1 SELECT car\_id, car\_name, car\_brand FROM car WHERE car\_price < (SELECT AVG(car\_price) FROM car);
- Result Table:

CAR_ID	CAR_NAME	CAR_BRAND
306	Ferrari Roma	Ferrari
307	Porsche Panamera Turbo S	Porsche
308	Aston Martin DB11 V8	Aston Martin
309	Lexus LS 500h	Lexus
310	Quattroporte GranTurismo	Maserati
- Buttons: Download CSV.
- Message: 5 rows selected.

Fig8.4: Single row subquery command and full table as a result.

### Multiple row subquery:

Showing the name, ID, and address of employees whose names are longer than 'Mohammed Doad'.

```
SELECT e_id, e_name, e_address FROM employee WHERE e_id = ANY (SELECT e_id FROM employee WHERE LENGTH(e_name)>LENGTH('Mohammad Doad'));
```

The screenshot shows a SQL worksheet interface with a dark theme. At the top, there are navigation links for Feedback, Help, and a user account (nazmussadatnuman92@gmail.com). Below the header are buttons for Clear, Find, Actions, Save, and Run.

```
1 SELECT e_id, e_name, e_address FROM employee WHERE e_id = ANY (SELECT e_id FROM employee WHERE LENGTH(e_name)>LENGTH('Mohammad Doad'));
```

The main area displays a table with four rows:

E_ID	E_NAME	E_ADDRESS
101	Hasibul Islam Hasib	Basundhara R/A
102	Adriyana Jannat	Gulshan
104	Shahanoor Islam	Dhanmondi
105	Redwanul Haque	Mirpur

Below the table are two buttons: "Download CSV" and "4 rows selected."

Fig8.5: Multiple row subquery command and full table as a result.

### Right outer join:

List of all cars and their details, along with the corresponding customer ID of those who bought them. If a car has not been bought, display NULL for the customer ID.

```
SELECT c.car_id, c.car_name, c.car_brand, c.car_color, b.c_id "Customer ID"
FROM car c, buy b WHERE c.car_id = b.car_id(+);
```

The screenshot shows a SQL worksheet interface with a dark theme. At the top, there are navigation links for Feedback, Help, and a user account (nazmussadatnuman92@gmail.com). Below the header are buttons for Clear, Find, Actions, Save, and Run.

```
1 SELECT c.car_id, c.car_name, c.car_brand, c.car_color, b.c_id "Customer ID" FROM car c, buy b WHERE c.car_id = b.car_id(+);
```

The main area displays a table with nine rows:

CAR_ID	CAR_NAME	CAR_BRAND	CAR_COLOR	CUSTOMER_ID
305	Range Rover SVAutobiography	Land Rover	Corris Grey	201
304	Mercedes-Maybach S-Class	Mercedes-Maybach	Obsidian Black	202
306	Ferrari Roma	Ferrari	Rosso Corsa	203
301	Lamborghini Huracán Evo	Lamborghini	Blu Asteria	204
310	Quattroporte GranTurismo	Maserati	Nero Ribelle	205
302	Rolls-Royce Ghost	Rolls-Royce	Black Sapphire	-
303	Bentley Flying Spur	Bentley	Beluga	-
309	Lexus LS 500h	Lexus	Deep Blue Mica	-
307	Porsche Panamera Turbo S	Porsche	Black	-
308	Aston Martin DB11 V8	Aston Martin	Quantum Silver	-

Fig8.6: Right outer join creation command and full table as a result.

## Self-join:

Self join of car table. Car table divided into 2 part: (a.car\_id, a.car\_name) and (b.car\_id, b.car\_brand, b.car\_price)

```
SELECT a.car_id, a.car_name, b.car_id, b.car_brand, b.car_price FROM car a,  
car b WHERE a.car_id = b.car_id;
```

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar: Includes "Live SQL" button, "Feedback", "Help", user email "nazmussadatnuman92@gmail.com", and a brightness icon.
- Header: "SQL Worksheet" with buttons for "Clear", "Find", "Actions", "Save", and a "Run" button.
- Query Editor: A code editor containing the SQL command: `1 SELECT a.car_id, a.car_name, b.car_id, b.car_brand, b.car_price FROM car a, car b WHERE a.car_id = b.car_id;`
- Result Table: A data grid displaying the results of the self-join. The columns are labeled: CAR\_ID, CAR\_NAME, CAR\_ID, CAR\_BRAND, and CAR\_PRICE. The data consists of 10 rows, each representing a car with its ID, name, brand, and price.

CAR_ID	CAR_NAME	CAR_ID	CAR_BRAND	CAR_PRICE
301	Lamborghini Huracán Evo	301	Lamborghini	17519880
302	Rolls-Royce Ghost	302	Rolls-Royce	15894760
303	Bentley Flying Spur	303	Bentley	14582340
304	Mercedes-Maybach S-Class	304	Mercedes-Maybach	13877000
305	Range Rover SVAutobiography	305	Land Rover	13757500
306	Ferrari Roma	306	Ferrari	12873200
307	Porsche Panamera Turbo S	307	Porsche	12342200
308	Aston Martin DB11 V8	308	Aston Martin	11811160
309	Lexus LS 500h	309	Lexus	11280000
310	Quattroporte GranTurismo	310	Maserati	10848840

Fig8.7: Self join creation command and full table as a result.

## Simple view:

Create a view named as customer\_view where car name, brand, color and price will be shown to the customer.

```
CREATE VIEW customer_view AS SELECT  
car_name,car_brand,car_color,car_price FROM car;
```

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar: Includes "Live SQL" button, "Feedback", "Help", user info "nazmussadatnuman92@gmail.com", and a light/dark mode switch.
- Section Header: "SQL Worksheet".
- Text Area:

```
1 CREATE VIEW customer_view AS SELECT car_name,car_brand,car_color,car_price FROM car;
```
- Status Bar: "View created."

Fig8.8: Simple view creation command.

DESCRIBE customer\_view;

The screenshot shows a SQL worksheet interface with the following details:

- Toolbar: Includes "Live SQL" button, "Feedback", "Help", user info "nazmussadatnuman92@gmail.com", and a light/dark mode switch.
- Section Header: "SQL Worksheet".
- Text Area:

```
1 DESCRIBE customer_payment_view;
```
- Table: Describes the CUSTOMER\_PAYMENT\_VIEW.

Column	Null?	Type
COUSTOMER_ID	NOT NULL	NUMBER(10,0)
COUSTOMER_NAME	-	VARCHAR2(30)
PAYMENT_ID	NOT NULL	NUMBER(10,0)
PAYMENT_DATE	-	VARCHAR2(30)
PAYMENT_TYPE	-	VARCHAR2(30)
- Text Area: "Download CSV"
- Status Bar: "5 rows selected."

Fig8.9: Description of the simple view.

```
SELECT * FROM customer_view;
```

The screenshot shows a SQL worksheet interface with the following details:

- Header: Live SQL, Feedback, Help, User: nazmussadatnuman92@gmail.com, Dark Mode toggle.
- Toolbar: Clear, Find, Actions, Save, Run.
- Query: 1 | `SELECT * FROM customer_view;`
- Result Table:

CAR_NAME	CAR_BRAND	CAR_COLOR	CAR_PRICE
Lamborghini Huracán Evo	Lamborghini	Blu Asteria	17519880
Rolls-Royce Ghost	Rolls-Royce	Black Sapphire	15894760
Bentley Flying Spur	Bentley	Beluga	14582340
Mercedes-Maybach S-Class	Mercedes-Maybach	Obsidian Black	13877000
Range Rover SVAutobiography	Land Rover	Corris Grey	13757500
Ferrari Roma	Ferrari	Rosso Corsa	12873200
Porsche Panamera Turbo S	Porsche	Black	12342200
Aston Martin DB11 V8	Aston Martin	Quantum Silver	11811160
Lexus LS 500h	Lexus	Deep Blue Mica	11280000
Quattroporte GranTurismo	Maserati	Nero Ribelle	10848840

Fig8.10: Result of the simple view as a whole table.

### Complex view:

Create a view named as customer\_payment\_view where Customer ID, Name and Payment ID, Date, Type will be shown.

```
CREATE VIEW customer_payment_view (Customer_ID, Customer_Name,  
Payment_ID, Payment_Date, Payment_Type) AS SELECT c.c_id, c.c_name,  
p.p_id, p.p_date, p.p_type FROM customer c, pay p WHERE c.c_id = p.c_id;
```

The screenshot shows a SQL worksheet interface with the following details:

- Header: Live SQL, Feedback, Help, User: nazmussadatnuman92@gmail.com, Dark mode switch.
- Toolbar: Clear, Find, Actions, Save, Run.
- SQL Worksheet area:

```
1 ✓ CREATE VIEW customer_payment_view (Customer_ID, Customer_Name, Payment_ID, Payment_Date, Payment_Type)
2     AS SELECT c.c_id, c.c_name, p.p_id, p.p_date, p.p_type FROM customer c, pay p WHERE c.c_id = p.c_id;
```
- Message bar: View created.

Fig8.11: Complex view creation command.

DESCRIBE customer\_payment\_view;

The screenshot shows a SQL worksheet interface with the following details:

- Header: Live SQL, Feedback, Help, User: nazmussadatnuman92@gmail.com, Dark mode switch.
- Toolbar: Clear, Find, Actions, Save, Run.
- SQL Worksheet area:

```
1 DESCRIBE customer_payment_view;
```
- Output area:

VIEW CUSTOMER\_PAYMENT\_VIEW

Column	Null?	Type
COUSTOMER_ID	NOT NULL	NUMBER(10,0)
COUSTOMER_NAME	-	VARCHAR2(30)
PAYMENT_ID	NOT NULL	NUMBER(10,0)
PAYMENT_DATE	-	VARCHAR2(30)
PAYMENT_TYPE	-	VARCHAR2(30)

Download CSV

5 rows selected.

Fig8.12: Description of the complex view.

```
SELECT * FROM customer_payment_view;
```

The screenshot shows a SQL worksheet interface with the following details:

- Header:** Live SQL, Feedback, Help, User: nazmussadatnuman92@gmail.com, Dark Mode switch.
- Toolbar:** SQL Worksheet, Clear, Find, Actions, Save, Run.
- Query:** 1 | SELECT \* FROM customer\_payment\_view;
- Result Table:** A table with columns COUSTOMER\_ID, COUSTOMER\_NAME, PAYMENT\_ID, PAYMENT\_DATE, and PAYMENT\_TYPE. The data is as follows:

COUSTOMER_ID	COUSTOMER_NAME	PAYMENT_ID	PAYMENT_DATE	PAYMENT_TYPE
201	Sheikh Hasina Wazed	401	10-Jan-2023	visa card
202	Khaleda Khanam Putul	402	20-Feb-2023	cash
203	Sayem Sobhan Anvir	403	30-Mar-2023	cash
204	Salman F Rahman	404	11-Apr-2023	cash
205	Shakib Al Hasan	405	30-Aug-2023	visa card

- Buttons:** Download CSV.
- Message:** 5 rows selected.

Fig8.13: Result of the complex view as a whole table.

## **Conclusion**

I have shown all the queries to create the tables in Live Oracle. Also, I showed the queries to insert the values and tested multiple queries then took screenshots of them. Here, I have established four different relationships between the entities. Due to the normalization process, my work has become easier.