INTRODUCTION TO DATABASE SYSTEMS

Name: Muhammad Saad (L1F22BSCS0937)

Section: D-16

Project: Fuel Pump Management System

Date of Submission: 19-06-2024

Submitted To: Sir Afham Nazir

TABLE OF CONENTS

Phase-01

Description:

The Fuel Pump Management System streamlines gas station operations by tracking pump details, sales transactions, and customer information. It ensures fuel availability and maIntains equipment through maIntenance records. Employees are associated with transactions and maIntenance tasks, while suppliers provide fuel and maIntenance services. Fuel types are categorized for efficient management, linking pumps, tank refills, and suppliers. This Integrated system enhances operational efficiency, customer service, and inventory control, optimizing the overall management of the gas station.

Entities:

1. Fuel pump

Attributes	Datatype	
Pump_ID	Int	Primary key
Location	VarChar	
Capacity	Float	
Fuel_type	Char	
Status	Char	

2. Transaction

Attributes	Datatype	
Transaction_ID	Int	Primary key
Date time	Date	
Fuel Amount	Float	
Payment_method	Char	

3. Customer

Attributes	Datatype	
Customer_ID	Int	Primary key
Name	Char	
Contact Number	Int	
Email	VarChar	

Address	VarChar	

4. Employee

Attributes	Datatype	
Employee_ID	Int	Primary key
Name	Char	
Contact Number	Int	
Email	VarChar	
Position	VarChar	

5. Fuel Tank

Attributes	Datatype	
Tank_ID	Int	Primary key
Location	Char	
Capacity	float	
Fuel_type	Char	
Current_level	Float	

6. Payment

Attributes	Datatype	
Payment_ID	Int	Primary key
Amount	float	
Payment method	Char	
Date_Time	DateTime	

7. Fuel Type

Attributes	Datatype	
Fuel_Type_ID	Int	Primary key
Type_Name	Char	

8. Tank Refill

Attributes	Datatype	
Refill_ID	Int	Primary key
Date_Time	DateTime	
Refill_Amount	Float	

9. MaIntenance Record

Attributes	Datatype	
MaIntenance _ID	Int	Primary key
Date_Time	DateTime	
Description	VarChar	
Cost	Float	

10. Supplier

Attributes	Datatype	
1. Supplier _ID	Int	Primary key
Name	Char	
Contact Number	Int	
Email	VarChar	
Address	VarChar	

Relationships

1. **Fuel Pump and Transactions:**
One-to-Many with Transaction: Each fuel pump can have multiple transactions.

2. **Fuel Pump and MaIntenance Records:**
One-to-One with MaIntenance Record: Each fuel pump can have multiple maIntenance records.
3. **Transaction and Fuel Pump**:
Many-to-One with Fuel Pump: Many transactions can be associated with one fuel pump.

4. **Transaction and Customer**:
Many-to-One with Customer: Many transactions can be associated with one customer.

5. **Transaction and Payment**:
One-to-One with Payment: Each transaction has one payment associated with it.

6. **Transaction and Fuel Type:**
   Many-to-One with Fuel Type: Each transaction involves a specific fuel type.

7. **Customer and Transaction**:
One-to-Many with Transaction: Each customer can have multiple transactions.

8. **Employee and Transaction**:
One-to-Many with Transaction: An employee can handle multiple transactions.

9. **Employee and MaIntenance Record:**
One-to-Many with MaIntenance Record: An employee can be associated with multiple maIntenance records.

10. **Supplier and Tank Refill:**
One-to-Many with Tank Refill: A supplier can provide fuel for multiple tank refills.

11. **Supplier and MaIntenance Record:**
One-to-Many with MaIntenance Record: A supplier can be associated with multiple maIntenance records (e.g., supplying parts or services).
12. **Supplier and Fuel Type**:
One-to-Many with Fuel Type: A supplier can provide multiple types of fuel.

13. **Fuel Tank and Tank Refill**:
One-to-Many with Tank Refill: Each fuel tank can have multiple tank refill records.

14. **Payment and Transaction**:
One-to-One with Transaction: Each payment is associated with one transaction.

15. **Fuel Type and Fuel Pump**:
    One-to-Many with Fuel Pump: Multiple fuel pumps can dispense the same type of fuel.

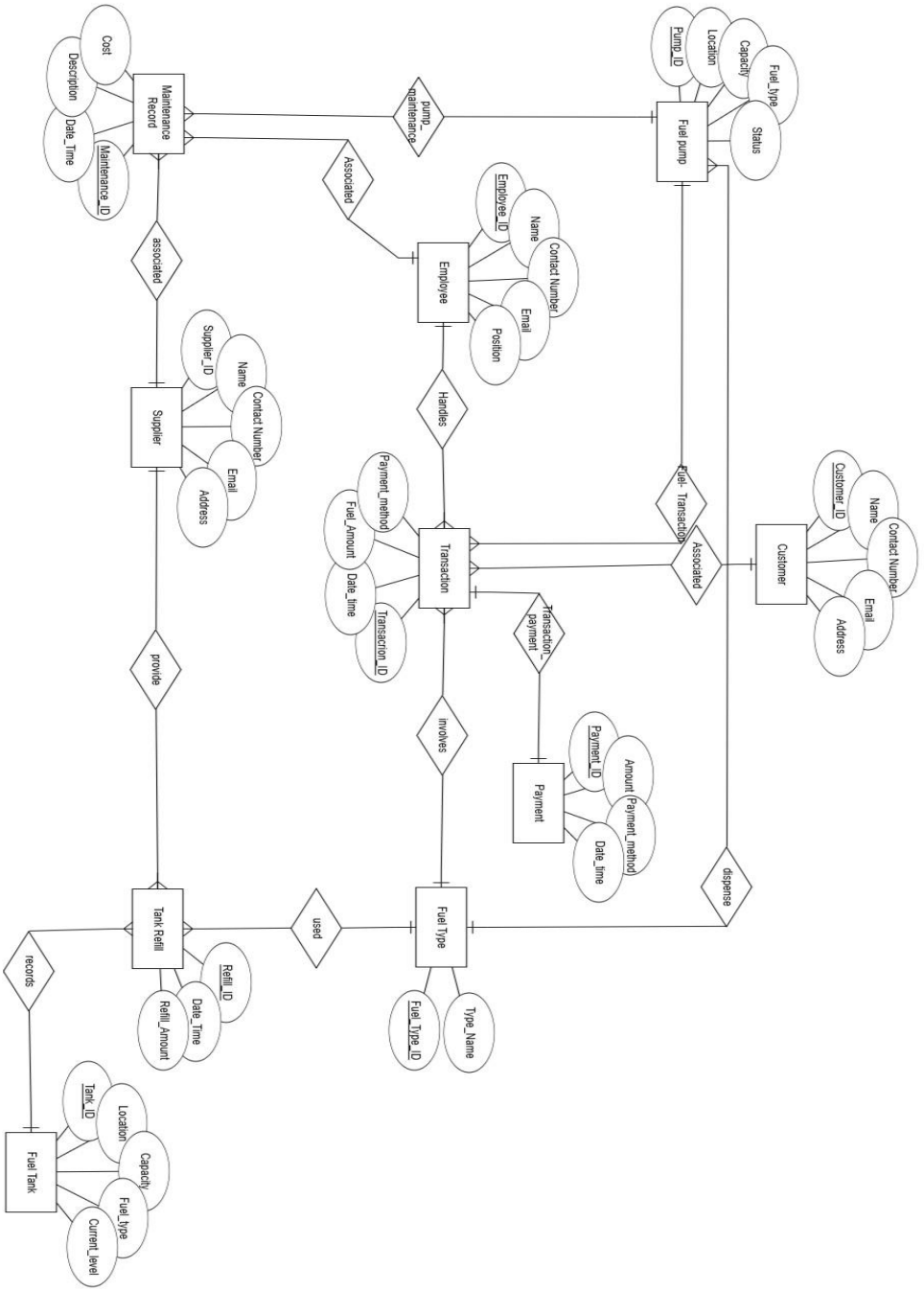
16. **Fuel Type and Tank Refill:**
    One-to-Many with Tank Refill: The same type of fuel can be used to refill multiple tanks.

17. **Tank Refill and Fuel Tank**:
Many-to-One with Fuel Tank: Many tank refills can be associated with one fuel tank.

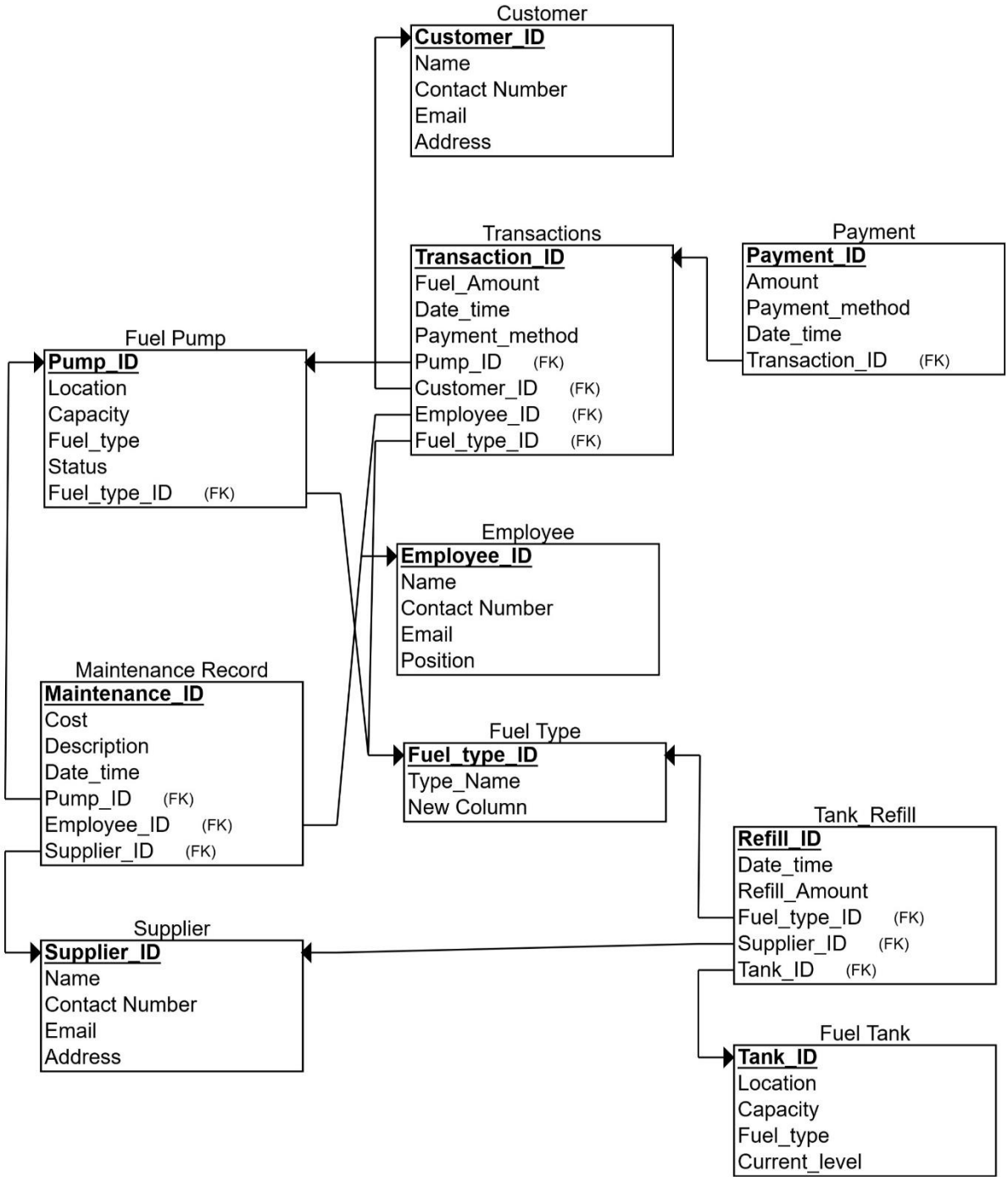
18. **MaIntenance Record and Fuel Pump**:
Many-to-One with Fuel Pump: Many maIntenance records can be associated with one fuel pump.

Phase-02

ERD:


Pump_ID
Location
Capacity
Fuel_type
Status
Fuel pump
pump_maintenance
Cost
Description
Date_Time
Maintenance_ID
Maintenance Record
Associated
Employee_ID
Name
Contact Number
Email
Position
Employee
associated
Supplier_ID
Name
Contact Number
Email
Address
Supplier
Handles
Payment_method
Fuel_Amount
Date_time
Transaction_ID
Transaction
Fuel_Transaction
Associated
Customer_ID
Name
Contact Number
Email
Address
Customer
Transaction_payment
provide
involves
Payment_ID
Amount (Payment_method)
Date_time
Payment
dispense
used
Tank Refill
Fuel Type
records
Refill_ID
Date_Time
Refill_Amount
Fuel_Type_ID
Type_Name
Tank_ID
Location
Capacity
Fuel_type
Current_level
Fuel Tank

Relational Schema:



Phase-03

DDL & DML

Queries:

1. Insertion:

```
create database project;

use project;

create table Fuel_pump(
Pump_ID Int(5),
Location varChar (20),
Capacity float(20),
Fuel_type Char(10),
status Char (20)
);

create table Transaction(
Transaction_ID Int(5),
Date_time Date,
Fuel_Amount float(20),
Payment_method Char(10)
);

create table customer(
Customer_ID Int(5),
Name Char(20),
Contact_Number Int(50),
Email varChar(25),
Address varChar(20)
);

create table Employee(
Employee_ID Int(5),
Name Char(20),
Contact_Number Int(50),
Email varChar(25),
Position varChar(10)
);

create table Fuel_Tank(
```

```
Tank_ID Int(5),
Location Char(20),
Capacity float(20),
Fuel_Type Char(10),
Current_level float(20)
);

create table Payment(
Payment_ID Int(5),
Amount float(20),
Payment_method Char(10),
Date_time datetime
);

create table Fuel_type(
Fuel_type_ID Int(5),
Type_name Char(10)
);

create table Tank_Refill(
Refill_ID Int(5),
Date_time datetime,
Refill_Amount float(10)
);

create table MaIntenance_Record(
MaIntenance_ID Int(5),
Date_time datetime,
Description varChar(20),
Cost float(20)
);

create table Supplier(
Supplier_ID Int(5),
Name Char(10),
Contact_Number Int(50),
Email varChar(25),
Address varChar(20)
);


insert Into Fuel_pump (Pump_ID, Location, Capacity, Fuel_type, Status) values
(1, 'Location_1', 892.95, 'Diesel', 'Inactive'),
(2, 'Location_2', 695.61, 'Petrol', 'Inactive'),
(3, 'Location_3', 704.34, 'Petrol', 'Inactive'),
(4, 'Location_4', 701.52, 'Gas', 'Inactive'),
(5, 'Location_5', 828.62, 'Diesel', 'Active'),
```

```
(6, 'Location_6', 544.03, 'Gas', 'Active'),
(7, 'Location_7', 952.81, 'Gas', 'Active'),
(8, 'Location_8', 615.62, 'Petrol', 'Active'),
(9, 'Location_9', 584.34, 'Petrol', 'Inactive'),
(10, 'Location_10', 831.40, 'Diesel', 'Inactive');
select*from Fuel_pump;

insert Into Transaction (Transaction_ID, Date_time, Fuel_Amount, Payment_method) values
(1, '2023-01-22', 86.40, 'Cash'),
(2, '2023-09-01', 96.64, 'Cash'),
(3, '2023-11-23', 82.48, 'Card'),
(4, '2023-10-09', 65.46, 'Cash'),
(5, '2023-05-10', 17.39, 'Card'),
(6, '2023-08-07', 82.46, 'Cash'),
(7, '2023-07-01', 41.96, 'Cash'),
(8, '2023-10-25', 31.30, 'Card'),
(9, '2023-08-08', 45.02, 'Cash'),
(10, '2023-10-02', 61.36, 'Card');
select*from Transaction;

insert Into Customer (Customer_ID, Name, Contact_Number, Email, Address) values
(1, 'Customer_1', 619993791, 'email1@example.com', 'Address_1'),
(2, 'Customer_2', 356731234, 'email2@example.com', 'Address_2'),
(3, 'Customer_3', 578755580, 'email3@example.com', 'Address_3'),
(4, 'Customer_4', 626374693, 'email4@example.com', 'Address_4'),
(5, 'Customer_5', 722520902, 'email5@example.com', 'Address_5'),
(6, 'Customer_6', 343480754, 'email6@example.com', 'Address_6'),
(7, 'Customer_7', 833398957, 'email7@example.com', 'Address_7'),
(8, 'Customer_8', 430518438, 'email8@example.com', 'Address_8'),
(9, 'Customer_9', 732227830, 'email9@example.com', 'Address_9'),
(10, 'Customer_10',436676712, 'email10@example.com', 'Address_10');
select*from customer;

insert Into Employee (Employee_ID, Name, Contact_Number, Email, Position) values
(1, 'Employee_1', 270863327, 'email1@company.com', 'Clerk'),
(2, 'Employee_2', 133568672, 'email2@company.com', 'Manager'),
(3, 'Employee_3', 191005173, 'email3@company.com', 'Clerk'),
(4, 'Employee_4', 151017849, 'email4@company.com', 'Attendant'),
(5, 'Employee_5', 254715837, 'email5@company.com', 'Clerk'),
(6, 'Employee_6', 130657187, 'email6@company.com', 'Attendant'),
(7, 'Employee_7', 030846552, 'email7@company.com', 'Manager'),
(8, 'Employee_8',857032973, 'email8@company.com', 'Manager'),
(9, 'Employee_9', 856267391, 'email9@company.com', 'Clerk'),
(10, 'Employee_10', 933766050, 'email10@company.com', 'Manager');
select*from Employee;

insert Into Fuel_Tank (Tank_ID, Location, Capacity, Fuel_Type, Current_level) values
```

(1, 'Location_1', 4851.44, 'Petrol', 2110.15),
(2, 'Location_2', 4033.89, 'Gas', 1053.55),
(3, 'Location_3', 2924.48, 'Diesel', 456.83),
(4, 'Location_4', 1190.37, 'Diesel', 509.62),
(5, 'Location_5', 3762.69, 'Gas', 1859.54),
(6, 'Location_6', 4126.03, 'Petrol', 1933.62),
(7, 'Location_7', 2455.72, 'Petrol', 2425.74),
(8, 'Location_8', 2843.82, 'Diesel', 1964.91),
(9, 'Location_9', 2781.35, 'Gas', 1488.78),
(10, 'Location_10', 2673.50, 'Petrol', 493.72);
select*from Fuel_Tank;

insert Into Payment (Payment_ID, Amount, Payment_method, Date_time) values
(1, 203.72, 'Cash', '2023-04-19 08:17:25'),
(2, 255.48, 'Card', '2023-06-08 14:43:15'),
(3, 73.98, 'Cash', '2023-10-28 05:01:11'),
(4, 276.20, 'Cash', '2023-02-13 10:30:47'),
(5, 475.72, 'Card', '2023-01-27 03:04:48'),
(6, 109.80, 'Card', '2023-11-03 11:32:01'),
(7, 438.24, 'Card', '2023-03-18 02:09:54'),
(8, 64.86, 'Cash', '2023-07-13 00:06:52'),
(9, 390.33, 'Cash', '2023-04-11 23:19:07'),
(10, 199.71, 'Card', '2023-09-16 00:51:57');
select*from Payment;

insert Into Fuel_type (Fuel_type_ID, Type_name) values
(1, 'Gas'),
(2, 'Petrol'),
(3, 'Petrol'),
(4, 'Petrol'),
(5, 'Diesel'),
(6, 'Gas'),
(7, 'Gas'),
(8, 'Diesel'),
(9, 'Diesel'),
(10, 'Gas');
select*from Fuel_type;

insert Into Tank_Refill (Refill_ID, Date_time, Refill_Amount) values
(1, '2023-11-25 19:00:37', 117.12),
(2, '2023-04-04 12:49:12', 567.15),
(3, '2023-08-12 10:21:25', 527.34),
(4, '2023-01-31 03:57:15', 406.93),
(5, '2023-08-23 01:37:21', 433.05),
(6, '2023-04-15 10:46:04', 468.33),
(7, '2023-01-14 22:28:51', 798.16),
(8, '2023-07-25 13:23:40', 112.49),

```
(9, '2023-06-04 17:25:58', 766.26),
(10, '2023-03-09 16:08:47', 637.31);
select*from Tank_Refill;

insert Into MaIntenance_Record (MaIntenance_ID, Date_time, Description, Cost) values
(1, '2023-09-01 19:00:02', 'Description_1', 909.85),
(2, '2023-11-22 04:33:49', 'Description_2', 310.92),
(3, '2023-07-19 02:45:46', 'Description_3', 495.51),
(4, '2023-05-18 07:44:41', 'Description_4', 678.57),
(5, '2023-04-27 21:28:46', 'Description_5', 586.72),
(6, '2023-09-14 11:32:51', 'Description_6', 864.85),
(7, '2023-12-19 15:29:04', 'Description_7', 498.55),
(8, '2023-06-19 13:28:17', 'Description_8', 273.12),
(9, '2023-11-30 06:06:25', 'Description_9', 364.54),
(10, '2023-04-18 15:47:51', 'Description_10', 831.34);
select*from MaIntenance_Record;

insert Into Supplier (Supplier_ID, Name, Contact_Number, Email, Address) values
(1, 'Supplier_1', 518149973, 'supplier1@mail.com', 'Address_1'),
(2, 'Supplier_2', 745876443, 'supplier2@mail.com', 'Address_2'),
(3, 'Supplier_3', 209385859, 'supplier3@mail.com', 'Address_3'),
(4, 'Supplier_4', 755352322, 'supplier4@mail.com', 'Address_4'),
(5, 'Supplier_5', 839647875, 'supplier5@mail.com', 'Address_5'),
(6, 'Supplier_6', 608036272, 'supplier6@mail.com', 'Address_6'),
(7, 'Supplier_7', 890998849, 'supplier7@mail.com', 'Address_7'),
(8, 'Supplier_8', 421819594, 'supplier8@mail.com', 'Address_8'),
(9, 'Supplier_9', 958091141, 'supplier9@mail.com', 'Address_9'),
(10, 'Suppler_10', 66812936, 'supplier10@mail.com', 'Address_10');
select*from Supplier;
```

2. Primary keys:

```
alter table Fuel_pump
add primary key (Pump_ID);

alter table Transaction
add primary key (Transaction_ID);

alter table customer
add primary key (customer_ID);

alter table Employee
add primary key (Employee_ID);

alter table Fuel_Tank
```

```
add primary key (Tank_ID);

alter table Payment
add primary key (Payment_ID);

alter table Fuel_type
add primary key (Fuel_type_ID);

alter table Tank_Refill
add primary key (Refill_ID);

alter table MaIntenance_Record
add primary key (MaIntenance_ID);

alter table Supplier
add primary key (Supplier_ID);
```

3. Foreign keys:

```
-- Transaction and Fuel Pump: Many-to-One with Fuel Pump
alter table Transaction
add foreign key (Pump_ID) references Fuel_pump(Pump_ID);

-- Transaction and Customer: Many-to-One with Customer
alter table Transaction
add foreign key (Customer_ID) references Customer(Customer_ID);

-- Transaction and Payment: One-to-One with Payment
alter table Payment
add foreign key (Transaction_ID) references Transaction(Transaction_ID);

-- Transaction and Fuel Type: Many-to-One with Fuel Type
alter table Transaction
add foreign key (Fuel_Type_ID) references Fuel_type(Fuel_type_ID);

-- Employee and Transaction: One-to-Many with Transaction
ALTER TABLE Transaction
ADD Employee_ID INT(5),
ADD FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID);


-- Employee and MaIntenance Record: One-to-Many with MaIntenance Record
alter table MaIntenance_Record
```

```
ADD M_Employee_ID INT(5),
add foreign key (M_Employee_ID) references Employee(Employee_ID);

-- Supplier and Tank Refill: One-to-Many with Tank Refill
alter table Tank_Refill
add foreign key (Supplier_ID) references Supplier(Supplier_ID);

-- Supplier and MaIntenance Record: One-to-Many with MaIntenance Record
ALTER TABLE MaIntenance_Record
ADD M_Supplier_ID INT(5),
ADD FOREIGN KEY (M_Supplier_ID) REFERENCES Supplier(Supplier_ID);


-- Supplier and Fuel Type: One-to-Many with Fuel Type
alter table Fuel_type
ADD Supplier_ID INT(5),
add foreign key (Supplier_ID) references Supplier(Supplier_ID);

-- Fuel Tank and Tank Refill: One-to-Many with Tank Refill
alter table Tank_Refill
add foreign key (Tank_ID) references Fuel_Tank(Tank_ID);

-- Fuel Type and Fuel Pump: One-to-Many with Fuel Pump
alter table Fuel_pump
ADD Fuel_type_ID INT(5),
add foreign key (Fuel_type_ID) references Fuel_type(Fuel_type_ID);

-- Fuel Type and Tank Refill: One-to-Many with Tank Refill
alter table Tank_Refill
ADD Fuel_type_ID INT(5),
add foreign key (Fuel_type_ID) references Fuel_type(Fuel_type_ID);

-- MaIntenance Record and Fuel Pump: Many-to-One with Fuel Pump
alter table MaIntenance_Record
add foreign key (Pump_ID) references Fuel_pump(Pump_ID);
```

4. Updation:

```
update Transaction
set Fuel_Amount = 150.00
```

```
where Transaction_ID = 1;
select* from Transaction;

update MaIntenance_Record
set Cost = 510.5
where MaIntenance_ID = 4;
select* from MaIntenance_Record;

update Employee
set Position = 'HR'
where Employee_ID = 2;
select* from Employee
```

5. Changing datatypes:

```
-- Change Supplier_ID from INT(5) to INT(10)
alter table Fuel_type
modify Supplier_ID Int(10);

-- Change Contact_Number from BIGINT(15) to VARCHAR(20)
alter table Supplier
modify Contact_Number varChar(20);

-- Change Capacity from FLOAT(20) to DECIMAL(10,2)
alter table Fuel_pump
modify Status varChar(50);

-- Change Current_level from FLOAT(20) to DECIMAL(8,2)
alter table Fuel_Tank
modify Location varChar(50);

-- Change Amount from FLOAT(20) to DECIMAL(10,2)
alter table Payment
modify Payment_method varChar(50);

-- Change Refill_Amount from FLOAT(10) to DECIMAL(8,2)
alter table Tank_Refill
modify Refill_Amount Int(50);

-- Change Cost from FLOAT(20) to DECIMAL(10,2)
alter table MaIntenance_Record
modify Description Char(50);
```

6. Rename table:

```
alter table Fuel_Tank rename to FuelTank;

alter table Transaction rename to TransactionRecord;

alter table Payment rename to PaymentInfo;

alter table Tank_Refill rename to TankRefill;

alter table MaIntenance_Record rename to MaIntenanceRecord;

alter table Employee rename to EmployeeInfo;

alter table Customer rename to CustomerInfo;

alter table Fuel_pump rename to FuelPump;

alter table Supplier rename to SupplierInfo;

alter table Fuel_type rename to FuelType;
```

7. Drop columns:

```
alter table FuelType
drop column Supplier_ID;

alter table SupplierInfo
drop column Contact_Number;

alter table FuelPump
drop column Status;

alter table CustomerInfo
drop column Email;

alter table EmployeeInfo
drop column Contact_Number;
```

```
alter table FuelTank
drop column Current_level;

alter table TransactionRecord
drop column Fuel_Amount,
drop column M_Employee_ID;


alter table PaymentInfo
drop column Payment_method;

alter table TankRefill
drop column Date_time;

alter table MaIntenanceRecord
drop column Supplier_ID;
```

DML

```
-- 1. Select FuelType with a specific ID:
SELECT * FROM FuelType WHERE Fuel_type_ID = 1;


-- 2. Select SupplierInfo where the name is not 'Supplier_5':
SELECT * FROM SupplierInfo WHERE Name <> 'Supplier_5';


-- 3. Select FuelPump with a capacity greater than 800:
SELECT * FROM FuelPump WHERE Capacity > 800;


-- 4. Select CustomerInfo where the Contact_Number is less than 500000000:
SELECT * FROM CustomerInfo WHERE Contact_Number < 500000000;


-- 5. Select EmployeeInfo where the position is 'Manager':
SELECT * FROM EmployeeInfo WHERE Position = 'Manager';


-- 6. Select FuelTank where the capacity is between 3000 and 5000:
```

```
SELECT * FROM FuelTank WHERE Capacity BETWEEN 3000 AND 5000;

-- 7. Select TransactionRecord where the payment method is 'Card':
SELECT * FROM TransactionRecord WHERE Payment_method = 'Card';


-- 8. Select PaymentInfo where the amount is greater than or equal to 200:
SELECT * FROM PaymentInfo WHERE Amount >= 200;


-- 9. Select TankRefill where the Refill_Amount is less than or equal to 500:
SELECT * FROM TankRefill WHERE Refill_Amount <= 500;
-- 10. Select MaIntenanceRecord where the cost is not equal to 600:
SELECT * FROM MaIntenanceRecord WHERE Cost <> 600;

-- 11. Select FuelType where the Type_name starts with 'P':
SELECT * FROM FuelType WHERE Type_name LIKE 'P%';

-- 12. Select SupplierInfo where the address contains 'Address':
SELECT * FROM SupplierInfo WHERE Address LIKE '%Address%';

-- 13. Select FuelPump where the location ends with '3':
SELECT * FROM FuelPump WHERE Location LIKE '%3';

-- 14. Select CustomerInfo where the name includes 'Customer':
SELECT * FROM CustomerInfo WHERE Name LIKE '%Customer%';


-- 15. Select EmployeeInfo where the email ends with 'company.com':
SELECT * FROM EmployeeInfo WHERE Email LIKE '%@company.com';

-- 16. Select FuelTank where the location starts with 'Location_1':
SELECT * FROM FuelTank WHERE Location LIKE 'Location_1%';

-- 17. Select TransactionRecord where the date is in the year 2023:
SELECT * FROM TransactionRecord WHERE YEAR(Date_time) = 2023;


-- 18. Select PaymentInfo where the datetime is after '2023-06-01 00:00:00':
SELECT * FROM PaymentInfo WHERE Date_time > '2023-06-01 00:00:00';
```

```
-- 19. Select TankRefill where the Refill_Amount is between 400 and 800:
SELECT * FROM TankRefill WHERE Refill_Amount BETWEEN 400 AND 800;

-- 20. Select MaIntenanceRecord where the description includes 'Description':
SELECT * FROM MaIntenanceRecord WHERE Description LIKE '%Description%';
```

Phase-04

Joins

Queries:

```
-- Inner joins
--  List all transactions along with customer names and the employee who managed the transaction.
SELECT Transaction.Transaction_ID,Customer.Name AS Customer_Name,
Employee.Name AS Employee_Name, Transaction.Fuel_Amount
FROM Transaction
INNER JOIN Customer ON Transaction.Customer_ID = Customer.Customer_ID
INNER JOIN Employee ON Transaction.Employee_ID = Employee.Employee_ID;

-- Find all tank refills with the supplier name and the tank location.
SELECT Tank_Refill.Refill_ID,Supplier.Name AS Supplier_Name, Fuel_Tank.Location
AS Tank_Location, Tank_Refill.Refill_Amount
FROM Tank_Refill
INNER JOIN Supplier ON Tank_Refill.Supplier_ID = Supplier.Supplier_ID
INNER JOIN Fuel_Tank ON Tank_Refill.Tank_ID = Fuel_Tank.Tank_ID;

-- List all payments made along with the transaction details and customer names.
SELECT
Payment.Payment_ID,Payment.Amount,Transaction.Fuel_Amount,Customer.Name AS
Customer_Name
FROM Payment
INNER JOIN Transaction ON Payment.Transaction_ID = Transaction.Transaction_ID
INNER JOIN Customer ON Transaction.Customer_ID = Customer.Customer_ID;
```

```
-- List all maIntenance records with the employee names who performed the maIntenance and the pump locations.
SELECT MaIntenance_Record.MaIntenance_ID,Employee.Name AS Employee_Name, Fuel_pump.Location AS Pump_Location,MaIntenance_Record.Description
FROM MaIntenance_Record
INNER JOIN Employee ON MaIntenance_Record.M_Employee_ID = Employee.Employee_ID
INNER JOIN Fuel_pump ON MaIntenance_Record.Pump_ID = Fuel_pump.Pump_ID;

-- Find all fuel types along with the supplier names who provide them.
SELECT Fuel_type.Type_name,Supplier.Name AS Supplier_Name
FROM Fuel_type
INNER JOIN Supplier ON Fuel_type.Supplier_ID = Supplier.Supplier_ID;


-- left outer join

-- List all customers and their transactions, including those who have not made any transactions.
SELECT Customer.Name AS Customer_Name,Transaction.Transaction_ID,Transaction.Fuel_Amount
FROM Customer
LEFT JOIN Transaction ON Customer.Customer_ID = Transaction.Customer_ID;


--  Find all fuel pumps and the maIntenance records associated with them, including pumps that have no maIntenance records.
SELECT Fuel_pump.Location AS Pump_Location,MaIntenance_Record.MaIntenance_ID,MaIntenance_Record.Descriptio
n
FROM Fuel_pump
LEFT JOIN MaIntenance_Record ON Fuel_pump.Pump_ID = MaIntenance_Record.Pump_ID;


-- List all suppliers and the tank refills they have provided, including suppliers with no refills.
SELECT Supplier.Name AS Supplier_Name,Tank_Refill.Refill_ID,Tank_Refill.Refill_Amount
FROM Supplier
LEFT JOIN Tank_Refill ON Supplier.Supplier_ID = Tank_Refill.Supplier_ID;
```

-- Show all employees and the transactions they managed, including employees who have not managed any transactions.
SELECT Employee.Name AS
Employee_Name,Transaction.Transaction_ID,Transaction.Fuel_Amount
FROM Employee
LEFT JOIN Transaction ON Employee.Employee_ID = Transaction.Employee_ID;

-- List all fuel tanks and the refills they have received, including tanks that have not been refilled.
SELECT Fuel_Tank.Location AS
Tank_Location,Tank_Refill.Refill_ID,Tank_Refill.Refill_Amount
FROM Fuel_Tank
LEFT JOIN Tank_Refill ON Fuel_Tank.Tank_ID = Tank_Refill.Tank_ID;

-- right outer join

-- List all transactions and the associated customer names, including transactions with no customer information.
SELECT Transaction.Transaction_ID,Transaction.Fuel_Amount,Customer.Name AS
Customer_Name
FROM Transaction
RIGHT JOIN Customer ON Transaction.Customer_ID = Customer.Customer_ID;

-- Show all fuel pumps and their maIntenance records, including pumps that have no associated maIntenance records.
SELECT Fuel_pump.Location AS
Pump_Location,MaIntenance_Record.MaIntenance_ID,MaIntenance_Record.Descriptio
n
FROM MaIntenance_Record
RIGHT JOIN Fuel_pump ON MaIntenance_Record.Pump_ID = Fuel_pump.Pump_ID;

-- List all tank refills and the associated supplier names, including refills with no supplier information.
SELECT Tank_Refill.Refill_ID,Tank_Refill.Refill_Amount,Supplier.Name AS
Supplier_Name
FROM Tank_Refill
RIGHT JOIN Supplier ON Tank_Refill.Supplier_ID = Supplier.Supplier_ID;

-- Show all employees and the maIntenance records they handled, including employees with no maIntenance records.

```
SELECT
MaIntenance_Record.MaIntenance_ID,MaIntenance_Record.Description,Employee.Nam
e AS Employee_Name
FROM MaIntenance_Record
RIGHT JOIN Employee ON MaIntenance_Record.M_Employee_ID =
Employee.Employee_ID;

-- List all transactions and the pump locations they occurred at, including transactions
with no pump information.
SELECT Transaction.Transaction_ID, Transaction.Fuel_Amount,Fuel_pump.Location
AS Pump_Location
FROM Transaction
RIGHT JOIN Fuel_pump ON Transaction.Pump_ID = Fuel_pump.Pump_ID;



-- natural join

-- List all suppliers and the tank refills they have provided.
SELECT Supplier.Name AS
Supplier_Name,Tank_Refill.Refill_ID,Tank_Refill.Refill_Amount
FROM Supplier
NATURAL JOIN Tank_Refill;

--  Show all maIntenance records and the suppliers who provided the services.
SELECT
MaIntenance_Record.MaIntenance_ID,MaIntenance_Record.Description,Supplier.Name
AS Supplier_Name
FROM MaIntenance_Record
NATURAL JOIN Supplier;

-- List all employees and the transactions they managed.
SELECT Employee.Name AS Employee_Name,
Transaction.Transaction_ID,Transaction.Fuel_Amount
FROM Employee
NATURAL JOIN Transaction;


--  List all transactions and the customers who made them.
SELECT Transaction.Transaction_ID,Customer.Name AS Customer_Name,
Transaction.Fuel_Amount
FROM Transaction
NATURAL JOIN Customer;
```

```
--  List all fuel types and the suppliers who provide them.
SELECT Fuel_type.Type_name,Supplier.Name AS Supplier_Name
FROM Fuel_type
NATURAL JOIN Supplier;


-- self join

-- Find pairs of customers who share the same address.
SELECT C1.Customer_ID AS Customer1_ID,C1.Name AS
Customer1_Name,C2.Customer_ID AS Customer2_ID,C2.Name AS
Customer2_Name,C1.Address
FROM Customer C1
JOIN Customer C2 ON C1.Address = C2.Address AND C1.Customer_ID <
C2.Customer_ID;

--  Find pairs of employees who have the same contact number.
SELECT E1.Employee_ID AS Employee1_ID,E1.Name AS
Employee1_Name,E2.Employee_ID AS Employee2_ID,E2.Name AS
Employee2_Name,E1.Contact_Number
FROM Employee E1
JOIN Employee E2 ON E1.Contact_Number = E2.Contact_Number AND
E1.Employee_ID < E2.Employee_ID;

-- List pairs of transactions that occurred on the same date.
SELECT T1.Transaction_ID AS Transaction1_ID,T2.Transaction_ID AS
Transaction2_ID,T1.Date_time
FROM Transaction T1
JOIN Transaction T2 ON T1.Date_time = T2.Date_time AND T1.Transaction_ID <
T2.Transaction_ID;


-- Find pairs of fuel tanks that have the same capacity.
SELECT FT1.Tank_ID AS Tank1_ID,FT2.Tank_ID AS Tank2_ID,FT1.Capacity
FROM Fuel_Tank FT1
JOIN Fuel_Tank FT2 ON FT1.Capacity = FT2.Capacity AND FT1.Tank_ID <
FT2.Tank_ID;


-- List pairs of fuel pumps that are at the same location.
SELECT FP1.Pump_ID AS Pump1_ID,FP2.Pump_ID AS Pump2_ID,FP1.Location
```

```
FROM Fuel_pump FP1
JOIN Fuel_pump FP2 ON FP1.Location = FP2.Location AND FP1.Pump_ID <
FP2.Pump_ID;

-- Full join

-- List all customers and their transactions, including customers with no transactions and
transactions with no customer information.
SELECT Customer.Customer_ID,Customer.Name AS
Customer_Name,Transaction.Transaction_ID,Transaction.Fuel_Amount
FROM Customer
LEFT JOIN Transaction ON Customer.Customer_ID = Transaction.Customer_ID
UNION
SELECT Customer.Customer_ID,Customer.Name AS
Customer_Name,Transaction.Transaction_ID,Transaction.Fuel_Amount
FROM Transaction
LEFT JOIN Customer ON Customer.Customer_ID = Transaction.Customer_ID;

-- List all fuel tanks and their refill records, including tanks with no refill records and
refill records with no tank information.
SELECT
Fuel_Tank.Tank_ID,Fuel_Tank.Location,Tank_Refill.Refill_ID,Tank_Refill.Refill_Amo
unt
FROM Fuel_Tank
LEFT JOIN Tank_Refill ON Fuel_Tank.Tank_ID = Tank_Refill.Tank_ID
UNION
SELECT
Fuel_Tank.Tank_ID,Fuel_Tank.Location,Tank_Refill.Refill_ID,Tank_Refill.Refill_Amo
unt
FROM Tank_Refill
LEFT JOIN Fuel_Tank ON Fuel_Tank.Tank_ID = Tank_Refill.Tank_ID;

-- List all fuel pumps and their maIntenance records, including pumps with no
maIntenance records and maIntenance records with no pump information.
SELECT
Fuel_pump.Pump_ID,Fuel_pump.Location,MaIntenance_Record.MaIntenance_ID,MaInt
enance_Record.Description
FROM Fuel_pump
LEFT JOIN MaIntenance_Record ON Fuel_pump.Pump_ID =
MaIntenance_Record.Pump_ID
UNION
```

```
SELECT
Fuel_pump.Pump_ID,Fuel_pump.Location,MaIntenance_Record.MaIntenance_ID,MaInt
enance_Record.Description
FROM MaIntenance_Record
LEFT JOIN Fuel_pump ON Fuel_pump.Pump_ID = MaIntenance_Record.Pump_ID;

--  List all suppliers and their supplied fuel types, including suppliers with no fuel types
and fuel types with no supplier information.
SELECT Supplier.Supplier_ID,Supplier.Name AS
Supplier_Name,Fuel_type.Fuel_type_ID,Fuel_type.Type_name
FROM Supplier
LEFT JOIN Fuel_type ON Supplier.Supplier_ID = Fuel_type.Supplier_ID
UNION
SELECT Supplier.Supplier_ID,Supplier.Name AS
Supplier_Name,Fuel_type.Fuel_type_ID,Fuel_type.Type_name
FROM Fuel_type
LEFT JOIN Supplier ON Supplier.Supplier_ID = Fuel_type.Supplier_ID;

-- List all employees and their transactions, including employees with no transactions and
transactions with no employee information.
SELECT Employee.Employee_ID,Employee.Name AS
Employee_Name,Transaction.Transaction_ID,Transaction.Fuel_Amount
FROM Employee
LEFT JOIN Transaction ON Employee.Employee_ID = Transaction.Employee_ID
UNION
SELECT Employee.Employee_ID,Employee.Name AS
Employee_Name,Transaction.Transaction_ID,Transaction.Fuel_Amount
FROM Transaction
LEFT JOIN Employee ON Employee.Employee_ID = Transaction.Employee_ID;
```

Nested & Correlated Queries

Queries:

```
-- Nested Queries
-- 1. Select FuelType with a specific ID:
SELECT * FROM Fuel_type WHERE Fuel_type_ID = (SELECT Fuel_type_ID FROM
Fuel_type WHERE Fuel_type_ID = 1);

-- 2. Select SupplierInfo where the name is not 'Supplier_5':
SELECT * FROM Supplier WHERE Name IN (SELECT Name FROM Supplier
WHERE Name <> 'Supplier_5');
```

```
-- 3. Select FuelPump with a capacity greater than 800:
SELECT * FROM Fuel_pump WHERE Capacity > (SELECT MIN(Capacity) FROM
Fuel_pump WHERE Capacity > 800);

-- 4. Select CustomerInfo where the Contact_Number is less than 500000000:
SELECT * FROM Customer WHERE Contact_Number < (SELECT
MAX(Contact_Number) FROM Customer WHERE Contact_Number < 500000000);

-- 5. Select EmployeeInfo where the position is 'Manager':
SELECT * FROM Employee WHERE Position IN (SELECT Position FROM Employee
WHERE Position = 'Manager');

-- 6. Select FuelTank where the capacity is between 3000 and 5000:
SELECT * FROM Fuel_Tank WHERE Capacity BETWEEN (SELECT MIN(Capacity)
FROM Fuel_Tank WHERE Capacity BETWEEN 3000 AND 5000) AND (SELECT
MAX(Capacity) FROM Fuel_Tank WHERE Capacity BETWEEN 3000 AND 5000);

-- 7. Select TransactionRecord where the transaction has a corresponding payment in
PaymentInfo:
SELECT * FROM Transaction t1 WHERE EXISTS (SELECT * FROM Payment p1
WHERE p1.Transaction_ID = t1.Transaction_ID);

-- 8. Select PaymentInfo where the amount is greater than or equal to 200:
SELECT * FROM Payment WHERE Amount >= (SELECT MIN(Amount) FROM
Payment WHERE Amount >= 200);

-- 9. Select TankRefill where the Refill_Amount is less than or equal to 500:
SELECT * FROM Tank_Refill WHERE Refill_Amount <= (SELECT
MAX(Refill_Amount) FROM Tank_Refill WHERE Refill_Amount <= 500);

-- 10. Select MaIntenanceRecord where the cost is not equal to 600:
SELECT * FROM MaIntenance_Record WHERE Cost <> (SELECT Cost FROM
MaIntenance_Record WHERE Cost = 600 LIMIT 1);

-- Correlated Subqueries
-- 1. Select FuelType with a specific ID:
SELECT * FROM Fuel_type t1 WHERE EXISTS (SELECT * FROM Fuel_type t2
WHERE t1.Fuel_type_ID = t2.Fuel_type_ID AND t2.Fuel_type_ID = 1);

-- 2. Select SupplierInfo where the name is not 'Supplier_5':
SELECT * FROM Supplier s1 WHERE NOT EXISTS (SELECT * FROM Supplier s2
WHERE s2.Name = 'Supplier_5' AND s1.Supplier_ID = s2.Supplier_ID);
```

```
-- 3. Select FuelPump with a capacity greater than 800:
SELECT * FROM Fuel_pump f1 WHERE EXISTS (SELECT * FROM Fuel_pump f2
WHERE f1.Pump_ID = f2.Pump_ID AND f2.Capacity > 800);

-- 4. Select CustomerInfo where the Contact_Number is less than 500000000:
SELECT * FROM Customer c1 WHERE EXISTS (SELECT * FROM Customer c2
WHERE c1.Customer_ID = c2.Customer_ID AND c2.Contact_Number < 500000000);

-- 5. Select EmployeeInfo where the position is 'Manager':
SELECT * FROM Employee e1 WHERE EXISTS (SELECT * FROM Employee e2
WHERE e1.Employee_ID = e2.Employee_ID AND e2.Position = 'Manager');

-- 6. Select FuelTank where the capacity is between 3000 and 5000:
SELECT * FROM Fuel_Tank ft1 WHERE EXISTS (SELECT * FROM Fuel_Tank ft2
WHERE ft1.Tank_ID = ft2.Tank_ID AND ft2.Capacity BETWEEN 3000 AND 5000);

-- 7. Select TransactionRecord where the transaction has a corresponding payment in
PaymentInfo:
SELECT * FROM Transaction tr1 WHERE EXISTS (SELECT * FROM Payment p1
WHERE p1.Transaction_ID = tr1.Transaction_ID);

-- 8. Select PaymentInfo where the amount is greater than or equal to 200:
SELECT * FROM Payment p1 WHERE EXISTS (SELECT * FROM Payment p2
WHERE p1.Payment_ID = p2.Payment_ID AND p2.Amount >= 200);

-- 9. Select TankRefill where the Refill_Amount is less than or equal to 500:
SELECT * FROM Tank_Refill tr1 WHERE EXISTS (SELECT * FROM Tank_Refill tr2
WHERE tr1.Refill_ID = tr2.Refill_ID AND tr2.Refill_Amount <= 500);

-- 10. Select MaIntenanceRecord where the cost is not equal to 600:
SELECT * FROM MaIntenance_Record mr1 WHERE EXISTS (SELECT * FROM
MaIntenance_Record mr2 WHERE mr1.MaIntenance_ID = mr2.MaIntenance_ID AND
mr2.Cost <> 600);
```

The End