Database Queries for Assignment-A3

Creating database

```
CREATE DATABASE Database_A3;
USE Database_A3;
```

Creating tables and specify primary keys:

```
CREATE TABLE Account(acc_no INT, branch_name VARCHAR(50), balance INT, PRIMARY KEY (acc_no));
CREATE TABLE Branch(branch_name VARCHAR(50), branch_city VARCHAR(50), assets INT, PRIMARY KEY (branch_name));
CREATE TABLE Customer (cust_name VARCHAR(50), cust_street VARCHAR(50), cust_city VARCHAR(50), PRIMARY KEY (cust_name));
CREATE TABLE Depositor (cust_name VARCHAR(50), acc_no INT);
CREATE TABLE Loan (loan_no INT, branch_name VARCHAR(50), amount INT, PRIMARY KEY (loan_no));
CREATE TABLE Borrower (cust_name VARCHAR(50), loan_no INT);
```

Altering tables to specify foreign keys

```
ALTER TABLE Account ADD FOREIGN KEY (branch_name) REFERENCES Branch(branch_name);
ALTER TABLE Depositor ADD FOREIGN KEY (cust_name) REFERENCES Customer (cust_name);
ALTER TABLE Depositor ADD FOREIGN KEY (acc_no) REFERENCES Account (acc_no);
ALTER TABLE Loan ADD FOREIGN KEY (branch_name) REFERENCES Branch (branch_name);
ALTER TABLE Borrower ADD FOREIGN KEY (cust_name) REFERENCES Customer (cust_name);
ALTER TABLE Borrower ADD FOREIGN KEY (loan_no) REFERENCES Loan (loan_no);
```

Adding data

```
INSERT INTO Branch (branch_name, branch_city, assets) VALUES
("Pune_Station", "Pune", 5000),
("Hadapsar", "Pune", 20000),
("Dhole_Patil", "Mumbai", 7500),
("Nagarwala", "Nandurbar", 3200);

INSERT INTO Customer (cust_name, cust_street, cust_city) VALUES
("Kalas", "Airport Road", "Pune"),
("Mehul", "Shahdha", "Nandurbar"),
("Tanmay", "Porwal Road", "Pune"),
("Kshitij", "Hadapasar", "Pune"),
("Aditya", "Mira RD", "Mumbai"),
("Himanshu", "Smart City", "Nandurbar");

INSERT INTO Account (acc_no, branch_name, balance) VALUES
(2501, "Dhole_Patil", 5000),
(2511, "Pune_Station", 1500),
(2521, "Hadapsar", 2000),
(2512, "Nagarwala", 5000),
(2531, "Pune_Station", 1000);

INSERT INTO Loan (loan_no, branch_name, amount) VALUES
(155, "Dhole_Patil", 500),
(156, "Pune_Station", 250),
(157, "Hadapsar", 600),
(158, "Nagarwala", 1400),
(159, "Pune_Station", 25000);

INSERT INTO Borrower VALUES
("Kalas", 156),
("Mehul", 158),
("Tanmay", 155),
("Kshitij", 157),
("Aditya", 159),
("Himanshu", 158);

INSERT INTO Depositor VALUES
("Kalas", 2511),
("Mehul", 2512),
("Tanmay", 2501),
("Kshitij", 2521),
("Aditya", 2531),
("Himanshu", 2512);
```

Queries

Part 1 (Borrower):

1. Create a View1 to display List all customers in alphabetical order who have loan from Pune_Station branch

```
CREATE VIEW View1 AS
SELECT cust_name
FROM Borrower
INNER JOIN Loan ON Borrower.loan_no = Loan.loan_no
WHERE branch_name = "Pune_Station"
ORDER BY cust_name;
SELECT * FROM View1;
```

2. Create View2 on branch table by selecting any two columns and perform insert update delete operations.

```
CREATE VIEW View2 AS
SELECT branch_name, branch_city
FROM Branch;
SELECT * FROM View2;

-- Insert operation
INSERT INTO View2 (branch_name, branch_city) VALUES ('Yerwada', 'Pune');
SELECT * FROM View2;

-- Update operation
UPDATE View2 SET branch_name = 'Peachtree' WHERE branch_name = 'Yerwada';
SELECT * FROM View2;

-- Delete operation
DELETE FROM View2 WHERE branch_name = 'Peachtree';
SELECT * FROM View2;
```

3. Create View3 on borrower and depositor table by selecting any one column from each table perform insert update delete operations.

```
CREATE VIEW View3 AS
SELECT Borrower.cust_name, Depositor.acc_no
FROM Borrower JOIN Depositor ON Borrower.cust_name = Depositor.cust_name;
SELECT * FROM View3;

-- Insert operation
INSERT INTO Customer (cust_name, cust_street, cust_city) VALUES ("Macho", "Pedgaon", "Ahemadnagar");
INSERT INTO Account (acc_no, branch_name, balance) VALUES (2502, "Hadapsar", 3000);
INSERT INTO Loan (loan_no, branch_name, amount) VALUES (160, "Hadapsar", 500);
INSERT INTO Borrower (cust_name, loan_no) VALUES ("Macho", 160);
INSERT INTO Depositor(cust_name, Acc_no) VALUES("Macho", 2502);
SELECT * FROM View3;

-- Update operation
INSERT INTO Account (acc_no, branch_name, balance) VALUES (2566, 'Hadapsar', 3000);
UPDATE Depositor SET acc_no = 2566 WHERE cust_name = 'Macho';
SELECT * FROM View3;

-- Delete operation
DELETE FROM Borrower WHERE cust_name = 'Macho';
DELETE FROM Depositor WHERE cust_name = 'Macho';
SELECT * FROM View3;
```

4. Create Union of left and right joint for all customers who have an account or loan or both at bank

```
SELECT DISTINCT Customer.cust_name
FROM Customer
LEFT JOIN Depositor ON Customer.cust_name = Depositor.cust_name
LEFT JOIN Borrower ON Customer.cust_name = Borrower.cust_name
WHERE Depositor.acc_no IS NOT NULL OR Borrower.loan_no IS NOT NULL;
```

5. Display content of View1, View2, View3

```
SELECT * FROM View1;
SELECT * FROM View2;
SELECT * FROM View3;
```

6. Create Simple and Unique index

```
-- Simple Index
CREATE INDEX cust_ind ON Customer (cust_city);

-- Unique Index
CREATE UNIQUE INDEX branch_ind ON Branch (branch_name);
```

7. Display index Information

```
SHOW INDEX FROM Customer;
SHOW INDEX FROM Branch;
```

8. Truncate table Customer

```
DROP TABLE Depositor;
DROP TABLE Borrower;
TRUNCATE TABLE Customer;
```

Part 2:

Create tables

```
CREATE TABLE Companies(comp_id varchar(50), name varchar(50), cost int, year int);
CREATE TABLE Orders(comp_id varchar(50), domain varchar(50), quantity int);
```

Insert values

```
INSERT INTO Companies (comp_id, name, cost, year) VALUES
("C001", "ONGC", 2000, 2010),
("C002", "HPCL", 2500, 2012),
("C005", "IOCL", 1000, 2014),
("C006", "BHEL", 3000, 2015);

INSERT INTO Orders (comp_id, domain, quantity) VALUES
("C001", "Oil", 109),
("C002", "Gas", 121),
("C005", "Telecom", 115);
```

Queries

1. Find names, costs, domains and quantities for companies using inner join.

```
SELECT name, cost, domain, quantity FROM Companies INNER JOIN Orders on Companies.comp_id = Orders.comp_id;
```

2. Find names, costs, domains and quantities for companies using left outer join.

```
SELECT name, cost, domain, quantity FROM Companies lEFT OUTER JOIN Orders ON Companies.comp_id = Orders.comp_id;
```

3. Find names, costs, domains and quantities for companies using right outer join.

```
SELECT name, cost, domain, quantity FROM Companies RIGHT OUTER JOIN Orders ON Companies.comp_id = Orders.comp_id;
```

4. Find names, costs, domains and quantities for companies using Union operator.

```
SELECT name, cost FROM Companies UNION SELECT domain, quantity FROM Orders;
```

5. Create View View1 by selecting both tables to show company name and quantities.

```
CREATE VIEW view1 AS SELECT name, quantity FROM Companies JOIN Orders ON Companies.comp_id = Orders.comp_id;
SELECT * FROM view1;
```

6. Create View2 on branch table by selecting any two columns and perform insert update delete operations.

```
CREATE VIEW view2 AS SELECT name, cost FROM Companies;
SELECT * FROM view2;

-- Insert operation
INSERT INTO view2 (name, cost) VALUES ("BCCC", 3100);
SELECT * FROM view2;

-- Update operation
UPDATE view2 SET cost = 3500 WHERE name = "BCCC";
SELECT * FROM view2;

-- Delete operation
DELETE FROM view2 WHERE name = "BCCC";
SELECT * FROM view2;
```

7. Display content of View1, View2.

```
SELECT * from view1;
SELECT * FROM view2;
```
