

# Database Queries for Assignment-A2

This file contains all the queries required to build the [database \(https://git.kska.io/sppu-te-comp-content/DatabaseManagementSystems/src/branch/main/Practical/Assignment-A2/Database\\_A2.sql\)](https://git.kska.io/sppu-te-comp-content/DatabaseManagementSystems/src/branch/main/Practical/Assignment-A2/Database_A2.sql).

## Creating database

```
CREATE DATABASE Database_A2;  
USE Database_A2;
```

## Creating tables and specify primary keys:

```
CREATE TABLE Account (  
    acc_no INT,  
    branch_name VARCHAR(255),  
    balance INT,  
    PRIMARY KEY (acc_no)  
);  
  
CREATE TABLE Branch (  
    branch_name VARCHAR(255),  
    branch_city VARCHAR(255),  
    assets INT,  
    PRIMARY KEY (branch_name)  
);  
  
CREATE TABLE Customer (  
    cust_name VARCHAR(255),  
    cust_street VARCHAR(255),  
    cust_city VARCHAR(255),  
    PRIMARY KEY (cust_name)  
);  
  
CREATE TABLE Depositor (  
    cust_name VARCHAR(255),  
    acc_no INT  
);  
  
CREATE TABLE Loan (  
    loan_no INT,  
    branch_name VARCHAR(255),  
    amount INT,  
    PRIMARY KEY (loan_no)  
);  
  
CREATE TABLE Borrower (  
    cust_name VARCHAR(255),  
    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
('Akurdi', 'Pune', 5000000),
('Nigdi', 'Pune', 3000000),
('Kharadi', 'Pune', 4000000),
('Hadapsar', 'Pune', 3500000),
('Viman Nagar', 'Pune', 4500000);

INSERT INTO Account (acc_no, branch_name, balance) VALUES
(101, 'Akurdi', 15000),
(102, 'Nigdi', 8000),
(103, 'Kharadi', 20000),
(104, 'Hadapsar', 12000),
(105, 'Viman Nagar', 5000);

INSERT INTO Customer (cust_name, cust_street, cust_city) VALUES
('Kalas', 'Street 1', 'Pune'),
('Mehul', 'Street 2', 'Pune'),
('Tanmay', 'Street 3', 'Pune'),
('Kshitij', 'Street 4', 'Pune'),
('Aditya', 'Street 5', 'Pune');

INSERT INTO Depositor (cust_name, acc_no) VALUES
('Kalas', 101),
('Mehul', 102),
('Tanmay', 103),
('Kshitij', 101),
('Aditya', 104);

INSERT INTO Loan (loan_no, branch_name, amount) VALUES
(201, 'Akurdi', 15000),
(202, 'Nigdi', 13000),
(203, 'Kharadi', 25000),
(204, 'Hadapsar', 18000),
(205, 'Akurdi', 1400);

INSERT INTO Borrower (cust_name, loan_no) VALUES
('Mehul', 202),
('Tanmay', 203),
('Aditya', 205);
```

## Running queries

1. Find the names of all branches in loan relation.

```
SELECT DISTINCT branch_name FROM Loan;
```

2. Find all loan numbers for loans made at Akurdi Branch with loan amount > 12000.

```
SELECT loan_no FROM Loan WHERE branch_name = 'Akurdi' AND amount > 12000;
```

3. Find all customers who have a loan from bank. Find their names, loan\_no and loan amount.

```
SELECT * from Borrower;  
SELECT Borrower.cust_name, Borrower.loan_no, Loan.amount FROM Borrower INNER JOIN Loan ON Borrower.loan_no = Loan.loan_no;
```

4. List all customers in alphabetical order who have loan from Akurdi branch.

```
SELECT cust_name FROM Borrower INNER JOIN Loan ON Borrower.loan_no = Loan.loan_no WHERE branch_name = 'Akurdi' ORDER BY cust_name;
```

5. Find all customers who have an account or loan or both at bank.

```
SELECT cust_name FROM Depositor UNION SELECT cust_name FROM Borrower;
```

6. Find all customers who have both account and loan at bank.

```
SELECT cust_name FROM Depositor INTERSECT SELECT cust_name FROM Borrower;
```

7. Find all customers who have account but no loan at the bank.

```
SELECT cust_name FROM Depositor WHERE cust_name NOT IN (SELECT cust_name FROM Borrower);  
  
-- OR YOU CAN RUN THE BELOW COMMAND IF YOU ARE FEELING FANCY  
-- SELECT Customer.cust_name FROM Customer INNER JOIN Depositor ON Customer.cust_name = Depositor.cust_name LEFT JOIN Borrower ON Cu
```

8. Find the average account balance at each branch

```
SELECT branch_name, AVG(balance) FROM Account GROUP BY branch_name;
```

9. Find no. of depositors at each branch.

```
SELECT branch_name, COUNT(*) FROM Account INNER JOIN Depositor ON Account.acc_no = Depositor.acc_no GROUP BY branch_name;
```

10. Find name of Customer and city where customer name starts with Letter K.

```
SELECT cust_name, cust_city FROM Customer WHERE cust_name LIKE 'K%';
```

11. Display distinct cities of branch.

```
SELECT DISTINCT branch_city FROM Branch;
```

12. Find the branches where average account balance > 1200

```
SELECT branch_name FROM Account GROUP BY branch_name HAVING AVG(balance) > 12000;
```

13. Find number of tuples in customer relation.

```
SELECT COUNT(*) FROM Customer;
```

14. Calculate total loan amount given by bank.

```
SELECT SUM(amount) FROM Loan;
```

15. Delete all loans with loan amount between 1300 and 1500.

```
DELETE FROM Borrower WHERE loan_no IN (SELECT loan_no FROM Loan WHERE amount BETWEEN 1300 AND 1500);  
DELETE FROM Loan WHERE amount BETWEEN 1300 AND 1500;
```

16. Delete all tuples at every branch located in Nigdi.

```
DELETE FROM Borrower WHERE loan_no IN (SELECT loan_no FROM Loan WHERE branch_name = 'Nigdi');  
DELETE FROM Loan WHERE branch_name = 'Nigdi';  
DELETE FROM Depositor WHERE acc_no IN (SELECT acc_no FROM Account WHERE branch_name = 'Nigdi');  
DELETE FROM Account WHERE branch_name = 'Nigdi';  
DELETE FROM Branch WHERE branch_name = 'Nigdi';
```