Data Base Management Systems

LAB 4

Name :Anuvind M P

Roll no : AM.EN.U4AIE22010

Question 1

DESCRIPTION:

The following relations keep track of a banking enterprise. Create the tables with proper primary key and references.

1. BRANCH (branch-name: varchar(10), branch-city:varchar(10), assets:numeric(8,2))

2. ACCOUNT (accno:int, branch-name:varchar(10), balance:numeric(8,2))

3. CUSTOMER (customer-no: varchar(5), customer-name:varcha1), customer-street:varchar(15), customer-city:varchar(10))

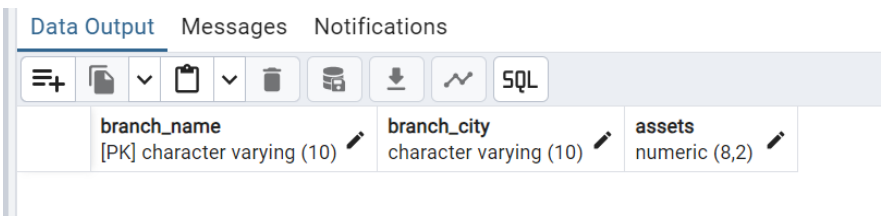
4. LOAN (loan-number:int, branch-name:varchar(10), amount:numeric(8,2))

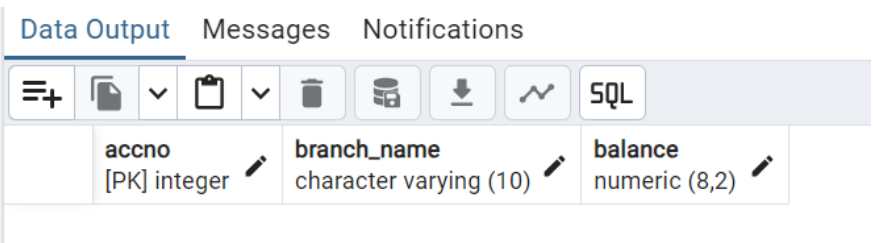
5. DEPOSITOR (customer-no:varchar(5), accno:int)

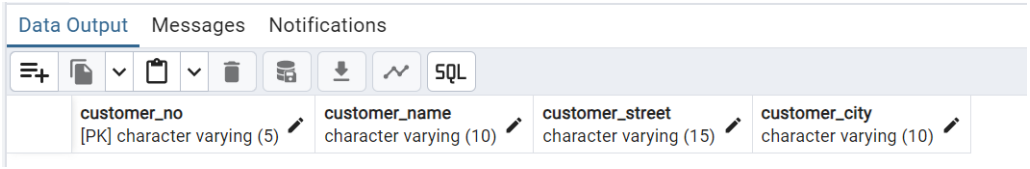
6. BORROWER (customer-no:varchar(5), loan-number:int)

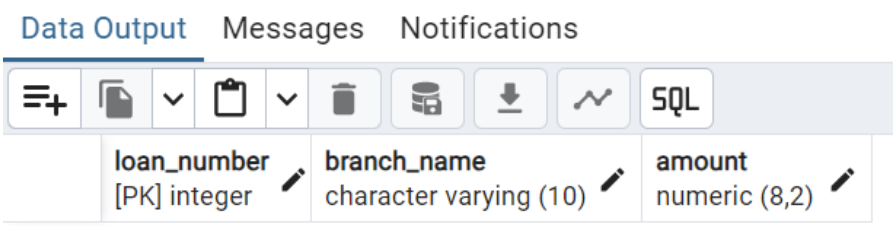
```
CREATE TABLE BRANCH (
    branch_name VARCHAR(10) PRIMARY KEY,
    branch_city VARCHAR(10),
    assets NUMERIC(8,2)
);

CREATE TABLE ACCOUNT (
    accno INT PRIMARY KEY,
    branch_name VARCHAR(10),
    balance NUMERIC(8,2),
    FOREIGN KEY (branch_name) REFERENCES BRANCH(branch_name)
);

CREATE TABLE CUSTOMER (
    customer_no VARCHAR(5) PRIMARY KEY,
    customer_name VARCHAR(10),
```

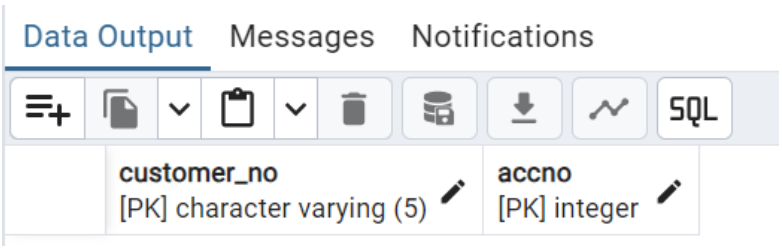
```
    customer_street VARCHAR(15),
    customer_city VARCHAR(10)
);

CREATE TABLE LOAN (
    loan_number INT PRIMARY KEY,
    branch_name VARCHAR(10),
    amount NUMERIC(8,2),
    FOREIGN KEY (branch_name) REFERENCES BRANCH(branch_name)
);

CREATE TABLE DEPOSITOR (
    customer_no VARCHAR(5),
    accno INT,
    PRIMARY KEY (customer_no, accno),
    FOREIGN KEY (customer_no) REFERENCES CUSTOMER(customer_no),
    FOREIGN KEY (accno) REFERENCES ACCOUNT(accno)
);

CREATE TABLE BORROWER (
    customer_no VARCHAR(5),
    loan_number INT,
    PRIMARY KEY (customer_no, loan_number),
    FOREIGN KEY (customer_no) REFERENCES CUSTOMER(customer_no),
    FOREIGN KEY (loan_number) REFERENCES LOAN(loan_number)
);
```

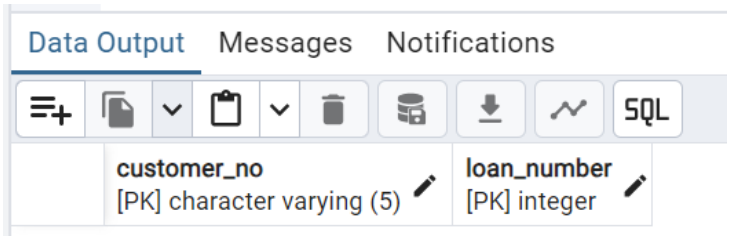












Queries:

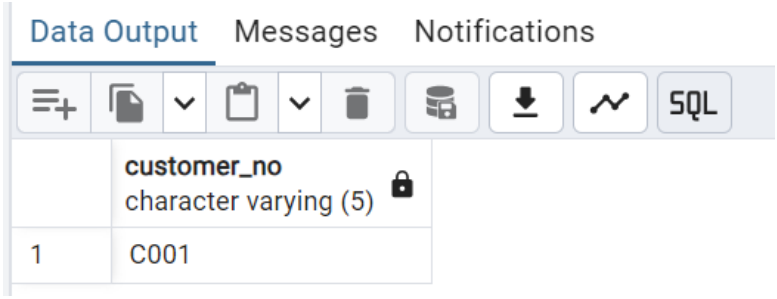
Enter at least three tuples for each relation and write each of the following queries in SQL.

```
INSERT INTO BRANCH (branch_name, branch_city, assets)
VALUES
('Main', 'NewYork', 100000.00),
('West', 'LosAngeles', 75000.50),
('East', 'Boston', 50000.75),
('Kollam', 'Kerala', 80000.00);


INSERT INTO ACCOUNT (accno, branch_name, balance)
VALUES
(1001, 'Main', 2000.00),
(1002, 'Main', 3500.25),
(1003, 'West', 1500.75),
(1004, 'East', 4500.50),
(1005, 'Kollam', 12000.00),
(1006, 'Kollam', 18000.00);
```

```
INSERT INTO CUSTOMER (customer_no, customer_name, customer_street, customer_city)
VALUES
('C001', 'John Doe', '1st Ave', 'NewYork'),
('C002', 'Jane Smith', 'Sunset Blvd', 'LosAngeles'),
('C003', 'Sam Wilson', 'Park St', 'Boston'),
('C004', 'Alan Brown', 'MG Road', 'Kerala');

INSERT INTO LOAN (loan_number, branch_name, amount)
VALUES
(2001, 'Main', 5000.00),
(2002, 'West', 8000.50),
(2003, 'East', 3000.25),
(2004, 'Kollam', 15000.75);

INSERT INTO DEPOSITOR (customer_no, accno)
VALUES
('C001', 1001),
('C001', 1002),
('C002', 1003),
('C003', 1004),
('C004', 1005),
('C004', 1006);

INSERT INTO BORROWER (customer_no, loan_number)
VALUES
('C001', 2001),
('C002', 2002),
('C003', 2003),
('C004', 2004);
```

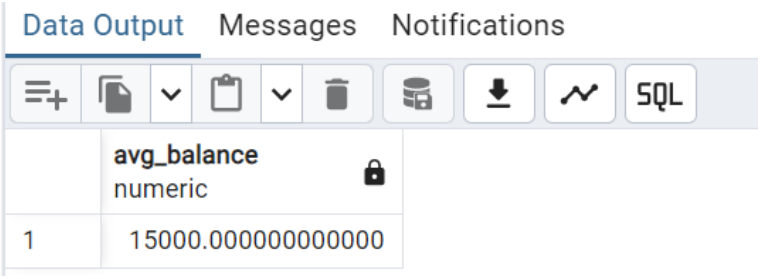
1. Find all the customers who have at least two accounts at the '*Main*' branch.

```
select customer_no from account where branch_name = 'Main' group by customer_no
having count(accno)>=2
```



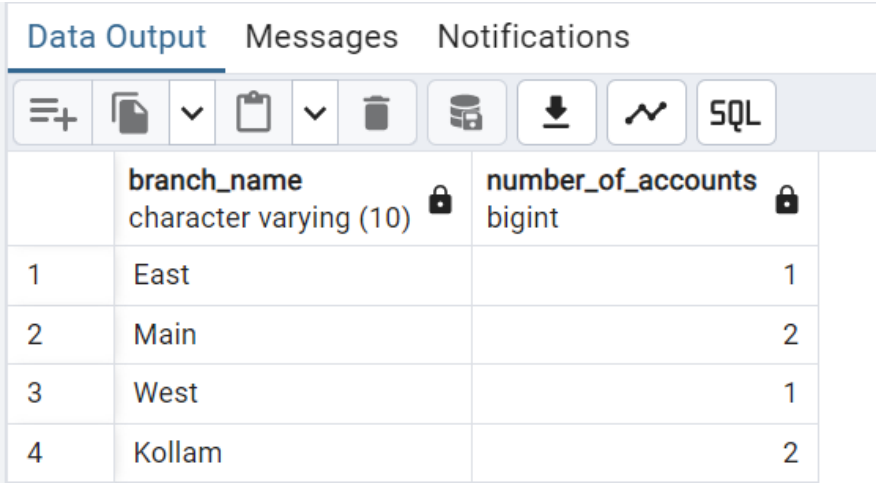
2. Find the average account balance at the *'Kollam'* branch.

```
SELECT AVG(balance) AS avg_balance FROM ACCOUNT WHERE branch_name = 'Kollam';
```



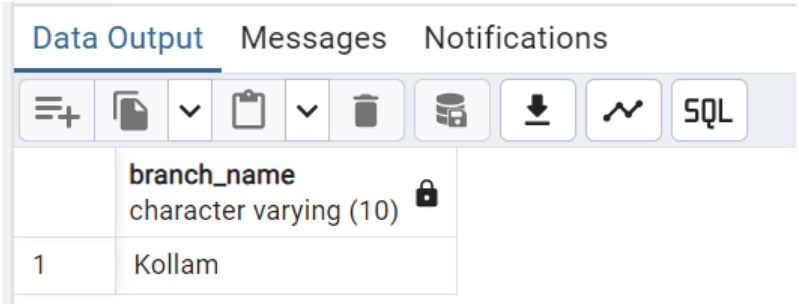
3. Find the number of depositors for each branch.

```
SELECT branch_name, COUNT(accno) AS number_of_accounts FROM ACCOUNT GROUP BY
branch_name;
```



4. Find the names of all branches where the average account balance is more than RS. 1,2000.

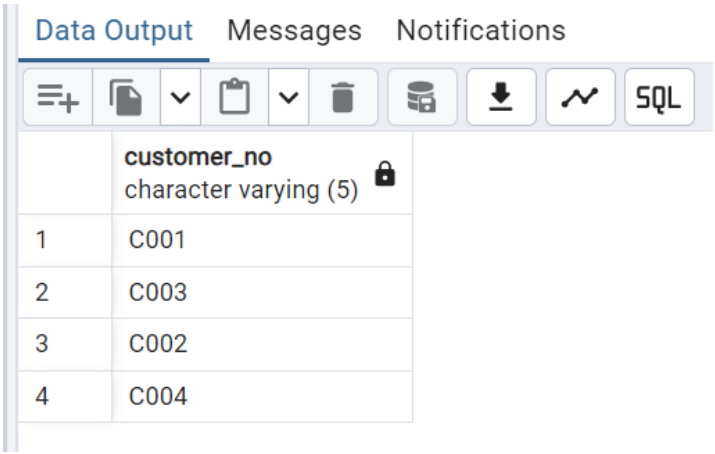
```
SELECT branch_name FROM ACCOUNT GROUP BY branch_name HAVING AVG(balance) > 12000;
```



	branch_name character varying (10)
1	Kollam

5. Find all customers who have a loan, an account, or both.

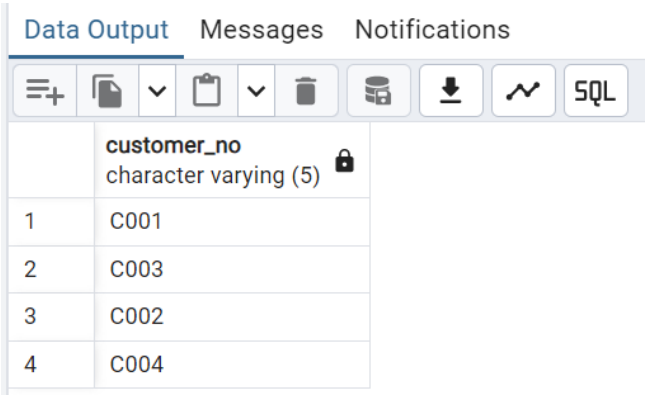
```
SELECT DISTINCT customer_no FROM (SELECT customer_no FROM DEPOSITOR UNION SELECT customer_no FROM BORROWER) AS customers;
```



	customer_no character varying (5)
1	C001
2	C003
3	C002
4	C004

6. Find all customers who have both a loan and an account.

```
SELECT customer_no FROM DEPOSITOR INTERSECT SELECT customer_no FROM BORROWER;
```



	customer_no character varying (5)
1	C001
2	C003
3	C002
4	C004

7. Find the number of branches that currently have loans.

```
SELECT COUNT(DISTINCT branch_name) AS num_branches_with_loans FROM LOAN;
```

Data Output | Messages | Notifications

	num_branches_with_loans bigint
1	4

8. Find the average loan amount for each branch.

```
SELECT branch_name, AVG(amount) AS avg_loan_amount FROM LOAN GROUP BY
branch_name;
```

Data Output | Messages | Notifications

	branch_name character varying (10)	avg_loan_amount numeric
1	East	3000.2500000000000000
2	Main	5000.0000000000000000
3	West	8000.5000000000000000
4	Kollam	15000.7500000000000000

9. Find all customers with more than one loan.

```
SELECT customer_no FROM BORROWER GROUP BY customer_no HAVING COUNT(loan_number) >
1;
```

Data Output | Messages | Notifications

	customer_no character varying (5)

10. Find the total of all loan amounts

```
SELECT SUM(amount) AS total_loan_amount FROM LOAN;
```

Data Output | Messages | Notifications

	total_loan_amount numeric
1	31001.50