AHMED M. S. ALBREEM

Std.id : 210041258

COURSE.NO : CSE4308

DATE OF SOLUTION : 12-09-2023

ASSIGNMENT 4 OF DBMS LAB

`FIRST TASK : "showing specific city of the customer which have end as d and f some where in the string (city) "

Code : "

```
SELECT customer_city
FROM customer
WHERE customer_city LIKE '%f%' AND customer_city LIKE '%d';
```

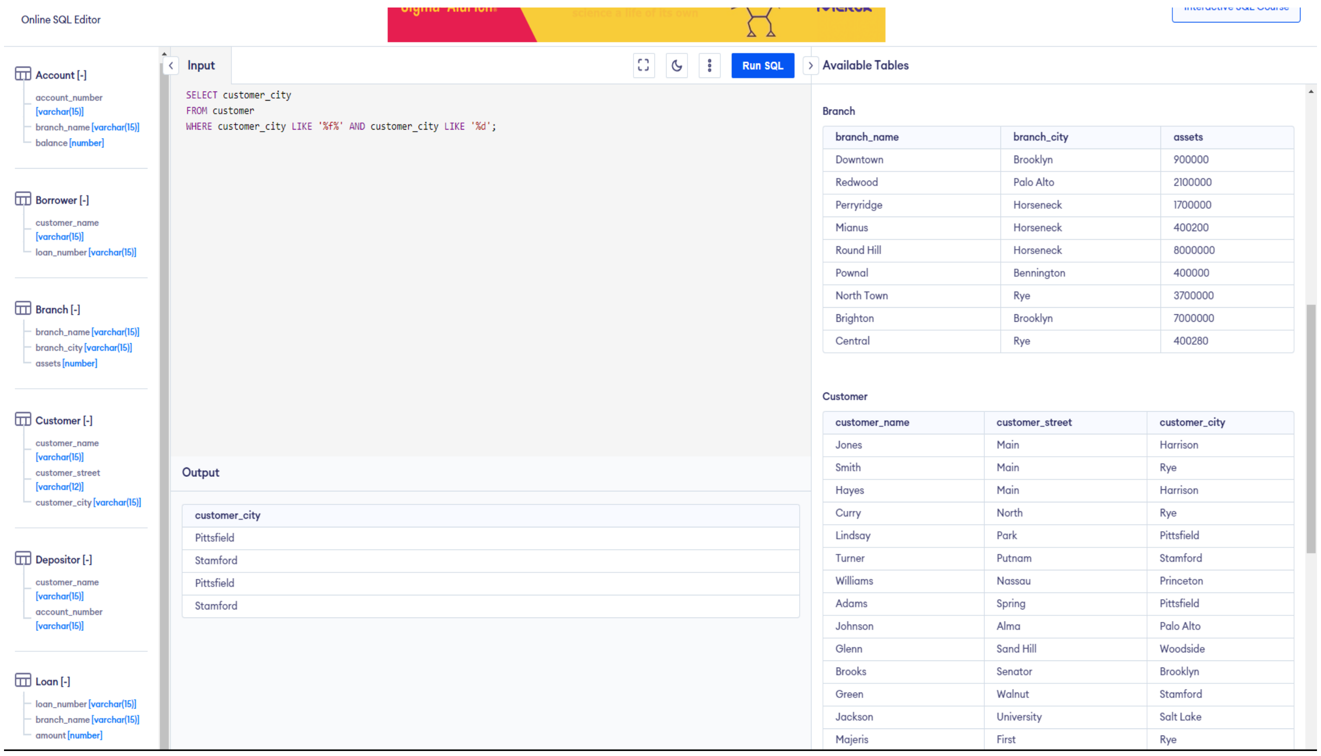
"

Explaining : "

Showing city of the customer from customer if the city have any structure like this in end as d and in the anywhere else it is will be as 'f'

"

The screenshot :


Online SQL Editor

Account [-]
- account_number [varchar(15)]
- branch_name [varchar(15)]
- balance [number]

Borrower [-]
- customer_name [varchar(15)]
- loan_number [varchar(15)]

Branch [-]
- branch_name [varchar(15)]
- branch_city [varchar(15)]
- assets [number]

Customer [-]
- customer_name [varchar(15)]
- customer_street [varchar(12)]
- customer_city [varchar(15)]

Depositor [-]
- customer_name [varchar(15)]
- account_number [varchar(15)]

Loan [-]
- loan_number [varchar(15)]
- branch_name [varchar(15)]
- amount [number]

Input

```
SELECT customer_city
FROM customer
WHERE customer_city LIKE '%f%' AND customer_city LIKE '%d';
```

Output

customer_city
Pittsfield
Stamford
Pittsfield
Stamford

Available Tables

Branch

branch_name	branch_city	assets
Downtown	Brooklyn	900000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400200
Round Hill	Horseneck	8000000
Pownal	Bennington	400000
North Town	Rye	3700000
Brighton	Brooklyn	7000000
Central	Rye	400280

Customer

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn
Green	Walnut	Stamford
Jackson	University	Salt Lake
Majeris	First	Rye

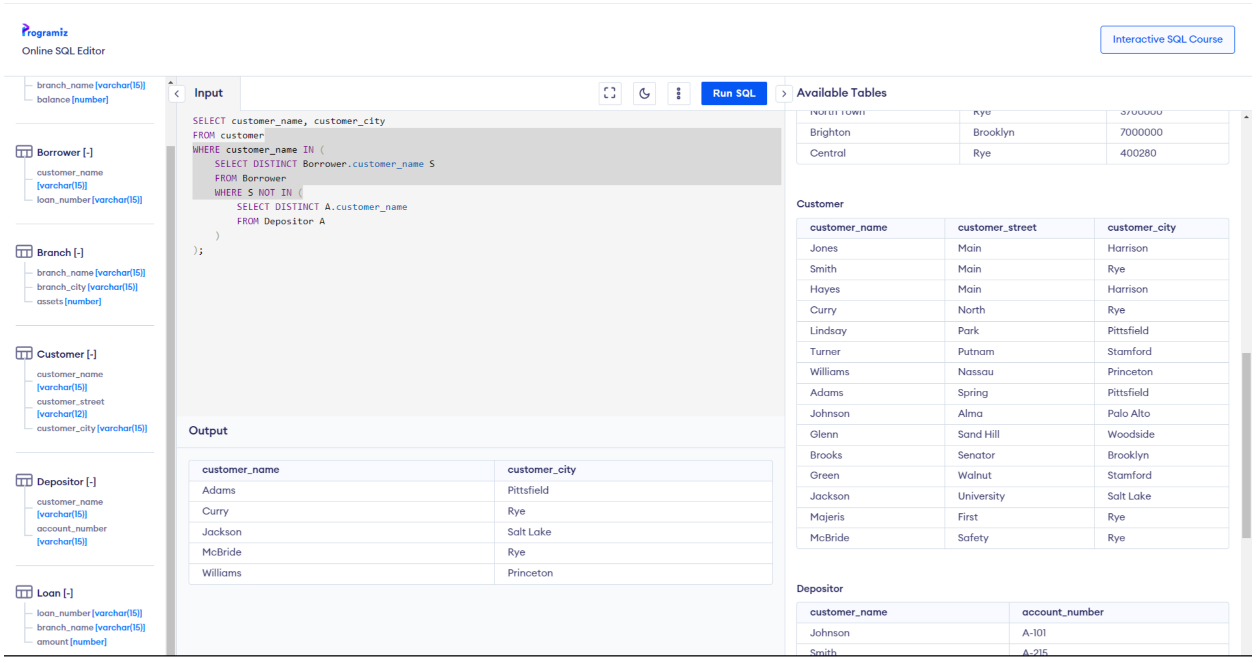
Second command : “Find all customer names and their cities who have a loan but not an account. ”

Explaining : “ showing the city and name from customer table where his name in the loan table (brower) where also in the account table (depositor)”

Code : ”

```
SELECT customer_name, customer_city
FROM customer
WHERE customer_name IN (
    SELECT DISTINCT L.customer_name
    FROM loan L
    WHERE L.customer_name NOT IN (
        SELECT DISTINCT A.customer_name
        FROM account A
    )
);
```
”

Screenshot :



Programiz
Online SQL Editor

branch_name [varchar(15)]
balance [number]

Borrower [-]
customer_name [varchar(15)]
loan_number [varchar(15)]

Branch [-]
branch_name [varchar(15)]
branch_city [varchar(15)]
assets [number]

Customer [-]
customer_name [varchar(15)]
customer_street [varchar(12)]
customer_city [varchar(15)]

Depositor [-]
customer_name [varchar(15)]
account_number [varchar(15)]

Loan [-]
loan_number [varchar(15)]
branch_name [varchar(15)]
amount [number]

```
SELECT customer_name, customer_city
FROM customer
WHERE customer_name IN (
    SELECT DISTINCT Borrower.customer_name S
    FROM Borrower
    WHERE S NOT IN (
        SELECT DISTINCT A.customer_name
        FROM Depositor A
    )
);
```

Output

customer_name	customer_city
Adams	Pittsfield
Curry	Rye
Jackson	Salt Lake
McBride	Rye
Williams	Princeton

Brighton	Brooklyn	7000000
Central	Rye	400280

Customer

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn
Green	Walnut	Stamford
Jackson	University	Salt Lake
Majeris	First	Rye
McBride	Safety	Rye

Depositor

customer_name	account_number
Johnson	A-101

The third command : “ Find all customer-related information about who has an account or a loan. ”

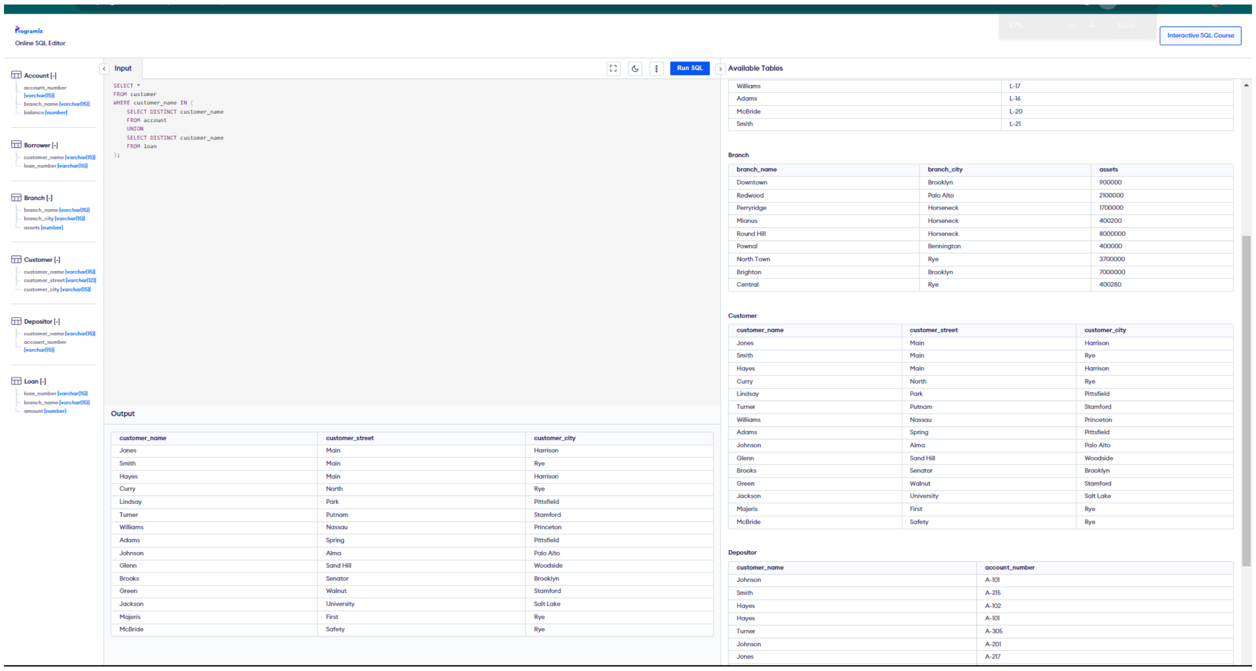
Code : “`SELECT *`

```
FROM customer
WHERE customer_name IN (
    SELECT DISTINCT customer_name
    FROM account
    UNION
    SELECT DISTINCT customer_name
    FROM loan
);
```

”

Explaining : “showing all the information of the customer from his table where his name showen in anyone of these table the account or the loan ”

Screenshot :


Programiz
Online SQL Editor
Interactive SQL Course
Account [-]
account_number [varchar(15)]
branch_name [varchar(15)]
balance [number]
Borrower [-]
customer_name [varchar(15)]
loan_number [varchar(15)]
Branch [-]
branch_name [varchar(15)]
branch_city [varchar(15)]
assets [number]
Customer [-]
customer_name [varchar(15)]
customer_street [varchar(12)]
customer_city [varchar(15)]
Depositor [-]
customer_name [varchar(15)]
account_number [varchar(15)]
Loan [-]
loan_number [varchar(15)]
branch_name [varchar(15)]
amount [number]
Input
Run SQL
Available Tables

```
SELECT *
FROM customer
WHERE customer_name IN (
    SELECT DISTINCT customer_name
    FROM account
    UNION
    SELECT DISTINCT customer_name
    FROM loan
);
```

Output

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn
Green	Walnut	Stamford
Jackson	University	Salt Lake
Majeris	First	Rye
McBride	Safety	Rye

Williams	L-17
Adams	L-16
McBride	L-20
Smith	L-21

Branch

branch_name	branch_city	assets
Downtown	Brooklyn	900000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400200
Round Hill	Horseneck	8000000
Pownal	Bennington	400000
North Town	Rye	3700000
Brighton	Brooklyn	7000000
Central	Rye	400280

Customer

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn
Green	Walnut	Stamford
Jackson	University	Salt Lake
Majeris	First	Rye
McBride	Safety	Rye

Depositor

customer_name	account_number
Johnson	A-101
Smith	A-215
Hayes	A-102
Hayes	A-101
Turner	A-305
Johnson	A-201
Jones	A-217

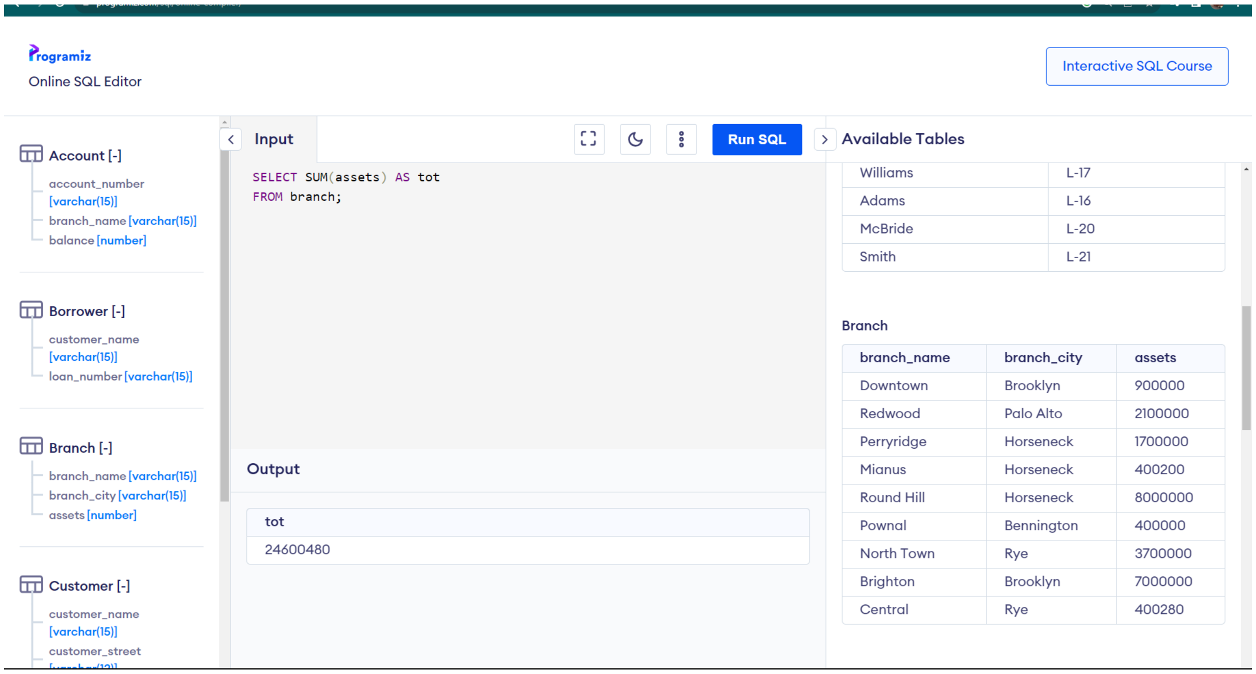
The 4th commend : “ Find the total assets of all branches. ”

Code : “`SELECT SUM(assets) AS tot`

`FROM branch;`**”**

Explaining : “showing their assets and count their values from the table of branch”

Screenshot :



Account [-]
account_number [varchar(15)]
branch_name [varchar(15)]
balance [number]

Borrower [-]
customer_name [varchar(15)]
loan_number [varchar(15)]

Branch [-]
branch_name [varchar(15)]
branch_city [varchar(15)]
assets [number]

Customer [-]
customer_name [varchar(15)]
customer_street

```
SELECT SUM(assets) AS tot
FROM branch;
```

tot
24600480

Williams	L-17
Adams	L-16
McBride	L-20
Smith	L-21

Branch

branch_name	branch_city	assets
Downtown	Brooklyn	900000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400200
Round Hill	Horseneck	8000000
Pownal	Bennington	400000
North Town	Rye	3700000
Brighton	Brooklyn	7000000
Central	Rye	400280

The 5th commend : “Find the name of the branch city and the total number of accounts at each branch city”

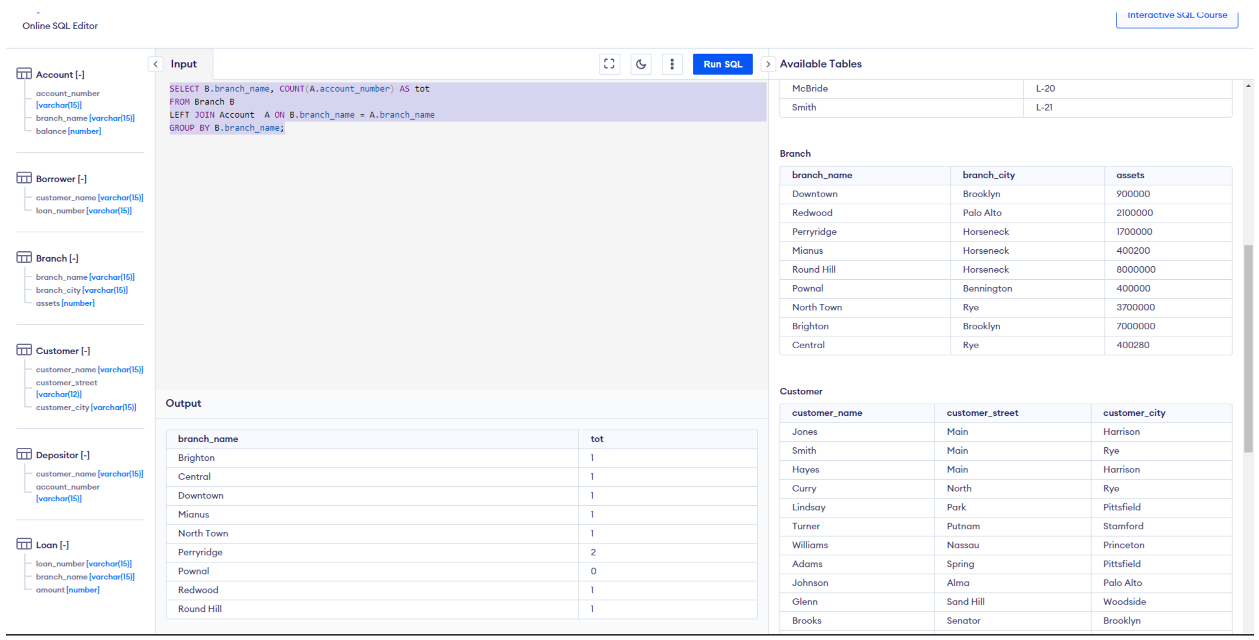
The code : “
```
SELECT B.branch_name, COUNT(A.account_number) AS tot
FROM branch B
LEFT JOIN account A ON B.branch_name = A.branch_name
GROUP BY B.branch_name;
```
”

The explanation : “

Showing branch name from branch table and the number of the account in the table of the accounts here using joining to implement two tables together with where the branch name of the account is same as the branch name which also will be grouped by branch name

”

The screenshot :



Account [-]
- account_number [varchar(15)]
- branch_name [varchar(15)]
- balance [number]

Borrower [-]
- customer_name [varchar(15)]
- loan_number [varchar(15)]

Branch [-]
- branch_name [varchar(15)]
- branch_city [varchar(15)]
- assets [number]

Customer [-]
- customer_name [varchar(15)]
- customer_street [varchar(12)]
- customer_city [varchar(15)]

Depositor [-]
- customer_name [varchar(15)]
- account_number [varchar(15)]

Loan [-]
- loan_number [varchar(15)]
- branch_name [varchar(15)]
- amount [number]

```sql
SELECT B.branch_name, COUNT(A.account_number) AS tot
FROM Branch B
LEFT JOIN Account  A ON B.branch_name = A.branch_name
GROUP BY B.branch_name;
```

Output

branch_name	tot
Brighton	1
Central	1
Downtown	1
Mianus	1
North Town	1
Perryridge	2
Pownal	0
Redwood	1
Round Hill	1

McBride	L-20
Smith	L-21

Branch

branch_name	branch_city	assets
Downtown	Brooklyn	900000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400200
Round Hill	Horseneck	8000000
Pownal	Bennington	400000
North Town	Rye	3700000
Brighton	Brooklyn	7000000
Central	Rye	400280

Customer

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn

The 6th commend : “Find the customer name and account number of the account that has the highest balance.”

The code : “select Depositor.customer_name , Depositor.account_number

from Account

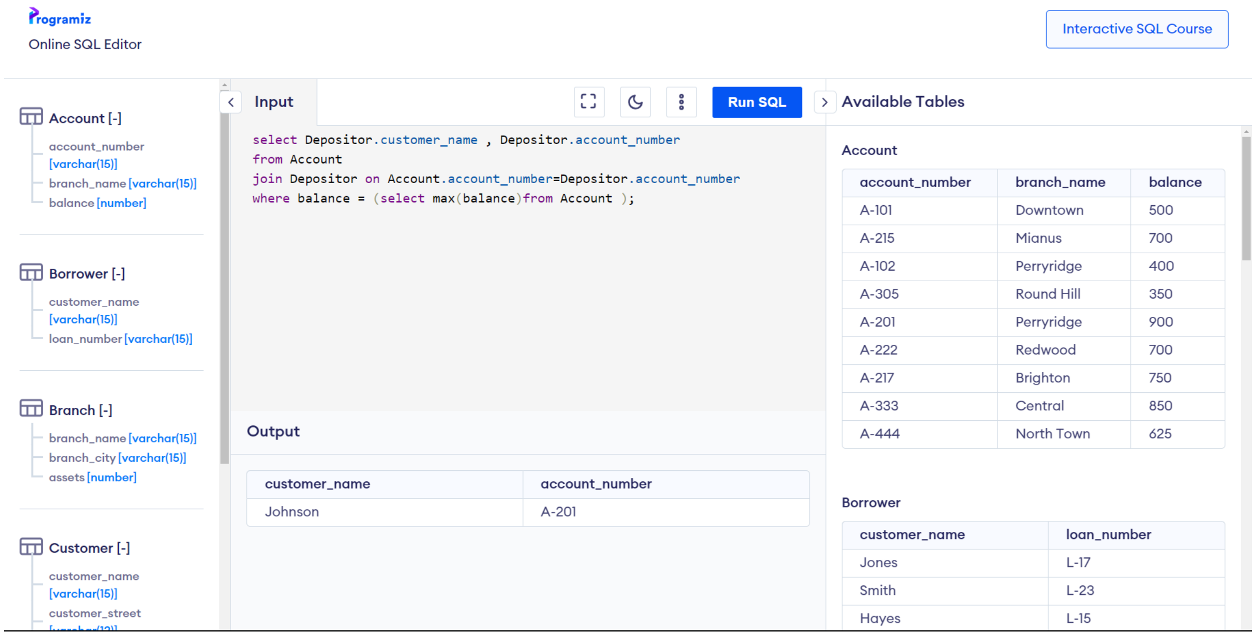
join Depositor on Account.account_number=Depositor.account_number

where balance = (select max(balance)from Account);

”

The explanation : “showing the customer name and his number account from the table of cusmtor and depositor where the balance equalled the max balance of all the table ”

The screenshot :



Account [-]
- account_number [varchar(15)]
- branch_name [varchar(15)]
- balance [number]

Borrower [-]
- customer_name [varchar(15)]
- loan_number [varchar(15)]

Branch [-]
- branch_name [varchar(15)]
- branch_city [varchar(15)]
- assets [number]

Customer [-]
- customer_name [varchar(15)]
- customer_street

```
select Depositor.customer_name , Depositor.account_number
from Account
join Depositor on Account.account_number=Depositor.account_number
where balance = (select max(balance)from Account );
```

Output

customer_name	account_number
Johnson	A-201

Account

account_number	branch_name	balance
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Perryridge	900
A-222	Redwood	700
A-217	Brighton	750
A-333	Central	850
A-444	North Town	625

Borrower

customer_name	loan_number
Jones	L-17
Smith	L-23
Hayes	L-15

The 7th commend : “. Find the average balance of accounts at each branch and display

them according to their

branch name in ascending order and the average balance in descending order.”

The code : “SELECT B.branch_name, AVG(A.balance) AS Averagebalance

FROM branch B

JOIN account A ON B.branch_name = A.branch_name

GROUP BY B.branch_name

ORDER BY B.branch_name ASC, Averagebalance DESC;

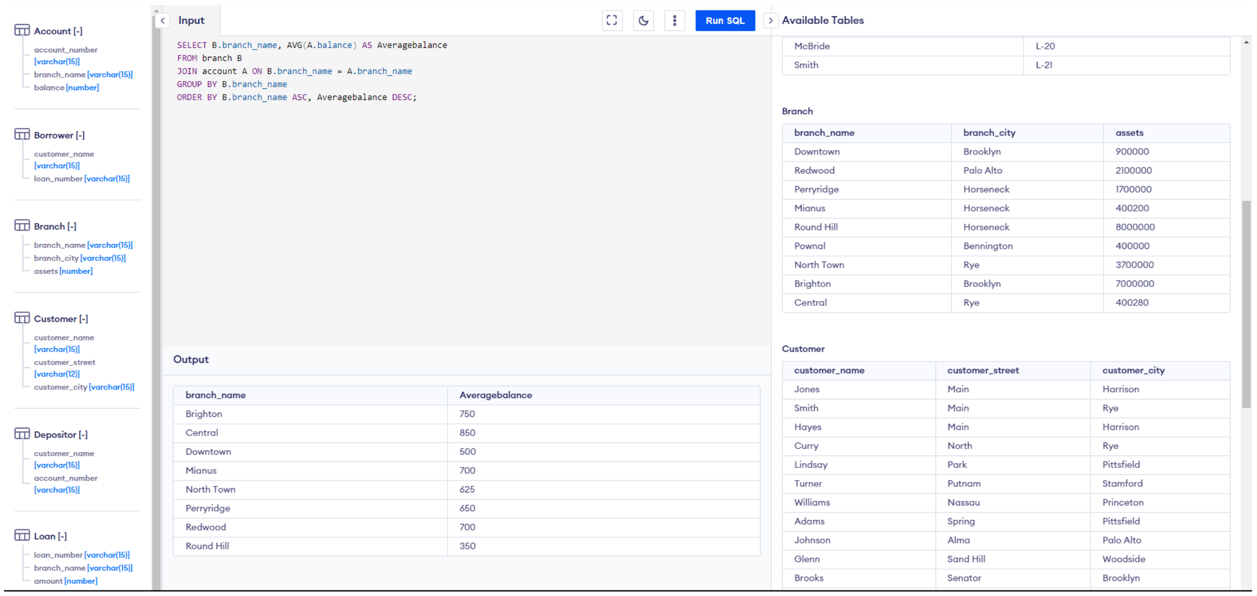
”

The explanation : “

Showing the branch name from the branch table and the average of balance of the accounts who are in the table i used the join with condition as branch name account must be as branch nameand group it and order each one of them according to their name asc if they are same then sort as averagebalance desc

”

The screenshot :



Account [-]
- account_number [varchar(15)]
- branch_name [varchar(15)]
- balance [number]

Borrower [-]
- customer_name [varchar(15)]
- loan_number [varchar(15)]

Branch [-]
- branch_name [varchar(15)]
- branch_city [varchar(15)]
- assets [number]

Customer [-]
- customer_name [varchar(15)]
- customer_street [varchar(12)]
- customer_city [varchar(15)]

Depositor [-]
- customer_name [varchar(15)]
- account_number [varchar(15)]

Loan [-]
- loan_number [varchar(15)]
- branch_name [varchar(15)]
- amount [number]

Input

```sql
SELECT B.branch_name, AVG(A.balance) AS Averagebalance
FROM branch B
JOIN account A ON B.branch_name = A.branch_name
GROUP BY B.branch_name
ORDER BY B.branch_name ASC, Averagebalance DESC;
```

Output

branch_name	Averagebalance
Brighton	750
Central	850
Downtown	500
Mianus	700
North Town	625
Perryridge	650
Redwood	700
Round Hill	350

Available Tables

McBride	L-20
Smith	L-21

Branch

branch_name	branch_city	assets
Downtown	Brooklyn	900000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400200
Round Hill	Horseneck	8000000
Pownal	Bennington	400000
North Town	Rye	3700000
Brighton	Brooklyn	7000000
Central	Rye	400280

Customer

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn

The 8th commend : “Find the number of customers of each customer_city who are depositors as well as have loans.”

The code : “select count(Customer.customer_name)"number of who have loan and have deposior",Customer.customer_city

from Customer

where Customer.customer_name IN (select Depositor.customer_name from Depositor join Account on Account.account_number = Depositor.account_number)

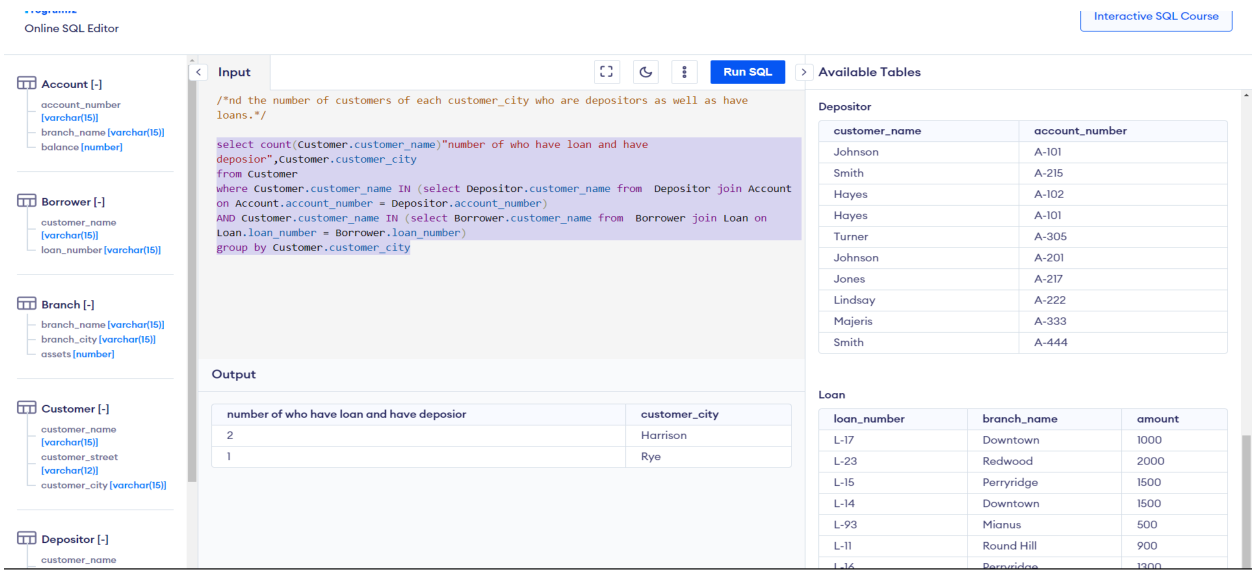
AND Customer.customer_name IN (select Borrower.customer_name from Borrower join Loan on Loan.loan_number = Borrower.loan_number)

group by Customer.customer_city

”

The explanation : “here is intersect using the and which it will do as the two group every group inside it number if that number inside the other number it will take it as info as record inside it and the second same it what it is the first but one for account and one for the loans ”

The screenshot :


Online SQL Editor
Interactive SQL Course
Account [-]
account_number [varchar(15)]
branch_name [varchar(15)]
balance [number]
Borrower [-]
customer_name [varchar(15)]
loan_number [varchar(15)]
Branch [-]
branch_name [varchar(15)]
branch_city [varchar(15)]
assets [number]
Customer [-]
customer_name [varchar(15)]
customer_street [varchar(12)]
customer_city [varchar(15)]
Depositor [-]
customer_name
Input
Run SQL
Available Tables

```
/*nd the number of customers of each customer_city who are depositors as well as have
loans.*/

select count(Customer.customer_name)"number of who have loan and have
deposior",Customer.customer_city
from Customer
where Customer.customer_name IN (select Depositor.customer_name from  Depositor join Account
on Account.account_number = Depositor.account_number)
AND Customer.customer_name IN (select Borrower.customer_name from  Borrower join Loan on
Loan.loan_number = Borrower.loan_number)
group by Customer.customer_city
```

Output

number of who have loan and have deposior	customer_city
2	Harrison
1	Rye

Depositor

customer_name	account_number
Johnson	A-101
Smith	A-215
Hayes	A-102
Hayes	A-101
Turner	A-305
Johnson	A-201
Jones	A-217
Lindsay	A-222
Majeris	A-333
Smith	A-444

Loan

loan_number	branch_name	amount
L-17	Downtown	1000
L-23	Redwood	2000
L-15	Perryridge	1500
L-14	Downtown	1500
L-93	Mianus	500
L-11	Round Hill	900
L-16	Perryridge	1300

The 9th commend : “Find the average loan amount at each branch. Do not include any branch which is located in a that has the substring, ’Horse’ in its name.”

The code : “SELECT B.branch_name, AVG(L.amount) AS Averageamount

FROM branch B

JOIN loan L ON B.branch_name = L.branch_name

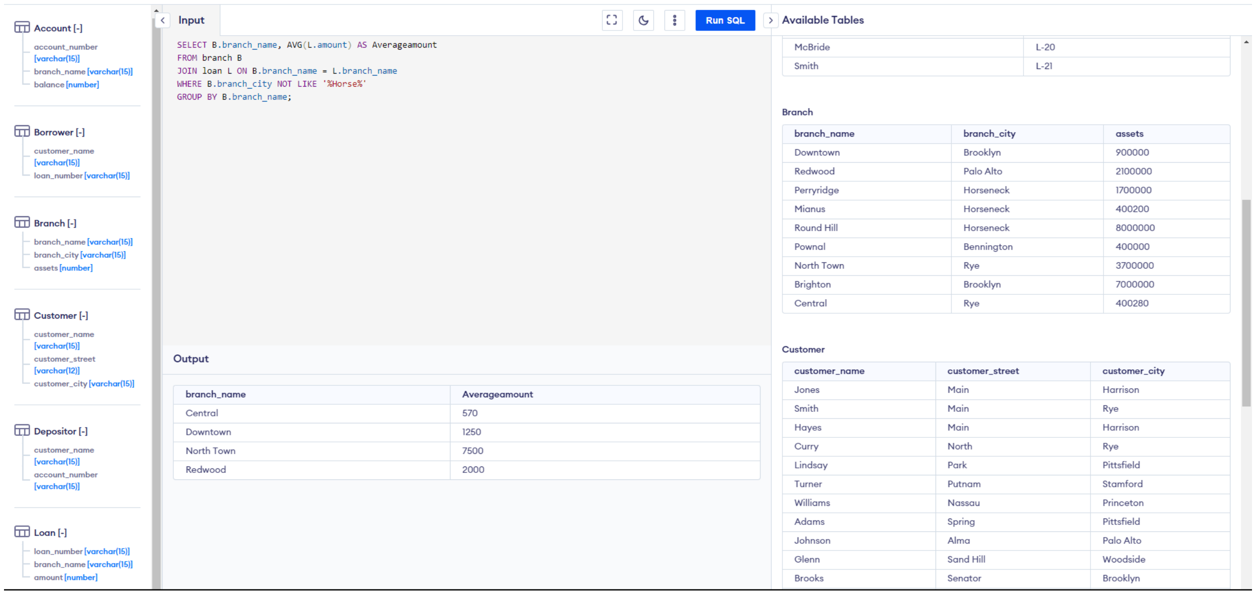
WHERE B.branch_city NOT LIKE '%Horse%'

GROUP BY B.branch_name;

”

The explanation : “it will show the name of the branch from branch and the average of the amount of the loan where it will join two table of the info where the branch name of the account be same as branch name and also will not have horse in the structure of the character inside it as branch city and finally it will be grouped as branch name ”

The screenshot :



Account [-]
account_number [varchar(15)]
branch_name [varchar(15)]
balance [number]
Borrower [-]
customer_name [varchar(15)]
loan_number [varchar(15)]
Branch [-]
branch_name [varchar(15)]
branch_city [varchar(15)]
assets [number]
Customer [-]
customer_name [varchar(15)]
customer_street [varchar(12)]
customer_city [varchar(15)]
Depositor [-]
customer_name [varchar(15)]
account_number [varchar(15)]
Loan [-]
loan_number [varchar(15)]
branch_name [varchar(15)]
amount [number]
Input
Run SQL
Available Tables

```
SELECT B.branch_name, AVG(L.amount) AS Averageamount
FROM branch B
JOIN loan L ON B.branch_name = L.branch_name
WHERE B.branch_city NOT LIKE '%Horse%'
GROUP BY B.branch_name;
```

Output

branch_name	Averageamount
Central	570
Downtown	1250
North Town	7500
Redwood	2000

McBride	L-20
Smith	L-21

Branch

branch_name	branch_city	assets
Downtown	Brooklyn	900000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400200
Round Hill	Horseneck	8000000
Pownal	Bennington	400000
North Town	Rye	3700000
Brighton	Brooklyn	7000000
Central	Rye	400280

Customer

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn

The 10th commend : “Find all customer-related information who have an account in a branch, located in the same

city as they live.”

The code : “select * from Customer join depositor on depositor.customer_name = customer.customer_name

where account_number in (select account_number from account join branch on

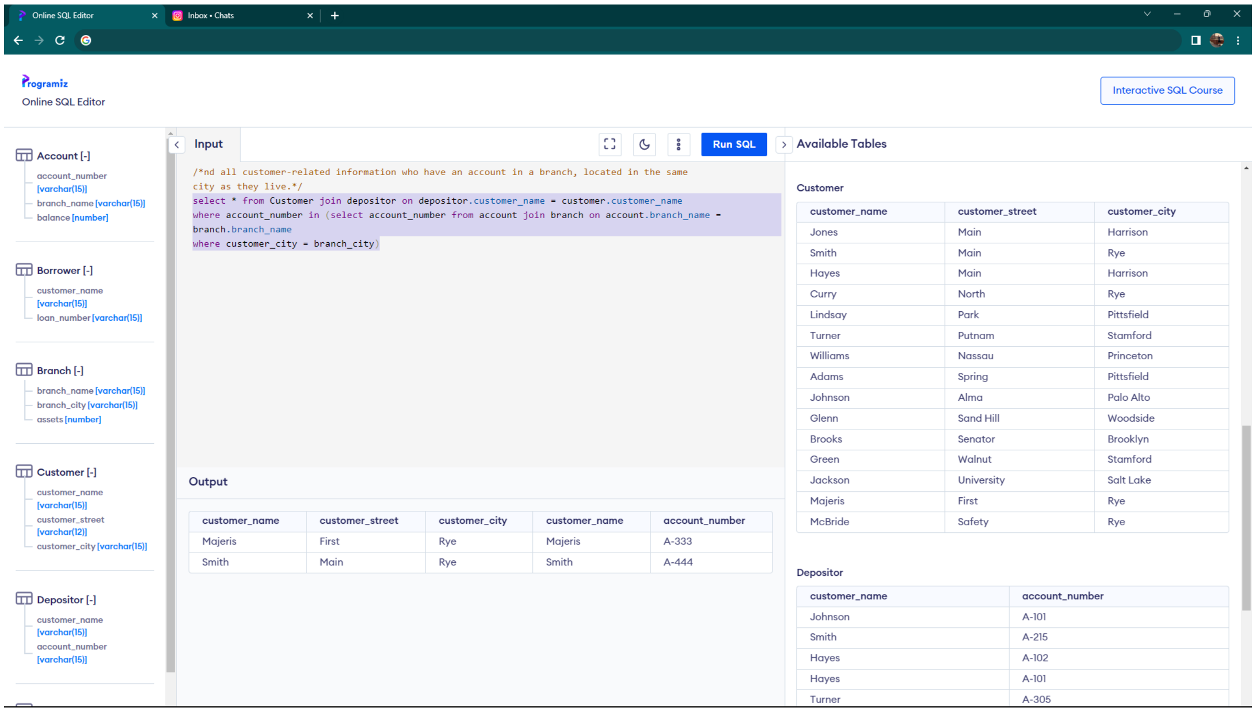
account.branch_name = branch.branch_name

where customer_city = branch_city)

”

The explanation : “it will show all of the info about these customers who have depositors where they will be taken from the customer city and they will be taken branch name after that if they are equealled it will record them ”

The screenshot :



Programiz
Online SQL Editor

Account [-]
- account_number [varchar(15)]
- branch_name [varchar(15)]
- balance [number]

Borrower [-]
- customer_name [varchar(15)]
- loan_number [varchar(15)]

Branch [-]
- branch_name [varchar(15)]
- branch_city [varchar(15)]
- assets [number]

Customer [-]
- customer_name [varchar(15)]
- customer_street [varchar(12)]
- customer_city [varchar(15)]

Depositor [-]
- customer_name [varchar(15)]
- account_number [varchar(15)]

Input

```
/*nd all customer-related information who have an account in a branch, located in the same
city as they live.*/
select * from Customer join depositor on depositor.customer_name = customer.customer_name
where account_number in (select account_number from account join branch on account.branch_name =
branch.branch_name
where customer_city = branch_city)
```

Output

customer_name	customer_street	customer_city	customer_name	account_number
Majeris	First	Rye	Majeris	A-333
Smith	Main	Rye	Smith	A-444

Available Tables

Customer

customer_name	customer_street	customer_city
Jones	Main	Harrison
Smith	Main	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn
Green	Walnut	Stamford
Jackson	University	Salt Lake
Majeris	First	Rye
McBride	Safety	Rye

Depositor

customer_name	account_number
Johnson	A-101
Smith	A-215
Hayes	A-102
Hayes	A-101
Turner	A-305

The 11th commend : “Find the average loan amount at each branch. Do not include any branch which is located in a that has the substring, ’Horse’ in its name.”

The code : “

```
SELECT B.branch_name
FROM branch B
JOIN account A ON B.branch_name = A.branch_name
GROUP BY B.branch_name
HAVING SUM(A.balance) > (
   SELECT AVG(Totalbalance)
   FROM (
      SELECT SUM(A.balance) AS Totalbalance
      FROM branch B
      JOIN account A ON B.branch_name = A.branch_name
      GROUP BY B.branch_name
   ) AS tot);
```

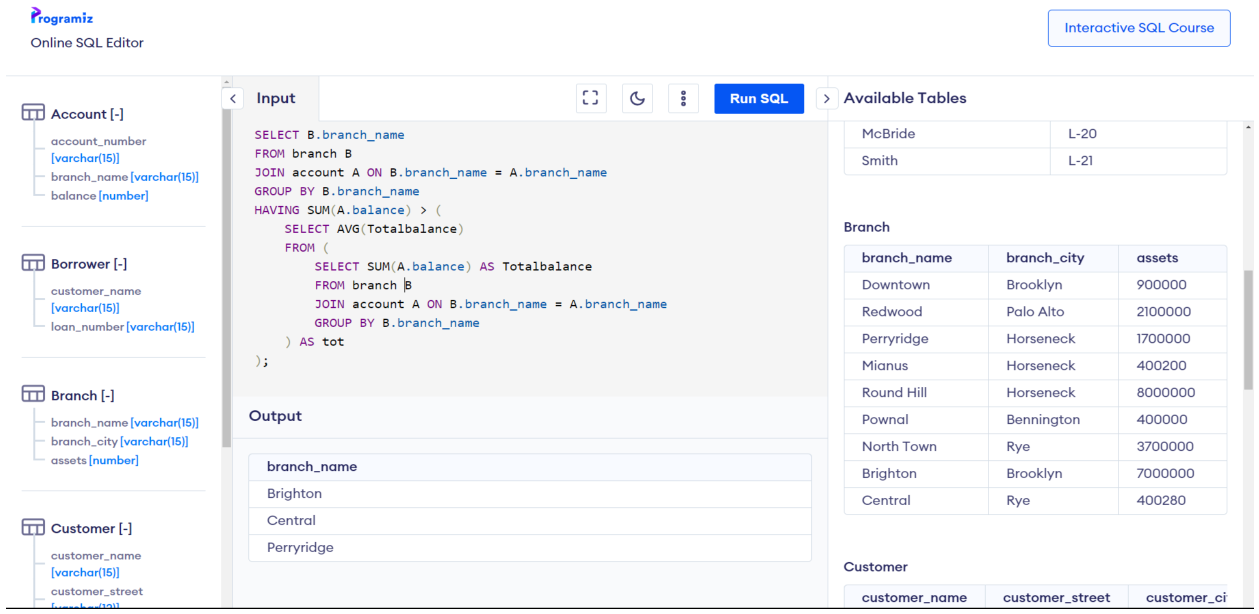
”

The explanation : “

Showing the branch name from branch table which branch name must be same as account branch name and will be grouped by branch name while the sum of his balance will be bigger than (showing the sum of the balance of that account)

”

The screenshot :



Account [-]
account_number [varchar(15)]
branch_name [varchar(15)]
balance [number]

Borrower [-]
customer_name [varchar(15)]
loan_number [varchar(15)]

Branch [-]
branch_name [varchar(15)]
branch_city [varchar(15)]
assets [number]

Customer [-]
customer_name [varchar(15)]
customer_street

```
SELECT B.branch_name
FROM branch B
JOIN account A ON B.branch_name = A.branch_name
GROUP BY B.branch_name
HAVING SUM(A.balance) > (
    SELECT AVG(Totalbalance)
    FROM (
        SELECT SUM(A.balance) AS Totalbalance
        FROM branch B
        JOIN account A ON B.branch_name = A.branch_name
        GROUP BY B.branch_name
    ) AS tot
);
```

Output

branch_name
Brighton
Central
Perryridge

McBride	L-20
Smith	L-21

Branch

branch_name	branch_city	assets
Downtown	Brooklyn	900000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400200
Round Hill	Horseneck	8000000
Pownal	Bennington	400000
North Town	Rye	3700000
Brighton	Brooklyn	7000000
Central	Rye	400280

Customer

customer_name	customer_street	customer_ci

The 12th commend : “Find those branch names where the total customer balance is greater than the total customer

loan amount.”

The code : “SELECT B.branch_name

FROM branch B

JOIN account A ON B.branch_name = A.branch_name

JOIN loan C ON A.branch_name = C.branch_name

GROUP BY B.branch_name

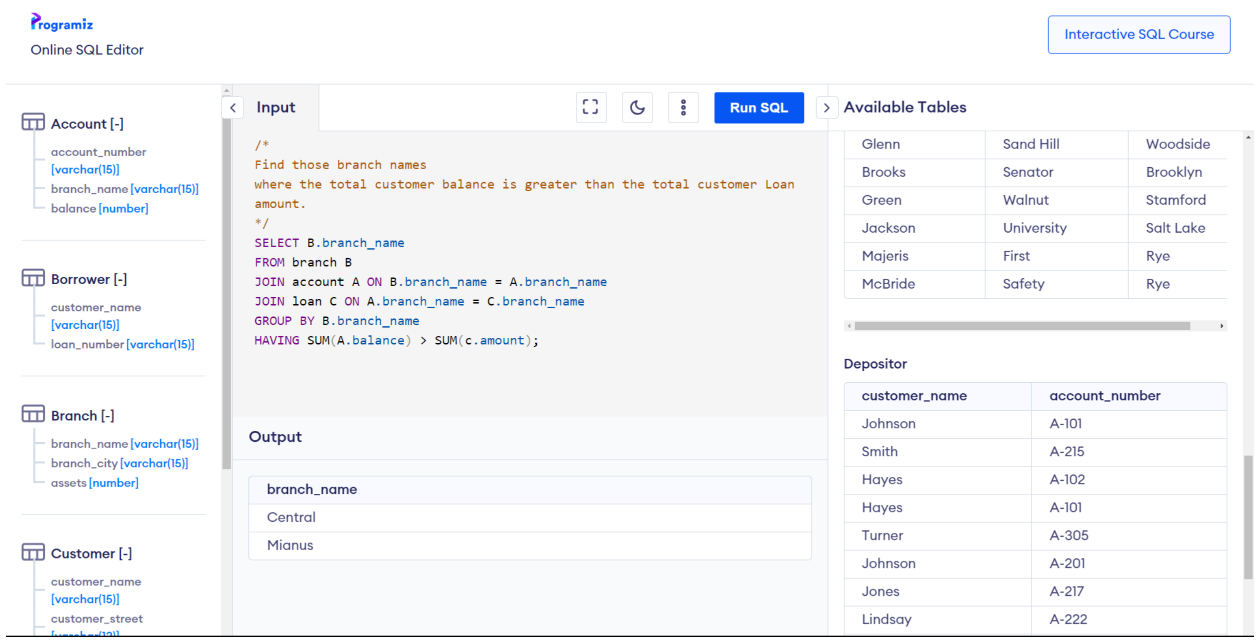
HAVING SUM(A.balance) > SUM(c.amount);

”

The explanation : “selecting the branch name and group by it where the balance bigger than amount which i made joining two table inside the branch because that’s will give me

what i need easily such as balance and amount also there is another method i thinked about it i use it before at 10th commend question it's a nested where "

The screenshot :



Account [-]
- account_number [varchar(15)]
- branch_name [varchar(15)]
- balance [number]

Borrower [-]
- customer_name [varchar(15)]
- loan_number [varchar(15)]

Branch [-]
- branch_name [varchar(15)]
- branch_city [varchar(15)]
- assets [number]

Customer [-]
- customer_name [varchar(15)]
- customer_street

```
/*
Find those branch names
where the total customer balance is greater than the total customer Loan
amount.
*/
SELECT B.branch_name
FROM branch B
JOIN account A ON B.branch_name = A.branch_name
JOIN loan C ON A.branch_name = C.branch_name
GROUP BY B.branch_name
HAVING SUM(A.balance) > SUM(c.amount);
```

Output

branch_name
Central
Mianus

Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn
Green	Walnut	Stamford
Jackson	University	Salt Lake
Majeris	First	Rye
McBride	Safety	Rye

Depositor

customer_name	account_number
Johnson	A-101
Smith	A-215
Hayes	A-102
Hayes	A-101
Turner	A-305
Johnson	A-201
Jones	A-217
Lindsay	A-222

The 13th commend : " Find the names of the customers who have at least one loan that can be paid off by his/her total balance."

The code : "SELECT DISTINCT C.customer_name

FROM customer C

JOIN borrower B ON C.customer_name = B.customer_name

JOIN loan L ON B.loan_number = L.loan_number

JOIN account A ON L.branch_name = A.branch_name

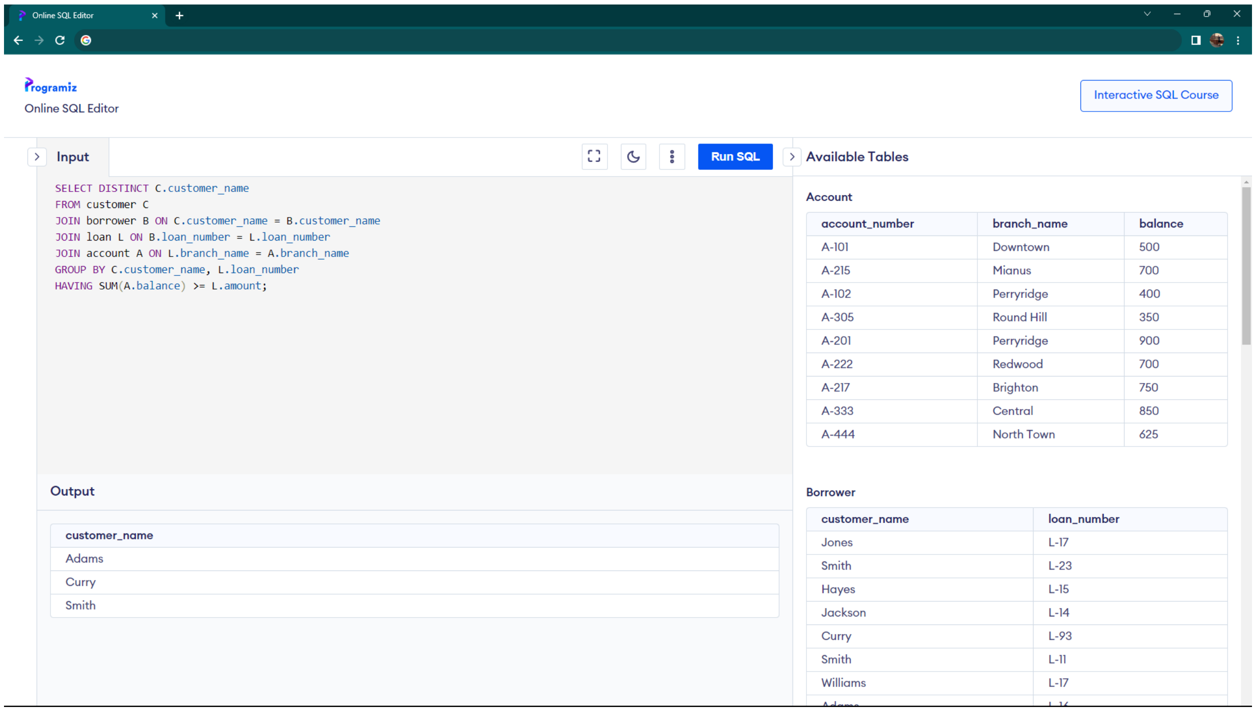
GROUP BY C.customer_name, L.loan_number

HAVING SUM(A.balance) >= L.amount;

"

The explanation : "showing customer name from table customer and join the borrower which his name showen in table of the customer and same as loan and accout joining tables after that group it by name of the customer and loan number who will have balance bigger or equalled to the loan amount it will be showen as uniqe value "

The screenshot :



```sql
SELECT DISTINCT C.customer_name
FROM customer C
JOIN borrower B ON C.customer_name = B.customer_name
JOIN loan L ON B.loan_number = L.loan_number
JOIN account A ON L.branch_name = A.branch_name
GROUP BY C.customer_name, L.loan_number
HAVING SUM(A.balance) >= L.amount;
```

Output

customer_name
Adams
Curry
Smith

Account

account_number	branch_name	balance
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Perryridge	900
A-222	Redwood	700
A-217	Brighton	750
A-333	Central	850
A-444	North Town	625

Borrower

customer_name	loan_number
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

The 14th commend : "Find those customers' names, balances and loan amounts who have accounts and loans but

neither in the branch of their own city."

The code : "SELECT DISTINCT C.customer_name, A.balance, L.amount

FROM customer C

JOIN depositor D ON C.customer_name = D.customer_name

JOIN account A ON D.account_number = A.account_number

JOIN borrower B ON C.customer_name = B.customer_name

JOIN loan L ON B.loan_number = L.loan_number

WHERE A.branch_name NOT IN (

SELECT branch_name

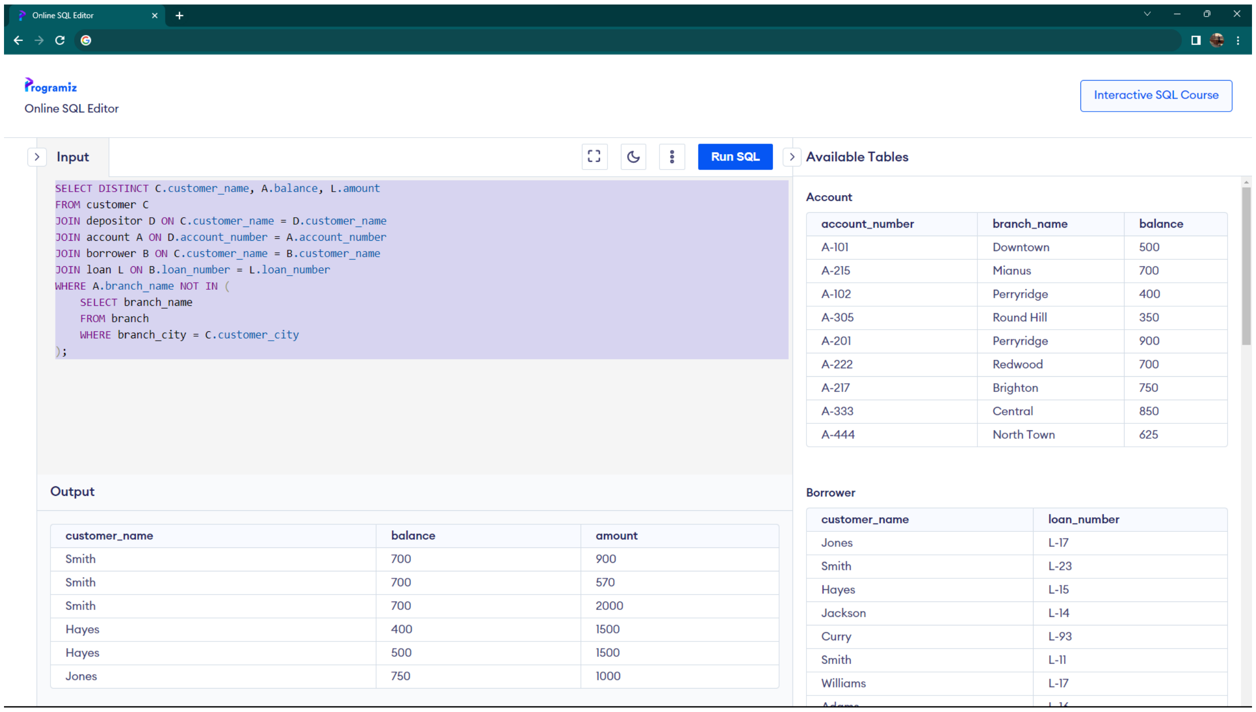
FROM branch

WHERE branch_city = C.customer_city

);”

The explanation : “showing the name of the customer and balance of his account and amount of his loan and joining these tables together which contains same info of same human when the branch city will not be same as the city of the branch ”

The screenshot :



```sql
SELECT DISTINCT C.customer_name, A.balance, L.amount
FROM customer C
JOIN depositor D ON C.customer_name = D.customer_name
JOIN account A ON D.account_number = A.account_number
JOIN borrower B ON C.customer_name = B.customer_name
JOIN loan L ON B.loan_number = L.loan_number
WHERE A.branch_name NOT IN (
    SELECT branch_name
    FROM branch
    WHERE branch_city = C.customer_city
);
```

Output

customer_name	balance	amount
Smith	700	900
Smith	700	570
Smith	700	2000
Hayes	400	1500
Hayes	500	1500
Jones	750	1000

Account

account_number	branch_name	balance
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Perryridge	900
A-222	Redwood	700
A-217	Brighton	750
A-333	Central	850
A-444	North Town	625

Borrower

customer_name	loan_number
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17

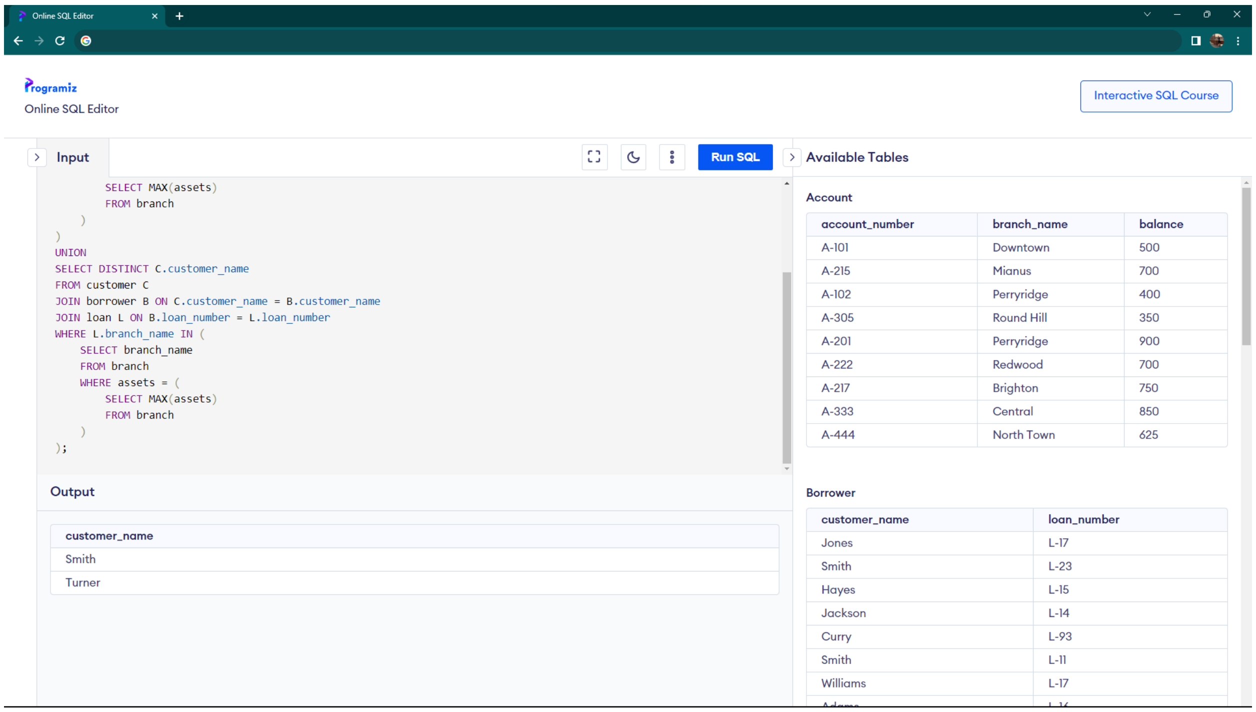
The 15th commend : “Find the customers’ names who have accounts or loans in the branch that has the highest assets.”

The code : “
```
SELECT DISTINCT C.customer_name
FROM customer C
JOIN depositor D ON C.customer_name = D.customer_name
JOIN account A ON D.account_number = A.account_number
WHERE A.branch_name IN (
   SELECT branch_name
   FROM branch
   WHERE assets = (
      SELECT MAX(assets)
      FROM branch
   )
)
UNION
SELECT DISTINCT C.customer_name
FROM customer C
JOIN borrower B ON C.customer_name = B.customer_name
JOIN loan L ON B.loan_number = L.loan_number
WHERE L.branch_name IN (
   SELECT branch_name
   FROM branch
   WHERE assets = (
       SELECT MAX(assets)
       FROM branch
   )
);
```

”

The explanation : “there is two set will include all of two showing records every table has his condition and his joining part it will search for the branch who will have the max of assets by their names the branches of course we dont have the main access to that one then we will connect it together by the relation we made it as join like that will have the access of all of it ”

The screenshot :



Programiz
Online SQL Editor

Input

```
        SELECT MAX(assets)
        FROM branch
    )
)
UNION
SELECT DISTINCT C.customer_name
FROM customer C
JOIN borrower B ON C.customer_name = B.customer_name
JOIN loan L ON B.loan_number = L.loan_number
WHERE L.branch_name IN (
    SELECT branch_name
    FROM branch
    WHERE assets = (
        SELECT MAX(assets)
        FROM branch
    )
);
```

Output

customer_name
Smith
Turner

Available Tables

Account

account_number	branch_name	balance
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Perryridge	900
A-222	Redwood	700
A-217	Brighton	750
A-333	Central	850
A-444	North Town	625

Borrower

customer_name	loan_number
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14
Curry	L-93
Smith	L-11
Williams	L-17