

**LAB-03**

**Title: Writing SQL Statements for Multi-table, Set and Aggregate.**

**Name: Azizul Abedin Azmi**

**ID: 2022-1-60-130**

**Section: 02**

**Course Code: CSE302**

**Course Title: (Database Systems)**

**Date: 29/08/2024**

**Course Instructor:**

**Mahmuda Rawnak Jahan**

**Lecturer**

**Department of Computer Science and Engineering**

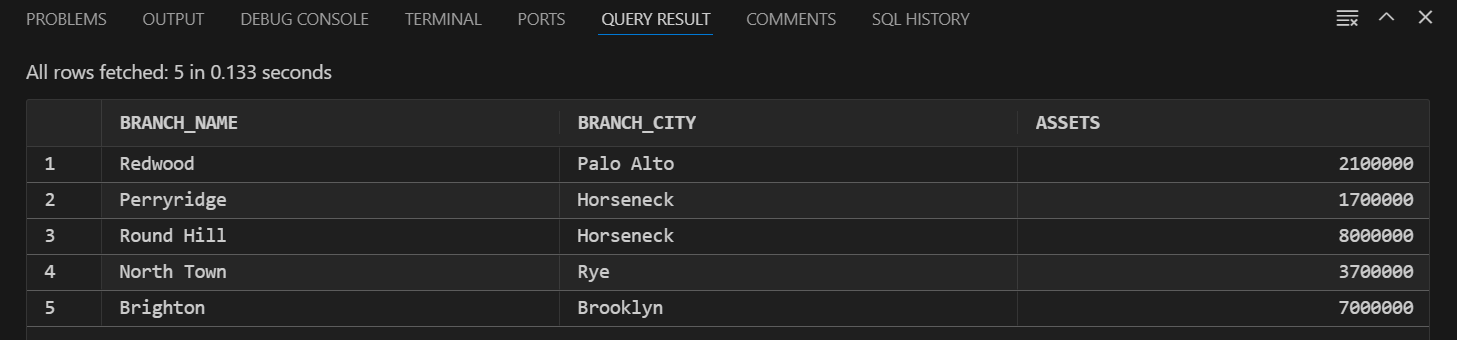
**Lab Tasks:**

1) Find all branch names and cities with assets more than 1000000. (on single table)

SELECT \*

FROM BRANCH

WHERE ASSETS > 1000000;



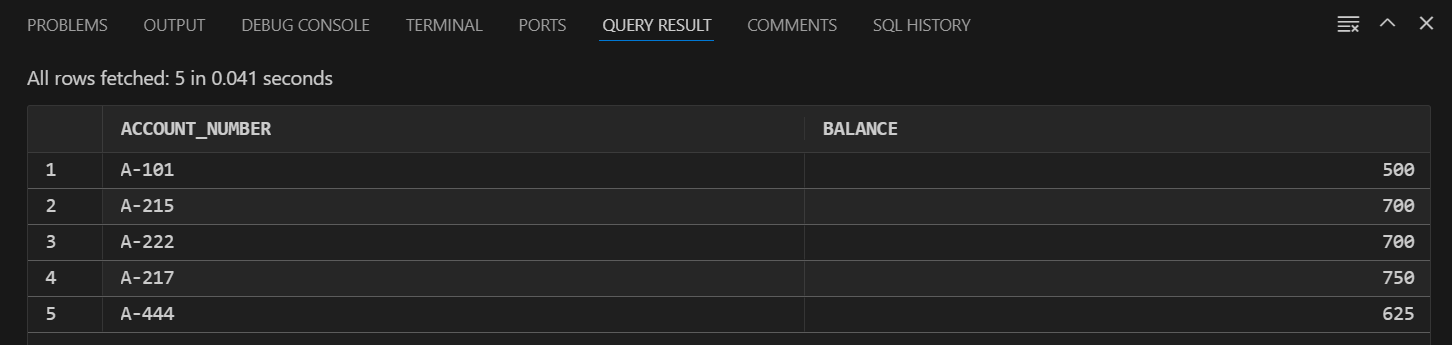
2) Find all account numbers and their balance which are opened in ‘Downtown’ branch or which have balance in between 600 and 750. (on single table)

SELECT account\_number, balance

FROM account

WHERE branch\_name = 'Downtown'

   OR balance BETWEEN 600 AND 750;

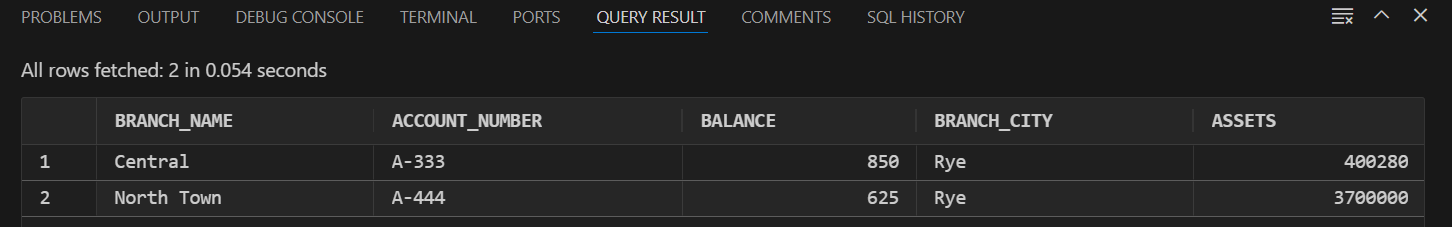


3) Find all account numbers which are opened in a branch located in ‘Rye’ city. (multiple tables)

SELECT \*

FROM account NATURAL JOIN BRANCH

WHERE branch\_city = 'Rye';



4) Find all loan numbers which have amount greater than or equal to 1000 and their customers are living in ‘Harrison’ city. (multiple tables)

SELECT loan.loan\_number

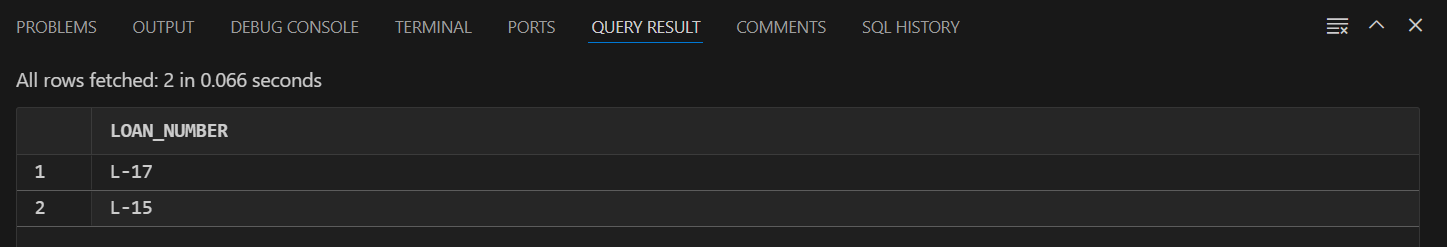
FROM loan

JOIN borrower ON loan.loan\_number = borrower.loan\_number

JOIN customer ON borrower.customer\_name = customer.customer\_name

WHERE loan.amount >= 1000

  AND customer.customer\_city = 'Harrison';

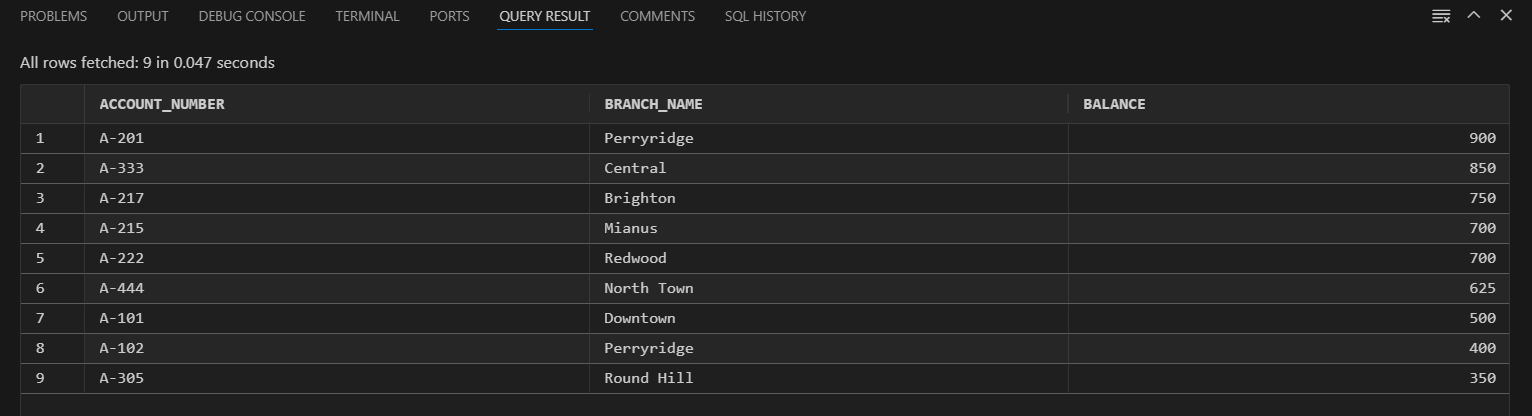


5) Display the account related information based on the descending order of the balance. (order by clause)

SELECT \*

FROM ACCOUNT

ORDER BY balance DESC;

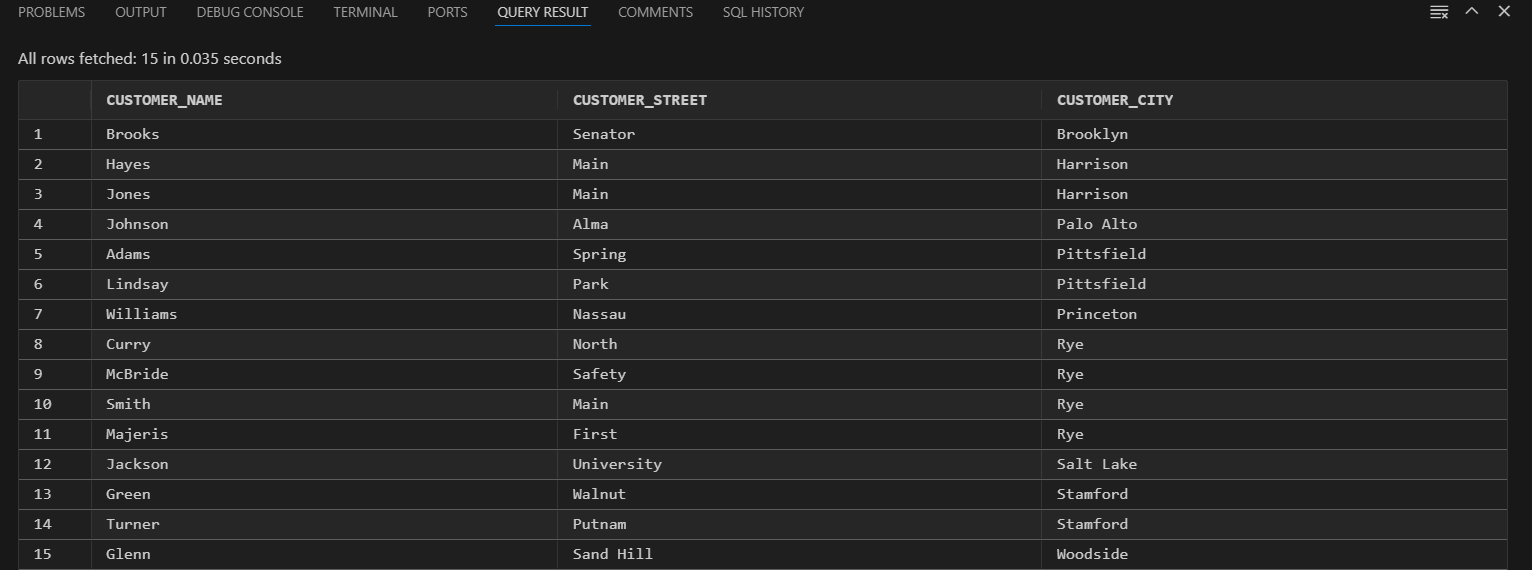


6) Display the customer related information in alphabetic order of customer cities. (order by clause)

SELECT \*

FROM CUSTOMER

ORDER BY customer\_city;



7) Find all customer names who have an account as well as a loan. (intersect)

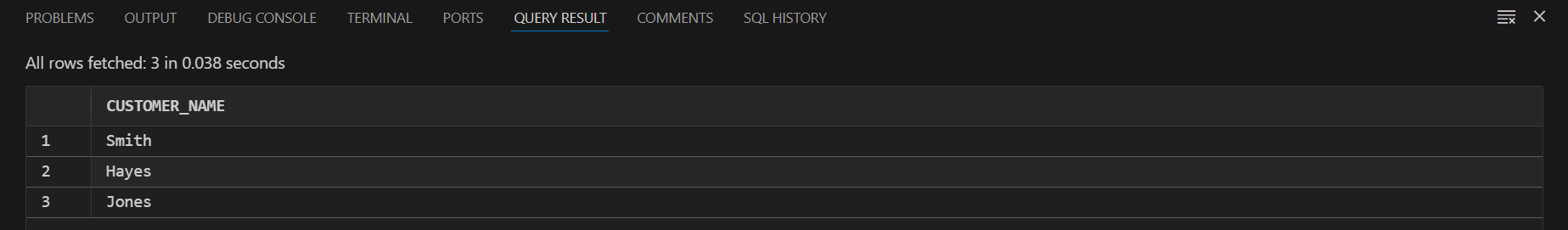
SELECT customer\_name

FROM depositor

INTERSECT

SELECT customer\_name

FROM borrower;



8) Find all customer related information who have an account or a loan. (union)

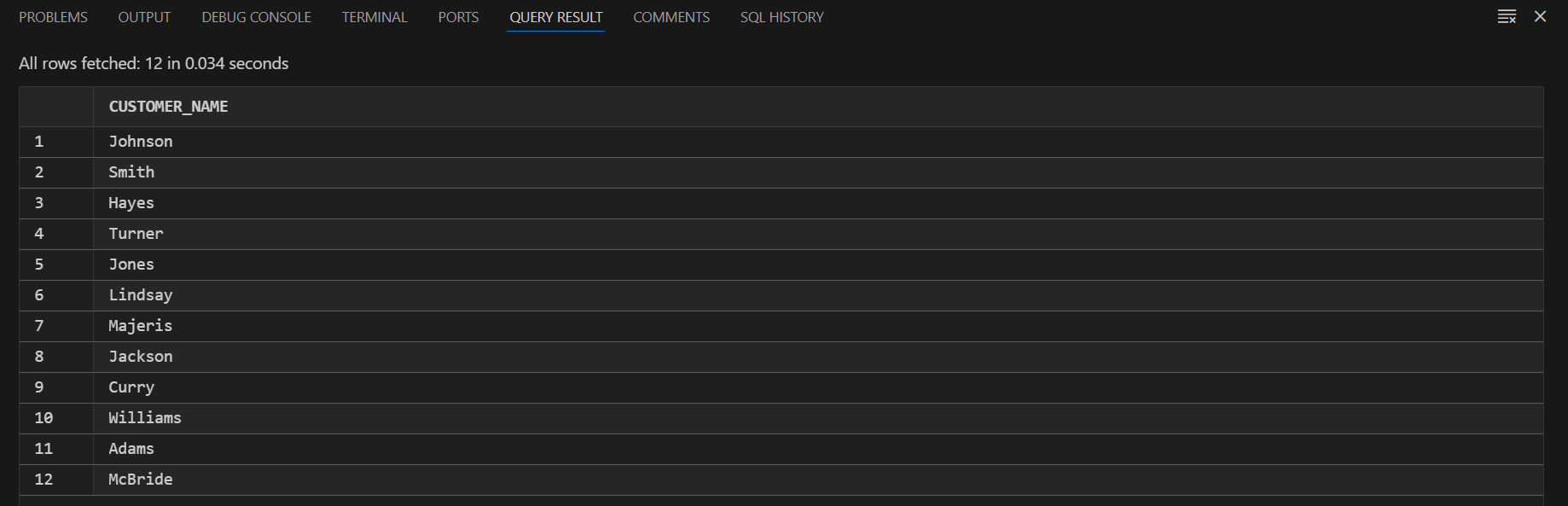
SELECT customer\_name

FROM depositor

UNION

SELECT customer\_name

FROM borrower;



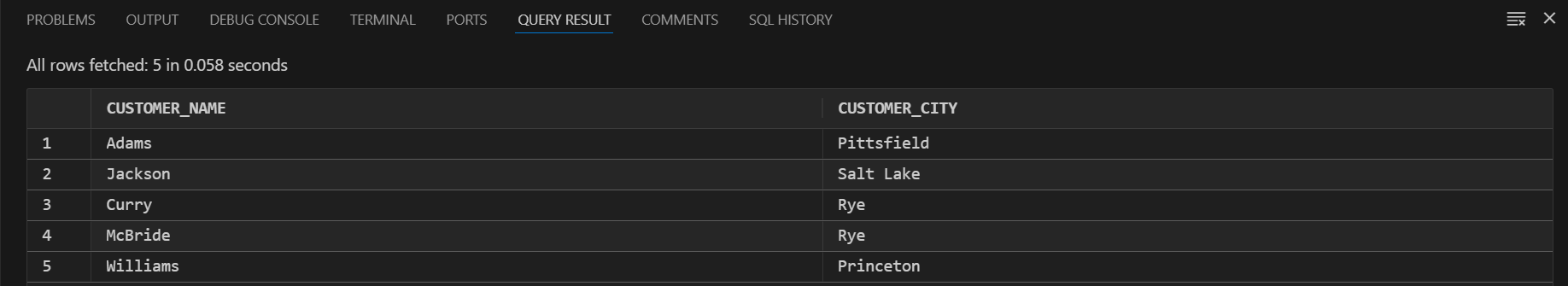
9) Find all customer names and their cities who have a loan but not an account. (minus)

SELECT customer.customer\_name, customer.customer\_city

FROM customer

JOIN borrower ON customer.customer\_name = borrower.customer\_name

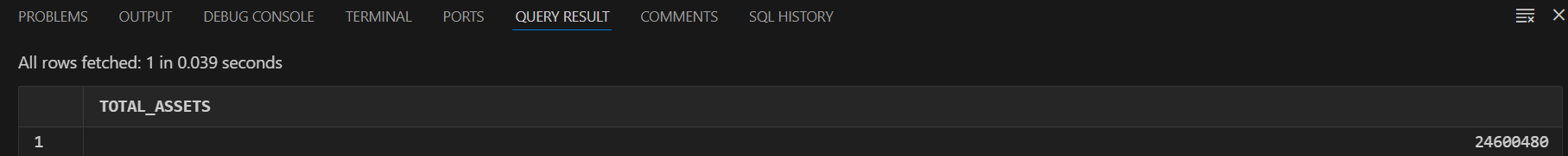
WHERE customer.customer\_name NOT IN (SELECT customer\_name FROM depositor);



10) Find the total assets of all branches. (aggregate function)

SELECT SUM(ASSETS) AS total\_assets

FROM branch;

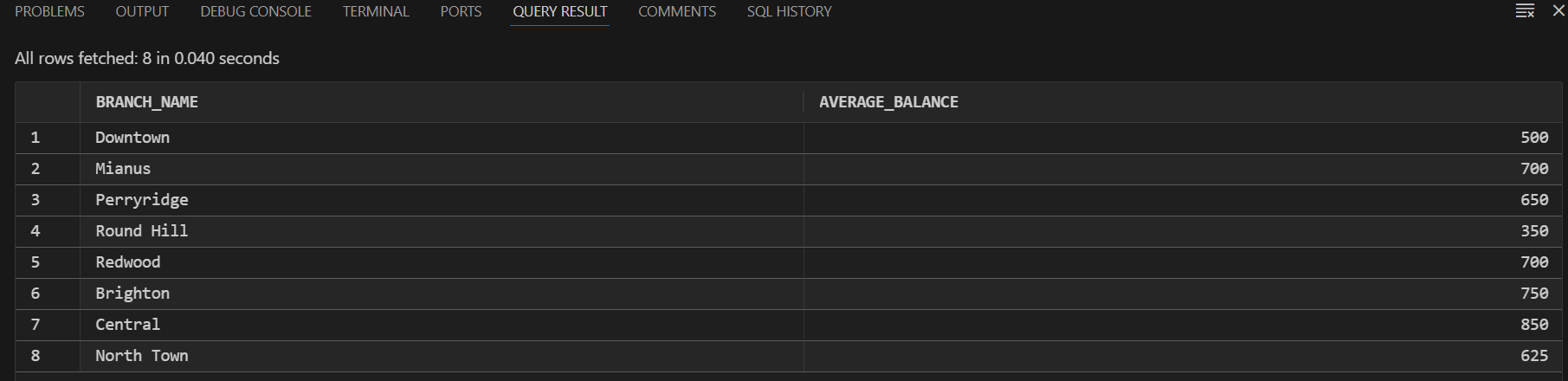


11) Find the average balance of accounts at each branch. (aggregate function)

SELECT branch\_name, AVG(balance) AS average\_balance

FROM account

GROUP BY branch\_name;



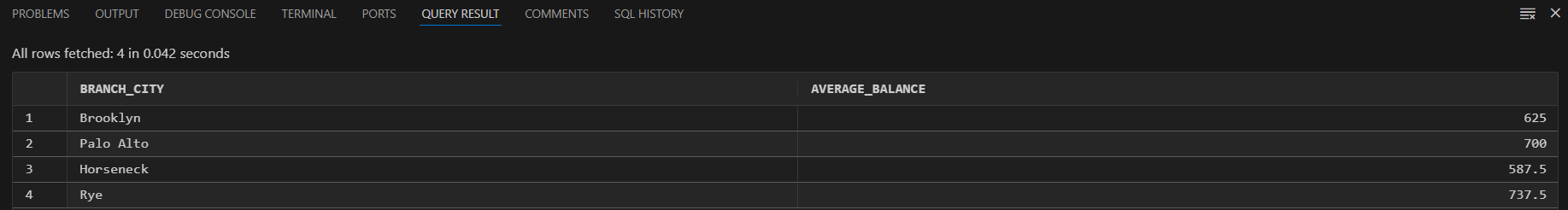
12) Find the average balance of accounts at each branch city. (aggregate function)

SELECT branch.branch\_city, AVG(account.balance) AS average\_balance

FROM account

JOIN branch ON account.branch\_name = branch.branch\_name

GROUP BY branch.branch\_city;

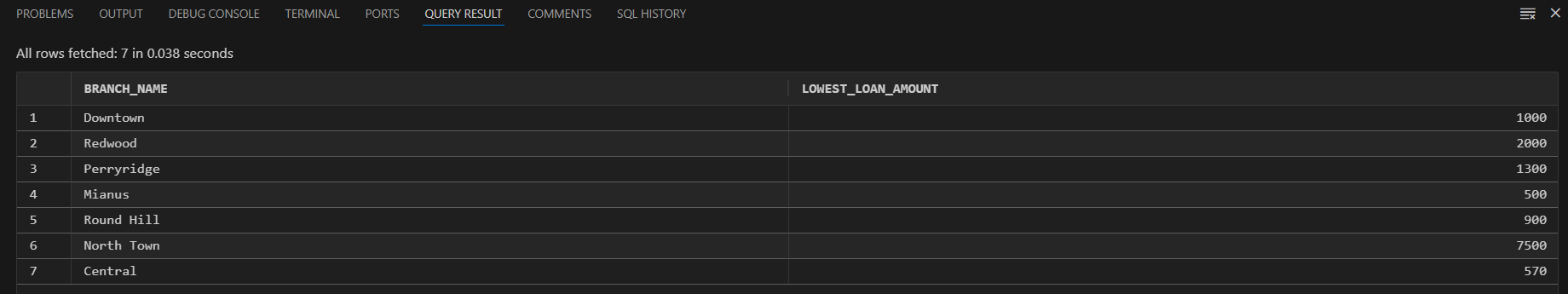


13) Find the lowest amount of loan at each branch. (aggregate function)

SELECT branch\_name, MIN(amount) AS lowest\_loan\_amount

FROM loan

GROUP BY branch\_name;

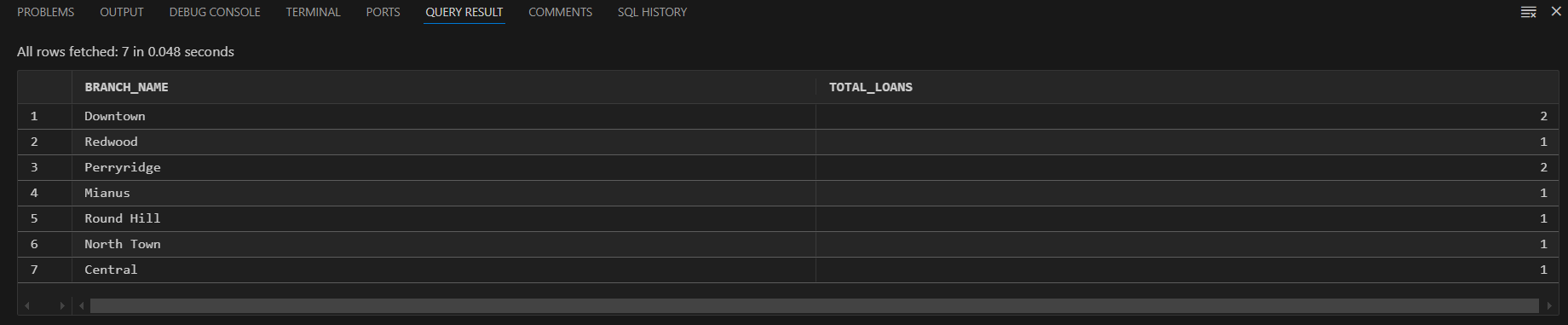


14) Find the total number of loans at each branch. (aggregate function)

SELECT branch\_name, COUNT(\*) AS total\_loans

FROM loan

GROUP BY branch\_name;



15) Find the customer name and account number of the account which has the highest balance. (aggregate function)

SELECT customer.customer\_name, account.account\_number

FROM depositor

JOIN account ON depositor.account\_number = account.account\_number

JOIN customer ON depositor.customer\_name = customer.customer\_name

WHERE account.balance = (SELECT MAX(balance) FROM account);

