Ross Bollinger

Dr. M. Owrang

28 February 2019

CSC-434

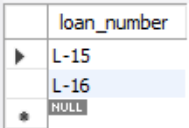
Homework #2

Retrieval Queries

1. **Find all loan numbers for loans made at the Perryridge branch with loan amounts greater than $1100.**

**Query:**  
select loan\_number from loan where amount > 1100

and branch\_name = 'Perryridge';

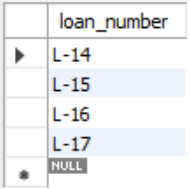
**Output:**

1. **Find the loan number of those loans with loan amounts between $1,000 and $1,500 (that is, >=$1,000 and <=$1,500).**

**Query:**

select loan\_number from loan where amount between 1000 and 1500;

**Output:**



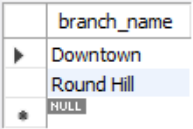
1. **Find the names of all branches that have greater assets than some branch located in Brooklyn.**  
    **Query:**

select distinct branch\_name from branch

where assets > some

(select assets from branch where branch\_city = 'Brooklyn');

**Output:**



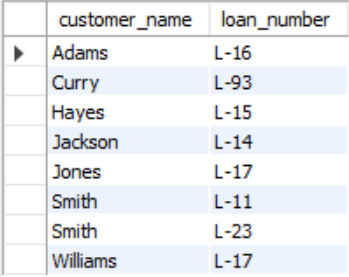
1. **Find the customer names and their loan numbers for all customers having a loan at some branch.**

**Query:**

select borrower.\* from borrower, loan

where borrower.loan\_number = loan.loan\_number;

**Output:**



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

**Query:**  
select customer\_name from depositor

union

select customer\_name from borrower;

**Output:**

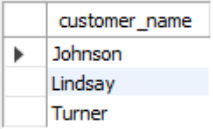
1. **Find all customers who have an account but no loan.**

**Query:**

select customer\_name from depositor

where customer\_name not in

(select customer\_name from borrower);

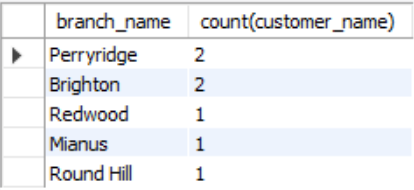
**Output:**  


1. **Find the number of depositors for each branch**

**Query:**  
select branch\_name, count(customer\_name) from account, depositor

where depositor.account\_number = account.account\_number

group by branch\_name;

**Output:**  


1. **Find the names of all branches where the average account balance is more than $500.**

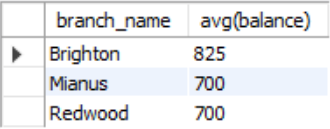
**Query:**

select branch\_name, avg(balance) from account

group by branch\_name

having avg(balance) > 500;

**Output:**



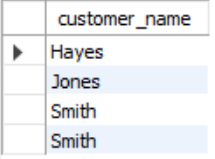
1. **Find all customers who have both an account and a loan at the bank.**

**Query:**  
select customer\_name from borrower

where customer\_name in

(select customer\_name from depositor);

**Output:**



1. **Find all customers who have a loan at the bank but do not have an account at the bank.**

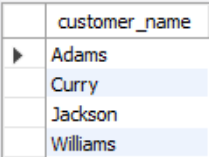
**Query:**

select customer\_name from borrower

where customer\_name not in

(select customer\_name from depositor);

**Output:**



1. **Find the names of all branches that have greater assets than all branches located in Horseneck. (using both non-nested and nested select statement)**

Could not figure out how to do the non-nested selection with this specific problem. Only did the nested. I really tried to figure it out, but nothing I tried actually worked.

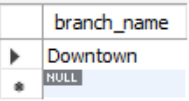
**Query:**

select distinct branch\_name from branch

where assets > all

(select assets from branch where branch\_city = 'Horseneck');

**Output:**



1. **1 query of your choice involving aggregate functions.**

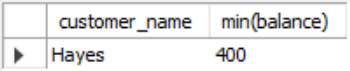
**Find the name of the borrower with the lowest balance in their account.**

**Query:**  
select borrower.customer\_name, min(balance) from borrower, account, depositor

where depositor.account\_number = account.account\_number

and depositor.customer\_name = borrower.customer\_name;

**Output:**



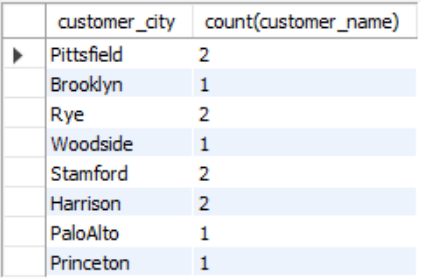
1. **1 query of your choice involving group by feature.  
     
   For each city, find the number of customers that live there.**

**Query:**

select customer\_city, count(customer\_name) from customer

group by customer\_city;

**Output:**

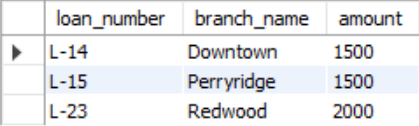


Insert Queries

1. **Create a HighLoan table with loan amount >=1500.  
     
   Query:**  
   create table if not exists HighLoan as

select \* from loan where amount >= 1500;

select \* from HighLoan;

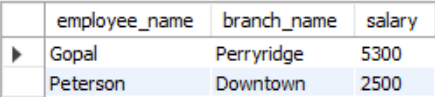
**Output:**  


1. **Create a HighSalaryEmployee table with employee having salary more than 2000.**  
     
   **Query:**  
   create table if not exists HighSalaryEmployee as

select \* from employee where salary > 2000;

select \* from HighSalaryEmployee;

**Output:**



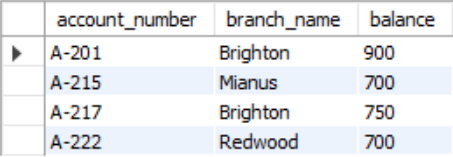
1. **1 more query (meaningful) of your choice on any table.**

**Create a HighBalance table of account holders with balances >= 700.**

**Query:**create table if not exists HighBalance as

select \* from account where balance >= 700;

select \* from HighBalance;

**Output:**  


Update Queries

1. **Increase all accounts with balances over $800 by 7%, all other accounts receive 8%.**

**Query:**

select \* from account;

update account

set balance = balance + (balance \* .07)

where balance > 800;

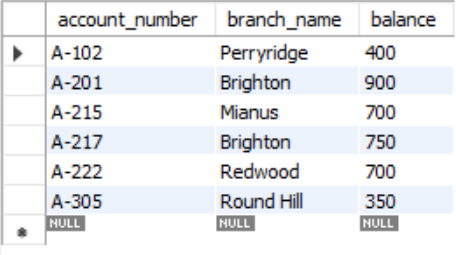
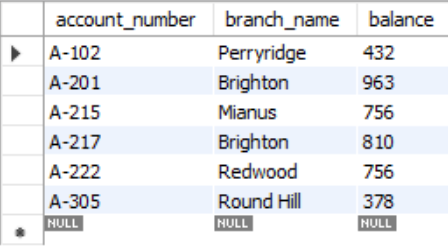
update account

set balance = balance + (balance \* .08)

where balance <= 800;

select \* from account;

**Output:**  
Before: After:

1. **Do 2 update queries, each involving 2 tables.**
2. **Change all A-101 account numbers of account holders and depositors to A-103**

**Query:**  
select \* from account;

select \* from depositor;

update account

set account\_number = 'A-103'

where account\_number = 'A-102';

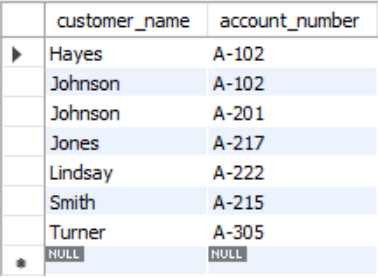
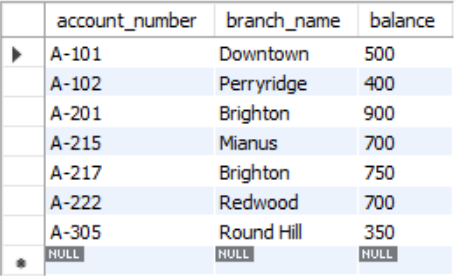
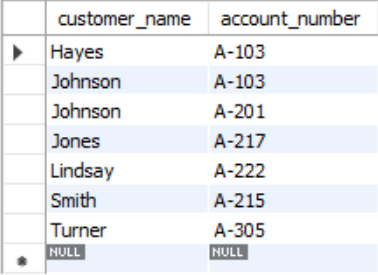
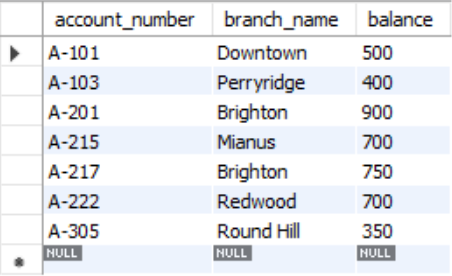
update depositor

set account\_number = 'A-103'

where account\_number = 'A-102';

select \* from account;

select \* from depositor;

**Output:**  
Before:  
After:  


1. **All accounts and employees at the Downtown branch were moved to the Brighton branch. This is because the Downtown branch is under maintenance, making the Brighton branch the only available branch in Brooklyn.**

**Query:**  
select \* from account;

select \* from employee;

update account

set branch\_name = 'Brighton'

where branch\_name = 'Downtown';

update employee

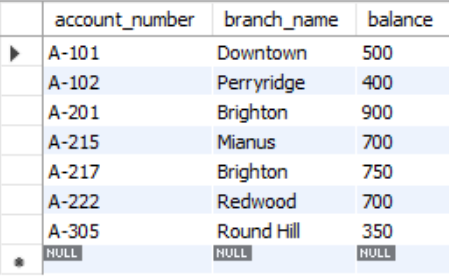
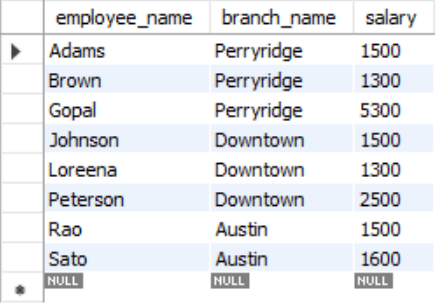
set branch\_name = 'Brighton'

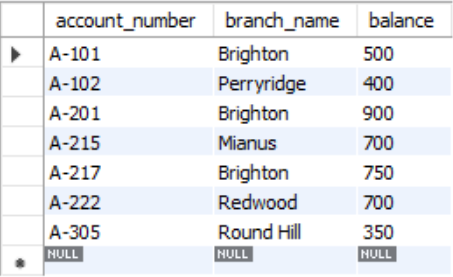
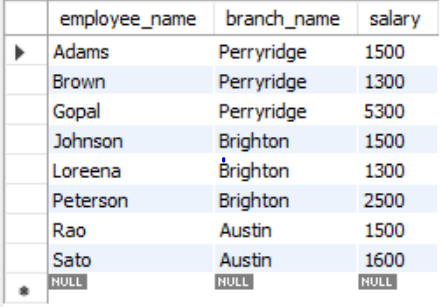
where branch\_name = 'Downtown';

select \* from account;

select \* from employee;

**Output:**  
Before:

After:  
 

1. **1 more update query of your choice on any table.**

**Decrease all assets by 4%.**

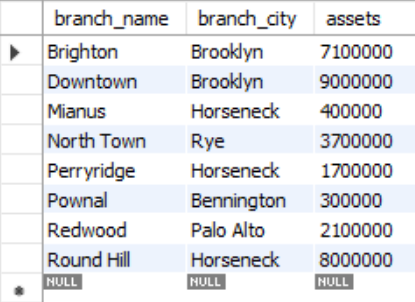
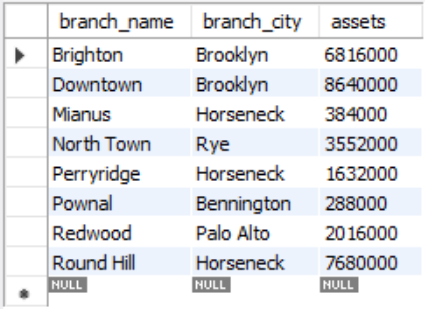
**Query:**  
select \* from branch;

update branch

set assets = assets - (assets \* .04);

select \* from branch;

**Output:**  
Before: After:

Delete Queries

1. **Delete the record of all accounts with balances below the average at the bank.**

**Query:**  
select \* from account;

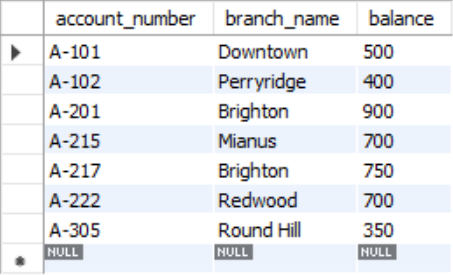
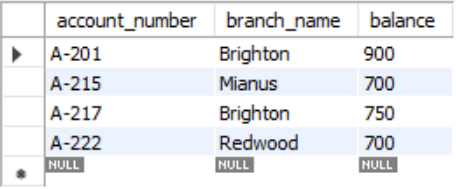
delete from account

where balance < (select \* from

(select avg(balance) from account) as t);

select \* from account;

**Output:**  
Before: After:

1. **Do 2 update queries, each involving 2 tables.**
2. **Delete the first account that has one of the most frequently occurring branch names in employee.**

**Query:**

select \* from account;

delete from account

where branch\_name =

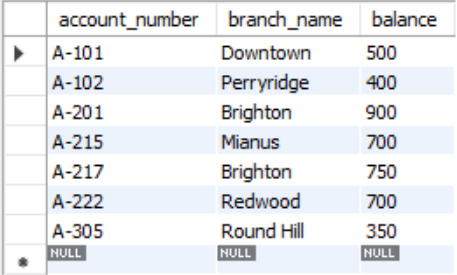
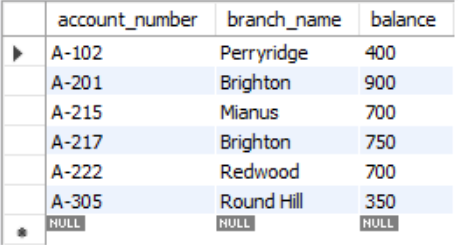
(select branch\_name from

(select max(num), branch\_name from

(select count(branch\_name) as num, branch\_name from account) as a) as b);

select \* from account;

**Output:**  
Before: After:

1. **Delete all customers who do not live in a city with a branch**  
     
   **Query:**  
   select \* from customer;

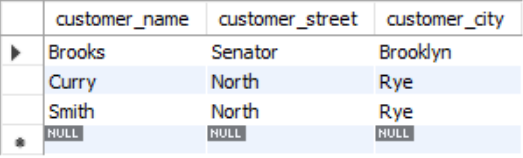
delete from customer

where customer\_city not in

(select branch\_city from branch);

select \* from customer;

**Output:**

Before: After:  
 

1. **1 more delete query of your choice from any table.**

**Delete all loans with transactions made at Perryridge.**

**Query:**select \* from loan;

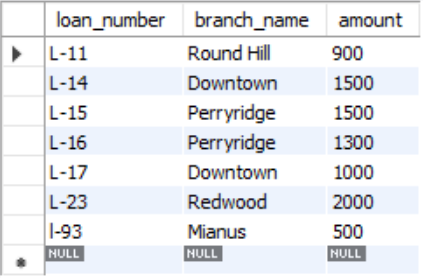
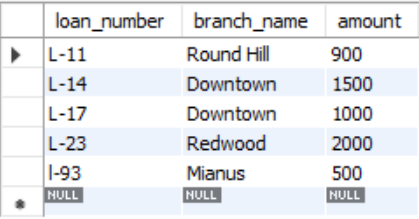
delete from loan

where branch\_name = 'Perryridge';

select \* from loan;

**Output:**

Before: After:

Views Queries

1. **A view consisting of branches and their customers.**

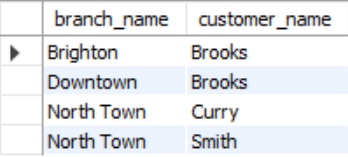
**Query:**

create view branchAndCustomer as

select branch\_name, customer\_name from customer, branch

where customer\_city = branch\_city;

select \* from branchAndCustomer;

**Output:**  


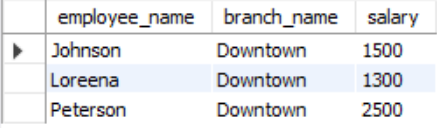
1. **Create a view of HQEmployee who work in downtown branch.**

**Query**:  
create view HQEmployee as

select \* from employee

where branch\_name = 'Downtown';

select \* from HQEmployee;

**Output:**  


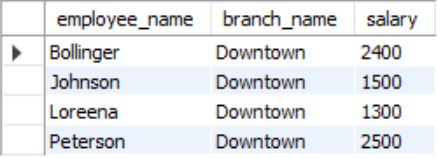
1. **Do one insert, delete, update, and select queries on HQEmployee view.**

**Insert. Insert a new HQ employee with name 'Bollinger' and salary 2400.**

**Query:**

insert into HQEmployee values('Bollinger', 'Downtown', 2400);

select \* from HQEmployee;

**Output:**

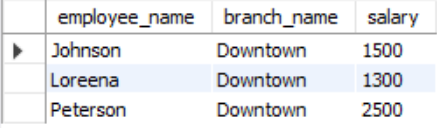
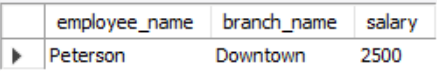
**Delete. Delete all HQ employees who have a salary that is less than 2000.**

**Query:**  
delete from HQEmployee

where salary < 2000;

select \* from HQEmployee;

**Output:**Before: After:

**Update. Increase all salaries of HQ employees with salaries less than 2000 by 10%.**

**Query:**  
update HQEmployee

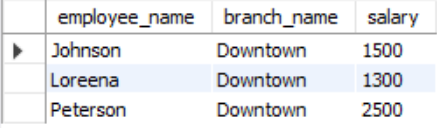
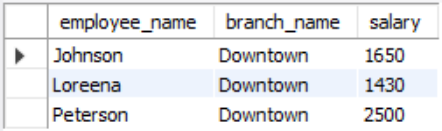
set salary = salary + (salary \* .1)

where salary < 2000;

select \* from HQEmployee;

**Output:**

Before: After:

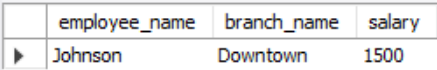
 

**Select. Select all HQ employees who have a loan.**

**Query:**

select distinct HQEmployee.\* from HQEmployee, depositor

where employee\_name = customer\_name;

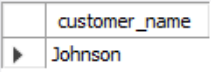
**Output:**  


Complex Queries

1. **1 select query involving 3 tables.  
     
   Find the names of all customers who are employees and have a loan.  
     
   Query:**select distinct customer.customer\_name from customer, employee, depositor

where customer.customer\_name = employee.employee\_name

and customer.customer\_name = depositor.customer\_name**;**

**Output:**  


1. **1 Delete query involving 3 tables.  
     
   Delete all branches that do not have an account of a customer that also has a loan.**

**Query:**select \* from branch;

delete from branch

where branch\_name not in

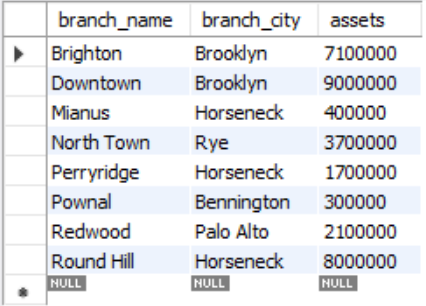
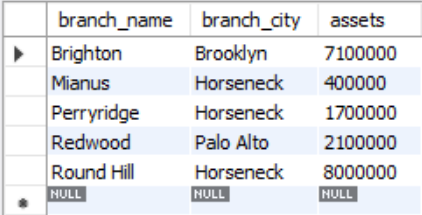
(select branch\_name from account

where account\_number in

(select account\_number from depositor));

select \* from branch;

**Output:**  
Before: After:

1. **1 Update query involving 3 tables.**

**Increase the salary of all employees with loans greater than 1000 by 20%.**

**Query:**

Select \* from employee;update employee

set salary = salary + (salary \* .2)

where employee\_name in

(select customer\_name from borrower

where loan\_number in

(select loan\_number from loan

where amount > 1000));

select \* from employee;

**Output:**  
Before: After:

