**COMP 3005  
Assignment #3   
Due: February 18**

**Instruction**

1. You should do the assignments independently. Copying is not allowed.
2. The assignment must be typed, completed on an individual basis, and submitted as a single Word/PDF file with your name as the filename to **brightspace**. Scanned handwritten documents *won’t* be accepted. Make sure your uploaded file can be opened and contain everything required.
3. It is based on the database you create in the first assignment where Lastname in Customer table is your last name. If your information is not shown correctly in the result, you will get 0 mark for the assignment.
4. You should directly do your assignment on this document and name the document with your last name followed by your first name so that it is easy for TAs.
5. You need to use either [Openstack](https://carleton.ca/scs/tech-support/scs-open-stack/openstack-technical-support/openstack-step-by-step-guide/) or [Oracle VM](https://git.scs.carleton.ca/downloads/CourseVirtualMachines/2022F-2023W/COMP3005-W23.ova) and SQLPLUS interface to Oracle DBMS for the SQL part of this assignment by entering the SQL query expressions, generating query results and putting the screenshots of the query together with the generated results on the assignment document. As we don’t’ have DRC and QBE interfaces to Oracle DBMS, you just put the DRC and QBE queries on the assignment document. If your SQL queries do not generate correct results, you can just provide correct results separately.

**Queries (120 marks)**

Use Domain Relational Calculus (DRC) and SQL to express the following 10 queries and use Query By Example (QBE) to express 1,2,3,5,6 based on the Bank-Customer database. For each question, including its number, two or three kinds of queries and the query results. Every query is 4 marks. Every query result is 2 marks.

**Bank**

|  |  |  |
| --- | --- | --- |
| **B#** | **Name** | **City** |
| B1 | England | London |
| B2 | America | New York |
| B3 | Royal | Toronto |
| B4 | France | Paris |

**Customer**

|  |  |  |  |
| --- | --- | --- | --- |
| **C#** | **Name** | **Age** | **City** |
| C1 | Adams | 20 | London |
| C2 | Blake | 30 | Paris |
| C3 | Clark | 25 | Paris |
| C4 | Lastname | 20 | Ottawa |
| C5 | Smith | 30 | Toronto |

**Account**

|  |  |  |  |
| --- | --- | --- | --- |
| **C#** | | **B#** | **Balance** |
| C1 | B1 | | 1000 |
| C1 | B2 | | 2000 |
| C1 | B3 | | 3000 |
| C1 | B4 | | 4000 |
| C2 | B1 | | 2000 |
| C2 | B2 | | 3000 |
| C2 | B3 | | 4000 |
| C3 | B1 | | 3000 |
| C3 | B2 | | 4000 |
| C4 | B1 | | 4000 |
| C4 | B2 | | 5000 |

1. Get the name of the bank that Lastname banks.

DRC> {BankN | (exists B#, C#) (Customer(C#, “Beg”, \_, \_) and Account(C#, B#, \_) and Bank(B#, BankN,\_))};

1. Get the name of the customer who banks in Royal bank.

DRC> {CustN | (exists B#, C#) (Bank(B#, “Royal”,\_) and Account(C#, B#, \_) and Customer(C#, CustN,\_,\_ ) )};

SQL:

Text

Description automatically generated

1. Get the name of the customer who has an account with balance less than 3000.

DRC> {CustB | (exists B#, C#, Balance) (Bank(B#,\_,\_) and Account(C#, B#, Balance) and Customer(C,N,\_,\_) and Balance > 3000)};

SQL:

Graphical user interface, text, application

Description automatically generated

1. Get the customer name/bank name pairs such that the indicated customer has an account in the indicated bank.

DRC> {CustN, BankN | (exists Balance) (Account (CustN, CustB, Balance))};

SQL:

Text

Description automatically generated

1. Get the name of the customer who does not have any bank account.

DRC> {CustN | (exists C#) (Customer(C#, Name,\_,\_) and not (exists C#) Account (C#,\_,\_))};

SQL:

Text

Description automatically generated

1. Get the name of the customer who has an account in every bank.

DRC> {CustN | (exists C#) (Customer(C#, Name, \_,\_) and (exists B#) Bank(B#,\_,\_) and

(forall C#) (not Account(C#,\_,\_) or Bank(C#, Name,\_,\_)))}

SQL:

Text

Description automatically generated

1. Get the name of the customer who has an account in every bank except France Bank.

{CustN | (exists C#)(Customer(C#,CName,\_,\_) and

(forall C#)

(if(exists B#) (Bank(B#, BName,\_) and Account(C#,B#,\_) and BName! = ‘France’ then Account(C#,B#,\_))

Or

(if(exists B#) (Bank(B#, BName,\_) and Account(C#,B#,\_) and BName = ‘France’ then not Account(C#,B#,\_)))};

SQL:

Text

Description automatically generated

1. Get the name of the customer who has an account in every bank that Clark banks.

{N | (exists C1#, B#, C#) (Customer(C1#,Name,\_,\_) and Bank(B#,\_,\_) and Name != ‘Clark’ and Customer(C#,’Clark’,\_,\_) and

(forall C#)

(if Account(C#,B#,\_) then Account(C1#,B#,\_)))};

SQL:

Text

Description automatically generated

1. Get the name of the customer who banks only in the banks that Clark banks.

{N | (exists C1#,B#,C#) (Customer(C1#,Name,\_,\_) and Bank(B#,\_,\_) and Name != ‘Clark’ and Customer(C#,’Clark’,\_,\_) and

(forall C#)

(if Account(C#,B#,\_) then Account(C1#,B#,\_)) and

(if not Account(C#,B#,\_) then not Account(C1#,B#,\_)))};

SQL:

Couldn’t figure this one out properly

Text

Description automatically generated

1. Get the name of the customer who banks in more than two banks.

{N | (exists C#,B#) (Customer(C,N,\_,\_) and Account(C1#,B#,\_) and Account(C2#,B#,\_) and Account(C3#,B#) and C1# != C2# and C2# != C3# and C1# != C3#)};

Text

Description automatically generated