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 or Oracle VM 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,_))};

2. 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:

```
SQL> Select Customer.C#, Customer.Name, Account.Balance, Bank.Name
  2  from Customer, Bank, Account
  3  where Bank.Name = 'Royal' and Customer.C# = Account.C# and Bank.B# = Account.B#;

C#         NAME          BALANCE NAME
---------- ---------- ---------- ----------
C1         Adams            3000 Royal
C2         Blake            4000 Royal

SQL>
```

3. 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:

```
SQL> Select distinct Customer.Name
  2  from Customer, Account
  3  Where Account.Balance<3000 and Customer.C#=Account.C#;

NAME
----------
Adams
Blake
```

4. 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:

```
SQL> Select Customer.C#, Customer.Name, Account.Balance, Bank.Name
  2  from Customer, Bank, Account
  3  where Customer.C# = Account.C# and Bank.B# = Account.B#;

C#         NAME           BALANCE NAME
---------- ---------- ---------- ----------
C1         Adams            4000 France
C1         Adams            3000 Royal
C1         Adams            2000 America
C1         Adams            1000 England
C2         Blake            4000 Royal
C2         Blake            3000 America
C2         Blake            2000 England
C3         Clark            4000 America
C3         Clark            3000 England
C4         Beg              4000 England

10 rows selected.
```

5. 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:

```
SQL> select Customer.C#, Customer.Name
  2  from Customer
  3  where C# not in (select Account.C#
  4  from Account);

C#         NAME
---------- ----------
C5         Smith
```

6. 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:

```
SQL> select Customer.C#, Customer.Name
  2  from Customer
  3  where not exists
  4  (select Bank.B# from Bank
  5  where not exists
  6  (select Account.B#
  7  from Account
  8  where Account.C# = Customer.C# and Bank.B# = Account.B#)
  9  );

C#         NAME
---------- ----------
C1         Adams
```

7. 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:

```
SQL> Select Customer.Name, Customer.C#
  2  from Customer
  3  where not exists
  4  (select Bank.B#
  5  from Bank
  6  where Bank.Name != 'France'
  7  and not exists
  8  (select Account.B#
  9  from Account
 10  where Account.B# = Bank.B# and Customer.C# = Account.C#)
 11  )
 12  and C# in (select Account.C#
 13  from Account
 14  group by C# having count(*)=
 15  (select count(*)
 16  from Bank)-1);

NAME       C#
---------- ----------
Blake      C2
```

8. 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:

```
SQL> Select Customer.C#, Customer.Name
  2  from Customer, Bank, Account
  3  where Account.C# = Customer.C# and Bank.B# = Account.B#
  4  and Customer.Name != 'Clark' and Bank.Name in
  5  (select Bank.Name
  6  from Bank, Account, Customer
  7  where Customer.C# = Account.C# and
  8  Bank.B# = Account.B# and Customer.Name = 'Clark');

C#         NAME
---------- ----------
C1         Adams
C1         Adams
C2         Blake
C2         Blake
C4         Beg
```

9. 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

```
SQL> Select distinct Customer.C#, Customer.Name
  2  from Customer, Bank, Account
  3  where Account.C# = Customer.C# and Bank.B# = Account.B#
  4  and Customer.Name = 'Clark' and Bank.Name in
  5  (select Bank.Name
  6  from Bank, Account, Customer
  7  where Bank.B# = Account.B# and
  8  Customer.C# = Account.C#
  9  and Customer.Name = 'Clark') and
 10  Customer.C# in
 11  (select Account.C#
 12  from Account
 13  group by C# having count(*) =
 14  (select count(*)
 15  from Account, Customer
 16  where Account.C# = Customer.C# and
 17  Customer.Name = 'Clark'));

C#         NAME
---------- ----------
C3         Clark
```

10. 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#)};

```
SQL> Select Customer.C#
  2  from Customer
  3  where C# in
  4  (select Account.C#
  5  from Account
  6  group by C# having count(*)>2);

C#
----------
C1
C2
```