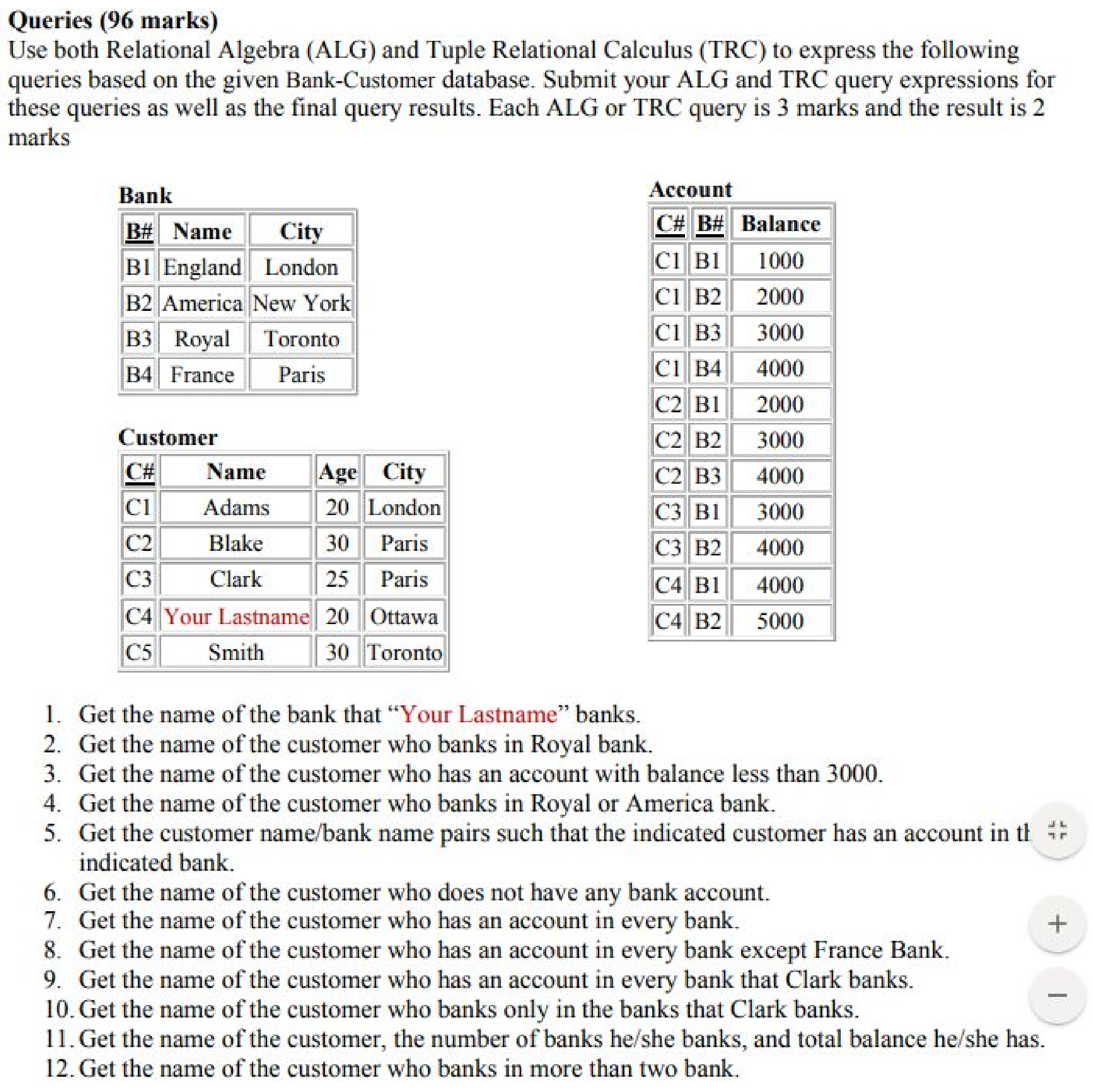
COMP 3005
Assignment #2
Krystian Wojcicki
101001444



Queries (96 marks)
Use both Relational Algebra (ALG) and Tuple Relational Calculus (TRC) to express the following queries based on the given Bank-Customer database. Submit your ALG and TRC query expressions for these queries as well as the final query results. Each ALG or TRC query is 3 marks and the 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	Your 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 "Your Lastname" banks.
2. Get the name of the customer who banks in Royal bank.
3. Get the name of the customer who has an account with balance less than 3000.
4. Get the name of the customer who banks in Royal or America bank.
5. Get the customer name/bank name pairs such that the indicated customer has an account in th indicated bank.
6. Get the name of the customer who does not have any bank account.
7. Get the name of the customer who has an account in every bank.
8. Get the name of the customer who has an account in every bank except France Bank.
9. Get the name of the customer who has an account in every bank that Clark banks.
10. Get the name of the customer who banks only in the banks that Clark banks.
11. Get the name of the customer, the number of banks he/she banks, and total balance he/she has.
12. Get the name of the customer who banks in more than two bank.

1)
ALG> t1 := select Name='Wojcicki' (Customer);
t2 := project B# (t1 njoin Account);
t3 := t2 njoin Bank;
project Name (t3);

```
TRC> { B.Name | B in Bank and (exists A in Account, C in Customer) ( C.C# = A.C# and A.B# =
B.B# and C.Name = ‘Wojcicki’)};
```

```
Name
England
America
```

2)
```
ALG> t1 := select Name='Royal' (Bank);
t2 := project C# (t1 njoin Account);
t3 := t2 njoin Customer;
project Name (t3);
```

```
TRC> { C.Name | C in Customer and (exists A in Account, B in Bank) ( C.C# = A.C# and A.B# =
B.B# and B.Name = ‘Royal’) };
```

```
Name
Adams
Blake
```

3)
```
ALG> t1 := (select Balance < 3000 (Account)) njoin Customer;
project Name (t1);
```

```
TRC> { C.Name | C in Customer and (exists A in Account) ( C.C# = A.C# and A.Balance <
3000)};
```

```
Name
Adams
Blake
```

4)
```
ALG> t1 := select Name='Royal' or Name='America' (Bank);
t2 := project C# (t1 njoin Account);
t3 := (t2 njoin Customer);
project Name (t3);
```

```
TRC>  { C.Name | C in Customer and (exists A in Account, B in Bank) ( C.C# = A.C# and A.B#
= B.B# and (B.name = ‘Royal’ or B.Name = ‘America’)) };
```

```
Name
Adams
Blake
```

```
Clark
Wojcicki
```

5)

```
ALG> t1 := rename  Name to cname (Customer);
t2 := rename Name to bname (Bank);
t3 := (t1 njoin Account) njoin t2;
project cname, bname (t3);
```

```
TRC> { C.Name, B.Name | C in Customer and B in Bank and (exists A in Account)( A.C# =
C.C# and B.B# = A.B#)};
```

```
cname, bname
Adams England
Adams America
Adams Royal
Adams France
Blake England
Blake America
Blake Royal
Clark England
Clark America
Wojcicki England
Wojcicki America
```

6)

```
ALG> t1 := project Name (Customer);
t2 := project Name (Customer njoin Account);
t1 minus t2;
```

```
TRC> { C.Name | C in Customer and not (exists A in Account) (A.C# = C.C#) };
```

```
Name
Smith
```

7)

```
ALG> t1 := project C#,B# (Account);
t2 := project B# (Bank);
t3 := t1 divideby t2;
project Name (t3 njoin Customer);
```

```
TRC> { C.Name | C in Customer and (forall B in Bank)(exists A in Account) (B.B# = A.B# and
A.C# = C.C#) };

Name
Adams
```

8)

```
ALG> t1 := project  C#,B# (Account);
t2 := project B# (select Name != 'France' (Bank));
t3 := t1 divideby t2;
t4 := project C# (select Name = 'France' (Bank njoin Account));
t5 := t3 minus t4;
project Name (Customer njoin t5);

TRC> { C.Name | C in Customer and (forall B in Bank)
(( B.Name = ‘France’ and not (exists A in Account)(A.B# = B.B# and C.C# = A.C#)) or
( B.Name != ‘France’ and (exists A in Account)(A.B# = B.B# and C.C# = A.C#)))};

Name
Blake
```

9)

```
ALG> t1 := project C# (select Name='Clark' (Customer));
t2 := ((project C#,B# Account) / t1);
t3 := ((project C#,B# Account) / t2);
t4 := select Name !='Clark' (t3 njoin Customer);
project Name (t4);

TRC>  { C.Name | C in Customer and C.Name != ‘Clark’ and (exists C1 in Customer)(C1.Name
= ‘Clark’ and (forall B in Bank)
(((exists A in Account)(A.B# = B.B# and C1.C# = A.C# ) and
 (exists A1 in Account)(A1.C# = B.B# and A1.C# = C.C#))
 or
not (exists A in Account)( C1.C# = A.C# and A.B# = B.B#))) };

Name
Adams
Blake
Wojcicki
```

10)

```
ALG> t1 := project C#,B# (Account);
t2 := project B# ((select Name=’Clark’ (Customer)) njoin t1);
```

```
t3 := t1 divideby t2;
t4 := (project B# Bank) minus t2;
t5 := (t4 njoin Account) njoin Customer;
t6 := project Name (t5);
t7 := project Name (select Name != 'Clark' (t3 njoin Customer))
t7 minus t6;
```

```
TRC> { C.Name | C in Customer and C.Name != ‘Clark’ and
(exists C1 in Customer)(C1.Name = ‘Clark’ and (forall B in Bank)
(((exists A in Account)(A.C# = C1.C# and A.B# = B.B#) and
   (exists A1 in Account)(A1.C# = C.C# and A1.B# = B.B#))
or
(not (exists A in Account)(A.C# = C1.C# and A.B# = B.B#) and
not (exists A1 in Account)(A1.C# = C.C# and A1.B# = B.B#))))};
```

```
Name
Wojcicki
```

11)

```
ALG> Aggregate Name, count(B#), sum(Balance) (Customer njoin Account);
```

```
TRC> { C.Name, count(A.b#), sum(A.Balance) | C in Customer and A in Account and C.C# =
A.C# };
```

```
Name  Count(B#)    Sum(Balance)
Adams 4             10000
Blake   3            9000
Clark   2           7000
Kuang 2             9000
```

12)

```
ALG> t1(Name, count) := aggregate Name; count(B#) (Customer njoin Account);
t2 := select count > 2 (t1);
project Name (t2);
```

```
TRC> { C.Name | C in Customer and (exists A in Account)(count(A.B#) > 2 and A.C# = C.C#)};
```

```
Name
Adams
Blake
```