Advanced Database Systems

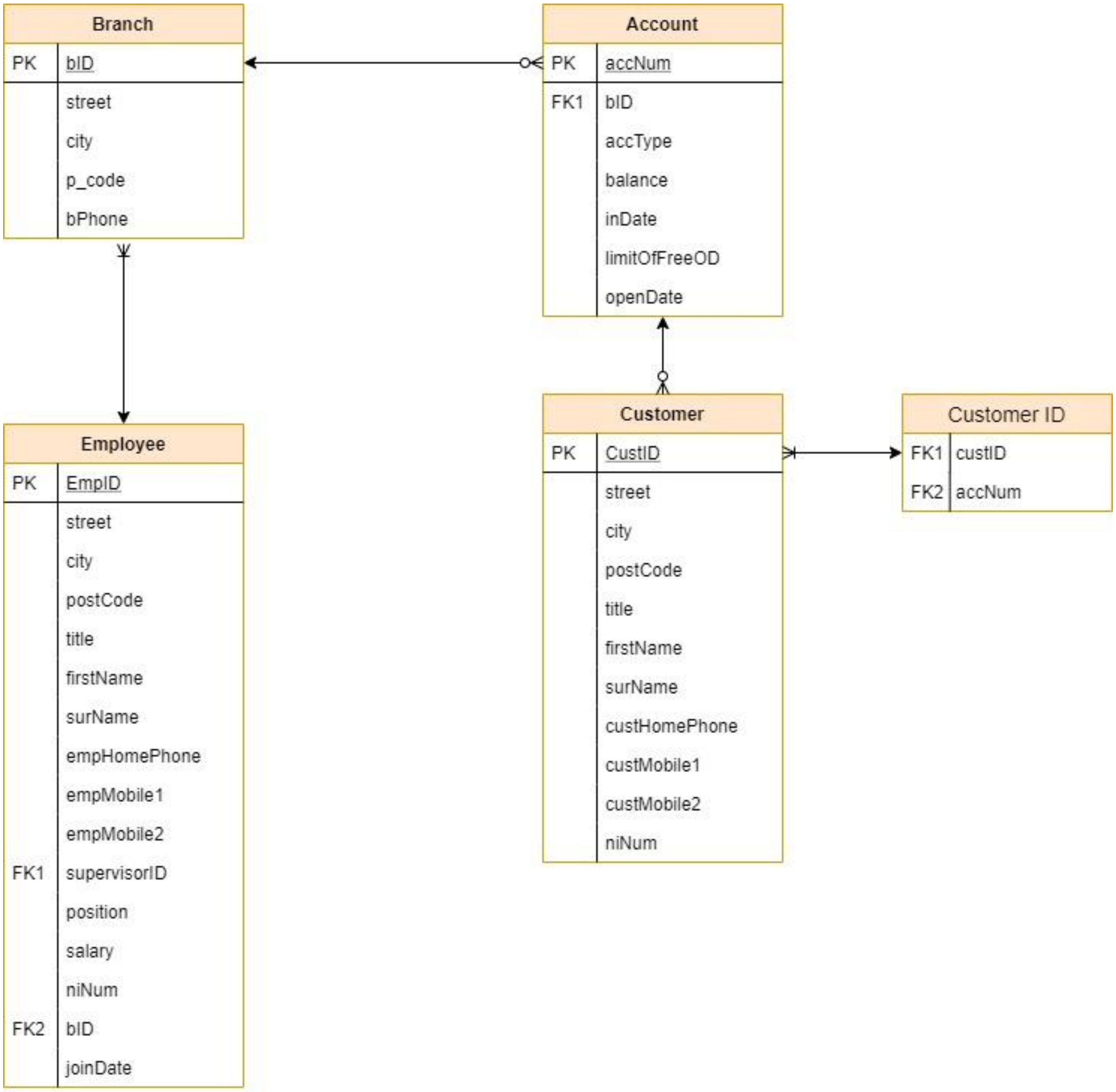
SET09107

Coursework

Bogoslava Dyankova

40323952@live.napier.ac.uk

Task 1. Draw an ER diagram corresponding to the relational database schema and the scenario



Branch	
PK	bID
	street
	city
	p_code
	bPhone

Account	
PK	accNum
FK1	bID
	accType
	balance
	inDate
	limitOfFreeOD
	openDate

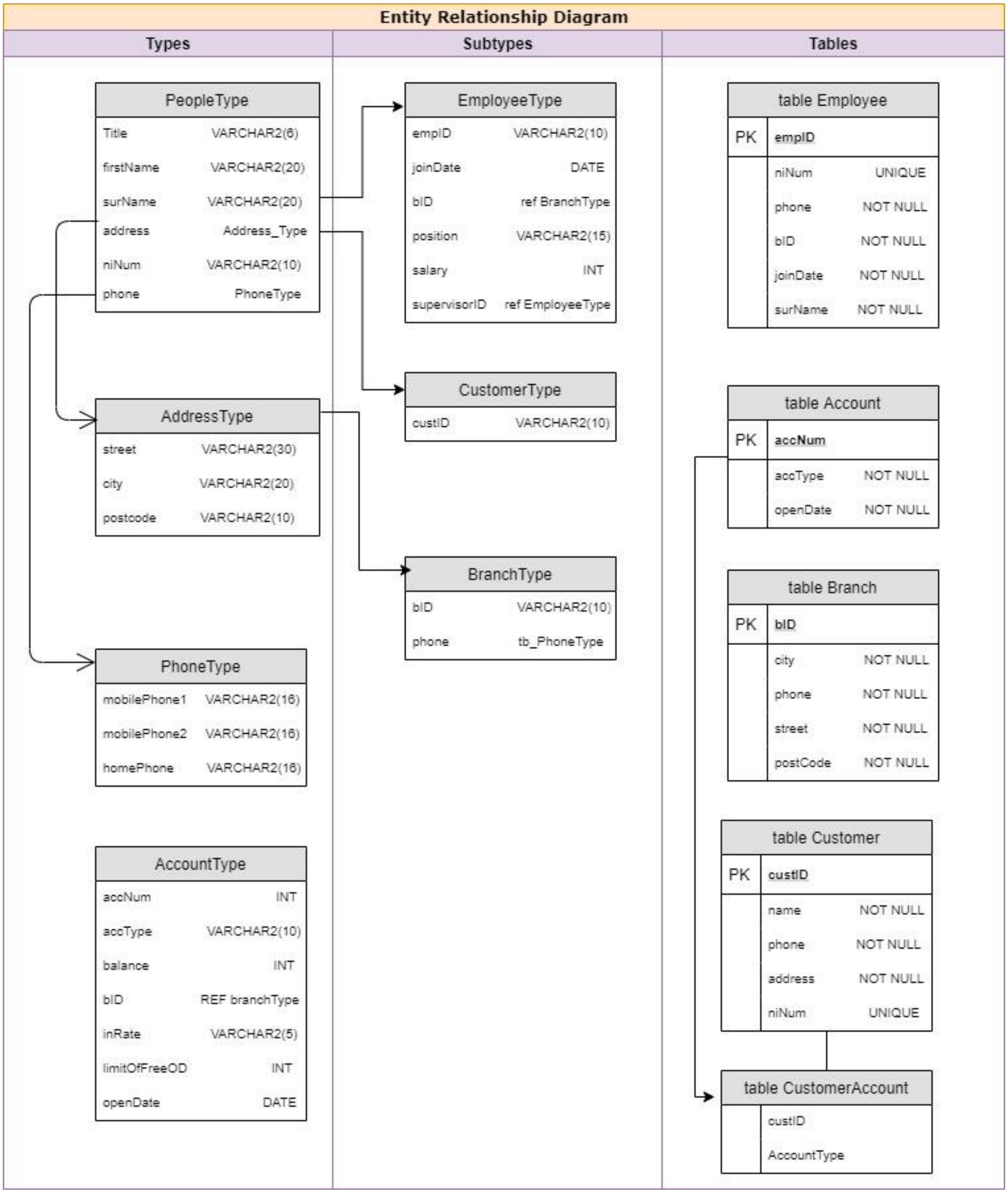
Employee	
PK	EmpID
	street
	city
	postCode
	title
	firstName
	surName
	empHomePhone
	empMobile1
	empMobile2
FK1	supervisorID
	position
	salary
	niNum
FK2	bID
	joinDate

Customer	
PK	CustID
	street
	city
	postCode
	title
	firstName
	surName
	custHomePhone
	custMobile1
	custMobile2
	niNum

Customer ID	
FK1	custID
FK2	accNum

Task 2. The purpose of this task is to analyze and re-design the database in order to have a bigger scope of the semantics of the application without losing the semantics of the previously implemented application, using object-relational features. For this, I created a new diagram (seen below), which I later on used to structure my code.

- Design of the new database



- Design description

  The design has been recreated by expanding the existing **types**, creating new **sub-types** and **tables**. It now takes an object-relational approach, using 4 object types, 3 sub-types and 5 tables. The types and sub-types are created first, after which they are used to create the tables, containing different properties. The properties are inherited by the tables, using the type from which it is obtained. Object types help promote a more streamlined code by preventing the existence of duplicate data. As an example, the People Type has two sub-types (Employee Type and Customer Type), which inherit the properties from the main type.

To include more applicable data in the tables, I used **references**. A reference is used when a table needs to use the data from another table and by inquiring the data with a reference, a 'join' function is not needed. Because of this, the queries are more optimized. A good example for this in the redesigned database is the Customer Account Table, referenced both towards Customer Table and Account Table as the Customer Account table needs the already inserted data in the other two tables.

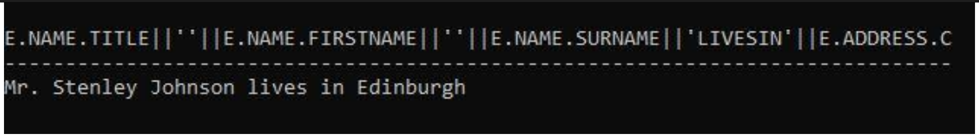
To prevent circumstances with null values, **nested tables** were used in the database. Nested tables are tables inside tables, also known as collections. As an example, the phone type is a nested table. A phone number is needed for customers, employees and branches, where some of them can have more than one phone in the database. Nested tables forbid empty values to be added to the database, such as a customer not having a phone number since it is a necessity to be able to be contacted if there are any problems. Another positive aspect of nested tables is that they don't have a limit on the number of entries.

Constraints were used in all the tables in the database in order to prevent the insertion of false data. They ensure there are no empty values added to the database, allowing a more efficient data management. Constraints are also used in order to apply a primary key if needed.

Task 3. SQL queries (input and output

a) Employees whose first name includes 'st'and live in Edinburgh

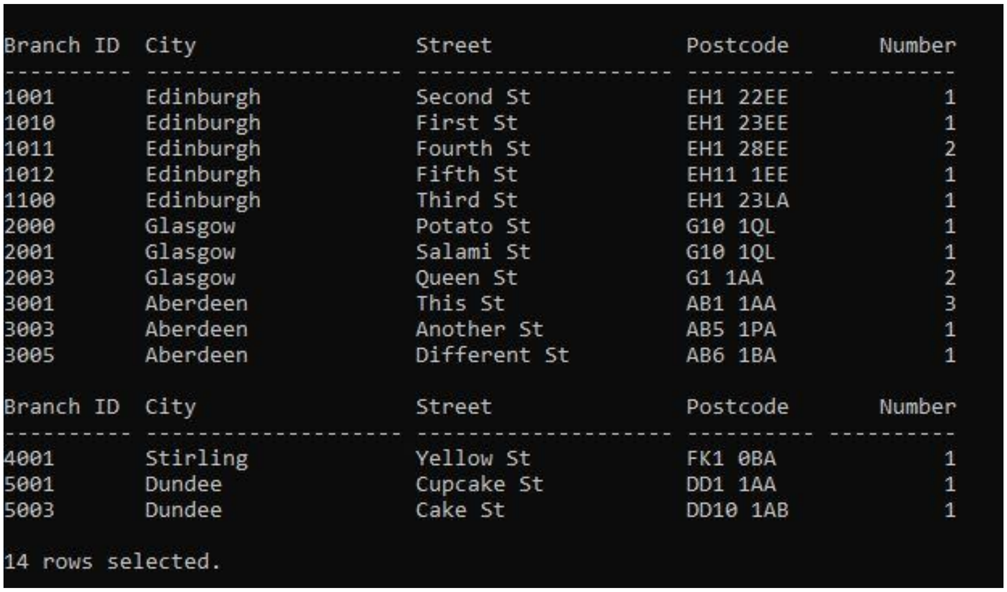
```
select e.name.title || ' ' ||
e.name.firstName || ' ' ||
e.name.surName || ' lives in ' ||
e.address.city
from tb_Employee e
where e.name.firstName like '%st%' or e.name.firstName like 'St%' and e.address.city = 'Edinburgh';
```



```
E.NAME.TITLE||''||E.NAME.FIRSTNAME||''||E.NAME.SURNAME||'LIVESIN'||E.ADDRESS.C
--------------------------------------------------------------------------------
Mr. Stenley Johnson lives in Edinburgh
```

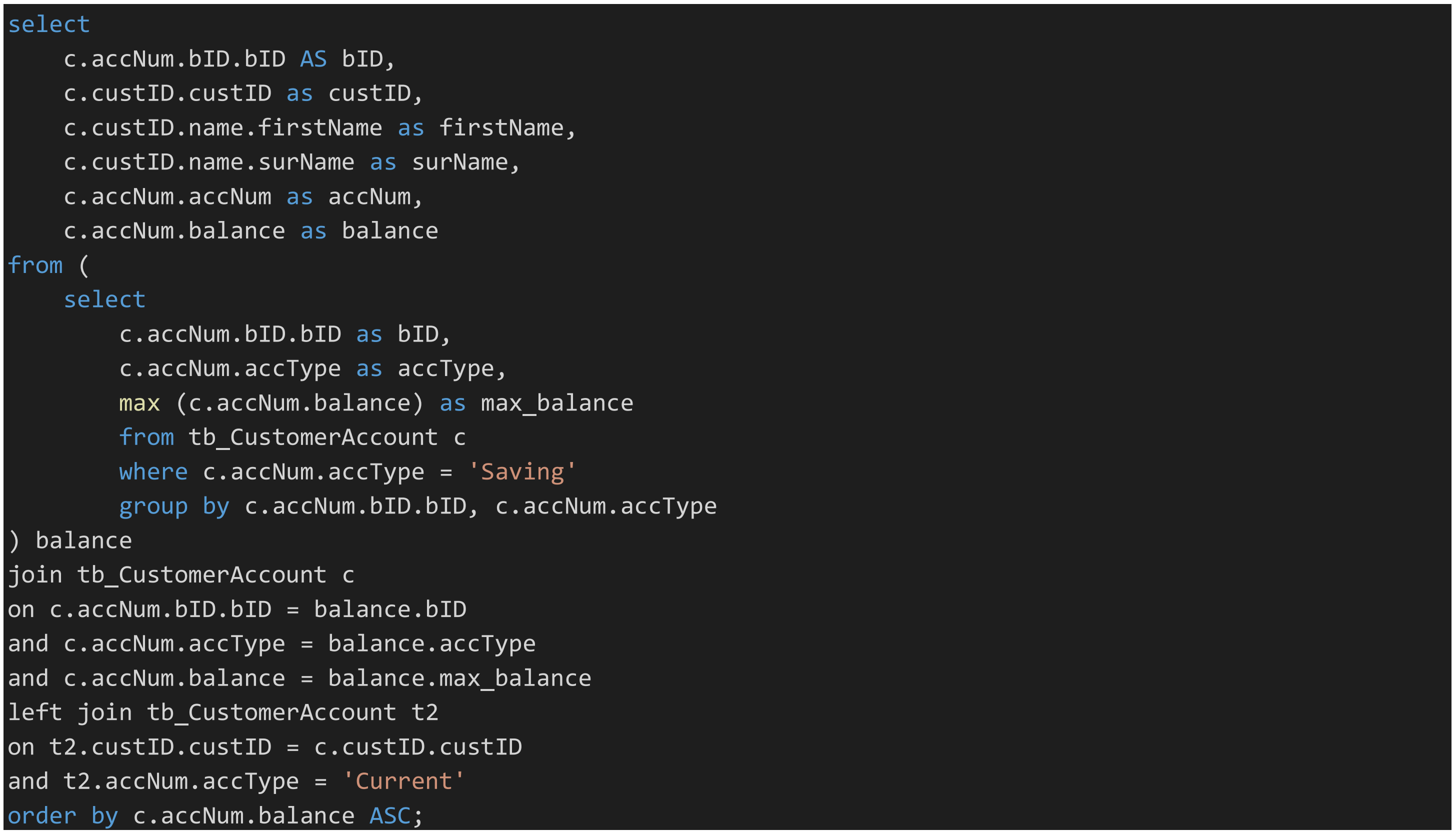
b) Number of saving accounts in each branch

```
select
 a.bID.bID as "Branch ID",
 a.bID.city as "City",
a.bID.street as "Street",
a.bID.postCode as "Postcode",
count(a.accType) as "Number"
from tb_Account a
where accType = 'Saving'
group by a.bID.bID, a.bID.city, a.bID.street, a.bID.postCode
order by a.bID.bID ASC;
```

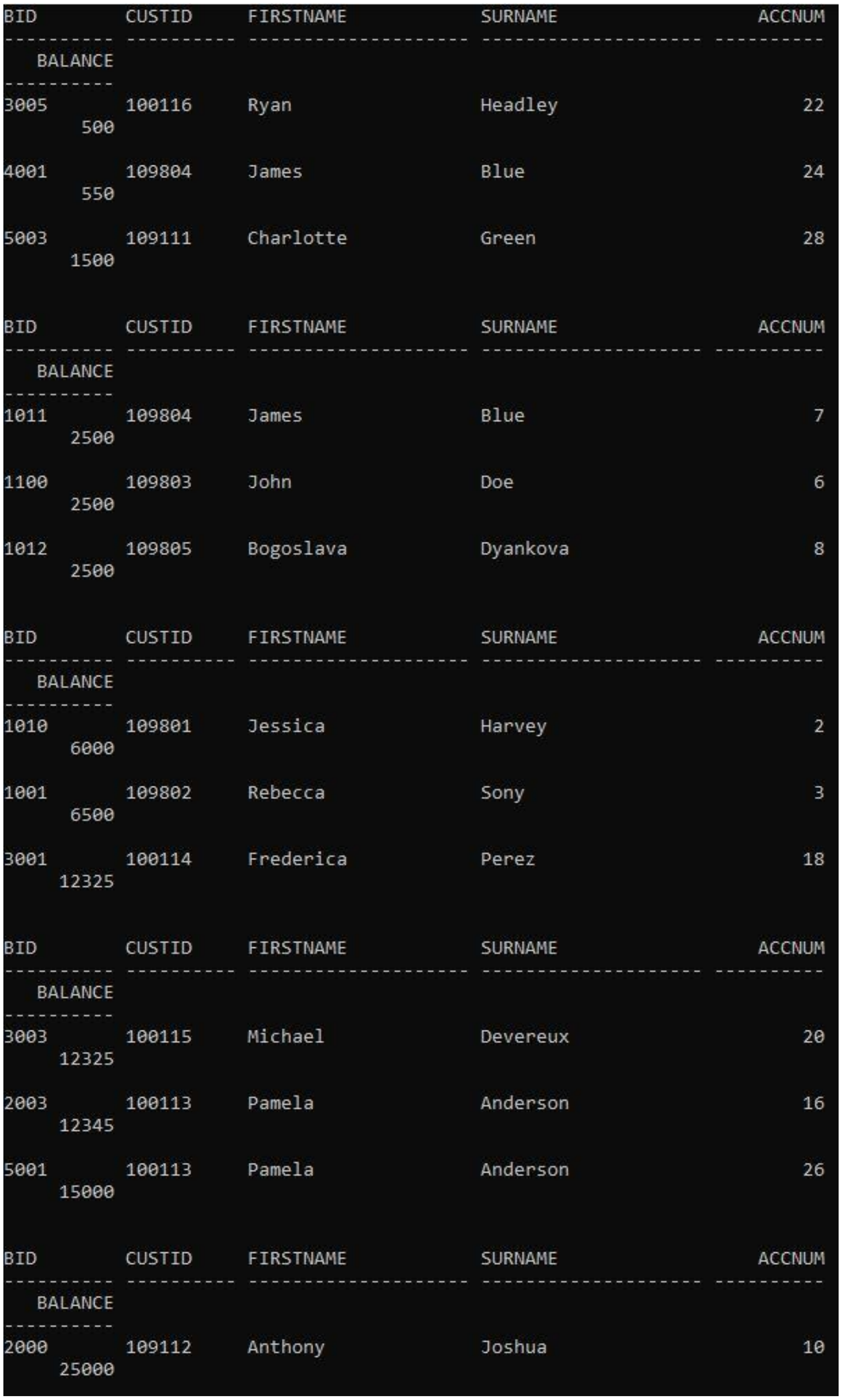


```
Branch ID  City                 Street               Postcode       Number
---------- -------------------- -------------------- ---------- ----------
1001       Edinburgh            Second St            EH1 22EE            1
1010       Edinburgh            First St             EH1 23EE            1
1011       Edinburgh            Fourth St            EH1 28EE            2
1012       Edinburgh            Fifth St             EH11 1EE            1
1100       Edinburgh            Third St             EH1 23LA            1
2000       Glasgow              Potato St            G10 1QL             1
2001       Glasgow              Salami St            G10 1QL             1
2003       Glasgow              Queen St             G1 1AA              2
3001       Aberdeen             This St              AB1 1AA             3
3003       Aberdeen             Another St           AB5 1PA             1
3005       Aberdeen             Different St         AB6 1BA             1

Branch ID  City                 Street               Postcode       Number
---------- -------------------- -------------------- ---------- ----------
4001       Stirling             Yellow St            FK1 0BA             1
5001       Dundee               Cupcake St           DD1 1AA             1
5003       Dundee               Cake St              DD10 1AB            1

14 rows selected.
```

c) Customers with highest balance in savings account

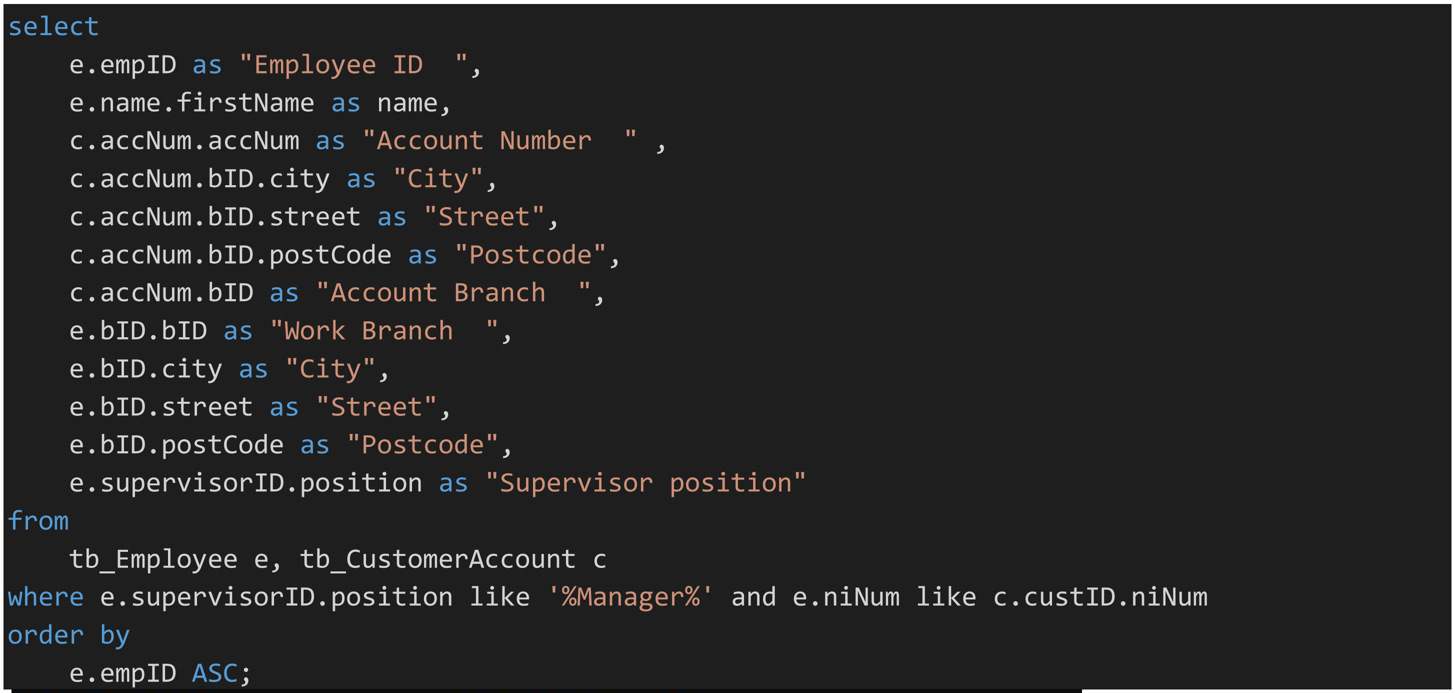


```
select
    c.accNum.bID.bID AS bID,
    c.custID.custID as custID,
    c.custID.name.firstName as firstName,
    c.custID.name.surName as surName,
    c.accNum.accNum as accNum,
    c.accNum.balance as balance
from (
    select
        c.accNum.bID.bID as bID,
        c.accNum.accType as accType,
        max (c.accNum.balance) as max_balance
        from tb_CustomerAccount c
        where c.accNum.accType = 'Saving'
        group by c.accNum.bID.bID, c.accNum.accType
) balance
join tb_CustomerAccount c
on c.accNum.bID.bID = balance.bID
and c.accNum.accType = balance.accType
and c.accNum.balance = balance.max_balance
left join tb_CustomerAccount t2
on t2.custID.custID = c.custID.custID
and t2.accNum.accType = 'Current'
order by c.accNum.balance ASC;
```

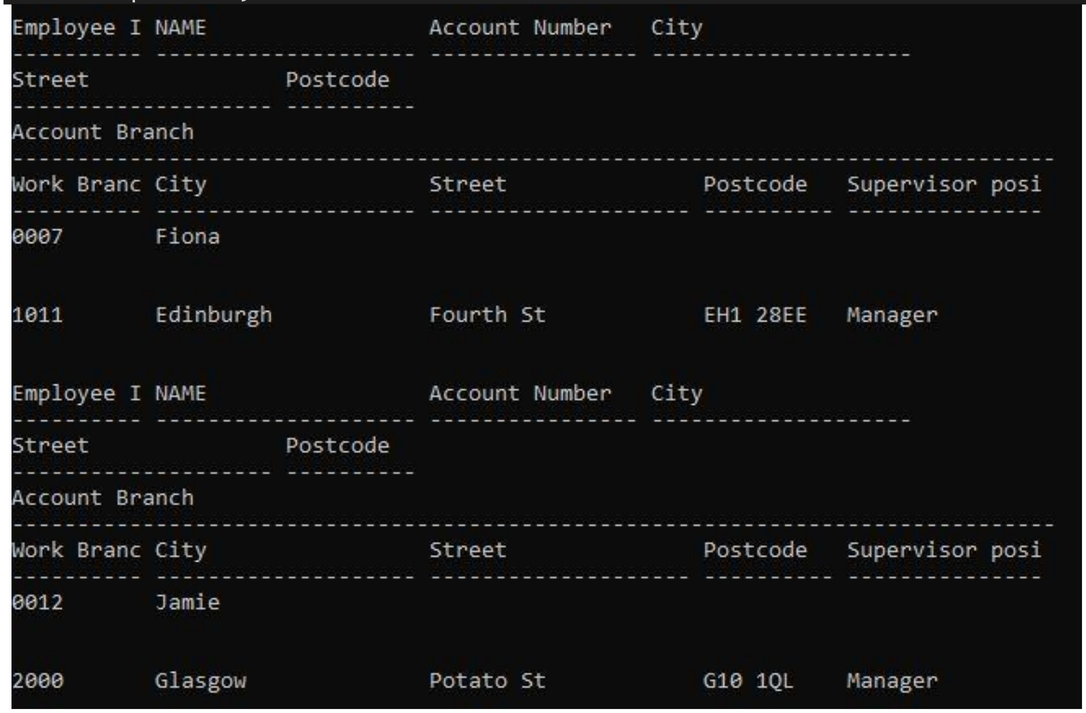


```
BID        CUSTID     FIRSTNAME            SURNAME                  ACCNUM
---------- ---------- -------------------- -------------------- ----------
   BALANCE
----------
3005       100116     Ryan                 Headley                      22
       500

4001       109804     James                Blue                         24
       550

5003       109111     Charlotte            Green                        28
      1500


BID        CUSTID     FIRSTNAME            SURNAME                  ACCNUM
---------- ---------- -------------------- -------------------- ----------
   BALANCE
----------
1011       109804     James                Blue                          7
      2500

1100       109803     John                 Doe                           6
      2500

1012       109805     Bogoslava            Dyankova                      8
      2500


BID        CUSTID     FIRSTNAME            SURNAME                  ACCNUM
---------- ---------- -------------------- -------------------- ----------
   BALANCE
----------
1010       109801     Jessica              Harvey                        2
      6000

1001       109802     Rebecca              Sony                          3
      6500

3001       100114     Frederica            Perez                        18
     12325


BID        CUSTID     FIRSTNAME            SURNAME                  ACCNUM
---------- ---------- -------------------- -------------------- ----------
   BALANCE
----------
3003       100115     Michael              Devereux                     20
     12325

2003       100113     Pamela               Anderson                     16
     12345

5001       100113     Pamela               Anderson                     26
     15000


BID        CUSTID     FIRSTNAME            SURNAME                  ACCNUM
---------- ---------- -------------------- -------------------- ----------
   BALANCE
----------
2000       109112     Anthony              Joshua                       10
     25000
```

d) Employees with supervisors and account in the bank

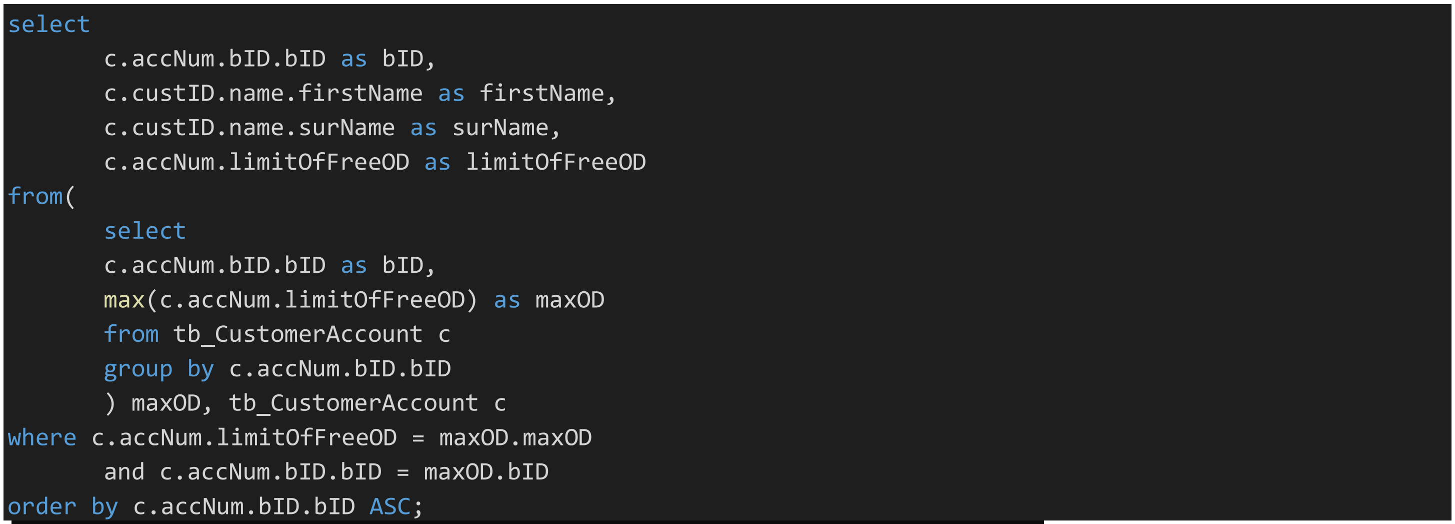


```sql
select
    e.empID as "Employee ID  ",
    e.name.firstName as name,
    c.accNum.accNum as "Account Number  " ,
    c.accNum.bID.city as "City",
    c.accNum.bID.street as "Street",
    c.accNum.bID.postCode as "Postcode",
    c.accNum.bID as "Account Branch  ",
    e.bID.bID as "Work Branch  ",
    e.bID.city as "City",
    e.bID.street as "Street",
    e.bID.postCode as "Postcode",
    e.supervisorID.position as "Supervisor position"
from
    tb_Employee e, tb_CustomerAccount c
where e.supervisorID.position like '%Manager%' and e.niNum like c.custID.niNum
order by
    e.empID ASC;
```

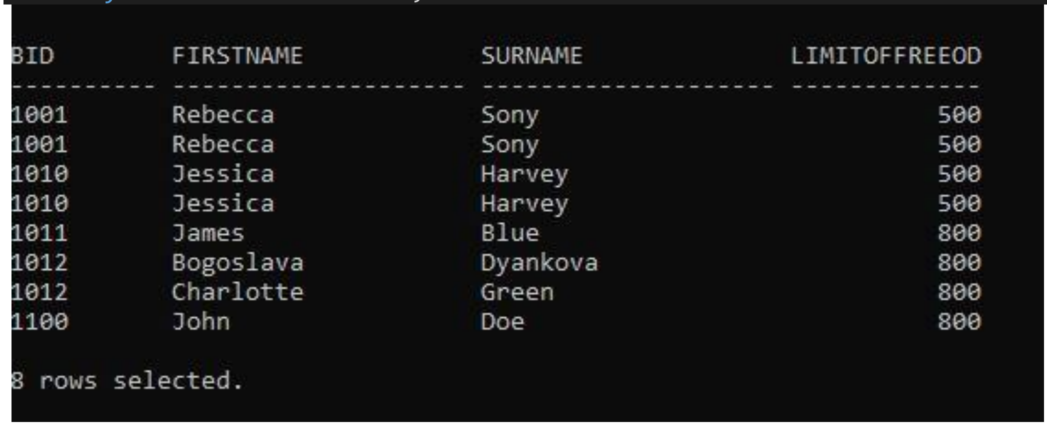


```
Employee I NAME                 Account Number   City
---------- -------------------- ---------------- --------------------
Street               Postcode
-------------------- ----------
Account Branch
--------------------------------------------------------------------------------
Work Branc City                 Street               Postcode   Supervisor posi
---------- -------------------- -------------------- ---------- ---------------
0007       Fiona


1011       Edinburgh            Fourth St            EH1 28EE   Manager


Employee I NAME                 Account Number   City
---------- -------------------- ---------------- --------------------
Street               Postcode
-------------------- ----------
Account Branch
--------------------------------------------------------------------------------
Work Branc City                 Street               Postcode   Supervisor posi
---------- -------------------- -------------------- ---------- ---------------
0012       Jamie


2000       Glasgow              Potato St            G10 1QL    Manager
```

e) Customers with highest free overdraft limit



```
select
       c.accNum.bID.bID as bID,
       c.custID.name.firstName as firstName,
       c.custID.name.surName as surName,
       c.accNum.limitOfFreeOD as limitOfFreeOD
from(
       select
       c.accNum.bID.bID as bID,
       max(c.accNum.limitOfFreeOD) as maxOD
       from tb_CustomerAccount c
       group by c.accNum.bID.bID
       ) maxOD, tb_CustomerAccount c
where c.accNum.limitOfFreeOD = maxOD.maxOD
       and c.accNum.bID.bID = maxOD.bID
order by c.accNum.bID.bID ASC;
```



```
BID        FIRSTNAME            SURNAME              LIMITOFFREEOD
---------- -------------------- -------------------- -------------
1001       Rebecca              Sony                           500
1001       Rebecca              Sony                           500
1010       Jessica              Harvey                         500
1010       Jessica              Harvey                         500
1011       James                Blue                           800
1012       Bogoslava            Dyankova                       800
1012       Charlotte            Green                          800
1100       John                 Doe                            800

8 rows selected.
```

Task 4. Advantages and disadvantages of the object-relational model against the entity relational model

The entity relational model represents both the data and the relationships between the data using a collection of tables. It also uses Primary and Foreign keys, allowing the user to create these relations. The advantages of the entity relational model are its simplicity, good store management and query tools. The simple design allows it to be easier to implement, thus being often chosen by businesses. The good storage management includes the backup and recovery of the data, which is essential whenever there is a system disruption so there is no loss of data. The disadvantages of the entity relational model are the insufficient asserting of data, defining types with nested relations, write methods, use entities with single units and run transactions which have a long duration. In conclusion, the ER model is simple and easy to understand. However, it lacks proper design functioning for dealing with obstacles such as duplicate data.

The object-relational model is an extension of the entity relational model, therefore covers all the disadvantages pointed out. Unlike the ER model, the OR model captures the semantics of the objects in object oriented programming. The object-relational database has clear-structured methods, types, inheritance, constraints and collections. Having this, it helps the user to create the tables more efficiently. One of the main disadvantages of the OR model is the complex structure which makes it less universally used, since it is not as time-efficient to pick up and support as the Entity Relational Model. The object relational model can shorten the number of columns in a table by using types. Although type has the ability to contain a variety of attributes, the values of these attributes cannot be moved in another column.

Another advantage of the OR model that the ER model doesn’t have is the skill to use member functions. Member functions are created by using an additional type which broadens the classic definition of a type.

In conclusion, the differences and similarities between Entity Relational and Object-relational Databases has been critically discussed in this analysis, understanding the advantages and disadvantages between them. To extend the semantics of the ER diagram, the OR model has included types, tables, references, nested tables and constraints.