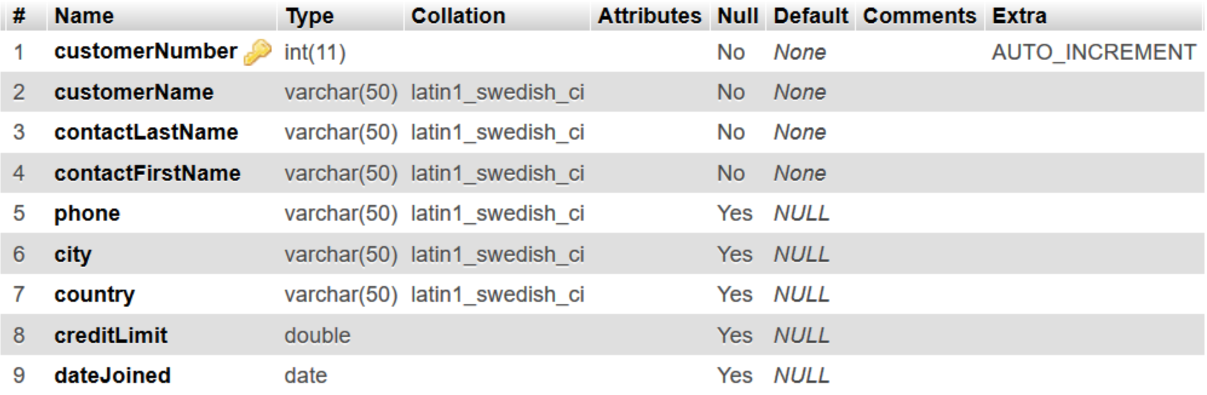
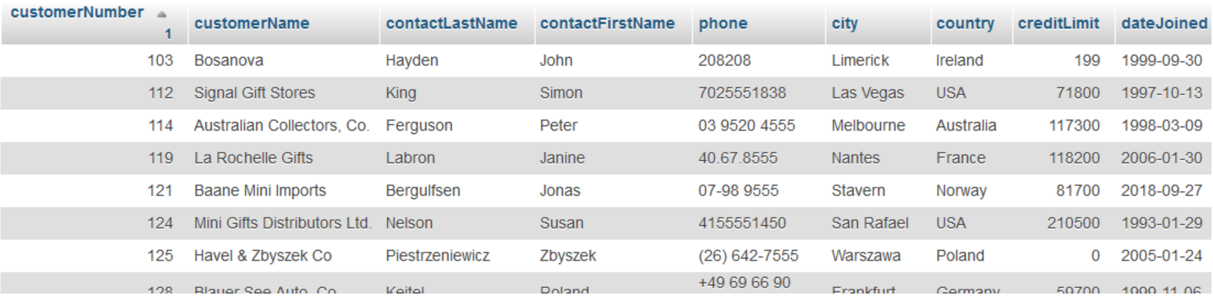
Download *company.sql* from Moodle. Execute this script on your local machine. The script will create a database called ‘company’ with a single table called ‘customers’.

The structure of this table is as follows:



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**customerNumber** 🔑	int(11)			No	*None*		AUTO_INCREMENT
2	**customerName**	varchar(50)	latin1_swedish_ci		No	*None*		
3	**contactLastName**	varchar(50)	latin1_swedish_ci		No	*None*		
4	**contactFirstName**	varchar(50)	latin1_swedish_ci		No	*None*		
5	**phone**	varchar(50)	latin1_swedish_ci		Yes	*NULL*		
6	**city**	varchar(50)	latin1_swedish_ci		Yes	*NULL*		
7	**country**	varchar(50)	latin1_swedish_ci		Yes	*NULL*		
8	**creditLimit**	double			Yes	*NULL*		
9	**dateJoined**	date			Yes	*NULL*		

There are 54 records in the customers table in total. Some sample records include the following:



customerNumber ▲ 1	customerName	contactLastName	contactFirstName	phone	city	country	creditLimit	dateJoined
103	Bosanova	Hayden	John	208208	Limerick	Ireland	199	1999-09-30
112	Signal Gift Stores	King	Simon	7025551838	Las Vegas	USA	71800	1997-10-13
114	Australian Collectors, Co.	Ferguson	Peter	03 9520 4555	Melbourne	Australia	117300	1998-03-09
119	La Rochelle Gifts	Labron	Janine	40.67.8555	Nantes	France	118200	2006-01-30
121	Baane Mini Imports	Bergulfsen	Jonas	07-98 9555	Stavern	Norway	81700	2018-09-27
124	Mini Gifts Distributors Ltd.	Nelson	Susan	4155551450	San Rafael	USA	210500	1993-01-29
125	Havel & Zbyszek Co	Piestrzeniewicz	Zbyszek	(26) 642-7555	Warszawa	Poland	0	2005-01-24
128	Blauer See Auto, Co.	Keitel	Roland	+49 69 66 90	Frankfurt	Germany	59700	1999-11-06

You can also download the starter code from Moodle - “Assignment3_starter.zip”. Once extracted, you will see that this project contains a class called “Customer.java” which you may find useful when developing your solution:

You must then carry out the following tasks writing and executing **SQL statements** as appropriate from **within** your code.

1. Display on screen, all records from the *customers* table in ascending order (on the *customerName* field). You must display all fields for each record on the screen. You must also display on screen how many records are returned from the database.

(25 marks)

2. Display on screen, the five records from the *customers* table that have the highest *creditLimit* – these records must be ordered by *creditLimit* in descending order. You must then display **just** the *customerNumber*, *customerName* and *creditLimit* for these five customers.

(20 marks)

3. Display on screen, all fields for every record for every record in the *customers* table who joined in 2018. You must also display on screen how many records are returned from the database.

(15 marks)

4. Insert three records into the *customers* table – specifying values for the *customerName*, *contactLastName*, *contactFirstName* and *dateJoined*. These three records must be inserted as part of a batch. The data to insert for each record is as follows:

   Data for Record 1:
   customerName: "Los Pollos Hermanos"
   contactLastName: "Fring"
   contactFirstName: "Gustavo"
   datejoined: **USE TODAYS DATE**

   Data for Record 2:
   customerName: "A1 Car Wash"
   contactLastName: "White"
   contactFirstName: "Walter"
   datejoined: **USE TODAYS DATE**

   Data for Record 3:
   customerName: "Saul Goodman & Associates*"*
   contactLastName: "Goodman"
   contactFirstName: "Saul"
   datejoined: **USE TODAYS DATE**

(25 marks)

5. Add a column called *'creditRisk'* to the *customers* table (by writing and executing a sql statement from within your code). This column will be used to track whether each customer is deemed a credit risk or not. You can decide on an appropriate type for this column yourself. Once the column has been added, it will initially have an empty/null value for each record. You must then ensure that all records which have a *creditLimit* of zero are recorded as a credit risk, while all others are not.

(10 marks)