Databases

Exercise set 1

Relation schemas

The exercises contained in this exercise set relate to the *music* database. The schemas for the relations in the database are shown here. Primary key attributes are italicised. The saleXY relation has multiple instances. One for each calender year.

customer(*custID* int,fname varchar(255),lname varchar(255),houseNum varchar(255),postCode varchar(255))
album(*albumID* int,artist varchar (255),title varchar (255),label varchar (255),year numeric(255), genre varchar(255),price decimal(6, 2))
artist(*artistID* int,name varchar(255),countryOfOrigin varchar(255))
label(*name* varchar(255),*region* varchar(255),country varchar(255))
genre(*name* varchar (255),description varchar (255))
saleXY(*salesRef* int,custID int, albumID int,saleDate date)

Using *Postgres*

The first exercises involve starting to use *Postgres*.

Exercise 1

Log in to a Linux machine in the lab. In your `work` folder you will find a file called 'db-password.txt'. This contains the password that you will need for the next step.

Connect to the database server by typing the following command in a terminal

```
psql -h mod-databases.cs.bham.ac.uk
```

`psql` is the terminal for working with the DBMS we are using, *Postgres*. The `-h` switch and its argument tell `psql` which database server to connect to.

When prompted for the password, enter the password contained in the file 'db-password.txt'.

You will now see the `psql` prompt. Your first task is to change your password. Passwords are not stored as securely in *Postgres* as they might be so **don't** use your regular university password. To change your password type

```
ALTER USER userid WITH PASSWORD 'new-password';
```

Where `userid` is your university username and 'new-password' is your new chosen password.
To switch to the actual database that we will be using type

```
\c music
```

The `\c` command is the `connect` command. The argument `music` is the actual database to connect to.

You are now connected to the music database.

Exercise 2

To see which relations exist in the `music` database type at the `psql` prompt

```
\dt
```

This will list the relations available in the current database. Explore all of the relations in the `music` database using the command

```
SELECT * FROM relation-name;
```

(replacing `relation-name` with the name of each relation in turn.)

Don't forget the semicolon at the end of the command otherwise `psql` will simply wait for further input. This is how you can enter commands across several lines.Compare the output from each command with the schemas defined at the top of this exercise sheet.

SQL commands

Exercise 3

Write an SQL query to list all of the albums by The Beatles.

Exercise 4

Write a query to list all of the artists from the USA.

Exercise 5

Use ORDER BY in a query that lists album titles in reverse alphabetical order.

Exercise 6

Write a query to find the titles and artists of albums released before 1970 but not including albums by The Beatles.

Exercise 7

Write a query to list the sales references (salesRef) of all 2015 sales of albums released from 1975 onwards.

Rewrite the above query to include the same data but from 2016.

More *Postgres*

Exercise 8

It can become tedious and error-prone to develop SQL commands at the command line, particularly as the commands get longer. It is better to develop the SQL commands in a file (where you can also comment them) and then load the file at the `psql` prompt.

In order to do this, create a folder in which you will keep your SQL commands. Firstly, in a Linux terminal, make sure you are in your `work` folder, and then type

`mkdir db` (or some other name like that)

Change to that folder using

```
cd db
```

Now open a text editor and create a file called `query.sql` (or other file name, but use the .sql extension). You can do this from the command line with an editor of your choosing. For example, to use `gedit` type

```
gedit query1.sql
```

In the text editor, type an SQL command such as

```
SELECT * FROM customer;
```

Save the file. You don't need to quit the editor.

In the `psql` terminal type

```
\i query1.sql
```

This will run the query contained in the file.

To redirect the output of the query to a file called out.txt, in the `psql` terminal type

```
\o out.txt
\i query1.sql
```

Output will be redirected to out.txt until you cancel it by entering the command

```
\o
```

You can output the results of queries entered at the command line too, with the same redirection. For example

```
\o out.txt
SELECT * FROM album;
\o
```

To quit `psql` type

```
\q
```