Database Fundamentals Assignment Part D - Implementation:

Gianni Pessotto (13199219) and Daniel Boustani (13207509) Tutorial:Wrk1 Activity:02
https://www.mcgrath.com.au/ - website to use for real data

Instructions-
1)
Choose THREE tables from your ERD to implement. The implemented tables should have RELATIONSHIPS (WITH EACH OTHER). We suggest the implementation of the associative relationship hence it requires an associative entity accompanied with two neighbour entities.
2) The Data Populate your database with suitable data for testing the SQL queries below (Section 3). Imaginary data for that web site is acceptable, but real data from that web site is preferred, where possible. You must provide the URL for the web site that inspired your project.
Each table should have at least 5 rows. Also, provide enough data so that the table rows demonstrate the relationships. For example, if there is a 1:M relationship between the 2 tables, ensure that there are at least two records in the M-side table that are related to a respective record in the 1-side table.
3) Queries Write the following queries for your database:
● Three SELECT * statements for three separate tables (see section 4.5).
● A query involving a “Group by”, perhaps also with a “HAVING” (see section 4.6)
● A query which uses "inner join" (see section 4.6).
● A query which uses a “subquery” (see section 4.6).
4) The Database Scripts Build up your database scripts based on the following instruction.
4.1. Provide the Required information
The file containing your SQL should begin with a comment header block (i.e. lines beginning with two dashes, or using /* <comment goes here> */). The first line of the header block should contain (“Database Fundamentals, Assignment Part D”), followed by lines providing your name and email address (but not your student number). The header block should then contain, in English, the nature of your database application. Do NOT use technical database language in this section. Write something simple. You must provide the URL for the website that inspired your project in this hearer block.
Example: See the first lines in the provided example file “dbpizza_Revised.txt”
4.2. Drop Each Table
Start the SQL Scripts with “DROP” commands for each of your tables, so you can run your script more than once.
Example: See the “drop table” statements in the provided example file “dbpizza_Revised.txt”
4.3. Create Each Table

Use CREATE statements to create your tables. Use “Constraint” to define primary and foreign keys.
Example: See the “create table” statements in the provided example file “dbpizza_Revised.txt”
4.4. Insert Data in Each Table
Use INSERT statements to insert data into your tables as required (see section 2).
Example: See the “insert” statements in the provided example file “dbpizza_Revised.txt”
4.5. Show the Inserted Data using Select * Statement
We need to check that the data that are inserted into your tables. To do this, you need to include “select * from TableName” statement in your database script with an English language description of what the SQL query does. Then provide the result table while the related lines to the result table, are provided as comment lines i.e. started with '-- ' (The space after the two dashes is required). Using this, when we run your script we will be able to check the result table with your provided result table in comment mode. You also need to comment out the English language description of the query with '-- ' so you do not get a syntax error when running your script

All designed SELECT statements, the corresponding SELECT statements and the generated result tables should be appended to your script right after the last INSERT statement of your script.
Example: Follow the structure provided in 3.a.1, 3.a.2 and 3.a.3 examples provided in the example file “dbpizza_Revised.txt”. You need to specify the question, the select statement and the result table.
4.6. Provide the Queries Using “Group by” “Inner Join” and “Sub Query”
Set out each of these queries as:
a. An English language description of what the SQL query does. Place each question right before the corresponding SELECT statement. Comment out the questions (with '-- ' as it is described in Section 4.5) so you do not get a syntax error when running your script!
b. Place the result table generated by each SELECT statement right after the SELECT statement. Comment out the result tables (with '-- ' as it is described in Section 4.5) so you do not get a syntax error when running your script!
Example: Follow the structure provided in 3.b, 3.c, 3.d, 3.e and 3.f examples provided in the example file “dbpizza_Revised.txt”. You need to specify the question, the select statement and the result table.

Tables to use:

Agents, Bookings and Customers. (relations)

Agents(AgentID, Division, Location, Image, StaffID*)
StaffID references Staff

Bookings(CustomerID*, OfficeID*, AgentID*, book_date, book_time)
CustomerID references Customers
OfficeID references Office
AgentID references Agent

Customers(CustomerID, email_address, first_name, last_name, Phone, Area, Street, Postcode, State, Suburb)

Relationships:
Agent - Bookings = mandatory 1 - mandatory 1
Customer- Bookings = mandatory 1 - mandatory 1

(It says all tables need 5 rows so might need to use offices instead of agents.)

```
 /* <Database Fundamentals Part D, 13199219 and 13207509> */


Create table agents
        (
        AgentID  numeric
        Email  varchar(20)
        Location  varchar(20)
        StaffID  numeric
        PhNo    numeric
        PRIMARY KEY (AgentID)

);


Insert into agents (001, Aaron.Bird@outlook.com, NSW, 0289, 0402434818)
Insert into agents (002, Alex.Mintorn@outlook.com, QLD, 0136, 0499442274)
Insert into agents (003, Bec.Myers@outlook.com, QLD, 0162, 0401859548)
Insert into agents (004, Ben.Cannon@outlook.com, NSW, 0245, 0413596496)
Insert into agents (005, Grace.Zhang@outlook.com, VIC, 0398, 0466387666)



Create table bookings
          (
          AgentID numeric
          CustomerID numeric
          Book_Date  dd/mm/yy
          Book_time    numeric
          Location  varchar(20)
          PRIMARY KEY(CustomerID, AgentID)
);


Insert into bookings ( 001, 001, 12/08/12, 1230, Customer House)
Insert into bookings ( 005, 004, 16/09/12, 1030, Customer House)
Insert into bookings ( 001, 002, 21/09/12, 1130, Property Location)
Insert into bookings ( 001, 005, 30/09/12, 1400, Office)
Insert into bookings ( 003, 003, 06/08/12, 1100, Office)

Create table customers
          (
          CustomerID numeric
          Email_address varchar(20)
          First_name  varchar(20)
          Last_name   varchar(20)
          Phno    numeric
          Street      varchar(40)
          City      varchar(20)
          State    varchar(20)
          PRIMARY KEY(CustomerID)
);


Insert into customers values (001, h1c@hotmail.com, Henry, Lawson, 04332880991,
Dawson Street, Sydney, NSW)
Insert into customers values (002, Kingston03@outlook.com, Kingston, Marsy,
0417890622, Monty Avenue, Sydney, NSW)
```

Insert into customers values (003, Br1g@gmail.com, Dallwod, Briggsy, 04657892987, Ty Street, Brisbane, QLD)
Insert into customers values (004, MZOINK@hotmail.com, Mary, Zoink, 0400243219, Downing Street, Melbourne, VIC)
Insert into customers values (005, Anthony.Tolliver@outlook.com, Anthony, Tolliver, 0463678791, Azork Avenue, Sydney, NSW)

/* what this does */
Select CustomerID from customers;

/* what this does */
Select CustomerID from customers where state is “NSW”
Having city = “Sydney”

/* what this does */
Select AgentID from agents where Location is “A%”
Group By Division