Populate and Extract Data from Your Database

1. Overview

In this lab, you will:

1. Check/revise your data model and/or marketing material (home page content) from last week's lab. You will work with two classmates to be sure your data model and web application functionality meet all the requirements.
2. Using MySql Workbench, you will populate all four of your database tables with data.
3. You will generate several SQL select statements.
4. You will put together a word document that shows all your work. (This will later be published to your web site.)

You may have to reference the tutorials from last lab:

- MySQL Workbench Tutorial 1: How to create database tables (with primary keys), enter data, and write single table select statements
- MySQL Workbench Tutorial 2 (continuation of previous tutorial): How to reverse engineer a database, create database relationships (with foreign keys), and how to write select statements that join data from more than one database table

2. Lab Requirements

1. Your **Data Model** shall:
    - meet the **check list** of the last lab (e.g., data types, PKs, FKs, Not Null vs. Nullable, etc.).
    - have **NO SQL keywords** as table names nor field names (google to see list of SQL keywords, but can't use these for sure: user, role, password, date).
    - be able to support the functionality described/marketed in your **home page content** (your web app cannot deliver data unless that data is stored in your database).

    If your data model from last lab does not meet the requirements above, fix it and/or modify your home page content. You are free to totally redo your idea from lab 1, but the new data model and home page content must meet the above requirements.

2. **Data Model Review from Classmates**. During a graded lab activity, you will have time to get feedback from (and give feedback to) two classmates check. The feedback should indicate whether or not your data model meets the requirements specified just above (or say why it does not). Bring a printout of your data model and your proposed web app functionality when you meet with your classmate(s) for the review. If you did not complete the lab activity, you'll have to find two classmates and get their feedback outside of class time. The submission requirements show what you need to submit to show that you got and gave feedback from two classmates.

3. Your database shall contain the following **realistic looking data**:
    a. 2-3 records in your role table, one of which is for the ADMIN role.

    b. 4-6 user records.
        - At least one record with all fields populated with data and at least one record with all of the optional fields empty.

    c. 4-6 records in your "other" table
        - At least one record with all fields populated with data and at least one record with all of the optional fields empty.

    d. 10-15 records in your associative table.
        - At least one record with all fields populated with data and at least one record with all of the optional fields empty.

    Note: you have to enter your data in order. For example, you cannot enter users before roles because user records reference role records. The same is true for the associative table - its data must be entered *after* you enter the user data and the "other" data.

4. Execute a **SELECT statement** for each of the following (and paste screen capture into word doc – see submission instructions for specifics on how you are to do this):

   a. List *all of the columns* of your "other" table, ordering the columns in a way that you think users would like to view the data (don't use "select *"). Sort the data by the first column. Include all records of the "other" table.

   b. Show all the records from your user table joined with your role table. Show the role name first, then the role id, then the email address, followed by all the rest of the columns of your user table (ordered in a way you think users would like to view the data – show the role id once, not twice.) Order the result set by role name, then email address (as a secondary sort). Include all records. There should be as many rows in your result set as you have records in your user table. If you have a lot more (and see duplication), you have forgotten the WHERE clause that joins the two tables together.

   c. Join your associative table with your "other" table. Include all the columns except PK and FK (id) columns. Order the columns in a way you think users would like to see them (not "select *"). Order the result set by the first two or three columns. There should be as many rows in your result set as there are records in your associative table.

   d. Join your associative table with your "other" table and with your user table. Include all the columns except PK and FK (id) columns. Order the columns in a way you think users would like to see them. Order the result set by the first two or three columns. There should be as many rows in your result set as there are records in your associative table. Hint: you need two conditions in your WHERE clause, one for each PK/FK relationship.

   e. Join your associative table with your "other" table and with your user table and with your role table. Include all the columns except PK and FK (id) columns. Order the columns in a way you think users would like to see them. Order the result set by the first two or three columns. There should be as many rows in your result set as there are records in your associative table. Hint: you need THREE conditions in your WHERE clause, one for each PK/FK relationship.

   f. Modify your SELECT STATEMENT from item "e", adding a condition to your WHERE so that you see only the data from one user (selected by user.user_id). Select a user such that your result set has at least 2 records in it.

   g. Modify your SELECT STATEMENT from item "e", so that it selects one user id, two or more "other" records (using LIKE keyword and % wildcard match), and some condition testing a field from the associative table. Come up with conditions such that your result set has at least 2 records in it. Example of using LIKE:

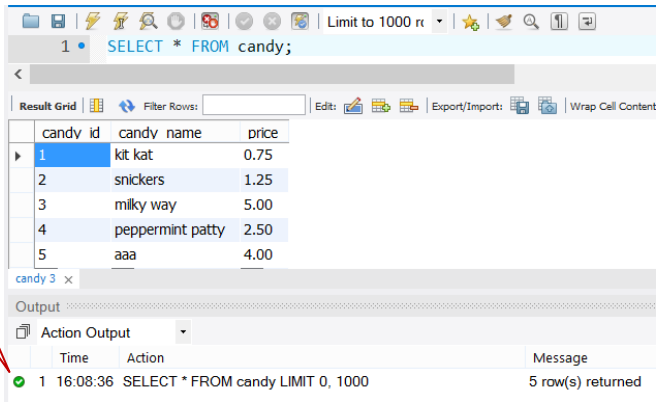
```
SELECT * FROM tableA, tableB
WHERE tableB.tableA_id = tableA.tableA_id AND tableB.name LIKE '%temple%'
```

3. Submission Requirements

Create and **submit into Blackboard**, a word or rtf document that contains the info described below. Include your last name in the file name of this document. Use landscape if that's easier to read.

1. **Your name**.
2. Heading "**Functionality**" followed by your web application's proposed functionality.
3. Heading "**Marketing Material**" followed by 1-2 paragraphs that attempt to entice people to become registered users and/or viewers of your web site. This will become the basis for the content area of your home page. In your paragraphs, do not explain to me what your web site will be able to do, write text that will entice users to visit your site.
4. Heading "**Data Model**" followed by a screen capture of your data model (created in MySQLWB).

   One way to get a screen capture is to press Alt-PrtSc (copies active window into the clipboard), paste into MsPaint (or any other image editor), then copy just the part of interest into your word/rtf document (if you paste the whole screen, it's too hard to read). Or use snippet tool.
5. Heading "**Table Designs**" followed by (for each of your four tables) a screen capture of the table design and **foreign key** tab of the table's design (if the table has a foreign key).
6. Heading "**Feedback from Classmates**" followed by the names and feedback from two classmates who reviewed your Data Model and proposed functionality.
7. Heading **"My Feedback"** followed by the names and feedback that you provided to two classmates.
8. Heading "**SELECT Statements**" followed by this for each of the 7 select statements:
   - Copy/paste the description of the SQL statement (from this doc).
   - Paste a screen capture of the SQL query window (we need the screen capture so we can be sure that your SQL ran properly). The screen capture shall include:
     - top area: syntactically correct SQL (no red/error messages),
     - middle area: all rows and all columns of result set,
     - bottom area (Action Output): confirmation message with number of rows returned (right click the green checkbox - clear before running the SQL that you are screen capturing).



Right click this checkmark and clear this before running SQL

For fast feedback, ***hand in a hard copy*** *of the first four sections this doc (name -> data model) to your instructor in lecture.*

4. Lab Grading

- **Professionalism:** When you answer the questions about your proposed web application, especially the last question which asks for "marketing material", we are looking for quality that would be acceptable by a "real company" that might be paying you to create their web site.
- **Check list:** We will check that your data model meets all of the requirements listed in lab 1 (requirements such as data type, PK, FK, null-able, unique). This is very important, since you will lose points in many future labs and the project if your data model is not as specified.
- **Consistency:** Your data model has to be able to support the functionality that you say your web application will offer.
- **Originality of Project and Data Model:** As in most lab assignments, points will be deducted if your answer is too similar to the sample or sample(s) provided. In this case, I gave you a lot of project ideas, so it's OK if your web application proposal is similar to one of them. However, your data model cannot be overly similar to the concert/band/venue example I provided. It also cannot be overly similar to any other student in the class.
- **Realistic Data:** You were asked to enter realistic data so that your web application looks good when it begins to display data on its pages.
- **SELECT Statements:** The select statements must be syntactically correct and produce the required results. The screen captures need to be readable and verifiable as described in the submission section.
- **Timeliness:** See lab due date schedule (in BB) for late policy for labs.

5. Suggested approach:

1. Self check your data model and functionality/marketing material.
2. Find two classmates with whom you will do mutual review of data model / functionality. Take note of your feedback for the two classmates (you will submit along with this lab). These students should also review your work.
3. Populate your database with realistic looking data.
4. Start working on the SQL select statements and getting the screen captures. If landscape works better to show your work, select that Page Layout. In Word, you might also click on "View – Web Layout" to be able to show information better.
5. Start building your document (as described in the Submission Requirements section).