SQL Continued

Before you begin please do the following:

1. Watch the video lecture posted. Many of the commands I will show you on the video.
2. You can also look at the SQLCheatsheet tab for many of the common SQL commands.
3. The below exercises correspond to the lecture titled SQL Continued.

Submission: Save a new Word document which will contain all SQL code and output results to be submitted to the folder titled SQL Continued in D2L Assignments tab.

I Create Table

1. Create a table, Castings, which contains the following columns:
    a. movieId (int)
    b. actorId (int)
    c. characterRole (varchar) *what character were they cast as*
    d. salary (decimal you need to supply parentheses)

Hint: For decimal see last week's lecture notes:
Decimal(precision, scale)
***Precision** = total number of digits including decimal places (excluding the .)*
***Scale** = the total number of decimal places.*
Example 12.345 has precision 5 and scale 3

II INSERT

2. a. Write the statement to insert at least 5 rows into the Castings table that are consistent with what you have in the movie and actor table. For example, you will want to choose an actorId and a movieId where that actor was really cast in that movie. (You may add additional rows to the Actor and Movie tables.) You can make up the salaries.

    b. **You must have at least one actor (with id #1) be listed there twice for different movies.** You can have more than one actor repeated for different movies, but one of them, must be actorId=1.

    c. Have at least one actor in the Actor table who is NOT in the Castings table.

III SELECT

3. Write the statement to select the salaries and actorId for everyone in the Cast table ordered by actorId.

IV Aggregates

4. Write the select statement to COUNT how many rows there are in the Castings table?
5. Paste the results of the above statement.

6. What is the AVERAGE salary, MINIMUM salary, MAXIMUM salary, and SUM of all salary for all of the salaries in the Castings table?
7. Paste the results of the above statement.

8. Write the select statement to count how many Castings belong to the actor with id number 1.
9. Paste the results of the above statement.

V Group by

10. In one statement select actorId, and their average salary. Group the results by actorId
    Hint: In addition to Group By you will need to use above aggregates
11. Paste the results of the above statement.

12. Select the actorId and the average salary, grouped by actorId, but ONLY display it when the actorId is 1.
    Hint:You cannot use 'where' in Group By when using Aggregates - use Having!
13. Paste the results of the above statement.

14. Do the same as the previous step (12-13) but display the sum instead of average. Write the statement and show the output.

VI Subqueries

15. Use a **subquery** to select the salary where you forgot the actorId, but you do know the actor's name! (Note: To receive credit this must be done using a subquery and a specific actorId **cannot** be explicitly written).
16. Paste the results of the above statement.

VII Joins

17. Select the actorId, actor first name, last name, and salary for all actors who are cast in the Castings table.
18. Paste the results.

19. Select the actorId, actor first name, last name, and salary whether or not they are listed in the Castings table. (All actors in the actor table should appear)
    Hint: Use Left or Right Join
20. Paste the results.

21. Select the actor's first and last names, characterRole, movie title, and the salary for all actors who are cast in the Castings table.
22. Paste the results.

VIII Additional Challenge

23. Try any SQL statement (join, subquery, etc.) that you are curious about.
24. Paste the results of the above statement.