**Software Developer Course Assessment**

**Quantitative Assessment Practice #1**

**Course Name: Database Programming and Data Processing**

**Current Week:** (DD/Mmm/YYYY)

**Introduction:**

The purpose of this assessment is to help us understand how the class is doing in terms of the review material that we have covered during the previous couple of weeks. The **only** purpose of this assessment is for us to improve our approach to review and ensure that what we’re currently doing is an effective strategy. Completion of this assessment is **mandatory - if you don’t submit a solution, it will be marked as incomplete. You must complete a minimum of 80% of your assigned QAPs per course – otherwise you will be marked as incomplete for that course no matter how good your other grades are.** If you do submit a solution, it will be marked as complete, as you will receive full marks no matter what your actual performance was – again this is a participation grade.  
  
**Again, the goal here is to help you** all in the best way that we can, so please do be honest when answering the questions related to how long it took, which resources you used, etc. And please ensure that you do your **own** work – don't just copy off a friend to get it done, earnestly do your best with it. If you can’t get it completely working, give us what you have. While it will be graded, the grade will not count against you, it’s just a way for us to see where everybody is, and to know which concepts, if any, we, as a class, may be struggling with.

**Deadline:** You will have until the end of the day on Friday (4:00pm) to submit your assessment solutions. Please ensure you answer all the questions outlined in the instructions portion of this document as well in your submission.

**Marking:** In this program core evaluation is marked with one of three possible marks: *Incomplete, Pass, Pass Outstanding.* For QAPs, though, where incomplete marks are more important for our own information as well as for the information of the student, we wanted to increase the resolution of our grading system. Therefore, QAPs are marked on a scale of 1-5. The details of this marking system are summarized in the table below.

|  |  |
| --- | --- |
| **Grade** | **Meaning** |
| 1 | *Incomplete.* Student shows severe lack of understanding of the material – solution is heavily incomplete, non-functional, or completely off base of what the assignment was asking for. |
| 2 | *Partially Complete.* Students show some understanding of the material. Solution may be non-functional or partially functional, but the approach is correct, albeit with some major bugs or missing features. |
| 3 | *Mostly Complete.* Student demonstrates understanding of the major ideas of the assignment. Solution is mostly working, albeit with a few small bugs or significant edge cases which were not considered. Shows a good understanding of the correct approach, and is either nearly a feature-complete solution, or is a feature-complete solution with some bugs. |
| 4 | *Complete (Equivalent to: Pass.)* Student shows complete understanding of assigned work and implemented all necessary features. Any bugs that are present are insignificant (for example aesthetic bugs when testing the functionality of code) and do not impact the core functionality in a significant way. All necessary objectives for the assignment are completed, and the student has delivered something roughly equivalent to the canonical solution in terms of features and approach. |
| 5 | *Complete with Distinction (Equivalent to: Pass Outstanding)* The student demonstrates a clear mastery of the subject matter tested by the QAP. The solution goes above and beyond in some way, makes improvements on the canonical solution, or otherwise demonstrates the student’s mastery of the subject matter in some way. A solution in this category would consider all reasonable edge cases and implement more than the necessary functionality required by the assignment. |

**Instructions:**

You are allowed to complete the assessment problems below in whatever way you can but please answer the following questions/points as part of your submission:

1. How many hours did it take you to complete this assessment? (Please keep try to keep track of how many hours you have spent working on each individual part of this assessment as best you can - an estimation is fine; we just want a rough idea.)
2. What online resources you have used? (My lectures, YouTube, Stack overflow etc.)
3. Did you need to ask any of your friends in solving the problems. (If yes, please mention name of the friend. They must be amongst your class fellows.)
4. Did you need to ask questions to any of your instructors? If so, how many questions did you ask (or how many help sessions did you require)?
5. Rate (subjectively) the difficulty of each question from your own perspective, and whether you feel confident that you can solve a similar but different problem requiring some of the same techniques in the future now that you’ve completed this one.

**Problem#1: ADMINISTERING**

When running and administering a database it is important to keep track of its current health and availability. Having information about its version and what databases are available becomes an important part of administering the database. Being able to perform these tasks from the command prompt also becomes an important professional skill.

A. Using the PostgreSQL command prompt to determine the following;

1. The running PostgreSQL version
2. A list of databases for your local instance
3. The current datetime

The display should be similar to the following;

Text

Description automatically generated

B. Using the pgAdmin GUI display the performance of your local instance of the PostgreSQL server. The display should look similar to the following;

Graphical user interface, application

Description automatically generated

**Deliverable #1:**

To complete problem#1 please take a screen capture of both the command line console showing all the executed commands and the pgAdmin GUI showing the dashboard. Save these two screen captures to individual files and store them in a folder that will be zipped up and sent to the instructor with other files completed during problems two and three.

**Problem #2: SELECT, SELECT, SELECT**

Developing an exemplary understanding of SQL will serve you very well as a software developer. During this problem you will execute several SELECT statements against the dvdrental database. Create and execute SQL SELECT statements using the following attributes and clauses; *(Note: Do not use the examples from the lecture, create your own SQL statements.*)

1. Reduce the number of columns displayed in the query result to four or less.
2. Reduce the number of rows displayed in the query result using a WHERE clause.
3. Group the results using a GROUP BY clause.
4. Order the results using an ORDER BY clause.
5. Join two or more tables using a JOIN clause.

**BONUS:** Complete this problem with only one SQL statement containing all the attributes and clauses. Yes, it is possible.

**Approach:**

I suggest you work in the pgAdmin GUI as you iterate with the development of your SQL statements. I believe the development environment found in the pgAdmin GUI is more intuitive than working in the psql console. Once you have the SQL working as desired save the SQL to a file with the file extension of .sql in the same directory as the image files from problem#1. Do a screen capture of the pgAdmin GUI showing your SQL and your query results. Save this screen capture to a file in the same directory as the previous step. Continue through all five items in Problem#2, save the query results as described above.

**Deliverable #2:**

Five .sql files **and** five screen captures showing completion of all five items in problem#2. If you complete the BONUS include it as another .sql file and separate screen capture.

Graphical user interface, text, application

Description automatically generated

**Problem #3: SQL CRUD**

SELECT is only one side of working with a database. Developing SQL skills to INSERT and UPDATE data is equally important. Use both the INSERT and UPDATE SQL to add and update records in the dvdrental database.

1. Add (or INSERT) a family of new customers to the dvdrental database. They should all have the same last name. Add them all to the same store and living at the same address.
2. The family moves, change their address using an UPDATE statement. Don’t over think the move… you should be able to add the new address and change one attribute for each family member to make the move.
3. DELETE the old family address

**Approach:**

Use the similar approach as described in problem#2.

**Deliverable #3:**

.sql files and screen captures showing completion of all items in problem#3.

**Problem #4: SQL CREATE**

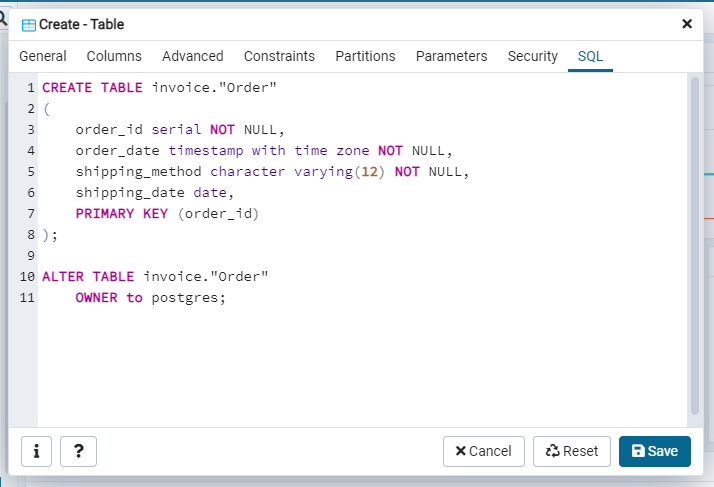
Creating new databases and adding items to the database is an important software development skill. Create both a new database and two database tables (entities) in the new database. The two database tables should have a one to many (1:M) relationship with each other. Imagine a business scenario that would require the 1:M relationship and create two entities (tables) that implement the 1:M. Even consider doing a Google search on “one to many relationship examples”.

**Approach:**

Use the similar approach as described in problem#2.

**Deliverable #4:**

.sql files and screen captures showing completion of all items in problem#4.



**Project Submission:**

Store all your solution files into a single working directory (folder) and archive them into a single zip file. Be sure you DO NOT include any other files than your solution files.

Submit this zip file to the team assignment page.