

(20 points) Milestone 3: DB setup and SQL

Due: 11:59pm, Sunday, April 07, 2024

Purpose:

- Set up the database to be used by your app
- Set all of your tables in your database
- Populate the database with the chosen dataset — be sure the data are practical and sufficient
- Write all SQL commands you plan to use in your app
- Write at least 2 different PL/SQL commands (such as trigger, assertion, check constraint, stored procedure, Function,...) to efficiently manage the business logic of your app
- Test and verify that your SQL commands work properly

In this milestone, you will work in your project group to set up a relational database using MySQL or your chosen DBMS and develop database queries to be used by your app.

Backup your database frequently!! If you use phpMyAdmin or Google Cloud SQL, use the export feature to back up your database often. In this way, if there is any problem with the database server, you can easily migrate your database to another server (using the import feature).

Minimum requirements:

- Set up a MySQL database
- Set all of your tables (from Milestone 1) in your MySQL database
- Populate the database with your chosen dataset. The data should be realistic and sufficient to demonstrate that your project is practical (**a minimum of 10 entries per tables**)
- Write all necessary SQL commands to be used by your app based on the UI design functionalities in Milestone 2. At this stage, you should include all the commands that have been used and the commands you plan to use in your app.
 - It is normal that you may update or write additional SQL commands as you continue developing your app.
 - **We grade this part based on the nature of your project, the quality of your SQL commands, and whether your SQL commands are reasonable for your project to provide the proposed services/functionalities.**
- Write at least two of the following PL/SQL commands to efficiently manage the business logic of your app. Later, these commands will be used by your app.
 - Trigger
 - Function
 - Assertion
 - CHECK constraint
 - Stored procedure

You are required to have at least two **"different"** advanced PL/SQL commands. Writing two triggers, for example, counts as one.

Please note, basic constraints and functions such as data types, NULL or NOT NULL, default values, primary keys, foreign keys, AUTO_INCREMENT, INDEX, and UNIQUE are not qualified for the PL/SQL command requirements.

- Test and verify that your SQL commands work properly.

Specific requirements:

- You should consider the feedback from graders (Milestone 1), revise your database design and finalize your tables as appropriate before writing SQL commands or setting your tables in the database server.
- To ensure consistency between your database design and implementation, submit a final version of your E-R diagram(s) and your normalized tables (written in Schema statements) along with the SQL commands.
- Please also note, as part of the final deliverables, your project must meet the minimum number of table requirements — our database must have all these tables and your app must use all these tables in some way.

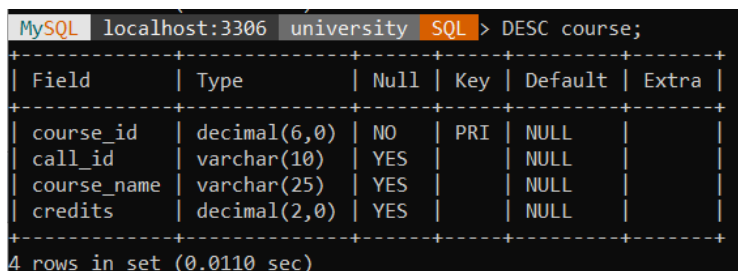
It is important to note that all SQL commands must be YOUR own. Using ChatGPT, Django or any SQL generation framework/tool/software/AI to help or create SQL commands does not satisfy the **"YOUR own SQL"** requirements.

What to submit

To demonstrate that you have completed the tasks and fulfilled these milestone requirements, document your database setup and SQL development in a report. You will submit milestone 3 in **three parts: (I) report, (II) .sql file containing all SQL commands, and (III) peer evaluation.**

I) Report *REPORT.pdf*: Submit a brief report in four parts

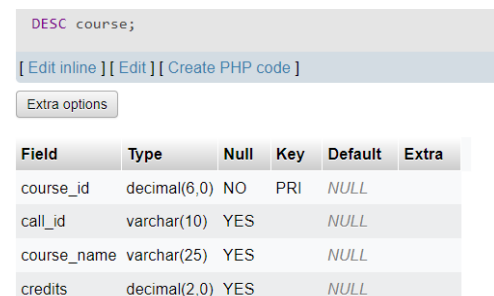
- 1) Database server:
 - a) Specify where you host your database
- 2) Table schemas — For each table,
 - a) Take a screenshot of the table schema and include it in your report.One way to view a table schema is to run a **DESC table-name** command.
The following figures shows examples of a table schema (depending on how you retrieve the table schema, your screen may look different)



```
MySQL localhost:3306 university SQL > DESC course;
```

Field	Type	Null	Key	Default	Extra
course_id	decimal(6,0)	NO	PRI	NULL	
call_id	varchar(10)	YES		NULL	
course_name	varchar(25)	YES		NULL	
credits	decimal(2,0)	YES		NULL	

4 rows in set (0.0110 sec)



```
DESC course;
```

[Edit inline] [Edit] [Create PHP code]

Extra options

Field	Type	Null	Key	Default	Extra
course_id	decimal(6,0)	NO	PRI	NULL	
call_id	varchar(10)	YES		NULL	
course_name	varchar(25)	YES		NULL	
credits	decimal(2,0)	YES		NULL	

- b) Run a **SELECT COUNT(*) from table-name** to get the total number of rows of data in the table. Include a screenshot showing the number of rows of data in your report.
- 3) Provide evidence showing that your data are practical and sufficient.
 - a) Include sample data of each table. The number of rows (records) of data in each table must be adequate (at least 10).
 - b) Showing a few rows of data is typically considered insufficient.
- 4) List of all SQL commands used in your app
 - a) Non-advanced SQL commands. Please refer to the Project proposal and UI design page for details and minimum requirements. These include commands to
 - Create tables
 - Retrieve data
 - Add data
 - Update data
 - Delete data
 - b) Advanced PL/SQL commands

II) Sql file **COMMANDS.sql** :

Submit all SQL commands that have been run against your database. To use the DBMS feature, go to your database (using phpMyAdmin or Google Cloud SQL), use the export feature to obtain a .sql file. This file will contain all SQL commands that have been run against your database. Submit this .sql file with your report; it will be used as a reference for grading purposes.

Why are you required to submit a .sql file when you have already documented them in your report? Here are some benefits of a .sql file submission:

- If something happens to your database (failures occur, a database server goes down, the service becomes unavailable, etc.), you can use this .sql file as a backup. You can import it to the database, so you do not need to create and set up the entire database.
- If the size of your data becomes very huge that it degrades the performance or you decide to migrate (or scale up) your database, you can use this .sql file to help you set up a new database.
- If you decide to evolve or enhance some functionalities of your app, you can use this .sql file to set up a separate development environment. This way, the changes will not impact the existing database.
- The teaching team can use this .sql file for grading purposes.

III) Peer evaluation (due when this milestone is due)

- **This is an individual task.** Do **not** include it in the project's report.
- The teaching team will consider this peer evaluation (along with the other peer evaluations and other deliverables) when assigning the project final grade to an individual team member. Each team member's grade may be adjusted by 0%-100% deduction, based on his/her contribution.
- Submit your peer evaluation:
https://docs.google.com/forms/d/e/1FAIpQLSfYSxIUOWoWsGDKaeEqZ1DlclrZF0ED0jPYyRzJf7WSnmc1cQ/viewform?usp=sf_link
 - **Everyone is required to submit the peer evaluation**
- You are required to enter the names and NetIDs of all team members
- Once this form is closed, the form will not be reopened and we have to assign a zero grade to this section of the rubric.

Grading Rubric [Total: 20 points]

- (3 points) — Successfully set up your database on a database server
- (3 points) — Include all tables from milestone 2 in your MySQL database
- (3 points) — Data is populated, practical, and sufficient
- (6 points) — Properly and correctly write all non-advanced SQL commands to be used by your app, including but not limited to SQL commands to
 - Create tables
 - Retrieve data
 - Add data
 - Update data
 - Delete data
- (5 points) — Properly and correctly write at least two different advanced PL/SQL commands
 - Trigger
 - Function
 - Assertion
 - CHECK constraint
 - Stored procedure
- (2 point part of the 5 points allocated for this requirement)— Submit peer evaluation, https://docs.google.com/forms/d/e/1FAIpQLSfYSxIUOWoWsGDKaeEqZ1DlclrZF0ED0jPYyRzJf7WSnmc1cQ/viewform?usp=sf_link (due when this milestone is due)

This is an individual task. Do not include it in the project's report.

Everyone is required to submit the peer evaluation, even though you are working on the project individually.

(- 2 points) — For submitting a report in a Word document or handwriting

(-5 points) — Not submit .sql file

Submission

- **Include all team member names and NetIDs in the report.**
- Save your report as a PDF. **No Word document. No handwriting.**
- Upload the following files to the Project Milestone 3 assignment on Gradescope.
 - Your report file (**REPORT.pdf**)
 - Your SQL file (**COMMANDS.sql**)
- Make sure you connect your partner to your group on Gradescope so that everyone receives credit.
- Each team submits only **one** copy.

Making your submission available to the course staff is **your** responsibility; if we cannot access or open your file, we have to assign a zero grade. Be sure to test access to your file before the due date.