**CPSC 50900 Database Systems Project**

All your efforts this semester will focus on this project to be done BY YOURSELF for which you will develop, document, implement, secure, and program with a database. You will post the **artifacts of work** (XML files, model diagrams, code) as a **GitHub repository**, and you will add to it steadily throughout the semester. You will document all your efforts in a single **Word document** that you'll write throughout the semester, sharing it with me periodically to review and provide comment.  
  
**To summarize: Create a report using Word, and store all of your writing there. For technical documents, post those to a GitHub account and include the link in your report at the end of your proposal.**   
  
The report must have a title page, and the total is worth 10 points for clarity and organization.  
Organizing your files in GitHub by type with clear and unique file names is worth another 10 points.

Your project will consist of the following sections:

**Initial Proposal**

In your report document:

You will describe the data you aim to store. What is your application or business? What data will be storing? What is the value of this data? Where will the data come from? Who will use this data?

Rubric: Your response to each of these five questions will be graded out of 3 points.

* 3 points: clear, complete descriptions that convey the importance and meaning of your data
* 2 points: mostly clear descriptions, although some additional data would have helped in some sections – If it sounds like you are writing without saying anything (“fluff”), then you are losing a point.
* 1 point: necessary details are lacking in many of your responses.

Be aware that this is not the final summary of what you will create in your summary. This is just an overview.

A sport played between two slides by 11 players with a spherical ball, well known as club soccer, soccer or soccer. Soccer is the most popular sport in the world. The

field player moves the ball on any part of the body except the hands or arms while the ball is being played. Only the goalkeeper can use his hand and only in his penalty area. If the ball is out of play on both sides of the field, the opponent who has not removed the ball may return the ball to play with his or her own hands. Both feet must stay on the ground behind the throwing line. Both hands must stay on the ball until it is released behind the thrower's head.

So in this project I intend to create a database which efficiently stores the football data and helps in making better decisions .

Files are located at:  
[www.github/mattsRobots](http://www.github/mattsRobots) (not a real address)

Your report might look similar, but please don’t copy what I wrote.

Total points possible: 15

**Relational Database Design Process**

Identify at least 5 entity sets and their attributes (at least 3 attributes per entity), and describe these entities in your **report**. This will look a lot like the description component of the Aquarium assignment. Then illustrate that you understand normalization and how to use it to reduce uncontrolled redundancy in your database design by creating a single ERD diagram using either UML or Crow’s Foot notation and upload that drawing to **GitHub**. The ERD must be a **physical** model (includes data types and key information).

Rubric: Your work will be graded as follows:

* 4 points for describing each of 5 entities and their attributes in your report
* 10 points for the physical model in GitHub

Total points possible: 25

## 1. Tables

### 1.1. Table Football\_league

#### 1.1.1. Columns

| **Column name** | **Type** | **Properties** | **Description** |
| --- | --- | --- | --- |
| FL\_id | int | PK |  |
| FL\_name | varchar(25) |  |  |
| FL\_year | int |  |  |

### 1.2. Table Football\_Team

#### 1.2.1. Columns

| **Column name** | **Type** | **Properties** | **Description** |
| --- | --- | --- | --- |
| Team\_id | int | PK |  |
| Team\_name | varchar(25) |  |  |
| Coach\_name | varchar(25) |  |  |
| Team\_captain | varchar(25) |  |  |
| set\_of\_players | int |  |  |
| City | varchar(25) |  |  |
| FL\_id | int |  |  |

### 1.3. Table player

#### 1.3.1. Columns

| **Column name** | **Type** | **Properties** | **Description** |
| --- | --- | --- | --- |
| Player\_id | int | PK |  |
| Player\_name | varchar(25) |  |  |
| Skill\_level | varchar(25) |  |  |
| Position | varchar(25) |  |  |
| Team\_id | int |  |  |

### 1.4. Table Game\_played

#### 1.4.1. Columns

| **Column name** | **Type** | **Properties** | **Description** |
| --- | --- | --- | --- |
| Game\_id | int | PK |  |
| opp\_team\_name | varchar(25) |  |  |
| date | varchar(10) |  |  |
| status | varchar(10) |  |  |
| Team\_id | int |  |  |

## 2. Reference

### 2.1. Reference Game\_played\_Football\_Team

#### 2.1.1. Columns

|  |  |  |
| --- | --- | --- |
| [Football\_Team](file:///C:\Users\pc\Downloads\football.html#tableFootballTeam) | 1..\* | [Game\_played](file:///C:\Users\pc\Downloads\football.html#tableGamePlayed) |
| Team\_id | <-> | Team\_id |

### 2.2. Reference player\_Football\_Team

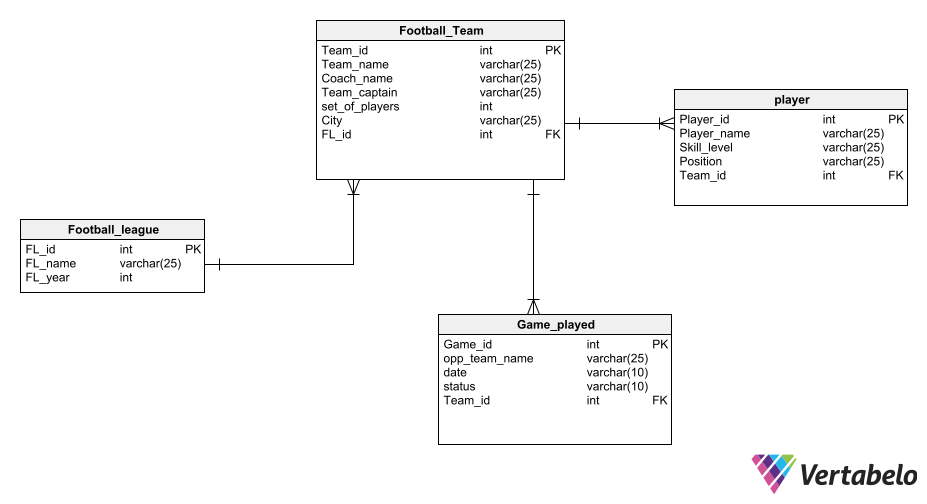
#### 2.2.1. Columns

|  |  |  |
| --- | --- | --- |
| [Football\_Team](file:///C:\Users\pc\Downloads\football.html#tableFootballTeam) | 1..\* | [player](file:///C:\Users\pc\Downloads\football.html#tablePlayer) |
| Team\_id | <-> | Team\_id |

### 2.3. Reference Football\_Team\_Football\_league

#### 2.3.1. Columns

|  |  |  |
| --- | --- | --- |
| [Football\_league](file:///C:\Users\pc\Downloads\football.html#tableFootballLeague) | 1..\* | [Football\_Team](file:///C:\Users\pc\Downloads\football.html#tableFootballTeam) |
| FL\_id | <-> | FL\_id |



**Data Sources**

Create examples (**three** records per entity) of all of your data in a parseable format such as XML, JSON, or delimited like comma separated values and submit the files to **GitHub**. In your **report**, explain the process of how your software stores each entity and attribute, what is the source of the attributes, and what data is default.  You don’t have to describe every attribute, but I want to know at least two details about each entity, and how that data enters the system.  
  
My example would include something like “As the business owner I would manually input my employee data (entity) which includes name, phone number, and address (specific details and attributes) into the database. The database updates the employee ID automatically (a default condition that defines the remaining attribute of my entity). However, I use an arglebargle software tool (and if I see arglebargle in your report, you are losing a lot of points!) to read product information (another entity) and format the part number, description, cost, and type (all the attributes of the product entity) into an XML data structure. All of these attributes come from the supplier. ” Notice, I was specific with these attributes and determined which were default, which were created, and where they all came from.

Rubric: Your work will be graded as follows:

* 5 points per populated file that has attributes which **match your design.** If you are missing one or two, then you lose a couple points. If you miss a lot, you may lose half or all the points.
* 5 points per entity: you described the contents of the data files in detail, including referencing their origin and explaining how they were structured.

Total points possible: 50

* Description of tables (10 points)
  + The tables include football league, football team, player, and game played.
  + The football league includes name and year attributes.
  + The football team includes team name, city, coach name, set of players, game id and football league name attributes.
  + The player table includes player id, team name, player name, position, and skill level.
  + The last table i.e., game played table include game id, host team, guest team, date, and score.
* Description of keys - Primary, foreign, composite

The primary keys are FL name, game id, player id and the foreign keys are fl name and game id which are present in football team table.

* Data types for each attribute (20 points)

As you can see, I’ve mostly chosen int and varchar for my attributes reason being all the ids store integer values and the rest i.e., team name, city, captain, etc. store letters.

* Description of relationships (20 points)

Football League - football team ( one to many)

Football team – game played (many to many)

Football team – player (one to many)

**Data Definition Language Scripts**

OLD INSTRUCTIONS:

First, use Vertabello to generate a script of SQL commands that build the database and its table structures. Or, you can write scripts or build Excel spreadsheets that take your data files and generate scripts of SQL insert statements from them. Use the MySQL *source* command to run the various scripts needed to build and populate the database in MySQL. Include the source code and / or Excel spreadsheets you use to manipulate and populate the data. Make sure all your tables have at least three records in them and that you've linked the tables through their foreign keys.

UPDATED INSTRUCTIONS:

To simplify the instructions, first you need a script that creates the table structures. Then, you need the ‘load’ command to import the data you saved to an XML or CSV file. You might have also used JSON or another format, but you’ll have to research that method yourself.

Assign columns as a **Primary key** or **Foreign key** (where needed) so that if you modify a row in one table, the changes update in the other tables. Obviously, do not assign columns as keys if they are not shared across tables.

In both of the following sections, I offer you CHOICES (by number)– not steps to complete. Pick your favorite ONE, and execute that choice.

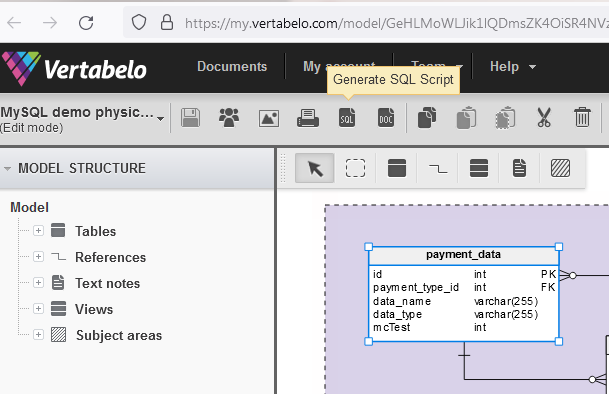
TO CREATE THE TABLE:

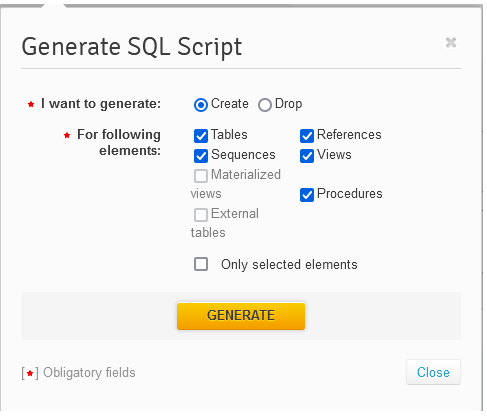
1. Manually write a script with CREATE and VALUE statements. You can either write this at the prompt (can be tedious if you make errors), copy/paste a script into the prompt, or save your script as a .sql file and use ‘SOURCE’ to load your script like you did when you loaded the nation database for the SQL#2 assignment.

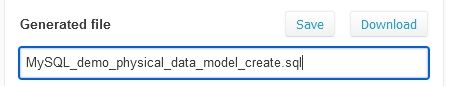
You may feel this option is easier/faster than using Vertebelo. However, if you drew your model in Vertebelo, then the script is just a 4 step point-and-click process.

2. Generate the script using a tool like Vertebelo. Here is the procedure:

a) Create the tables/database:



b) Using the SQL icon above the drawing area, choose to Generate SQL Script: 

c) Click on ‘Generate’ to export the SQL file:  


d) download, and copy the .sql file to your favorite directory where you administrate MariaDB. This is where you use the ‘SOURCE’ command to load the sql file like you did with the nation db example.

TO IMPORT DATA INTO MariaDB:

1. Use “LOAD DATA LOCAL INFILE ‘filename’ “ to load a delimited Excel file (usually a comma separated value, CSV).

<https://mariadb.com/kb/en/load-data-infile/>

You’ll notice the ‘set’ at the end gives you the option to manipulate data during the load. In fact, nearly all of those commands are optional since they are shown within square brackets, [ ]. Thus, this step is very simple! You may also redefine the delimiter character if you are not using a CSV.

2. If you are using XML files to store data, then see this example:

<https://mariadb.com/kb/en/load-xml/>

Again, a single command will load the contents of your file.

Rubric: Your work will be graded as follows:

* Database and table creation statements (manual or Vertebelo) saved as an sql script file. Upload to GitHub. 8 points
* The SQL commands for populating the tables **and** what they do (explain each of the commands and options that you used in the script) documented in your report: 8 points
* Screenshots of your successful attempts to populate each table with at least three records: 4 points

Total points possible: 20

The commands I’ve used in the script are

* Create database <database name> : it is used to create a database.
* Use <database name> : used for changing/accessing the specified database.
* Create table <table name> : it is used to create a table in the database.
* Insert into table <table name> : used for inserting records into the specified table.
* Source <file path> : for executing the statements written in the file.

Text

Description automatically generated

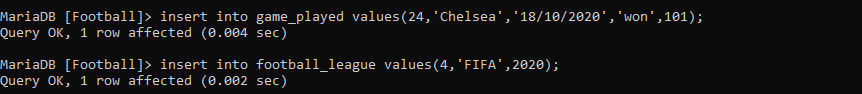
**Data Manipulation Language Scripts**

Write the SQL commands for twelve queries. Two queries should be insert statements, two should update statements, one should be a delete statement, one should be a simple select statement that selects a subset of the rows and columns from one table, two should be a select statements that select data from a joining of two tables, two should use summary functions to generate statistics about the data, one should be a multi-table query, and one should be another query of your choice. Show the queries and screenshots of the results in your Word document **report**, and save your queries in a commented sql script to GitHub. Points for the SQL statements will be scored in the scripts on GitHub, points for the screenshots will be scored in your report.

Rubric: Your work will be graded as follows:

* 1 point each for the two insert statements
* 1 point each for the two update statements
* 1 point for the delete statement
* 1 point for the simple select statement
* 2 points each for the 2 join statements
* 2 points each for the two that use summary statements
* 2 points for the multi-table query
* 2 points for the query of your choice.
* 12 points for showing the query and a screenshot of the corresponding result set back-to-back for each of these queries in your Word document.

Total points possible: 30





A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

**Indexes**

Improve the performance of your design by adding indexes to various tables. Show the SQL needed to add the indexes. Explain why you chose the ones you added. Explain how you would demonstrate the impact the indexes had on the performance of various queries. (what tests could you run? Remember the slow query log?) . You can put all of this information in your **report.**

Rubric: Your work will be graded as follows:

* 6 points for clearly defining at least three indexes and explaining why you chose them.
* 3 points for showing the sql needed to generate the indexes
* 3 points for explaining how you would demonstrate the performance improvement afforded by the indexes.

Total points possible: 12

create index League\_name on football\_league(fl\_name);

create index skill on player(skill\_level);

create index team on Football\_team(team\_id);

Text

Description automatically generated

the indexes will be on league name, skill level of player and football team id. I‘ve chosen them since they will be most searched on like what are the different leagues? How good the player is? What are the different teams? Etc

**Views**

Add two views to your database to provide easy access to combinations of data from multiple tables. Document the work and screenshots in your **report**.

Rubric: Your work will be graded as follows:

* 2 points for including the SQL for generating the two views in your Word document
* 2 points for including screenshots for the data contained in each view in your **Word document**
* 2 points for explaining why each view is a valuable addition to your database
* 2 points for explaining who might benefit most from having access to each view.

Total points possible: 8

create view Games\_Status as

select ft.team\_name, gp.opp\_team\_name, gp.date, gp.status from football\_team ft

inner join game\_played gp on gp.team\_id=ft.team\_id;

Text

Description automatically generated

create view team\_captains as

select fl.fl\_name, t.team\_name, t.team\_captain from football\_league fl

inner join football\_team t on t.fl\_id=fl.fl\_id;

Text

Description automatically generated

The 1st view shows the matches won/lost by different teams. It is essential since it can be a measure of team’s performance. While 2nd view shows the captains of every team. It can be useful to know different captains and who has changed.

Both views will be useful for the football association manager.

**Triggers**

Add a trigger to a table so that data will be updated when a certain event occurs. Document your triggers and screenshots in your **report**.

Rubric: Your work will be graded as follows:

* 2 points for including the SQL for the trigger in your Word document
* 2 points for clearly explaining the purpose of the trigger
* 2 points for a screenshot and explanation that shows the trigger in action.

Total points possible: 6

**Transactions**

Demonstrate that you know how to define and use a transaction. Why are transactions important for ensuring ACID behavior? Document all of this in your **report**.

Rubric: Your work will be graded as follows:

* 3 points for clearly explaining the importance of transactions to ensuring ACID behavior
* 3 points for including a screenshot and accompanying explanation of a MySQL transaction.

Total points possible: 6

Transactions are an important aspect for maintaining the acid properties of the database.

The acid properties are

Atomicity : all operations did while in the transactions will be executed in a single step.

Consistency : the data will be consistent with every transaction.

Isolation : each transaction will be isolated from other transaction.

Durability : the data will be durable after the execution of a transaction.

Text

Description automatically generated

**Security**

Identify the different kinds of users who will use your database. Write GRANT statements to define the privileges for these different kinds of users. Document all of this in your **report.**

Rubric: Your work will be graded as follows:

* 6 points for clearly identifying and describing the various kinds of users who will use the databases and identifying and justifying what privileges each should have.
* 4 points for writing GRANT statements that assign privileges to these different kinds of users.
* 4 points for demonstrating with screenshots that your GRANT statements do distinguish among different kinds of users in regard to what they can do with the database.

Total points possible: 14

The different users that use this database are football association manager who has total control over the database and

Public user who can only see the contents of the table.

GRANT all on Football.\* to ‘football\_manager’@’localhost’;

GRANT select on Football.\* to ‘user’@’localhost’;

Text

Description automatically generated

**Locking**

Explain the purpose of locking tables and show how to do that to prevent inconsistencies that may arise in your data when concurrent transactions take place. Document all of this in your **report.**

Rubric: Your work will be graded as follows:

* 3 points for clearly explaining an example that shows why you should lock tables to prevent inconsistencies.
* 3 points for providing a screenshot and accompanying explanation of locking tables.

Total points possible: 6

If two updates are being done on a same player record then an inconsistency with that record will occur. Hence to avoid that we use transactions.

Text

Description automatically generated

**Backup**

How you will back up your database. What commands will you issue? How frequently will the commands run? How can they be automated? Where will the backups be stored? Document all of this in your **report.**

Rubric: Your work will be graded as follows:

* 12 points for clearly explaining and justifying your database backup strategy, including the frequency with which you will back up the database, how you will automate backups, where you will store them, and how you will secure them. You will earn three points for addressing each factor (frequency, location, automation, and security)
* 3 points for providing a screenshot of the command you would issue to back up the database.

Total points possible: 15

Frequency - for every 200 operations on the database/tables.

Location - in the cloud or separate hard disk.

Automation - by creating a .bat script and scheduling it using windows scheduler.

Security - I would secure it using the various encryption algorithms.



Graphical user interface, text

Description automatically generated

**Python Programming**

Write a Python program that generates a report that contains a subset of the data from your database. Include the code for your Python program in your Word document **report**, and also post the program to your **GitHub** repository.  
  
The report is just a SELECT \* command that displays the contents of each table. If you have many records, you might want to look into the LIMIT option to restrict your output. Since the previous parts of this assignment specify 3 records, your output should be anywhere from 3 to 10 records per table.

Rubric: Your work will be graded as follows:

* 12 points for writing a Python script (and including its code in the Word doc) that will pull data from a database and store it to a text file and present it to the screen. Your code (the .py file) must have comments in it that explain how it works. You will be awarded 3 points for successfully connecting to the database, 3 points for successfully querying it, and 4 points for presenting the data to the screen and to a file. Internal comments count for 2 points.
* 2 points for posting the code to GitHub
* 4 points for showing a screenshot of your running the script and showing the results it produces on the screen.

Total points possible: 18

**--**

import mysql.connector  
from mysql.connector import errorcode  
from mysql.connector.errors import DataError  
  
*# Method to get the records from the database to store in a file and to present it.*def WriteRows(conn,Query,file):  
 cur = conn.cursor()  
 cur.execute(Query)  
 q = Query.split(' ')  
 file.write(q[3]+" Table:\n")  
 print(q[3]+" Table:")  
 for rec in cur.fetchall():  
 line=""  
 for r in rec:  
 line+=str(r)+","  
 file.write(line+"\n")  
 print(line)  
 file.write("\n")  
 print("\n")  
  
try:  
 *# Connecting to Database* conn = mysql.connector.connect(  
 user = "root",  
 password = "maria@123",  
 host = "localhost",  
 database = "football"  
 )  
  
 *# Opening a file to write* file = open('football\_tables.txt','w')  
 print("database and file are opened ")  
  
 *# Querying the Tables* WriteRows(conn,"select \* from football\_league",file)  
 WriteRows(conn,"select \* from football\_team",file)  
 WriteRows(conn,"select \* from game\_played",file)  
 WriteRows(conn,"select \* from game\_played",file)  
 print("Football tables are created in the football\_tables.txt file")  
   
except mysql.connector.Error as err:  
 print("database didn't connect")  
 exit()  
except IOError:  
 print("File not found or path incorrect")  
 exit()  
finally:  
 conn.close()

output:

**Graphical user interface, text, application

Description automatically generated**

**BONUS POINTS:**

If you complete all of the other sections, then you may try this section for bonus points. Feel free to do the PHP or the “Suggested Future Work”, or both!  
 **PHP Programming**

Build an HTML form that enables the user to specify criteria to search by. Use PHP to show the results of the query on a resulting web page. Make sure you include protections against an SQL injection attack. Include your HTML and PHP code in your Word document, and also post the files to your GitHub repository.

Rubric: Your work will be graded as follows:

* 4 points for writing an HTML form the user will use to enter search criteria
* 8 points for a PHP script that uses the search criteria and returns results
* 4 points for an HTML page that shows the results
* 4 points for explaining what SQL injection might be run on your website and explaining how you prevented it.
* 4 points for providing screen shots of your PHP website in action.
* 2 points for posting your code to GitHub

Total points possible: 26

**Suggested Future Work**

Describe the limitations of your current database and explain how you or someone else could improve the design to address these shortcomings. Also describe how you might take advantage of leverage cloud services to increase the performance and availability of your database. Finally, explain the advantages and disadvantages of storing your data in a NoSQL format instead.  
  
This requires researching what NoSQL is, what makes it different from MySQL, explaining how the 2 systems are different, and providing APA citations of your sources. You don’t need to create a separate bibliography or references section, but do provide the formatted source at the bottom of the document.

Tip: Purdue has an automatic citation generator page that is incredibly useful for this function.

Rubric: Your work will be graded as follows:

* 3 points for clearly describing the limitations of your database design
* 3 points for explaining how you would address these shortcomings
* 3 points for explaining how you might migrate the database to the cloud and describing what advantages you might gain from doing that.
* 3 points for explaining the advantages and disadvantages of storing your data in a document-based NoSQL format instead.
* 2 points for the source references.

Total points possible: 14

Grand Total:

245 + 40 bonus