Requirements Document

IT6037 Data Access and Management

Team: SWESLBS

# Business Requirements

## Introduction

The school Education Power Plus has been around for a short time now and is still setting up all the classrooms as well as the databases and procedures that are needed to run a fully working school. The school is only running three subjects right now which are: art, mathematics and technology.

This is just the start for Education Power Plus, as soon as more classroom are finished and more teachers are hired, other subject can be added in.

## Statement of Problem or Need

The problem of students not able to access articles on the subjects/topics that they are studying, who must search for the articles which may not exist anymore, and our tutors, who spend significantly more time copying and printing articles out for students, the impact of which is more time used by students and tutors which could be spent doing other tasks, and increased cost as tutors need to work more hours. A successful solution would be to implement a digital database that students can search, and tutors can add or modify articles.

## Business Requirements

### List of stakeholders

The **(CPP)Tutor** who is the client

The **Students** who read the articles

The **Tutors** who add/modify the articles

The **Administrators** who add/remove/modify the articles

### Client

(CPP)Tutor

### List of business requirements

* Students should be able to browse articles by category
* Students should be able to browse articles by a keyword in the title
* Tutors should be able to add articles
* Tutors should be able to modify articles
* Administrators should be able to add articles
* Administrators should be able to remove articles
* Administrators should be able to modify articles

### Quality requirements

List of non-functional requirements, i.e. security requirements and other e.g. quality requirements (e.g. availability, portability)

Availability

* Be available 99.999% of the time
* Maintenance outages to be done between 2am and 4am, weekdays

Security

* Must be able to protect all information
* Must prevent and protect the system from security threats and attacks

Usability

* Should be clear and easy to use
* Web accessibility – those with disabilities being able to use the site easily

Portability

* Be available and run on all different types of devices, operating systems and browsers (e.g. PC, mobile phone, tablet, Android, Mac OS, IOS, Google Chrome, Firefox, etc)

# Business Solution

## Options considered

* MongoDB  
    
  Mongo is a document database rather than a relational database, it keeps all of the related information contained in the same object. Mongo is simply designed to be extremely friendly to developers. The JSON representation makes it easy to parse thought and understand the database contents. Mongo has no predefined schema, which can store different types of data in the same collection. However, working without a schema introduces the potential for data integrity issues. The client needs to be responsible for making sure the data is correct. It’s more difficult to impose restrictions on the data in the collections when you don’t have a schema.
* MySQL Server  
    
  MySQL is a relational database management system. It uses a client server model meaning the database typically runs on a server, and the data is accessed over the network by clients and workstations. MySQL uses the structured query language to define, update and query the database system. MySQL allows you to present data in extremely precise ways using queries. Queries allow you to store less data. In a typical database, a piece of data is stored in only one place and retrieved by queries when needed. When storing data in MySQL, you’ll have to choose the type of each piece of data as well as how it’s indexed.

## Recommended Solution

### Solution statement

For the tutors and students who is reading or using articles related to the subject or topic they are studying/teaching, the MySQL a relational database management system is a solution, that presents data in a precise way using queries, and is easy to set up different access levels for different users, unlike MongoDB which is more difficult to impose restrictions on the data. MySQL is also popular and well-established, almost any problem will be answered online, often by multiple people and in great detail.

### Main Features

* Relational database system
* Client/server architecture
* SQL compatibility
* Stored procedures
* Replication
* Triggers

### Unique value proposition / Justification

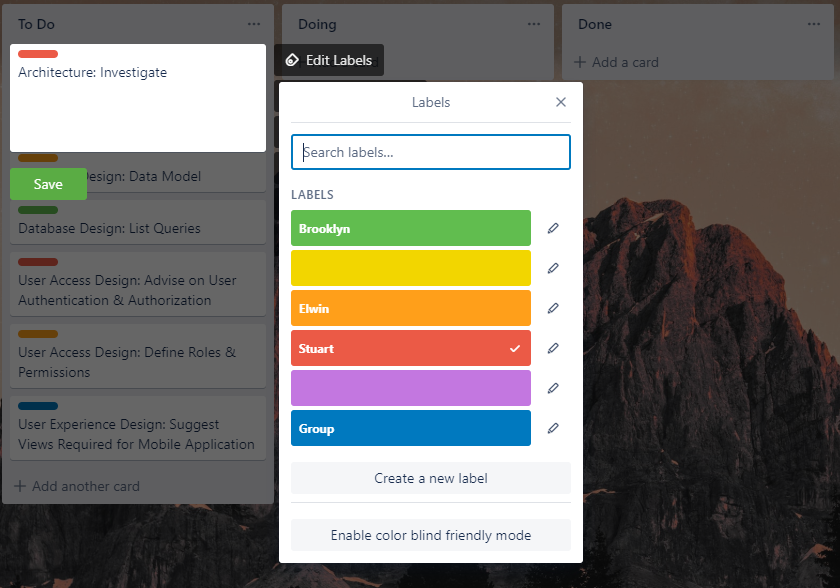
* There are several convenient user interfaces for administering a MySQL server.
* Full-text search simplifies and accelerates the search for words that are located within a text field.
* Replication allows the contents of a database to be copied onto a number of computers, to increase protection against system failure and to improve the speed of database queries.
* There are quite a number of application programming interfaces and libraries for the development of MySQL applications

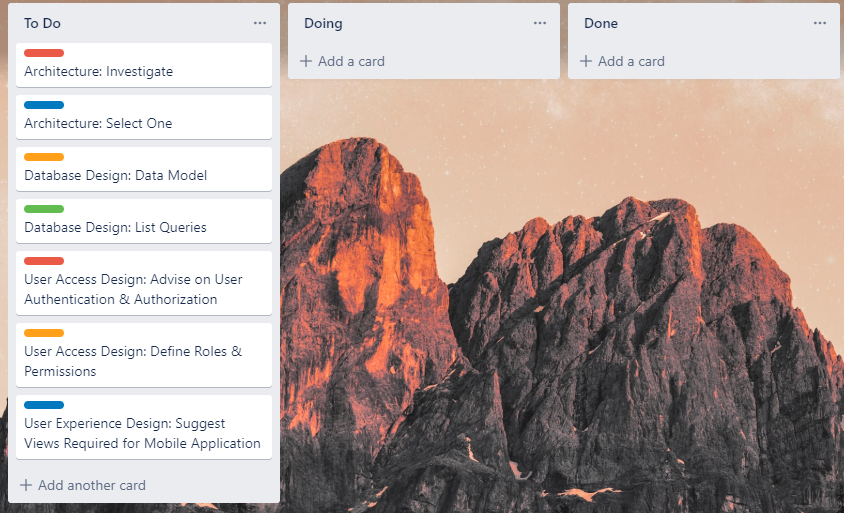
# Scope

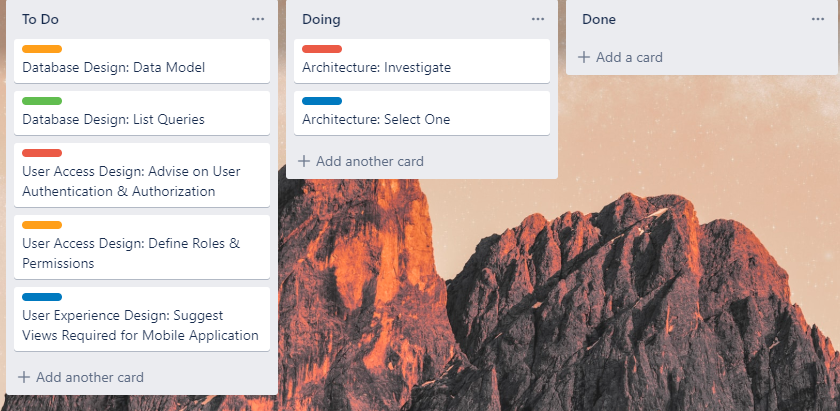
|  |  |
| --- | --- |
| **Iteration** | **Aimed Finish Date** |
| Iteration 1 | End of the 18/09/2019 |
| Iteration 2 | End of the 19/09/2019 |
| Iteration 3 | End of the 23/09/2019 (includes testing/debugging/fixing) |
| Iteration 4 | End of the 24/09/2019 |

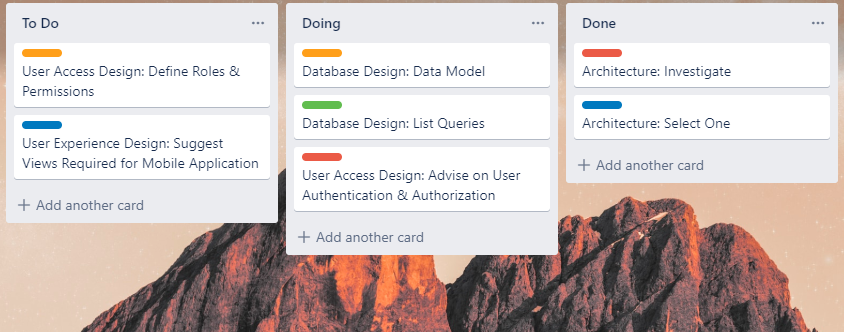
## Iteration 1: Developing Systems Design document

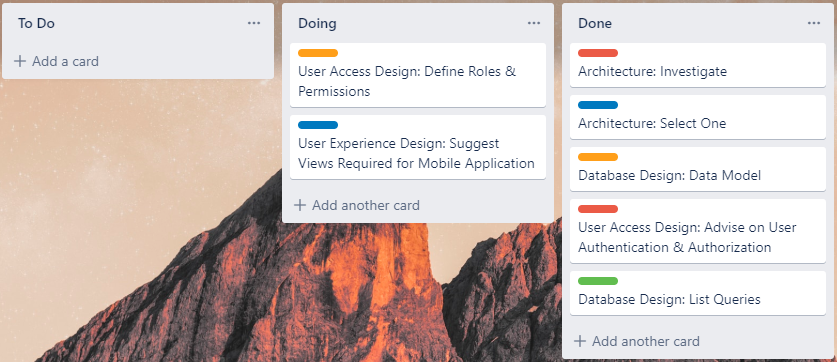
* Architecture: Investigate possible database management system.
* Architecture: Select one that meets the requirements and justify the selection.
* Database Design: Design data model with main entities, data type and relationships.
* Database Design: list appropriate queries.
* User Access Design: Advise on user authentication and authorisation.
* User Access Design: Define user roles and permissions.
* User Experience Design: Suggest views required for mobile application.

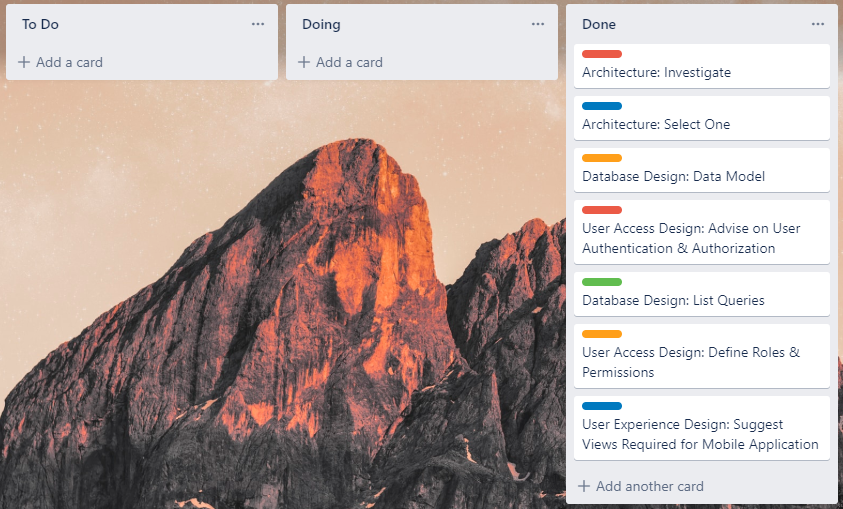






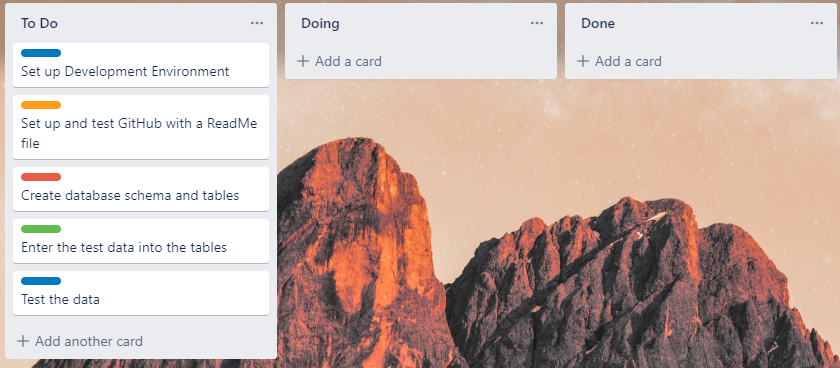


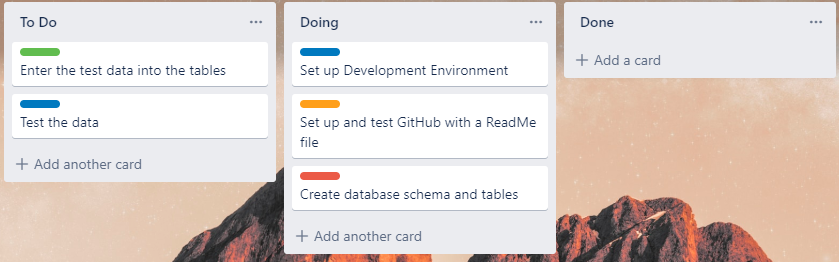


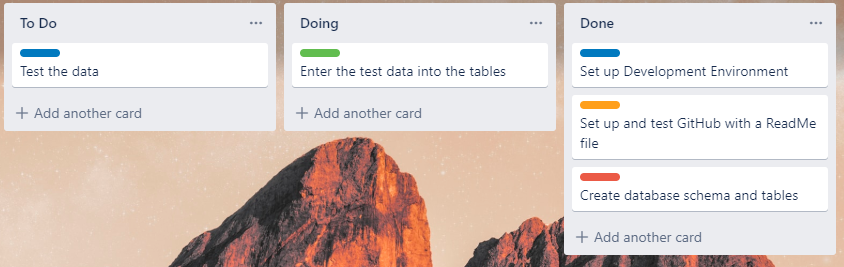


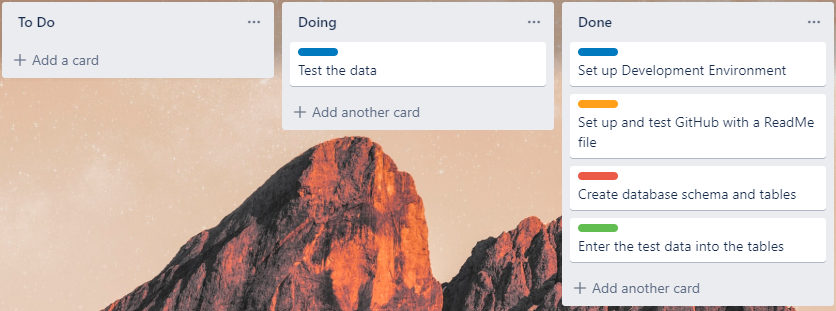
## Iteration 2: Developing the Database

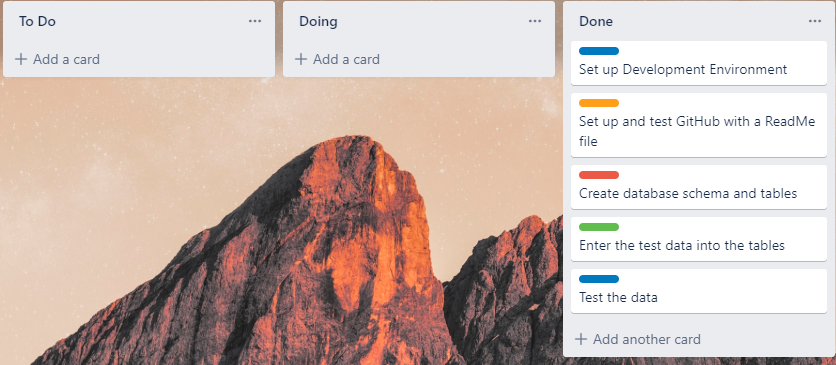
* Set up development environments.
* Set up and test GitHub repository with a ReadMe file.
* Create database schema and tables.
* Enter the test data into the tables.
* Test the data displayed correctly.





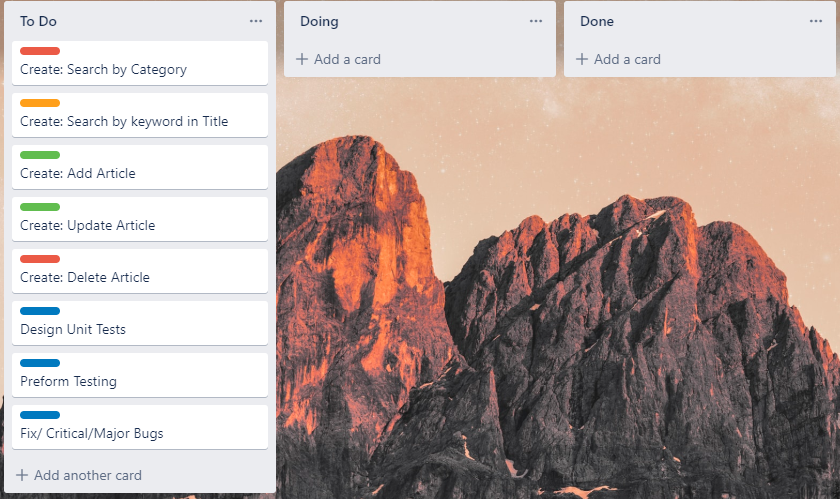




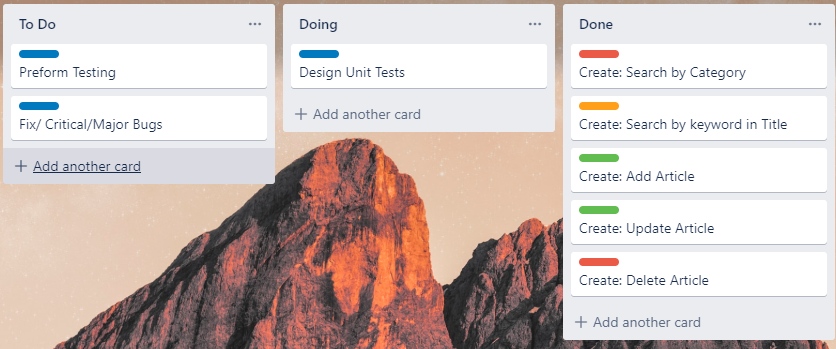


## Iteration 3: Developing Database Queries

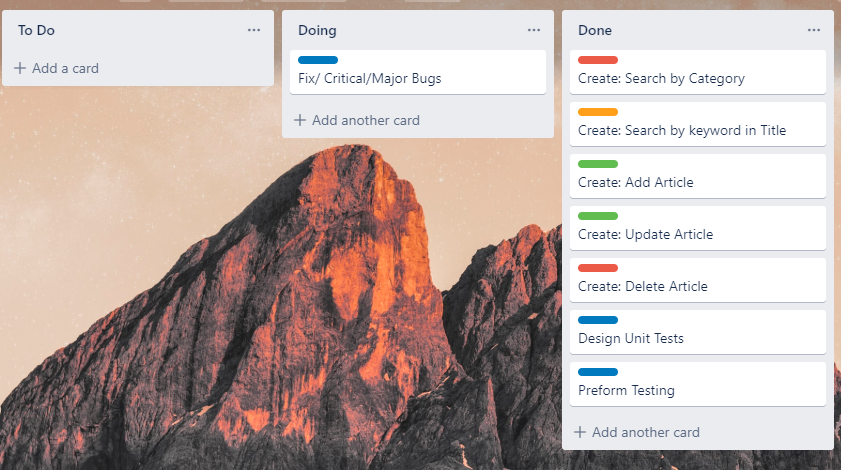
* Design queries to support user stories per client's requirements  
  1. Search by Category  
  2. Search by keyword in Title  
  3. Add articles  
  4. Update articles  
  5. Delete articles
* Design unit test for the queries
* Preform functional testing using a test plan
* Fix critical/major bugs

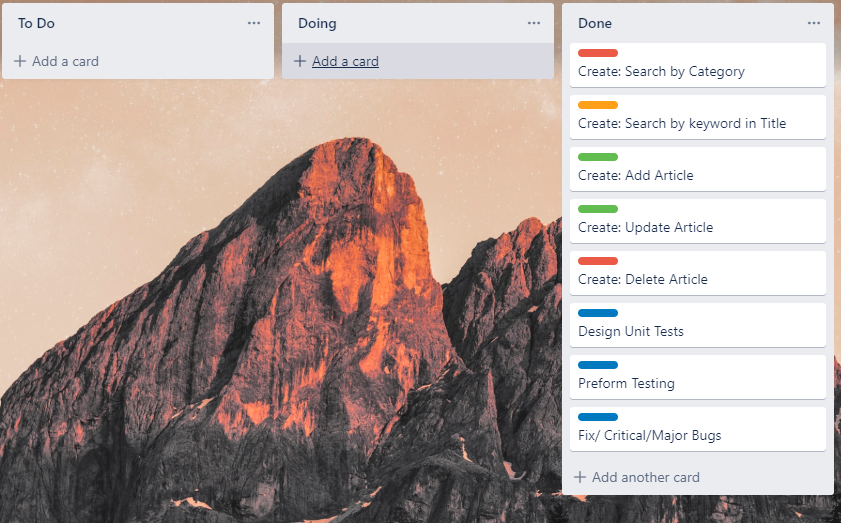








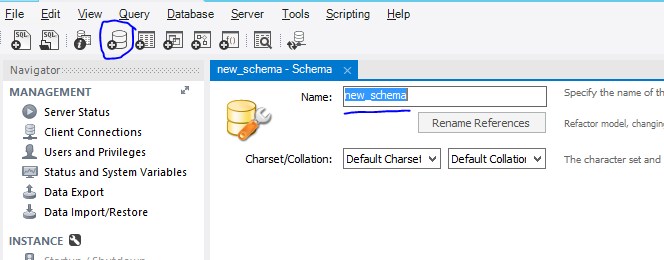




## Iteration 4: User Access Implementation Suggestions

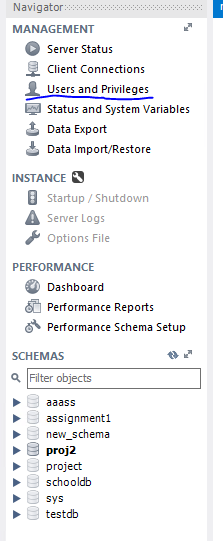
* Describe the process to follow to create users and assign permissions in MySQL

Once you have opened and logged into MySQL Workbench, the first thing to do is create a schema. To create a schema, click on the icon circled in blue. After that, rename the schema to suit the database.

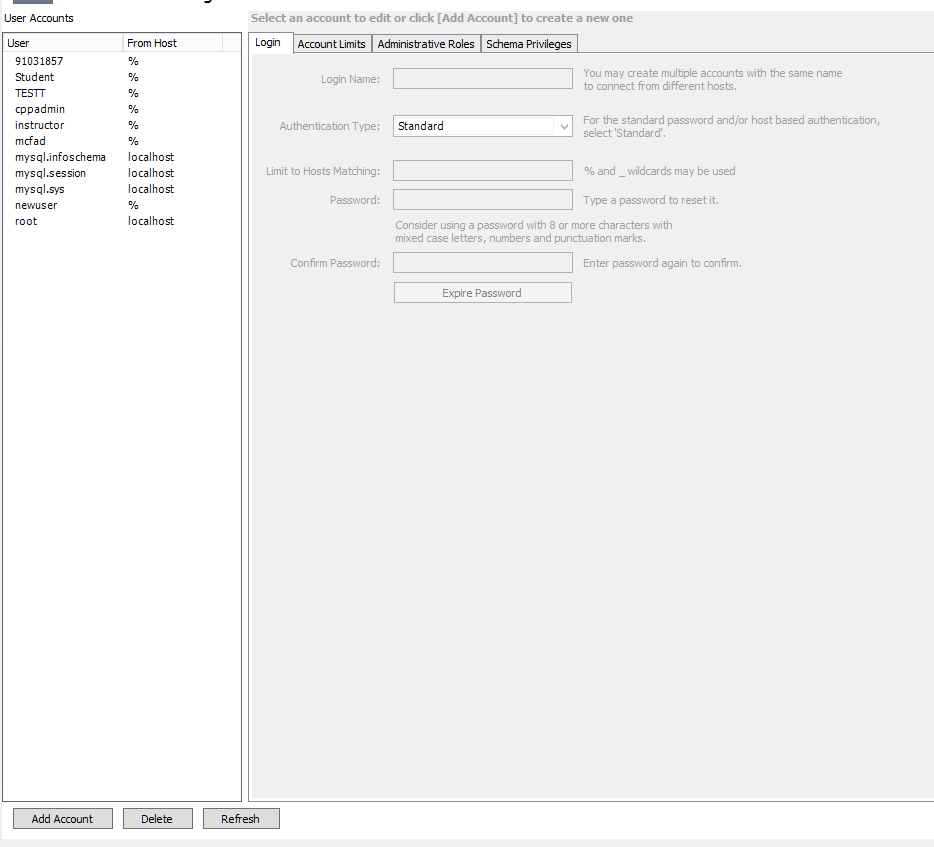


Then, click apply at the bottom right, which will prompt you with another screen showing you the DDL (Data Definition Language) of the SQL script you just wrote. Click apply, then finish.

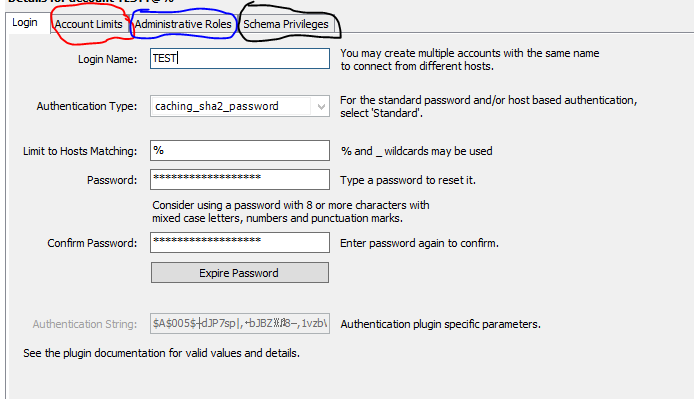
Once you have created the schema, on the navigator sidebar located on the left, click on ‘Users and Privileges’.



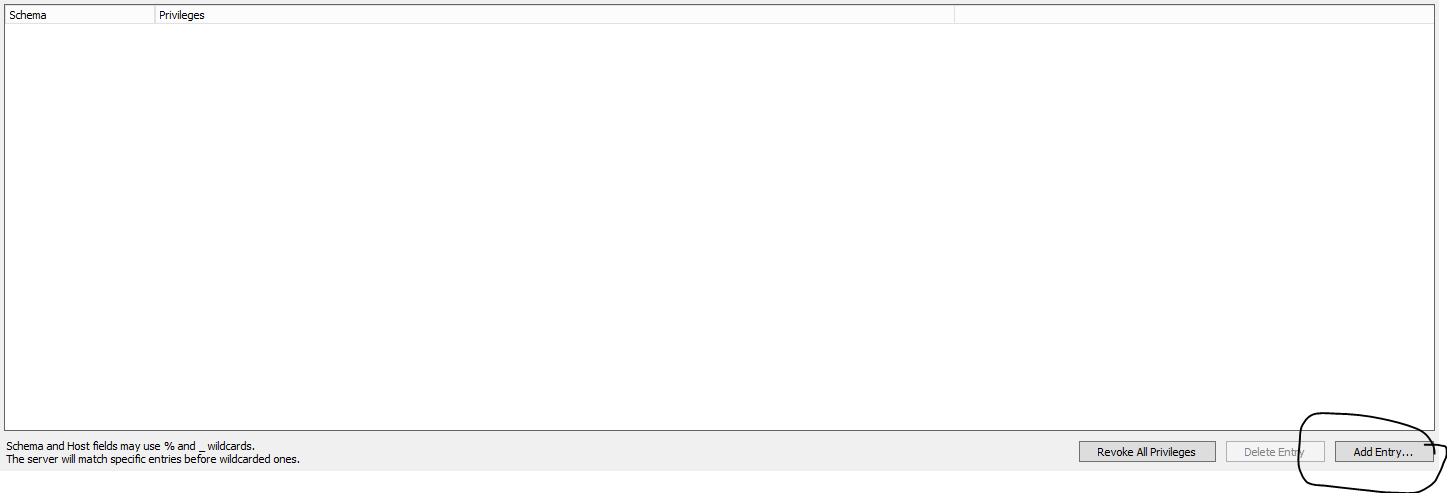
This will then prompt up on your screen.

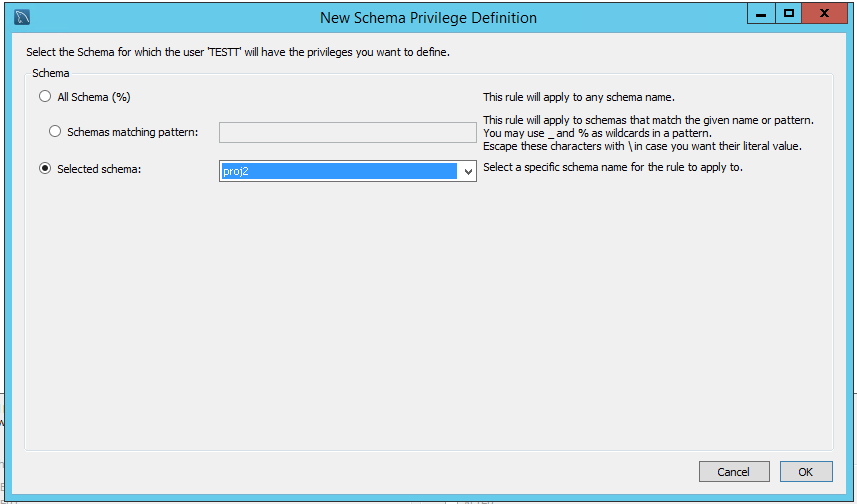


Click on ‘Add Account’, at the bottom of User Accounts. This will enable you to edit the information on the right side. Here is where you create your users.



First you will give them a login name. This is what the user will use if they want to log into the database. After that, you assign them with a password so that they can log onto their account. After that, click apply at the bottom right.

Before you can assign privileges to your users, they first must be able to access a specific database. You can set this up by clicking on ‘Schema Privileges’ circled in black.To start, click on ‘Add Entry’ circled in black, which will then prompt this up:

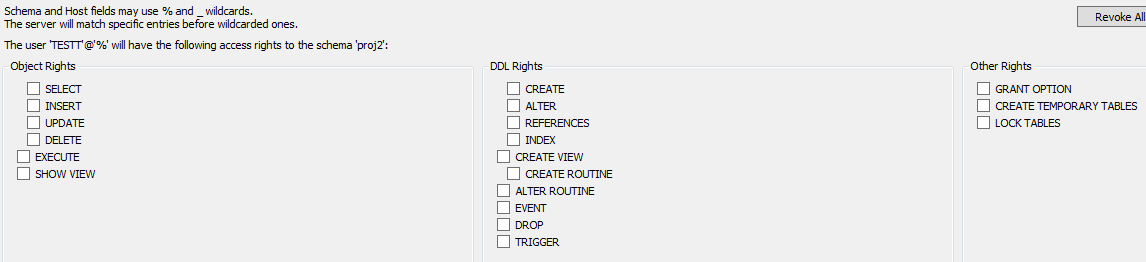


In here, you can specify what type of schema the user can access. There are 3 options. All Schema (%), Schemas matching pattern: and selected schema:

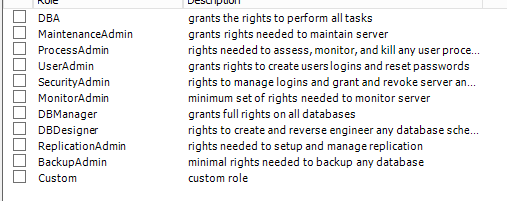
All Schema (%) means the user can access the schemas and any future schemas that are created in the database

Schemas matching pattern means that the user can access all the schemas that match the specific pattern stated.

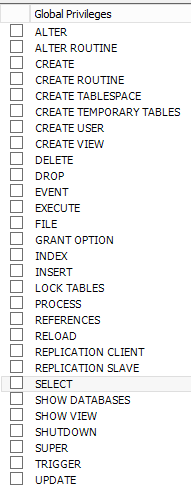
Selected Schema means you can select a specific schema that the user can have access to. You can only assign 1 schema to the user with this option, so if you want the user to have access to multiple schemas but also be restricted to not viewing others, using schema matching pattern is best suited.

Once you have selected what type of schema privilege you chose, click ‘OK’ the bottom right. After this, you can then select what type of rights the user can have access to within the schema and hit apply once you are done.

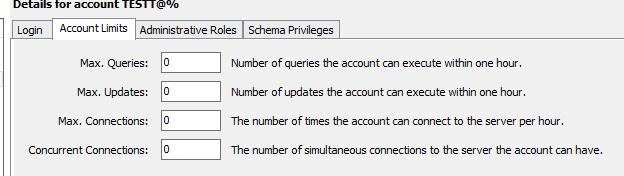
The next tab, ‘Administrative Roles’ circled in blue is where you can assign the user with what specific actions, they can perform. Below shows the different roles and the description on what they do.



Alternatively, if none if these roles suit what you are looking for, you can select the ‘Custom’ role and specifically pick out the restrictive privileges you want to assign your users to. You can do this by clicking on the specific privileges down below.



After this, click on ‘Account Limits’ circled in red. When you click on it, it will show you the following:



This tab is for restricting users on how many queries, updates they can execute. How many times they can connect to the server with that account every hour, and how many times a user can connect to the server on multiple computers with the same account.