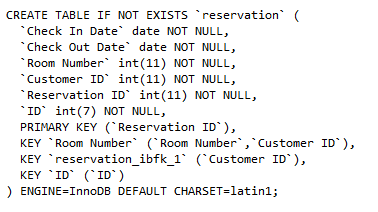
*INTRODUCTION*

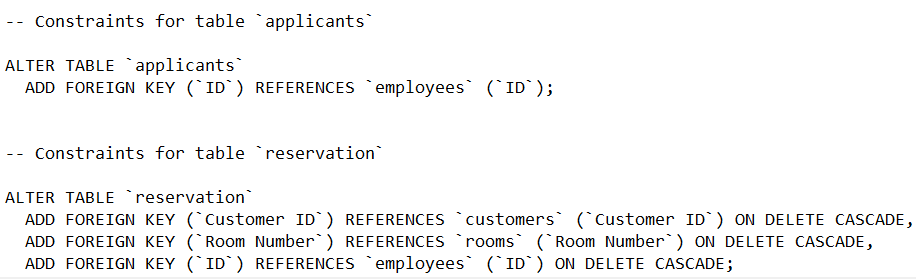
Our aim is to design and create a database suitable for a hotel to use in its daily operations. This database needs to store data concerning guests, reservations, rooms, employees, and employee applications. In addition to storing the data, the database must also be designed with constraints such as primary and foreign keys in mind. The key thing to keep in mind is how all the data relates to the other data within the database. We will achieve the goal of designing and structuring this database using relational design principles and helpful database creation tools. Included in this report are a description of our work, a list of tables used in the database, a diagram showing the relationship between the tables in the database, a user manual, and a conclusion.

*DESCRIPTION*

First, was the creation of the tables, the following code fragment shows how the reservation table was created. This table has columns that are created right after creating the table for which the data type was specified and whether or not it was going to accept null values. After creating the columns, the "Room Number" was set as the primary key and the columns that would be Foreign Keys were specified to establish the relationship between this table and the following tables (clients, rooms, employees). Tables and their attributes were created in this same manner for customers, reservations, rooms, employees, and applicants.



Here you can see how the constraints were created on the applicants and the reservation tables. The constraints establish the relationship that will exist between the tables, for example, the relationship between the reservation and the customers tables matches the primary key in the customers table (customer ID) with a foreign key in the reservation table (customer ID).



**Applicants** **Table**:



**Customers** **Table**:



**Rooms** **Table**:



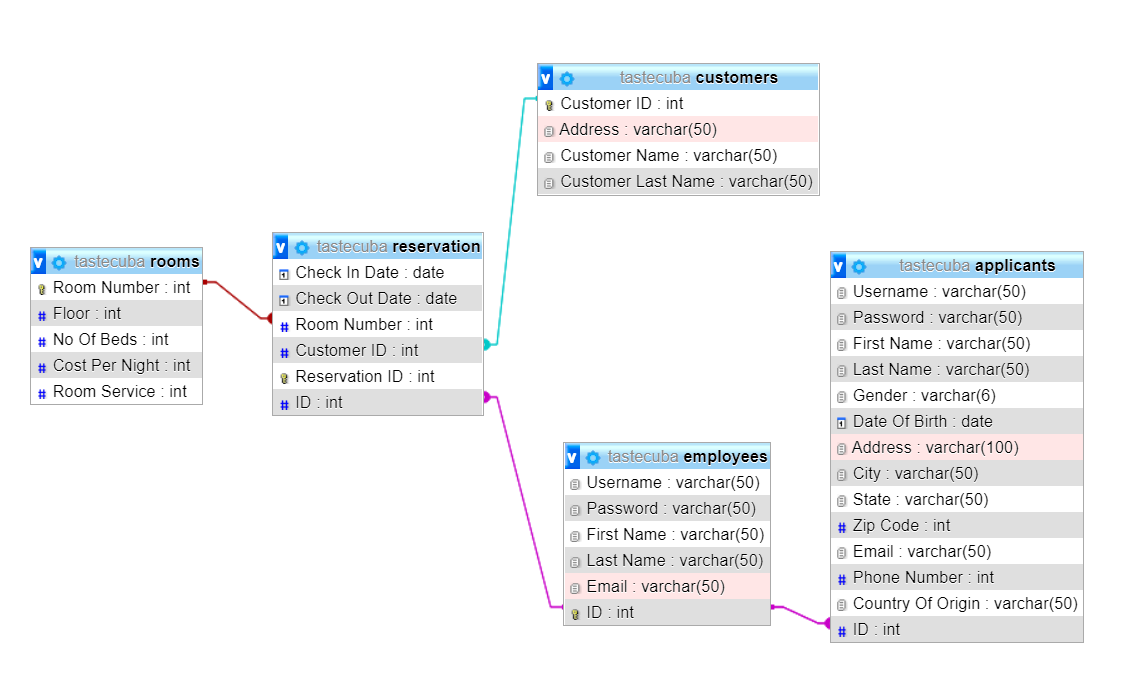
**Employees** **Table**:



**Reservation** **Table**:



**Relational Diagram**



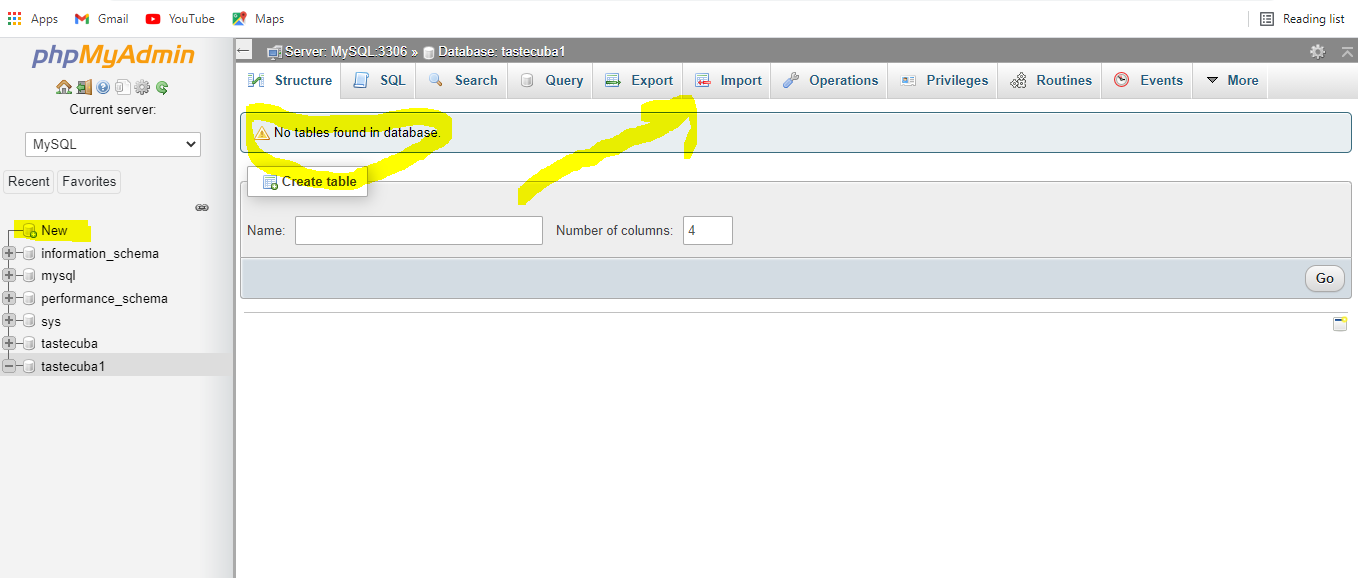
*USER MANUAL*

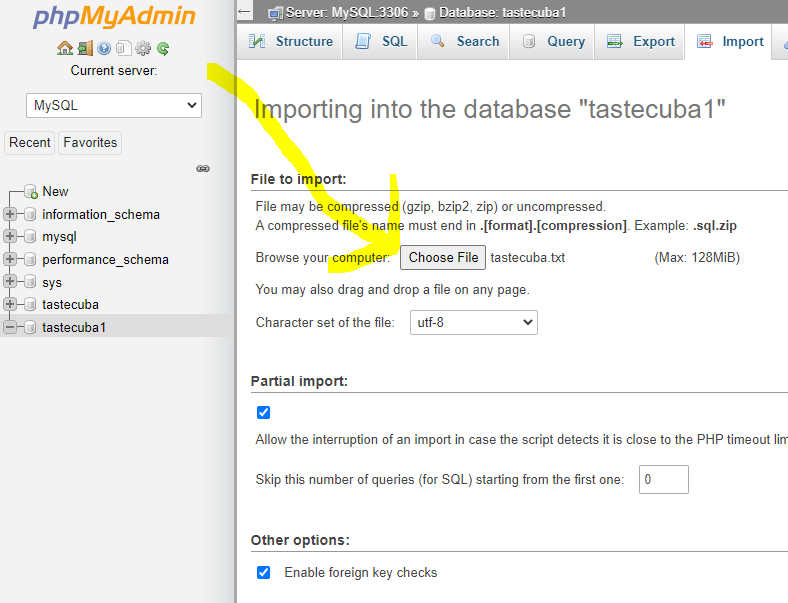
Follow the following steps to run the attached SQL scripts to implement our Relational diagram in MySQL database using phpMyAdmin.

1. Download the .zip file to a location where you can find it
2. Right click the .zip file and click Extract All…
3. The folder is now unzipped and the files within can be accessed.
4. Then, open the phpMyAdmin.
   1. If you don’t have WAMPSERVER installed on your PC you can follow the steps described on the following video

[How to Download and Install Wamp Server on Windows 10 - Bing video](https://www.bing.com/videos/search?q=how+to+download+wamp+server+in+windows+10&view=detail&mid=0912C2137CBF32749AB10912C2137CBF32749AB1&FORM=VIRE)

1. Once you are in phpMyAdmin, you need to create a new database where all the features declared in the script will be stored.
2. Next, you are going to import the document that you previously extracted from the .zip file.





1. Once you have selected the file as shown above and clicked go, you will see the confirmation that the import has been successfully finished and you will see the tables created as follows.



*CONCLUSION*

We achieved our goal of creating a database for a hotel. The various tables, diagrams, and scripts we designed resulted in the successful creation of a database. We learned a great deal about relational design as we created our database. There were specific challenges around preventing the tables from violating cardinality rules which taught us about the importance of design a database thoroughly before attempting to implement it. These challenges were overcome by making some minor corrections throughout its creation. It is possible that the database will need additional corrections or changes as it is used and expanded. The key here is that the database was created in such a way that it can be expanded easily in the future. As a business’s needs change, they ought to be able to adapt their databases to those needs. This was kept at the forefront of our thought as we developed this database. Especially because we knew we will be working on expanding and perfecting it soon.