**SQL Cheatsheet-MIDTERM**

This is a quick reference you can use when creating your database using mySQL.

**Database Creation**

You will need to create your own database from the ground up for this project. After you have logged into mySQL

mysql -u root -p;

execute a create database command naming your database appropriately.

mysql > CREATE DATABASE testdb;

In this case we now have a database named myFirstDatabase in which we can add tables and data as needed.

**User Creation**

You can create sql users that have different permission levels on specific databases. Root will always have admin responsibilities so it is good practice to create a user specific to each application that has slightly lower permissions.

CREATE user 'test'@'localhost' IDENTIFIED BY 'wombat1';

* This command creates a user test with password wombat1

GRANT SELECT, INSERT, UPDATE, DELETE ON testdb.\* TO 'test'@'localhost';

* This command grants SELECT, INSERT, UPDATE, DELETE privileges for testdb to a user named test. testdb.\* means all tables within the testdb database.
* It is good practice to not give the application user permission to alter/drop/create tables.

**Table Creation**

Once you have created your database select the database using the use command:

mysql > USE testdb;

You are now able to manipulate the testdb as you see fit. To create a table in your schema use the CREATE TABLE command. With this command you will have to specify the column names as well as the datatypes associated with the data inside this column. For example in your employees table the following command was executed to create the departments table:

CREATE TABLE `departments` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(30) NOT NULL,

`manager\_id` int(11) NOT NULL,

`location\_id` int(11) NOT NULL,

PRIMARY KEY (`id`),

FOREIGN KEY (`manager\_id`) REFERENCES `employee` (`id`),

FOREIGN KEY (`location\_id`) REFERENCES `location` (`id`)

)

* Alternatively we can set the foreign keys with alter table statements.

ALTER TABLE departments ADD CONSTRAINT department\_mgr\_fk

FOREIGN KEY (manager\_id) REFERENCES employee (id);

ALTER TABLE departments ADD CONSTRAINT department\_loc\_fk

FOREIGN KEY (location\_id) REFERENCES location (id);

ALTER TABLE departments ADD CONSTRAINT departments\_mgr\_u UNIQUE (manager\_id);

**Adding info to Tables**

Once a table is created you can insert data into the columns using the insert into statement as follows. When inserting you can omit columns that are not specified as NOT NULL in the table creation.

insert into departments (name, location\_id) values ('Research', 4);

insert into departments (name, location\_id) values ('Administration', 2);

insert into departments (name, location\_id) values ('Software Development', 3);

insert into departments (name, location\_id) values ('Test And Integration', 3);

insert into departments (name, location\_id) values ('Sales', 1);

insert into departments (name, location\_id) values ('HR', 2);

insert into departments (name, location\_id) values ('Operations', 1);

insert into departments (name, location\_id, manager\_id) values ('Documentation', 2, 1035);

**SQL Dump Files**

Once your database is build it is important to keep a record of the database in a working state. If changes are made that break the table you can simple drop the database and rebuild it using a sql dump file. We have used mySql dump files before when we created the testdb database. Remember this command?

mysql -u root -p < testdb.sql

This executed the dump file, that was generate by the database creator, and rebuilt the test database in your local mySQL based on the file that was exported. Once you have your projects database in a working state you would like to preserve you can create one of these files in your terminal execute the

mysqldump -u root -p testdb > dumpFile.sql

This command will create a file called dumpFile.sql in your current directory with auto generated create and insert statements based on your constructed database. Save these files somewhere so you can rebuild in case of errors, or when you are looking to deploy your db on a server.

**Resources**

The mySQL docs are a valuable resource to get the proper syntax for the commands you are looking to execute. You can find them [here](http://dev.mysql.com/doc/). Also googling specific questions like "how to insert into a mysql database" will often lead you to the desired section of this documentation.