Technological Institute of the Philippines Quezon City Campus		
FirstName, LastName	Subject and Section	Date:
Activity No. 2	Database Management Systems 1	Topic: Retrieving Data from MySQL Database

Introduction: A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

Using the default SAKILA database, we'll try to learn and apply the basic SQL statements to execute actions you need to perform on a database.

Document all output of this activity on a Microsoft Word similar to this format. Save it as ***DBMS1_ACTIVITY2_LastName_FirstName*** in FTP://10.6.15.222 . Look for the folder named ***DBMS1_ACTIVITY2.***

STEP 1 – Connecting to the DATABASE. Connect to SAKILA database. Set SAKILA as the default schema. The syntax for selecting the database is USE *database_name;*

STEP 2 – SELECTING ALL RECORDS. This is done by using the wildcard `*`. Syntax for selecting all records on a table is: SELECT * FROM `*database_name*`.`*table_name*`;

a. Show the output for selecting the actor table
b. Show the output for selecting the address table
c. Show the output for selecting the category table
d. Show the output for selecting the city table
e. Show the output for selecting the country table
f. Show the output for selecting the customer table
g. Show the output for selecting the film table
h. Show the output for selecting the language table
i. Show the output for selecting the staff table
j. Show the output for selecting the store table

STEP 3 – OTHER WAYS OF SELECTIONS. Sample syntax is done in the following: SELECT *column_name1, column_name2* FROM *table_name;*

a. Show the output for selecting last_name and first_name from table named actor
b. Show the output for selecting last_name, email and create_date from table named customer
c. Show the output for selecting title and release_year from the table named film

We can also show one column by selecting only a column name from the table name:

a. Show the output for selecting column description from table film, limit the result to 10; this will show that only 10 rows are retrieved
b. Show the output for selecting column country from table country; limit the result from 10 to 15 rows

STEP 4 - Query data using WHERE clause. This is done using the following syntax: SELECT * FROM tableName WHERE condition; "SELECT * FROM tableName" is the standard SELECT statement. "WHERE" is the keyword that restricts our select query result set and "condition" is the filter to be applied on the results. The filter could be a range, single value or sub query.

a. Show the output of selecting all records from table actor where last_name starts with “A”
b. Show the output of selecting all records from table film where release_year is equal to 1990

STEP 5 - WHERE clause combined with AND logical operator. The WHERE clause when used together with the AND logical operator, is only executed if ALL filter criteria specified are met. Example syntax: SELECT * FROM `movies` WHERE `category_id` = 2 AND `year_released` = 2008;

a. Show the output of selecting all records from table category where category_id is equal to 3 and name starts with “B”
b. Show the output of selecting all records from table customer where store_id is equal to 3 and column active is set to active

STEP 6 - WHERE clause combined with OR logical operator. The WHERE clause when used together with the OR operator, is only executed if any or the entire specified filter criteria is met.

a. The following script gets all the movies in either category 1 or category 2. SELECT * FROM `movies` WHERE `category_id` = 1 OR `category_id` = 2;

STEP 7 - WHERE clause combined with *COMPARISON OPERATORS. = EQUAL to, > GREATER THAN, < LESS THAN and <> NOT EQUAL TO.* Example: *SELECT * FROM `payments` WHERE `amount_paid` > 2000;*

a. Select all records from table payments where amount is equal to 500
b. Select all records from table payments where amount is greater than 500
c. Select all records from table payments where amount is not equal to 500

STEP 8 - SQL ORDER BY Syntax -

SELECT column_name, column_name
FROM table_name
ORDER BY column_name ASC|DESC, column_name ASC|DESC;

a. Show the output of selecting last_name and first_name from the table actor and order the last_name in ascending order, while the last_name should be shown in descending order.