MySQL SELECT Statement with Examples

What is a SELECT query in MySQL?

SELECT QUERY is used to fetch the data from the MySQL database. Databases store data for later retrieval. The purpose of MySQL Select is to return from the database tables, one or more rows that match a given criteria. Select query can be used in scripting languages like PHP, Ruby, or you can execute it via the command prompt.

SQL SELECT statement syntax

It is the most frequently used SQL command and has the following general syntax

```
SELECT [DISTINCT|ALL ] { * | [fieldExpression [AS
newName]} FROM tableName [alias] [WHERE condition][GROUP
BY fieldName(s)]  [HAVING condition] ORDER BY
fieldName(s)
```

HERE

- **SELECT** is the SQL keyword that lets the database know that you want to retrieve data.

- **[DISTINCT | ALL]** are optional keywords that can be used to fine tune the results returned from the SQL SELECT statement. If nothing is specified then ALL is assumed as the default.
- **{*| [fieldExpression [AS newName]}** at least one part must be specified, "*" Selecting all the fields from the specified table name, fieldExpression performs some computations on the specified fields such as adding numbers or putting together two string fields into one.
- **FROM** tableName is mandatory and must contain at least one table, multiple tables must be separated using commas or joined using the JOIN keyword.
- **WHERE** condition is optional, it can be used to specify criteria in the result set returned from the query.
- **GROUP BY** is used to put together records that have the same field values.
- **HAVING** condition is used to specify criteria when working using the GROUP BY keyword.
- **ORDER BY** is used to specify the sort order of the result set.

The Star symbol is used to select all the columns in the table. An example of a simple SELECT statement looks like the one shown below.

```
SELECT * FROM `members`;
```

The above statement selects all the fields from the members table. The semicolon is a statement terminated. It's not mandatory but is considered a good practice to end your statements like that.

Practical examples

We use the myflix DB used for practical examples.
You can learn to import the .sql file into MySQL WorkBench

The Examples are performed on the following two tables

Table 1: **members** table

membership_number	full_names	gender	date_of_birth	physical_address	postal_address	contct_number	email
1	Janet Jones	Female	21-07-1980	First Street Plot No 4	Private Bag	0759 253 542	janetjones@yagoo.com
2	Janet Smith Jones	Female	23-06-1980	Melrose 123	NULL	NULL	jj@fstreet.com
3	Robert Phil	Male	12-07-1989	3rd Street 34	NULL	12345	rm@tstreet.com
4	Gloria Williams	Female	14-02-1984	2nd Street 23	NULL	NULL	NULL

Table 2: **movies** table

movie_id	title	director	year_released	category_id

1	Pirates of the Caribbean 4	Rob Marshall	2011	1
2	Forgetting Sarah Marshall	Nicholas Stoller	2008	2
3	X-Men	NULL	2008	NULL
4	Code Name Black	Edgar Jimz	2010	NULL
5	Daddy's Little Girls	NULL	2007	8
6	Angels and Demons	NULL	2007	6
7	Davinci Code	NULL	2007	6
9	Honey mooners	John Schultz	2005	8
16	67% Guilty	NULL	2012	NULL

Getting members listing

Let's suppose that we want to get a list of all the registered library members from our database, we would use the script shown below to do that.

```
SELECT * FROM `members`;
```

Executing the above script in MySQL workbench produces the following results.

membership_number	full_names	gender	date_of_birth	physical_address	postal_address	contct_number	email

1	Janet Jones	Female	21-07-1980	First Street Plot No 4	Private Bag	0759 253 542	janetjones@yagoo.com
2	Janet Smith Jones	Female	23-06-1980	Melrose 123	NULL	NULL	jj@fstreet.com
3	Robert Phil	Male	12-07-1989	3rd Street 34	NULL	12345	rm@tstreet.com
4	Gloria Williams	Female	14-02-1984	2nd Street 23	NULL	NULL	NULL

Our above query has returned all the rows and columns from the members table.

Let's say we are only interested in getting only the full_names, gender, physical_address and email fields only. The following script would help us to achieve this.

```
SELECT `full_names`,`gender`,`physical_address`, `email` FROM `members`;
```

Executing the above script in MySQL workbench produces the following results.

full_names	gender	physical_address	email
Janet Jones	Female	First Street Plot No 4	janetjones@yagoo.cm

Janet Smith Jones	Female	Melrose 123	jj@fstreet.com
Robert Phil	Male	3rd Street 34	rm@tstreet.com
Gloria Williams	Female	2nd Street 23	NULL

Getting movies listing

Remember in our above discussion that we mention expressions being used in SELECT statements. Let's say we want to get a list of movies from our database. We want to have the movie title and the name of the movie director in one field. The name of the movie director should be in brackets. We also want to get the year that the movie was released. The following script helps us do that.

```
SELECT Concat(`title`, ' (', `director`, ')') ,
`year_released` FROM `movies`;
```

HERE

- The Concat () MySQL function is used to join the column values together.
- The line "Concat (`title`, ' (', `director`, ')') gets the title, adds an opening bracket followed by the name of the director then adds the closing bracket.

String portions are separated using commas in the Concat () function.

Executing the above script in MySQL workbench produces the following result set.

Concat(`title`, ' (', `director`, ')')	**year_released**
Pirates of the Caribbean 4 (Rob Marshall)	2011
Forgetting Sarah Marshall (Nicholas Stoller)	2008

NULL	2008
Code Name Black (Edgar Jimz)	2010
NULL	2007
NULL	2007
NULL	2007
Honey mooners (John Schultz)	2005
NULL	2012

Alias field names

The above example returned the Concatenation code as the field name for our results. Suppose we want to use a more descriptive field name in our result set. We would use the column alias name to achieve that. The following is the basic syntax for the column alias name

```
SELECT `column_name|value|expression` [AS] `alias_name`;
```

HERE

- ***"SELECT ` column_name|value|expression `"*** is the regular SELECT statement which can be a column name, value or expression.
- ***"[AS]"*** is the optional keyword before the alias name that denotes the expression, value or field name will be returned as.
- ***"`alias_name`"*** is the alias name that we want to return in our result set as the field name.

The above query with a more meaningful column name

```
SELECT Concat(`title`, ' (', `director`, ')') AS 'Concat', `year_released` FROM `movies`;
```

We get the following result

Concat	year_released
Pirates of the Caribean 4 (Rob Marshall)	2011
Forgetting Sarah Marshall (Nicholas Stoller)	2008
NULL	2008
Code Name Black (Edgar Jimz)	2010
NULL	2007
NULL	2007
NULL	2007
Honey mooners (John Schultz)	2005
NULL	2012

Getting members listing showing the year of birth

Suppose we want to get a list of all the members showing the membership number, full names and year of birth, we can use the LEFT string function to extract the year of birth from the date of birth field. The script shown below helps us to do that.

```
SELECT `membership_number`,`full_names`,LEFT(`date_of_birth`,4) AS `year_of_birth` FROM members;
```

HERE

- ***"LEFT(`date_of_birth`,4)"*** the **LEFT string function** accepts the date of birth as the parameter and only returns 4 characters from the left.
- ***"AS `year_of_birth`"*** is the **column alias name** that will be returned in our results. Note the ***AS keyword is optional***, you can leave it out and the query will still work.

Executing the above query in MySQL workbench against the myflixdb gives us the results shown below.

membership_number	full_names	year_of_birth
1	Janet Jones	1980
2	Janet Smith Jones	1980
3	Robert Phil	1989
4	Gloria Williams	1984

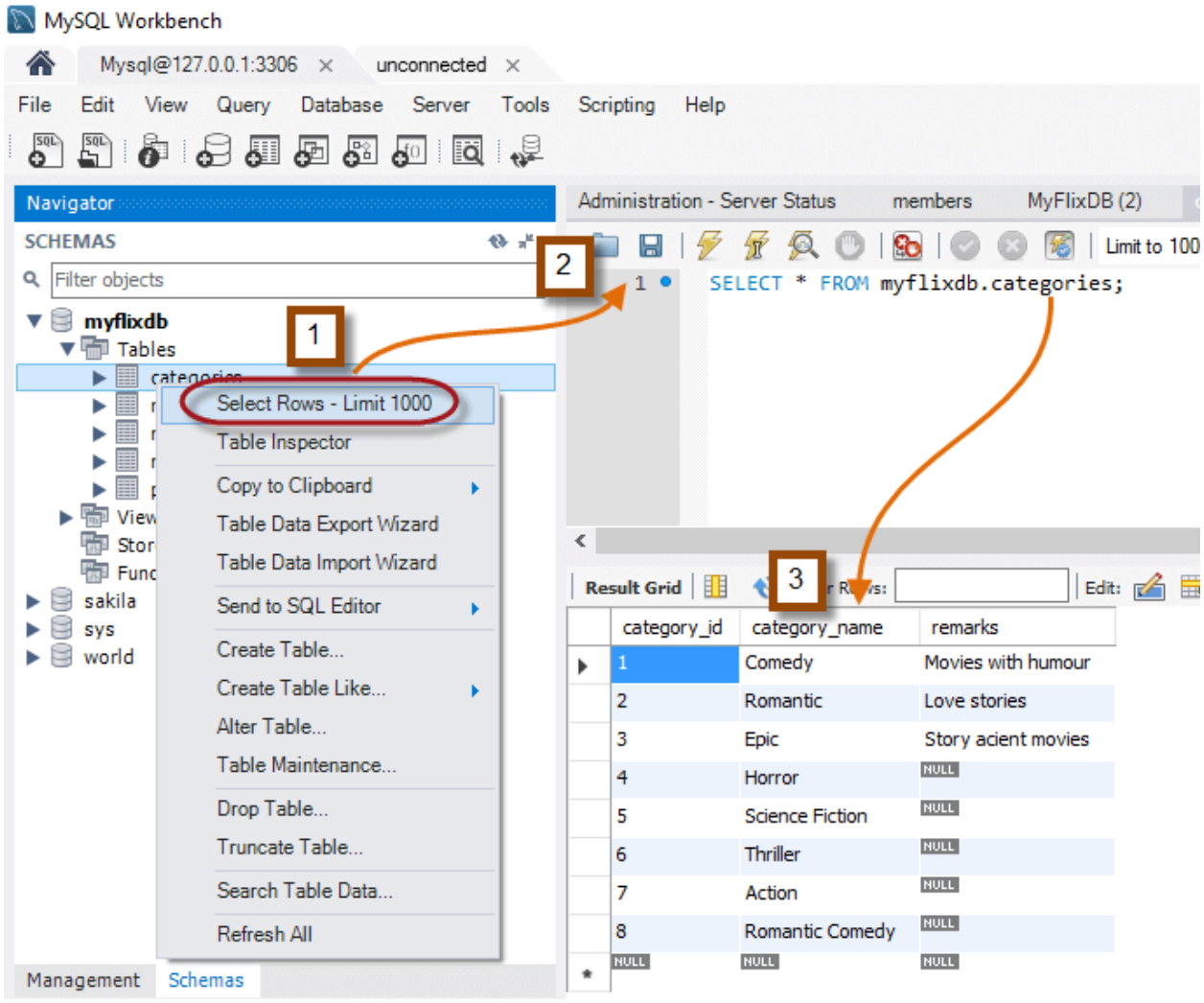
SQL using MySQL Workbench

We are now going to use MySQL workbench to generate the script that will display all the field names from our categories table.

1. Right Click on the Categories Table. Click on "Select Rows - Limit 1000"

2. MySQL workbench will automatically create a SQL query and paste in the editor.

3. Query Results will be show



Notice that we didn't write the SELECT statement ourselves. MySQL workbench generated it for us.

Why use the SELECT SQL command when we have MySQL Workbench?

Now, you might be thinking why learn the SQL SELECT command to query data from the database when you can simply use a tool such as MySQL workbench to get the same results without knowledge of the SQL language. Of course that is possible, but **learning how to use the SELECT command** gives you more **flexibility** and **control** over your **SQL SELECT statements**.

MySQL workbench falls in the category of "**Query by Example**" QBE tools. It's intended to help generate SQL statements faster to increase the user productivity.

Learning the SQL SELECT command can enable you to create **complex queries** that cannot be easily generated using Query by Example utilities such as MySQL workbench.

To improve productivity you can **generate the code using MySQL workbench** then **customize** it to **meet your requirements**. This can only happen if you understand how the SQL statements work!

Summary

- The SQL SELECT keyword is used to query data from the database and it's the most commonly used command.
- The simplest form has the syntax "SELECT * FROM tableName;"
- Expressions can also be used in the select statement . Example "SELECT quantity + price FROM Sales"
- The SQL SELECT command can also have other optional parameters such as WHERE, GROUP BY, HAVING, ORDER BY. They will be discussed later.
- MySQL workbench can help develop SQL statements, execute them and produce the output result in the same window.