**Querying the Database - SELECT**

The most common, important and complex task is to query a database for a subset of data that meets your needs - with the SELECT command. The SELECT command has the following syntax:

-- List all the rows of the specified columns

SELECT *column1Name*, *column2Name*, ... FROM *tableName*

--List all the rows of ALL columns,\* is a wildcard denoting all columns

SELECT *\** FROM *tableName*

-- List rows that meet the specified *criteria* in WHERE clause

SELECT *column1Name*, *column2Name*,... FROM *tableName* WHERE *criteria*

SELECT *\** FROM *tableName* WHERE *criteria*

For examples,

-- List all rows for the specified columns

mysql> **SELECT name, price FROM products;**

+-----------+-------+

| name | price |

+-----------+-------+

| Pen Red | 1.23 |

| Pen Blue | 1.25 |

| Pen Black | 1.25 |

| Pencil 2B | 0.48 |

| Pencil 2H | 0.49 |

+-----------+-------+

5 rows in set (0.00 sec)

-- List all rows of ALL the columns. The wildcard \* denotes ALL columns

mysql> **SELECT \* FROM products;**

+-----------+-------------+-----------+----------+-------+

| productID | productCode | name | quantity | price |

+-----------+-------------+-----------+----------+-------+

| 1001 | PEN | Pen Red | 5000 | 1.23 |

| 1002 | PEN | Pen Blue | 8000 | 1.25 |

| 1003 | PEN | Pen Black | 2000 | 1.25 |

| 1004 | PEC | Pencil 2B | 10000 | 0.48 |

| 1005 | PEC | Pencil 2H | 8000 | 0.49 |

+-----------+-------------+-----------+----------+-------+

5 rows in set (0.00 sec)

**SELECT without Table**

You can also issue SELECT without a table. For example, you can SELECT an expression or evaluate a built-in function.

mysql> **SELECT 1+1;**

+-----+

| 1+1 |

+-----+

| 2 |

+-----+

1 row in set (0.00 sec)

mysql> **SELECT NOW();**

+---------------------+

| NOW() |

+---------------------+

| 2012-10-24 22:13:29 |

+---------------------+

1 row in set (0.00 sec)

// Multiple columns

mysql> **SELECT 1+1, NOW();**

+-----+---------------------+

| 1+1 | NOW() |

+-----+---------------------+

| 2 | 2012-10-24 22:16:34 |

+-----+---------------------+

1 row in set (0.00 sec)

**Comparison Operators**

For numbers (INT, DECIMAL, FLOAT), you could use comparison operators: '=' (equal to), '<>' or '!=' (not equal to), '>' (greater than), '<' (less than), '>=' (greater than or equal to), '<=' (less than or equal to), to compare two numbers. For example, price > 1.0, quantity <= 500.

mysql> SELECT name, price FROM products **WHERE price < 1.0**;

+-----------+-------+

| name | price |

+-----------+-------+

| Pencil 2B | 0.48 |

| Pencil 2H | 0.49 |

+-----------+-------+

2 rows in set (0.00 sec)

mysql> SELECT name, quantity FROM products **WHERE quantity <= 2000**;

+-----------+----------+

| name | quantity |

+-----------+----------+

| Pen Black | 2000 |

+-----------+----------+

1 row in set (0.00 sec)

CAUTION: Do not compare FLOATs (real numbers) for equality ('=' or '<>'), as they are not precise. On the other hand, DECIMAL are precise.

For strings, you could also use '=', '<>', '>', '<', '>=', '<=' to compare two strings (e.g., productCode = 'PEC'). The ordering of string depends on the so-called *collation* chosen. For example,

mysql> SELECT name, price FROM products **WHERE productCode = 'PEN'**;

-- String values are quoted

+-----------+-------+

| name | price |

+-----------+-------+

| Pen Red | 1.23 |

| Pen Blue | 1.25 |

| Pen Black | 1.25 |

+-----------+-------+

3 rows in set (0.00 sec)

**String Pattern Matching - LIKE and NOT LIKE**

For strings, in addition to full matching using operators like '=' and '<>', we can perform *pattern matching* using operator LIKE (or NOT LIKE) with wildcard characters. The wildcard '\_' matches any single character; '%' matches any number of characters (including zero). For example,

* 'abc%' matches strings beginning with 'abc';
* '%xyz' matches strings ending with 'xyz';
* '%aaa%' matches strings containing 'aaa';
* '\_\_\_' matches strings containing exactly three characters; and
* 'a\_b%' matches strings beginning with 'a', followed by any single character, followed by 'b', followed by zero or more characters.

-- "name" begins with 'PENCIL'

mysql> SELECT name, price FROM products **WHERE name LIKE 'PENCIL%'**;

+-----------+-------+

| name | price |

+-----------+-------+

| Pencil 2B | 0.48 |

| Pencil 2H | 0.49 |

+-----------+-------+

-- "name" begins with 'P', followed by any two characters,

-- followed by space, followed by zero or more characters

mysql> SELECT name, price FROM products **WHERE name LIKE 'P\_\_ %'**;

+-----------+-------+

| name | price |

+-----------+-------+

| Pen Red | 1.23 |

| Pen Blue | 1.25 |

| Pen Black | 1.25 |

+-----------+-------+

MySQL also support regular expression matching via the REGEXE operator.