

Introduction to SQL

SQL is a very high level language which avoids data manipulation details needed in procedural languages like C++ or Java.

Here are some useful commands:

Selecting a database:

```
mysql> USE database;
```

Listing databases:

```
mysql> SHOW DATABASES;
```

Listing tables in a db:

```
mysql> SHOW TABLES;
```

Describing the format of a table:

```
mysql> DESCRIBE table;
```

Creating a database:

```
mysql> CREATE DATABASE db_name;
```

Creating a table:

```
mysql> CREATE TABLE table_name (field1_name TYPE(SIZE), field2_name TYPE(SIZE));
```

```
Ex: mysql> CREATE TABLE pet (name VARCHAR(20), sex CHAR(1), birth DATE);
```

Load tab-delimited data into a table:

```
mysql> LOAD DATA LOCAL INFILE "infile.txt" INTO TABLE table_name;
```

(Use \n for NULL)

Inserting one row at a time:

```
mysql> INSERT INTO table_name VALUES ('MyName', 'MyOwner', '2002-08-31');
```

(Use NULL for NULL)

Retrieving information (general):

```
mysql> SELECT from_columns FROM table WHERE conditions;
```

All values: SELECT * FROM table;

Some values: `SELECT * FROM table WHERE rec_name = "value";`

Multiple criteria: `SELECT * FROM TABLE WHERE rec1 = "value1" AND rec2 = "value2";`

Reloading a new data set into existing table:

`mysql> SET AUTOCOMMIT=1; # used for quick recreation of table`

`mysql> DELETE FROM pet;`

`mysql> LOAD DATA LOCAL INFILE "infile.txt" INTO TABLE table;`

Fixing all records with a certain value:

`mysql> UPDATE table SET column_name = "new_value" WHERE record_name = "value";`

Selecting specific columns:

`mysql> SELECT column_name FROM table;`

Retrieving unique output records:

`mysql> SELECT DISTINCT column_name FROM table;`

Sorting:

`mysql> SELECT col1, col2 FROM table ORDER BY col2;`

Backwards: `SELECT col1, col2 FROM table ORDER BY col2 DESC;`

Date calculations:

`mysql> SELECT CURRENT_DATE, (YEAR(CURRENT_DATE)-YEAR(date_col)) AS time_diff [FROM table];`

`MONTH(some_date)` extracts the month value and `DAYOFMONTH()` extracts day.

Pattern Matching:

`mysql> SELECT * FROM table WHERE rec LIKE "blah%";`

(% is wildcard - arbitrary # of chars)

Find 5-char values: `SELECT * FROM table WHERE rec like "_____";`

(_ is any single character)

Extended Regular Expression Matching:

`mysql> SELECT * FROM table WHERE rec RLIKE "^b$";`

(. for char, [...] for char class, * for 0 or more instances

^ for beginning, {n} for repeat n times, and \$ for end)

(RLIKE or REGEXP)

To force case-sensitivity, use "REGEXP BINARY"

Counting Rows:

```
mysql> SELECT COUNT(*) FROM table;
```

Grouping with Counting:

```
mysql> SELECT owner, COUNT(*) FROM table GROUP BY owner;
```

(GROUP BY groups together all records for each 'owner')

Selecting from multiple tables:

(Example)

```
mysql> SELECT pet.name, comment FROM pet, event WHERE pet.name = event.name;
```

(You can join a table to itself to compare by using 'AS')

Currently selected database:

```
mysql> SELECT DATABASE();
```

Maximum value:

```
mysql> SELECT MAX(col_name) AS label FROM table;
```

Adding a column to an already-created table:

```
mysql> ALTER TABLE tbl ADD COLUMN [column_create syntax] AFTER col_name;
```

Removing a column:

```
mysql> ALTER TABLE tbl DROP COLUMN col;
```

(Full ALTER TABLE syntax available at mysql.com.)

Batch mode (feeding in a script):

```
# mysql -u user -p < batch_file
```

(Use -t for nice table layout and -vvv for command echoing.)

Alternatively:

```
mysql> source batch_file;
```

Backing up a database with mysqldump:

```
# mysqldump --opt -u username -p database > database_backup.sql
```

(Use 'mysqldump --opt --all-databases > all_backup.sql' to backup everything.)