Introduction to SQL on GRAHAM

ED ARMSTRONG
SHARCNET
AUGUST 2018
Background Information
Background Information
What is a (Relational) Database

- Dynamic collection of information.
- Organized into **tables**, **rows**, and **columns**.
- Often indexed to improve access time.
- They exist in a variety of flavours.
Background Information

Types of Databases

- Distributed / NoSQL
- Object Oriented / PostgreSQL
- Relational / MySQL, MariaDB

- Dynamic collection of information.
- Organized into tables, rows, and columns.
- Often indexed to improve access time.
- They exist in a variety of flavours.
Background Information
What is SQL

- **Structured Query Language**
- The standard for accessing & manipulating relational databases.
- There is a standard for how SQL works.
Requesting a Database

Send a request to support@computecanada.ca with the following information:

- Your Compute Canada username.
- Amount of database space needed for your project.
- The system you would like an account on (Graham / Cedar).

We will create an account with a randomly generated password. The necessary information will be stored in a '.my.cnf' file in your home directory.
MySQL Configuration

[client]
ssl
ssl-cipher=DHE-RSA-AES256-SHA:AES128-SHA
user=your_username
password=YyG1ZJYRxmldfV0U
database=your_username
host=199.241.163.99
Create, Use, Delete Databases
Create, Use, Delete Databases

$ ssh graham@computecanada.ca
$ mysql (with .my.cnf)
$ mysql -h hostname -u username (w/o .my.cnf)
$ mysql --local-infile=1
CREATE DATABASE my_database;
SHOW DATABASES;
USE my_database;
DROP DATABASE my_database; * You won't receive a warning
Tables
A database is a collection of tables.
A database is a collection of tables.
A table is a collection of data entries (tuples).
Tables

A database is a collection of tables. A table is a collection of data entries (tuples). An entry is a row.
Tables

A database is a collection of tables.
A table is a collection of data entries (tuples).
An entry is a row.
A data point (type) is a column.
The table schema describes the contents of a table.

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
<th>BREED</th>
<th>COLOR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The table schema describes the contents of a table.

<table>
<thead>
<tr>
<th>NAME: STRING</th>
<th>AGE: NUMBER</th>
<th>BREED: STRING</th>
<th>COLOR: STRING</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SQL Data Types
## SQL Data Types

<table>
<thead>
<tr>
<th>Text Data Types</th>
<th>Number Data Types</th>
<th>Date Data Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>TINYINT</td>
<td>DATE</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>SMALLINT</td>
<td>DATETIME</td>
</tr>
<tr>
<td>TINYTEXT</td>
<td>MEDIUMINT</td>
<td>TIMESTAMP</td>
</tr>
<tr>
<td>BLOB</td>
<td>INT</td>
<td>TIME</td>
</tr>
<tr>
<td>MEDIUMTEXT</td>
<td>BIGINT</td>
<td>YEAR</td>
</tr>
<tr>
<td>LONGTEXT</td>
<td>FLOT</td>
<td></td>
</tr>
<tr>
<td>LONGBLOB</td>
<td>DOUBLE</td>
<td></td>
</tr>
<tr>
<td>ENUM</td>
<td>DECIMAL</td>
<td></td>
</tr>
<tr>
<td>SET</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

```sql
CREATE TABLE Employees (name CHAR(50), age TINYINT);
```
## SQL Data Types

<table>
<thead>
<tr>
<th>Text Data Types</th>
<th>Number Data Types</th>
<th>Date Data Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>TINYINT</td>
<td>DATE</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>SMALLINT</td>
<td>DATETIME</td>
</tr>
<tr>
<td>TINYTEXT</td>
<td>MEDIUMINT</td>
<td>TIMESTAMP</td>
</tr>
<tr>
<td>BLOB</td>
<td>INT</td>
<td>TIME</td>
</tr>
<tr>
<td>MEDIUMTEXT</td>
<td>BIGINT</td>
<td>TIMESTAMP</td>
</tr>
<tr>
<td>LongTEXT</td>
<td>FLOAT</td>
<td>YEAR</td>
</tr>
<tr>
<td>LONGTEXT</td>
<td>DOUBLE</td>
<td></td>
</tr>
<tr>
<td>LONGBLOB</td>
<td>DECIMAL</td>
<td></td>
</tr>
<tr>
<td>ENUM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SET</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SQL Data Types

- CHAR
- VARCHAR
- INT
- DOUBLE
- DATE
SQL Data Types

- **CHAR / VARCHAR**
- INT
- DOUBLE
- DATE

Variables in CHAR are fixed length string up to 255 characters in length. Variable in VARCHAR are variable length strings up to 65,535* characters in length.

*Shared across all columns.*
# SQL Data Types

- **CHAR/VARCHAR**
- **INT**
- **DOUBLE**
- **DATE**

<table>
<thead>
<tr>
<th>Type</th>
<th>Storage (Bytes)</th>
<th>Minimum Value Signed</th>
<th>Minimum Value Unsigned</th>
<th>Maximum Value Signed</th>
<th>Maximum Value Unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>TINYINT</td>
<td>1</td>
<td>-128</td>
<td>0</td>
<td>127</td>
<td>255</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>2</td>
<td>-32768</td>
<td>0</td>
<td>32767</td>
<td>65535</td>
</tr>
<tr>
<td>MEDIUMINT</td>
<td>3</td>
<td>-8388608</td>
<td>0</td>
<td>8388607</td>
<td>16777215</td>
</tr>
<tr>
<td>INT</td>
<td>4</td>
<td>-2147483648</td>
<td>0</td>
<td>2147483647</td>
<td>4294967295</td>
</tr>
<tr>
<td>BIGINT</td>
<td>8</td>
<td>-2^{63}</td>
<td>0</td>
<td>2^{63}-1</td>
<td>2^{64}-1</td>
</tr>
</tbody>
</table>

*Required Storage and Range for Integer Types Supported by MySQL*

SQL Data Types

- CHAR/VARCHAR
- INT
- DOUBLE
- DATE

The FLOAT and DOUBLE data types are APPROXIMATE. If you require an exact decimal value, such as for currency, use DECIMAL.
SQL Data Types

- VARCHAR
- INT
- DOUBLE
- DATE

The DATE data type represents a calendar value. There are a number of interpretation rules that MySQL uses, as such you should stick to the SQL standard format (YYYY-MM-DD).
Creating a Table
Creating a Table

CREATE TABLE employees (  
  name varchar(64),
  id int,
  start date
);
Setting and Getting Values

- A PREVIEW

```sql
INSERT INTO employees VALUES ('Adam',
   1,
   '2018-07-07'
);

SELECT * FROM employees;

INSERT INTO employees (name) VALUES ('Adam');
```
Creating a Table: Unsigned Integer

- Table name
- Table column names
- Table column data types

```sql
CREATE TABLE employees (
    name varchar(64),
    id int UNSIGNED,
    start date
);
```
Creating a Table:
- Reserved Keywords & Spaces

CREATE TABLE employees ( `first name` varchar(64), `index` int UNSIGNED, start date );
Creating a Table:
- Reserved Keywords

```
CREATE TABLE employees (  
`first name` varchar(64),  
`index` int UNSIGNED,  
start date  
);
```
Inserting Data Into a Table
INSERT INTO employees (name, id, start) values ('Adam', 1, '2018-07-07');
INSERT INTO employees(
    name,
    id,
    start
) VALUES(
    'Adam',
    1,
    '2018-07-07'
);
Inserting Multiple Data

```sql
INSERT INTO employees(name, id, start)
VALUES(‘Adam’, 1, ‘2018-07-07’)
 , (‘Steve’, 2, ‘2016-06-04’)
 , (‘Craig’, 3, ‘2016-06-04’);
```
Retrieving Your Data

**SELECT**
Retrieving Your Data

SELECT * FROM employees;
Retrieving Your Data

\textbf{SELECT} * \textbf{FROM} employees;

SQL Keywords
Retrieving Your Data

```
SELECT * FROM employees;
```

Column Selector

* means all

Table Selector

* means all
Retrieving Your Data

SELECT * FROM employees;
SELECT name FROM employees;
SELECT name, id FROM employees;
SELECT name, id, start FROM employees;
SELECT id, start, name FROM employees;
Terminating a Command
\c
Selecting Rows by Content

WHERE
Selecting Rows by Content

SELECT * FROM employees WHERE id = 3;
SELECT * FROM employees WHERE name = 'Adam';
SELECT * FROM employees WHERE name = 'ADAM';
SELECT * FROM employees WHERE name = 'A%';
SELECT * FROM employees WHERE binary name='Adam';
SELECT * FROM employees WHERE name='Adam'
    AND id = 7;
Change Existing Data

**UPDATE**
UPDATE employees SET name='Chris'
WHERE name='Adam';

UPDATE employees SET start='2018-05-09'
WHERE id='1';

UPDATE employees SET start='2000-01-01'
WHERE start IS null;
Removing Data

DELETE
Removing Data

DELETE FROM employees WHERE name='Adam';
DELETE FROM employees WHERE start=end;
DELETE FROM employees;
Executing SQL Files

SOURCE
$ mysql < instructions.sql
$ mysql --verbose < instructions.sql
$ mysql -e "select * from table"
mysql> source instructions.sql
Importing Data

LOAD DATA
LOAD DATA LOCAL INFILE 'file' INTO TABLE table
LOAD DATA LOCAL INFILE 'file' INTO TABLE table

fields
    terminated by '\t'
    enclosed by ""
    escaped by '\\'
lines
    terminated by '\n'
    starting by "";
LOAD DATA LOCAL INFILE 'file' INTO TABLE table
fields
  terminated by '\t'
  enclosed by ''
  escaped by '\\'
lines
  terminated by '\n'
  starting by '';
CREATE TABLE employees(
    first_name VARCHAR(64),
    last_name VARCHAR(64),
    id int AUTO_INCREMENT,
    start DATE,
    finish DATE,
    PRIMARY KEY(id)
)

LOAD DATA LOCAL INFILE 'employeeData.tab'
    INTO TABLE employees
    (first_name, last_name, @ignore, start, finish);
Save Data

```
mysql -ss -e "select * from employees" > data.tab
```