<https://dev.mysql.com/doc/>

How is data accessed from a database?

Using SQL statements

• SQL stands for Structured Query Language

• SQL is pronounced “sequel”

SQL statements fall into four categories

Data Manipulation Language (DML)

Query and Alter the data in a table

**SELECT, INSERT, UPDATE, DELETE**

Data Definition Language (DDL)

Create or Alter tables, views…

**CREATE, ALTER, DROP**

Data Control Language (DCL)

Add/Remove privileges to the database

**GRANT, REVOKE**

Database Definitions

* **RDBMS** (Relational Database Management System) – Software that stores and manipulates data arranged in relational database tables.
* **Database** – Collection of tables that serve a common purpose (Student records, Accounts Payable, General Ledger, Order Entry, Inventory)
* **Table** – The set of data that defines an entity – Orders, OrderItems, Customers, Students
* **Column** – The attributes that define an entity –

Table Possible columns you would find in that table

* + Students: StudentId, Major, BirthDate, HomeState
  + Classes: ClassId, Title, Description, CreditHrs
  + Schedules: StudentId, ClassId, ClassRoom, MeetingTime
  + A column has a name and type. The name should be descriptive. Type is the type of data it will hold. numbers, characters, dates, int,
* **Row** – Represents a single entity in a table. All attributes that define that entry
  + OrderId Integer, Unique for each Order
  + BillingAddress Character, 60 characters wide
  + BirthDate Date – MM/DD/YYYY

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Basic MySQL Commands | | | | What | **How** | **Example(s)** | | List all databases | SHOW DATABASES; | SHOW DATABASES; | | Create database | CREATE DATABASE *database*; | CREATE DATABASE PhoneDB; | | Use a database | USE *database*; | USE PhonDB; | | List tables in the database | SHOW TABLES; | SHOW TABLES; | | Show the structure of a table | DESCRIBE *table*; SHOW COLUMNS FROM *table*; | DESCRIBE Animals; SHOW COLUMNS FROM Animals; | | Delete a database (*Careful!*) | DROP DATABASE *database*; | DROP DATABASE PhoneDB; | |

|  |  |
| --- | --- |
| **Data Type** | **Description** |
| int(size)  integer()  bigint(), smallint(), tinyint() | Integers only. The maximum number of digits are specified in parenthesis. |
| decimal(**size**, dec)  numeric(**size**, dec) | Decimal numbesr. (**size**: max # of digits,  dec: max # of digits to the right of the decimal). |
| Float  double |  |
| char(size) | fixed length string |
| varchar(size) | variable length string |
| date(yyyymmdd) datetime(yyyymmdd hhmmss) | Date  Date and Time |

Getting data from a table in the DB **SELECT**

(All queries will be from the World database)

**SELECT** Almost always!!! The columns in the table from the **FROM**

Can be as simple as

**SELECT 1;**

**SELECT "Hello", 3.14159, NOW();**

Most interesting – get all data contained in a table or get specific columns

**SELECT \* SELECT Name, CountryCode**

**FROM City; FROM City;**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Name | Country Code | District | Population | Name | CountryCode |
| 1 | Kabul | AFG | Kabol | 1780000 | Kabul | AFG |
| 2 | Qandahar | AFG | Qandahar | 237500 | Qandahar | AFG |
| 3 | Herat | AFG | Herat | 186800 | Herat | AFG |
| 4 | Mazar-e-Sharif | AFG | Balkh | 127800 | Mazar-e-Sharif | AFG |
| 5 | Amsterdam | NLD | Noord-Holland | 731200 | Amsterdam | NLD |
| 6 | Rotterdam | NLD | Zuid-Holland | 593321 | Rotterdam | NLD |
| 7 | Haag | NLD | Zuid-Holland | 440900 | Haag | NLD |

Filtering the amount of data returned **WHERE**

**SELECT Name, Population SELECT Title, Description, Rating**

**FROM City FROM Film**

**WHERE CountryCode = 'USA' Where Rating ! = ‘NC-17’**

|  |  |  |  |
| --- | --- | --- | --- |
| Operators | |  |  |
| **Operator** | **Description** |  |  |
| **=** | Equal | **<>** | Not equal |
| **>** | Greater than | **>** | Greater than or equal |
| **<** | Less than | **< =** | Less than or equal |
| **BETWEEN** | Between an **inclusive** range | **IN** | if exact values are known |
| **LIKE** | to match a set [0-9], [a-z], [aeiou] | **"%"**  **"\_"** | matches Zero or **more**  matches **ONE** character |

|  |  |
| --- | --- |
| Some aggregate functions | |
| **Function** | **Returns** |
| AVG(column) | average value of a column |
| COUNT(column) | number of rows (without a NULL value) of a column |
| MAX(column) | highest value of a column |
| MIN(column) | lowest value of a column |
| SUM(column) | total sum of a column |

<https://dev.mysql.com/doc/refman/5.7/en/func-op-summary-ref.html>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | SQL Commands: Querying | | | | What | **How** | **Example(s)** | | All cols | SELECT \* FROM *table*; | SELECT \* FROM Film; | | Some cols | SELECT *column1*, *column2*, ... FROM *table*; | SELECT Title, Rating  FROM Film; | | Some rows/cols | SELECT *column1*, *column2*, ... FROM *table* [WHERE *condition(s)*]; | SELECT Title, Rental\_Duration FROM Film WHERE Description LIKE '%Epic%'; | | No Dups | SELECT [DISTINCT] *column(s)* FROM *table*; | SELECT DISTINCT Rating FROM Film; | | Ordering | SELECT *column1*, *column2*, ... FROM *table* [ORDER BY *column(s)* [DESC]]; | SELECT Name, Continent, Region, HeadOfState  FROM Country ORDER BY Continent, Region; | | Column Aliases | SELECT *col1* [AS *alias1*],  *col2* [AS *alias2*], ... FROM *table1*; | SELECT Name, HeadOfState AS Leader FROM Country; | | Grouping | SELECT *column1*, *column2*, ... FROM *table* [GROUP BY *column(s)*]; | SELECT HeadOfState, COUNT(\*) FROM Country GROUP BY HeadOfState; | | Group Filtering | SELECT *column1*, *column2*, ... FROM *table* [GROUP BY *column(s)*] [HAVING *condition(s)*]; | SELECT HeadOfState, COUNT(\*) FROM Country GROUP BY HeadOfState HAVING HeadOfState like '%II'; | | Joins | SELECT *column1*, *column2*, ... FROM *table1*, *table2*, ... [WHERE *condition(s)*]; | SELECT C.Name, K.Name, K.HeadOfState  FROM Country AS K, City AS C  WHERE C.ID = K.Capital; | | Joins | SELECT t1.*col1*, t2.*col2*, ... FROM *table1* AS t1 [JOIN *table2* AS t2 ON t1.ed = t2.id]; | SELECT C.Name, K.Name, K.HeadOfState  FROM Country AS K  JOIN City AS C ON C.ID = K.Capital; | | Table  Aliases | SELECT t1.*col1*, t2.*col2*, ... FROM *table1* AS t1 [JOIN *table2* AS t2 ON t1.ed = t2.id]; | SELECT C.Name, K.Name, K.HeadOfState  FROM Country AS K  JOIN City AS C ON C.ID = K.Capital; | | Everything | SELECT [DISTINCT]  *column1* [AS *alias1*],  *column2* [AS *alias2*], ... FROM *table1* [*alias1*],  *table2* [*alias2*], ... [WHERE *condition(s)*] [GROUP BY *column(s)*] [HAVING *condition(s)*] [ORDER BY *column(s)* [DESC]]; | SELECT HeadOfState, COUNT(\*) AS Realms FROM Country AS K  GROUP BY HeadOfState HAVING HeadOfState like '%II'  ORDER BY Realms Desc; | | |
| |  |  | | --- | --- | | What | How | | Count rows per group | COUNT(*column* | \*) | | Average value of group | AVG(*column*) | | Min value of group | MIN(*column*) | | Max value of group | MAX(*column*) | | Sum values in a group | SUM(*column*) | | Absolute value | abs(*number*) | | Rounding numbers | round(*number*) | | Integer part | floor(*number*) | | Next largest integer | ceiling(*number*) | | Square root | sqrt(*number*) | | *n*th power | pow(*base*, *exponent*) | | random # *n*, 0< = *n* < 1 | rand() | | Sin, cos, etc. | sin(*number*) | | |  |  | | --- | --- | | What | How | | Convert to lower case | lower(*string*) | | Convert to upper case | upper(*string*) | | Left-trim whitespace  (also rtrim, trim) | ltrim(*string*) | | Substring of string | substring(*string*, *start*, *length*) | | Encrypt password | password(*string*) | | Encode string | encode(*string*, *key*) | | Decode string | decode(*string*, *key*) | | Get date | GetDate() | | Day name from date | **DATENAME**  (WEEKDAY, GETDATE()) | | Day # from date | DAYOFYEAR, GETDATE()) | | Month from date | MONTH, GETDATE()) | |

|  |  |  |
| --- | --- | --- |
| SQL Commands: Modifying | | |
| What |  | **Example(s)** |
| Insert data |  | INSERT INTO Students VALUES         ('Smith', 'John', 123456789, 'Math', 'Selleck'); INSERT INTO Students SET         FirstName = 'John',       LastName = 'Smith',          StudentID = 123456789,       Major = 'Math'; INSERT INTO Students         (StudentID, FirstName, LastName)         VALUES (123456789, 'John', 'Smith'); |
| Insert/Select |  | INSERT INTO Students         (StudentID, FirstName, LastName)         SELECT StudentID, FirstName, LastName          FROM OtherStudentTable;         WHERE LastName like '%son'; |
| Delete data |  | DELETE FROM Students      WHERE LastName = 'Smith'; DELETE FROM Students          WHERE LastName like '%Smith%';         AND FirstName = 'John'; DELETE FROM Students; # Danger Danger! |
| Updating Data |  | UPDATE Students SET        LastName = 'Jones'  WHERE         StudentID = 987654321; UPDATE Students SET  LastName = 'Jones', Major = 'Theatre'          WHERE StudentID = 987654321         OR (MAJOR = 'Art' AND FirstName = 'Pete'); |
| Insert column |  | ALTER TABLE Students  ADD COLUMN      Hometown varchar(20); |
| Delete column |  | ALTER TABLE Students          DROP COLUMN Dorm; |
| Delete table (*Careful!*) |  | DROP TABLE Animals; |