SQL In Action, SQL Getting Started Guide

Use SQL To

- Store
- Organize
- Analyze

> SQL can do many of the same things a spreadsheet can but on a larger scale.

- Consider using only using a spreadsheet if your data isn't much bigger then 100 or so rows/columns
- If On the other hand your data is so massive it seems to go on forever, then you are going to want to use a **SQL Database.**

Query

- A request for data or information from a database

Basic structure of a SQL Query

1. **FROM**
    - [From the appropriate **TABLE**]
2. **SELECT**
    - [choose the **COLUMN(S)** you want]
3. **WHERE**
    - [a certain condition is met]

- The suggested order in which you write your SQL queries; Start big (data table) and go smaller (specific conditions)

> ## Sample SQL syntax
>
> `SELECT *`
>
> - The asterisk means all data in the the table
>
> `FROM data_set.table-name`
>
> - Calls up the data table

```
WHERE Column__#_ = 'Name'
```

- Filters out everything but what is being called up.

Reinforcement Notes on SQL

What is a query

- A query is a request for data or information from a database. When you query database, you use SQL to communicate your quest or request. You and the database can always exchange information as long as you speak the same language.
- Every programming language, including SQ:, follows a unique set of guidelines know as *syntax*. ***Syntax is the predetermined structure of the language that includes all required words, symbols, and punctuation, as well as their proper placement.*** As soon as you enter your search criteria using the correct syntax, the query starts working to pull the data you've requested from the target database.

> The syntax of every SQL query is the same
>
> - Use **SELECT** to choose the columns you want to return
> - Use **FROM** to choose the tables where the columns you want are located
> - Use **WHERE** to filter for certain information

A SQL query is like filling in a template. You will find that if you are writing a SQ: query from scratch, it's helpful to start a query by writing the keywords in the following format

1. SELECT
2. FROM
3. WHERE

Next, enter the table name after the **FROM**, the table columns you want after the **SELECT**, and finally the conditions you want to place on your query after the **WHERE**. Make sure to add a new line and indent when adding these.

SELECT

- Columns you want to look at

FROM

- Table the data is stored in

WHERE

- Certain condition is met

Following this method each time makes it easier to write SQL queries. It can also help you make fewer syntax errors.

Example of a query using BigQuery (a data warehouse on the Google Cloud Platform)

```
SELECT
   first_name

FROM
   customer_data.customer_name

WHERE
   first_name = 'Tony'
```

The above query uses three commands to locate customers with the first name "Tony":

1. **SELECT** the column named **first_name**
2. **FROM** a table named **customer_name** (in a dataset named **customer_data**)
    - (The dataset name is always followed by a dot, and then the table name)
3. But only return the data **WHERE** the *first_name* is **Tony**

Multiple columns in a query

In real life, you will need to work with more data beyond customers named Tony. Multiple columns that are chosen by the same **SELECT** command can be indented and grouped together.
If you are requesting multiple data fields from a table, you need to include these columns in your **SELECT** command. Each column is separated by a comma as shown

```
SELECT
   ColumnA,
   ColumnB,
   ColumnC
FROM
   Table Where the data lives
WHERE
   Certain condition is met
```

Here is an example of how it would appear in BigQuery

```
SELECT
   customer_id,
   first_name,
   last_name
FROM
   customer_data.customer_name
WHERE
   first_name = 'Tony'
```

The above uses three commands to locate customers with the first name Tony

1. **SELECT** the columns named **customer_id, first_name, last_name**
2. **FROM** a table named **customer_name** (in a dataset named **customer_data**)
    - (The dataset name is always followed by a dot, and then the table name)
3. But only return the data **WHERE** the *first_name* is **Tony**

The only difference between this query and the previous one is that more data columns are selected. The previous query selected the first*name only while this query selects _customer_id and last_name* in addition to *first_name*. In general, it is a more efficient use of resources to select only the columns that you need. For example, it makes sense to select more columns if you will actually use the additional fields in your **WHERE** clause. If you have multiple conditions in your **WHERE** clause, they may be written like this:

```
SELECT
   ColumnA,
   ColumnB,
   ColumnC
FROM
   Table where the data lives
WHERE
   Condition 1
   AND Condition 2
   AND Condition 3
```

Notice that unlike the **SELECT** command that uses a **comma** to separate *Fields, Variables, Parameters*, the **WHERE** command uses the **AND statement** to connect conditions. As you become a more advanced write of queries, you will make use of other connectors/operators such as OR and NOT.

Here is a BigQuery example with multiple fields used in a **WHERE** clause

```
SELECT
   customer_id,
   first_name,
   last_name
FROM
```

```
    customer_data.customer_name
WHERE
    customer_id > 0
    AND first_name = 'Tony'
    AND last_name = 'Magnolia'
```

The above query uses three commands to locate customers with a valid (greater then 0) customer ID whose first name is Tony and last name is Magnolia.

1. SELECT the columns named customer_id, first_name, and last_name
2. FROM a table named customer_name (in a dataset named customer_data) (The dataset name is always followed by a dot, and then the table name.)
3. But only return the data **WHERE customer_is is greater then 0, first_name is Tony and last_name is Magnolia**

Note that one of the conditions is a logical condition that checks to see if the customer_id is greater then zero.

If only one customer is named Toy Magnolia, the results from the query could be

customer_id	first_name	last_name
1967	Tony	Magnolia

If more then one customer has the same name, the results from the query could be

customer_id	first_name	last_name
1967	Tony	Magnolia
7689	Tony	Magnolia

Key Takeaway

- The most important thing to remember is how to use **SELECT, FROM, WHERE** in a query. Queries with multiple fields will become simpler after your practice writing your own SQL query later in the program.