Intro to SQL

What is SQL

SQL stands **S**tructured **Q**uery **L**anguage. It is the language you use to interact with a database. It allows you to write out what you want to search for, goes to a database that you specify, then returns those results to you. As a Data Analyst you will be able to look at, and perform calculations on the data, but not make any permanent changes.

Keywords

The **Structured** part of SQL refers to the format and keywords that make up the **Query.**

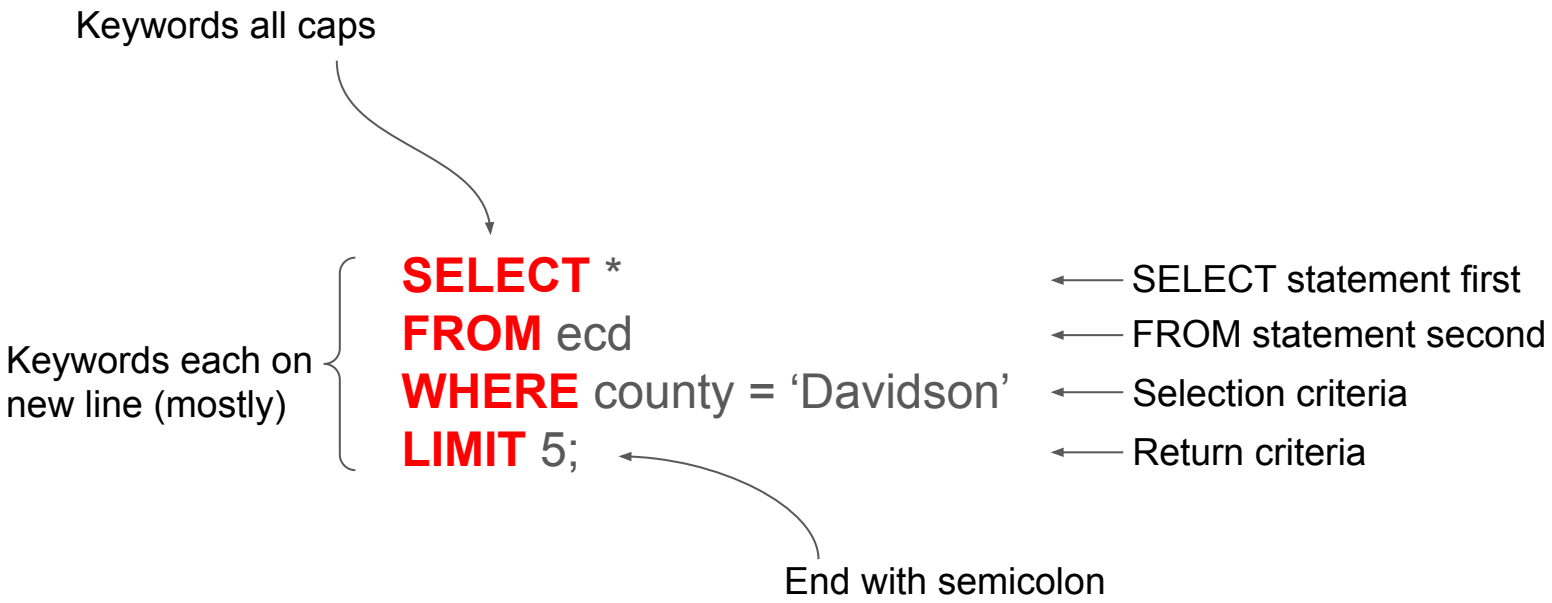
- SELECT
- FROM
- AS
- LIMIT
- DISTINCT
- COUNT
- WHERE
- AND
- OR
- BETWEEN
- IN
- (NOT) NULL
- LIKE
- AVG(), SUM(), MAX(), MIN(), etc.
- ORDER BY
- GROUP BY
- HAVING

Concepts

The **Keywords** allow you to perform operations such as:

- Selecting all columns from a table
- Selecting single columns from a table
- Aggregating data
- Finding unique values
- Slicing data (with multiple criteria)
- Selecting/avoiding null values
- Math
- Aliasing
- Organizing output results

Format of a Query



Let’s walk through a few examples using the **ecd** table

company	landed	capital_investment	new_jobs	project_type	county	county_tier	fjtap	fidp	ed	grants_total
ALSAC St Jude Children's	2016-11-30	$1,000,000,000.00	1800	Expansion	Shelby	2	NULL	NULL	$36,000,000.00	$36,000,000.00
Hankook Tire Co., Ltd	2013-10-14	$800,000,000.00	1800	Recruitment	Montgomery	1	$16,000,000.00	$19,600,000.00	NULL	$35,600,000.00
Tyson Foods, Inc.	2017-11-20	$320,000,000.00	1600	Expansion New Location	Gibson	3	NULL	$14,000,000.00	$6,000,000.00	$20,000,000.00
Denso Manufacturing Tennessee, Inc.	2017-10-06	$1,000,000,000.00	1000	Expansion	Blount	1	NULL	NULL	$20,000,000.00	$20,000,000.00

Selecting all columns from a table

When writing a query you indicate what columns you want back. These directions go in the **SELECT** statement. A shorthand to **SELECT ALL** is to use a *:

```
SELECT *
FROM ecd;
```

company	landed	capital_investment	new_jobs	project_type	county	county_tier	fjtap	fidp	ed	grants_total
ALSAC St Jude Children's	2016-11-30	$1,000,000,000.00	1800	Expansion	Shelby	2	NULL	NULL	$36,000,000.00	$36,000,000.00
Hankook Tire Co., Ltd	2013-10-14	$800,000,000.00	1800	Recruitment	Montgomery	1	$16,000,000.00	$19,600,000.00	NULL	$35,600,000.00
Tyson Foods, Inc.	2017-11-20	$320,000,000.00	1600	Expansion New Location	Gibson	3	NULL	$14,000,000.00	$6,000,000.00	$20,000,000.00
Denso Manufacturing Tennessee, Inc.	2017-10-06	$1,000,000,000.00	1000	Expansion	Blount	1	NULL	NULL	$20,000,000.00	$20,000,000.00

Selecting single columns from a table

You can also specify individual columns to return, each separated by a ‘,’:

```
SELECT company, new_jobs
FROM ecd;
```

company	new_jobs
ALSAC St Jude Children's	1800
Hankook Tire Co., Ltd	1800
Tyson Foods, Inc.	1600
Denso Manufacturing Tennessee, Inc.	1000

Aggregating Data

In certain instances you will want to summarize your data in different ways. For example you could **COUNT**, **SUM**, **AVERAGE**, or find the **MAX** or **MIN**:

```
SELECT COUNT(company)
FROM ecd
```

count
902

```
SELECT AVG(new_jobs)
FROM ecd
```

avg
152.3558758

Aggregating Data

The **GROUP BY** keyword will subdivide the table based on the specified columns. You can then perform aggregations on the subgroups:

```
SELECT SUM(capital_investment)
FROM ecd
GROUP BY county;
```

sum
$3,459,500.00
$1,465,460,355.00
$661,250.00
$391,441,723.00

Finding unique values

Sometimes a particular column or a calculation will result in duplicate values. To get just unique values:

```
SELECT DISTINCT(county_tier)
FROM ecd
```

county_tier
3
1
2
4

Slicing data (with multiple criteria)

Many times you will want to slice your data to perform a calculation or to return only a subset of your data. There are many keywords you can use to slice your data:

```
SELECT *
FROM ecd
WHERE county = 'Davidson'
AND (capital_investment > '$10,000,000' OR capital_investment < '$100,000')
AND county_tier IN (1, 2, 3)
AND new_jobs BETWEEN 1000 AND 2000
AND project_type LIKE 'Expansion%';
```

company	landed	capital_investment	new_jobs	project_type	county	county_tier	fjtap	fidp	ed	grants_total
Community Health Systems Inc.	2015-05-14	$66,150,000.00	1500	Expansion New Location	Davidson	1	NULL	NULL	$6,750,000.00	$6,750,000.00
UBS	2013-08-28	$36,500,000.00	1000	Expansion New Location	Davidson	1	NULL	NULL	$5,000,000.00	$5,000,000.00

Selecting/avoiding null values

Null values will likely exist in any data set you work with. It will be useful to identify or exclude records with null values:

```
SELECT fjtap, fidp, ed, grants_total
FROM ecd
WHERE fjtap IS NOT NULL;
```

fjtap	fidp	ed	grants_total
$16,000,000.00	$19,600,000.00	NULL	$35,600,000.00
$10,899,831.00	NULL	$4,000,000.00	$14,899,831.00
$13,000,000.00	NULL	NULL	$13,000,000.00
$12,000,000.00	NULL	NULL	$12,000,000.00

Math

The ability to perform mathematical functions can allow you to adjust values to a more understandable or relevant range and/or combine columns on the fly:

```
SELECT capital_investment/1000000
FROM ecd;
```

?column?
$1,000.00
$800.00
$320.00
$1,000.00

Aliasing

As queries and calculations become more complex, it may be useful to use aliasing to give a short hand to a subset or calculation so that you can reference it later:

```
SELECT capital_investment/'$1,000,000' AS cap_invest_millions
FROM ecd;
```

cap_invest_millions
$1,000.00
$800.00
$320.00
$1,000.00

Organizing output results

It may be easier to interpret the data if they are organized or limited in a particular way:

```
SELECT DISTINCT(county)
FROM ecd
ORDER BY county
LIMIT 2;
```

county
Anderson
Bedford

Exercises

1. What counties are represented in the **ecd table**?

2. How many companies did not have Economic Development grants (**ed**)? Alias as **ed_companies**.

3. What is the total **capital_investment**, in millions, when there is an **fjtap**? Call the column **fjtap_cap_invest_mil**.

4. What is the average number of new jobs for each **county_tier**?

5. How many companies are **LLC**s (combine **COUNT()** and **DISTINCT()**)? Call this value **llc_companies**.