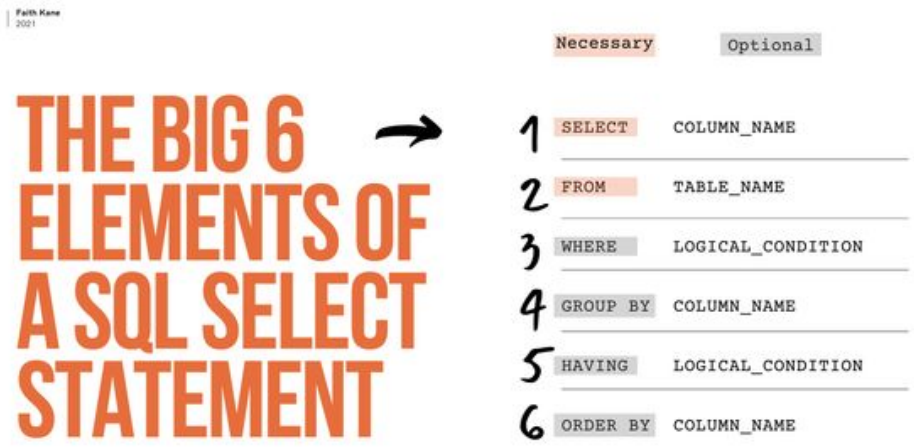
SQL `GROUP BY` Clause

Lesson Goals

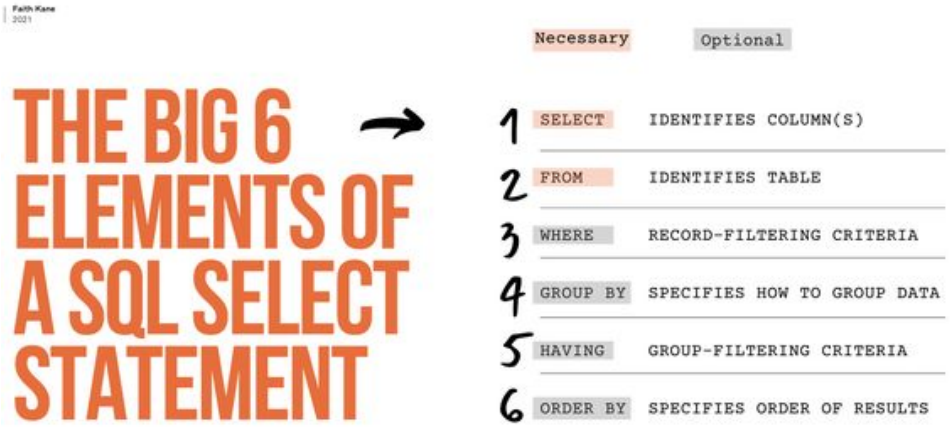
- Understand how to use the `GROUP BY` clause
- Understand how to use aggregate functions with the `GROUP BY` clause to return summary statistics about a column
  - `COUNT()`, `MIN()`, `MAX()`, `AVG()`, `SUM()`, `STDDEV()`, `VARIANCE()`

What Are the Elements of the SQL Select Statement?



So What Does Each Element Do?

- Using a `GROUP BY` clause removes the duplicates from a column just like when we use the `DISTINCT SELECT` statement.
- Think of using `GROUP BY` as creating segments of your data and performing analysis on those segments instead of just extracting records.
- `GROUP BY` allows us to change the level of granularity of our data.
  - For example, changing each row in your return set from representing a single passenger on the Titanic to representing all females or males or all third-class, second-class, and first-class passengers.
- Think of the column(s) that I group by as the dimension(s) of my data and the aggregate function(s) as the measures(s) for my dimension(s).
  - Dimension: A description of what is being measured - discrete value. (can be sorted)
  - Measure: A calculation - continuous value. (can't be sorted)



Now What?

- Let's code!

Using the `GROUP BY` Clause

- I can return the unique values from a single column or the unique combinations of values from multiple columns.
- Non-aggregated columns in your `SELECT` statement should also be in your `GROUP BY` clause unless they have a 1-to-1 relationship with each other or are at the same level of granularity.

```
USE titanic_db;

-- Returns 891 records.
SELECT *
FROM passengers;

-- Check out my data types.
DESCRIBE passengers;

-- What does a single row or record represent in the passengers table?
SELECT *
FROM passengers
LIMIT 10;
```

I can change the granularity or level of detail present in my result set by returning the unique values in a column or grouping by a specific dimension in my table. I can zoom in and out using GROUP BY.

```
-- Since passenger_id is my primary key, each record represents a pass
enger. How many passengers are in my table? I can do an overall count
 of the records or rows. (There are 891 passengers in my table.)
SELECT
    COUNT(*) AS number_of_passengers
FROM passengers;



-- Return only the unique values from the class column. Now what does
 a single row represent? Our result set contains much less detail. Zoo
m out.
-- Returns 3 rows.
SELECT
    DISTINCT class
FROM passengers;

-- The GROUP BY clause returns unique values in ascending order by def
ault.
SELECT
    class
FROM passengers
GROUP BY class;

-- If I want to reverse the order of the values, I can use the `DESC`
 keyword just like with ORDER BY.
SELECT
    class
FROM passengers
GROUP BY class DESC;

-- Return only the unique values from the sex column.
SELECT
    DISTINCT sex
FROM passengers;

-- The GROUP BY clause returns unique values in ascending order by def
ault.
SELECT
    sex
FROM passengers
GROUP BY sex;

-- I can select multiple columns to return all of the unique combinati
ons of values in the selected rows.
SELECT
    sex,
    class
FROM passengers
GROUP BY sex, class;
```

Using `GROUP BY` to Aggregate Data

- A `GROUP BY` clause can be combined with an aggregate function to turn a range of numbers into a single data point or summary metric based on a variety of mathematical operations.
  - average, minimum, maximum, count, variance, standard deviation
- If our column contains NULL values, the aggregate function ignores the NULL values.

```sql
-- What if I want to look at the number of rows in a column for each u
nique value? I can use an aggregate function.
SELECT
    sex,
    COUNT(*) as number_of_passengers
FROM passengers
GROUP BY sex;

-- The * returns the count of non-NULL values in the column; I can use
the specific column name if I'm not concerned about NULL values.
SELECT
    sex,
    COUNT(sex) as number_of_passengers
FROM passengers
GROUP BY sex;

-- Check out the difference. Just be aware of this difference when you
decide what you pass to the COUNT function.
SELECT
    deck,
    COUNT(deck) as 'non-null-values',
    COUNT(*) as 'all_rows'
FROM passengers
GROUP BY deck;

-- What if we want to look at the number of rows for each unique combi
nation of values in multiple columns?
SELECT
    sex,
    class,
    COUNT(*) as number_of_passengers
FROM passengers
GROUP BY sex, class;

-- Let's choose another dimension and a couple of different measures t
o further analyze our data. I'm formatting using the ROUND() function.
SELECT
    sex,
    ROUND(AVG(fare), 2) AS average_fare_paid,
    MIN(fare) AS minimum_fare_paid,
    ROUND(MAX(fare), 2) AS maximum_fare_paid,
    ROUND(STDDEV(fare), 2) AS standard_deviation_in_fare
FROM passengers
GROUP BY sex;

-- Let's drill down one more layer by adding the sub-dimension class.
 I'm basically creating a table of summary statistics for my table.
SELECT
    sex,
    class,
    ROUND(AVG(fare), 2) AS average_fare_paid,
    ROUND(MIN(fare), 2) AS minimum_fare_paid,
    ROUND(MAX(fare), 2) AS maximum_fare_paid,
    ROUND(STDDEV(fare), 2) AS standard_deviation_in_fare
FROM passengers
GROUP BY sex, class;
```