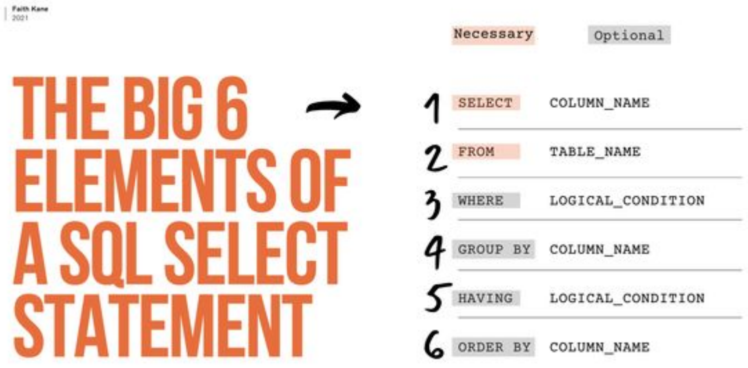
SQL `WHERE` Clause

Lesson Goals

- Introduce the `WHERE` clause for filtering
- Understand how to combine `WHERE` clauses with the `AND` and `OR` operators to create more complex queries
- Use keywords like `DISTINCT`, `LIKE`, and `BETWEEN` as well as the wildcard `%` in `WHERE` clauses to further filter

What Are the Elements of the SQL Select Statement?



Faith Kane
2021

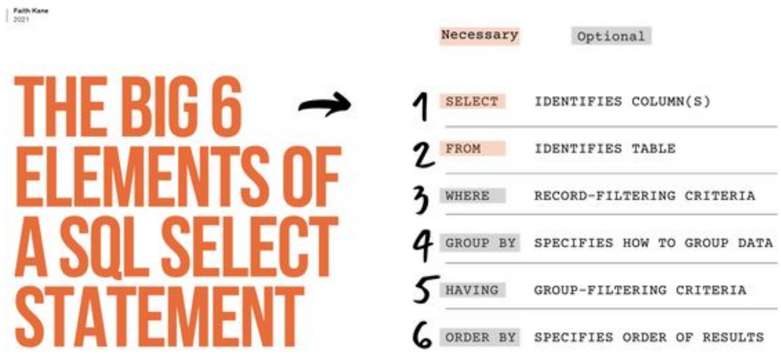
THE BIG 6 ELEMENTS OF A SQL SELECT STATEMENT

Necessary Optional

1	SELECT	COLUMN_NAME
2	FROM	TABLE_NAME
3	WHERE	LOGICAL_CONDITION
4	GROUP BY	COLUMN_NAME
5	HAVING	LOGICAL_CONDITION
6	ORDER BY	COLUMN_NAME

So What Does Each Element Do?

- I can add specific conditions to my queries using a `WHERE` clause to filter or limit the result set to records that satisfy my specific criteria or condition.
- I can use different keywords or operators with a `WHERE` clause to further filter the result set from my query. We will see this in our examples below.



Faith Kane
2021

THE BIG 6 ELEMENTS OF A SQL SELECT STATEMENT

Necessary Optional

1	SELECT	IDENTIFIES COLUMN(S)
2	FROM	IDENTIFIES TABLE
3	WHERE	RECORD-FILTERING CRITERIA
4	GROUP BY	SPECIFIES HOW TO GROUP DATA
5	HAVING	GROUP-FILTERING CRITERIA
6	ORDER BY	SPECIFIES ORDER OF RESULTS

Contents

Now What?

- Let's code!

Basic `WHERE` Clause

```sql
-- Use the albums database.
USE albums_db;

-- Return all of the rows and columns from the albums table. (returns 31 records)
SELECT *
FROM albums;

-- Return all of the unique values in the artist column. (returns 23 records)
SELECT DISTINCT artist
FROM albums;

-- Return all of the albums that were released in 1990. (returns 1 record)
SELECT name
FROM albums
WHERE release_date = 1990;
```

`WHERE` with `LIKE` Keyword and `%` Wildcard

```sql
-- Return all of the albums that contain the pattern 'at' somewhere in the titles. (returns 5 records)
SELECT name
FROM albums
WHERE name LIKE '%at%';

-- Return all of the albums wiht the word 'the' somewhere in them. (returns  9 records)
SELECT name
FROM albums
WHERE name LIKE '%the%';

-- Return all of the albums that end in the letter 'a'. (returns 2 records)
SELECT name
FROM albums
WHERE name LIKE '%a';

-- Return all of the albums that start with the letter 'a'. (returns 2  records)
SELECT name
FROM albums
WHERE name LIKE 'a%';

--Return all of the albums that begin and end with the letter 'a'. (returns 0 records)
SELECT name
FROM albums
WHERE name LIKE 'a%a';
```

Contents

WHERE with BETWEEN and IN Operators

```sql
-- Return all of the albums that were released in the 1990s. (returns 11 records)
SELECT
    name,
    release_date
FROM albums
WHERE release_date BETWEEN 1990 AND 1999;

-- Return the artist name, album title, and sales for all of the albums that sold between 10 and 20 million copies (returns 13 records)
SELECT
    artist,
    name,
    sales
FROM albums
WHERE sales BETWEEN 10 and 20;

-- Switch to the chipotle database for the rest of the demo.
USE chipotle;

-- Check out the data types of my orders table.
DESCRIBE orders;

-- Return all of the rows and columns from the orders table. (returns 4622 records)
SELECT *
FROM orders;

-- Return all of the unique values in the item_name column. (returns 50 records)
SELECT DISTINCT item_name
FROM orders;

-- Return all of the records that have chicken in the name. (returns 1560  records)
SELECT item_name
FROM orders
WHERE item_name LIKE '%chicken%';

-- Return only the non-repeating values for item_name that have chicken in the name. (returns 6 records)
SELECT DISTINCT item_name
FROM orders
WHERE item_name LIKE '%chicken%';

/*
Return only the records that have either 'Veggie Soft Tacos', 'Crispy Tacos', or 'Steak Bowl' as a value in item_name.
(returns 220 records)
*/
SELECT *
FROM orders
WHERE item_name IN ('Veggie Soft Tacos', 'Steak Bowl', 'Crispy Tacos');

-- Return only the records with order number 1, 7, or 10. (returns 8 records)
SELECT *
FROM orders
WHERE order_id IN (1, 7, 10);
```

Contents

WHERE with IS NULL and IS NOT NULL

```
-- Use my own db; yours will be your username database.
USE join_example_db;

-- Return all of the columns and rows from the users table. (returns 6 records)
SELECT *
FROM users;

-- Return only records that don't have NULL values in role_id; (returns 4 records)
SELECT *
FROM users
WHERE role_id IS NOT NULL;

-- Return only records that don't have NULL values in role_id; (returns 2 records)
SELECT *
FROM users
WHERE role_id IS NULL;
```

Contents

`WHERE` with `AND` & `OR` Operators

- We can use the `OR` operator to find values that match any of our criteria, while using the `AND` operator requires that a value match all of our criteria.
- 'Three words: order of operations. It's like you learned in math, certain operators take precedence over others (like multiplying comes before adding) unless you use parentheses to force it your way.' source (https://stackoverflow.com/questions/8172455/why-would-the-addition-of-parentheses-in-sql-query-cause-the-results-to-change/8172521)
- `NOT` has a higher precedence than `AND` which has a higher precedence than `OR` .
- Here (https://www.geeksforgeeks.org/mysql-operator-precedence/) is a very simple article reiterating what I'll demo below concerning operator precedence if you want a quick review later. Always nice to see things multiple times and in multiple ways.

Contents

```sql
/*
Return only the records that have either 'Veggie Soft Tacos', 'Crispy Tacos', or 'Steak Bowl' as a value in item_name chaining OR operators.
(returns 220 records)
*/
SELECT *
FROM orders
WHERE item_name = 'Veggie Soft Tacos' OR item_name = 'Steak Bowl' OR item_name = 'Crispy Tacos';

-- Return only the records with order number 1, 7, or 10 chaining OR operators. (returns 8 records)
SELECT *
FROM orders
WHERE order_id = 1 OR order_id = 7 OR order_id = 10;

-- Return only the records that have Chicken in the name OR are a part of order 10. (returns 1561 records)
SELECT *
FROM orders
WHERE item_name LIKE '%chicken%'
    OR order_id = 10;

-- Return only the records that have Chicken in the name AND are a part of order 10. (returns 1 record)
SELECT *
FROM orders
WHERE item_name LIKE '%chicken%'
    AND order_id = 10;

/*
Return only the records that have the name 'Veggie Soft Tacos' AND have order_id 304 or 322
OR any items that have the name 'Crispy Tacos'.
(returns 4 records)
*/
SELECT *
FROM orders
WHERE item_name = 'Veggie Soft Tacos'
    AND order_id IN (304, 322)
    OR item_name = 'Crispy Tacos';

/*
Return only the records that have either the order_id 304 or 322
OR have the name 'Crispy Tacos'
-->(conditions grouped with parentheses are evaluated first)
AND have the name 'Veggie Soft Tacos'.
(returns 2 records)
*/
SELECT *
FROM orders
WHERE item_name = 'Veggie Soft Tacos'
    AND (
        order_id IN (304, 322)
        OR item_name = 'Crispy Tacos'
        );

/*
The records returned have red in the description AND tacos in the name as well as any records that have soft in the name.
(returns 618 records)
```