SQL Basics Part 3 Notes

Team Treehouse

June 3, 2020

1 Searching Tables with 'WHERE'

- WHERE clause
  - **Syntax:** SELECT *columns* FROM *table name* WHERE *condition*;
  - **Syntax (Condition):** *Columns Operator Value*
- Equality Operator
  - **Syntax:** SELECT *columns* FROM *table name* WHERE *column name* = *value*;

Examples:

```
SELECT * FROM contacts WHERE first_name = "Andrew";


SELECT first_name, email FROM users WHERE last_name = "Chalkley";


SELECT name AS "Product Name" FROM products WHERE stock_count = 0;


SELECT title "Book Title" FROM books WHERE year_published = 1999;

```

- Inequality Operator
  - **Syntax:** SELECT *columns* FROM *table name* WHERE *column name* != *value*;

Examples:

```sql
SELECT * FROM contacts WHERE first_name != "Kenneth";


SELECT first_name, email FROM users WHERE last_name != "L:one";


SELECT name AS "Product Name" FROM products WHERE stock_count != 0;


SELECT title "Book Title" FROM books WHERE year_published != 2015;

```

- Greater than/ Less than Operator
  - **Syntax (less than):** SELECT *columns* FROM *table name* WHERE *column name* < *value*;
  - **Syntax (greater than):** SELECT *columns* FROM *table name* WHERE *column name* > *value*;
- Cheat Sheet: Link

2 Exercise 1

- Solution included in *exercise_1.sql*

3 Filtering by Comparing Values

- **Syntax (Less than):** SELECT *columns* FROM *table name* WHERE *column name* < *value*;
- **Syntax (Less than or equal):** SELECT *columns* FROM *table name* WHERE *column name* <= *value*;
- **Syntax (Greater than):** SELECT *columns* FROM *table name* WHERE *column name* > *value*;
- **Syntax (Greater than or equal):** SELECT *columns* FROM *table name* WHERE *column name* >= *value*;

Example:

```sql
SELECT first_name, last_name FROM users WHERE date_of_birth < '1998-12-01';


SELECT title AS "Book Title", author AS Author FROM books WHERE year_released <= 2015;


SELECT name, description FROM products WHERE price > 9.99;


SELECT title FROM movies WHERE release_year >= 2000;

```

4 Exercise 2

- Solution included in *exercise_2.sql*

5 Filtering on More than One Condition

- Is used when filtering with multiple conditions
- Can be done using *AND* and/or *OR* operator
- **Syntax (AND):** SELECT *columns* FROM *table name* WHERE ¡condition 1¿ AND ¡condition 2¿ ...;
- **Syntax (OR):** SELECT *columns* FROM *table name* WHERE ¡condition 1¿ OR ¡condition 2¿ ...;

Examples:

```sql
SELECT username FROM users WHERE last_name = "Chalkley" AND first_name = "Andrew";


SELECT * FROM products WHERE category = "Games Consoles" AND price < 400;


SELECT * FROM movies WHERE title = "The Matrix" OR title = "The Matrix Reloaded" OR title = "The Matrix Revolutions";

```

```
9
10    SELECT country FROM countries WHERE population < 1000000 OR
      population > 100000000;
11
```

6 Exercise 3

- Solution included in *exercise_3.sql*

7 Filtering By Dates

- Is done using comparison operators (same as part 3).
- **Syntax (Less than):** SELECT *columns* FROM *table name* WHERE *column name* < *value*;
- **Syntax (Less than or equal):** SELECT *columns* FROM *table name* WHERE *column name* <= *value*;
- **Syntax (Greater than):** SELECT *columns* FROM *table name* WHERE *column name* > *value*;
- **Syntax (Greater than or equal):** SELECT *columns* FROM *table name* WHERE *column name* >= *value*;

Examples:

```
1     SELECT first_name, last_name FROM users WHERE date_of_birth < '
      1998-12-01';
2
3
4     SELECT title AS "Book Title", author AS Author FROM books WHERE
      year_released <= 2015;
5
6
7     SELECT name, description FROM products WHERE price > 9.99;
8
9
10    SELECT title FROM movies WHERE release_year >= 2000;
11
```

8 Exercise 4

- Solution included in *exercise_4.sql*

9 Searching Within a Set of Values

- Returns results with matching sets of values in a columns
- Is similar to Python's *x in [Value1, value2, ...]*
- **Syntax:** SELECT *columns* FROM *table name* WHERE *column name* IN (*value 1*, *value 2*, ...);
- **Syntax (Negation):** SELECT *columns* FROM *table name* WHERE *column name* NOT IN (*value 1*, *value 2*, ...);

Examples:

```
1   SELECT name FROM islands WHERE id IN (4, 8, 15, 16, 23, 42);
2
3
4   SELECT * FROM products WHERE category IN ("eBooks", "Books", "
    Comics");
5
6
7   SELECT title FROM courses WHERE topic IN ("JavaScript", "Databases
    ", "CSS");
8
9
10  SELECT * FROM campaigns WHERE medium IN ("email", "blog", "ppc");
11
12
13  SELECT * FROM products WHERE category NOT IN ("Electronics");
14
15
16  SELECT title FROM courses WHERE topic NOT IN ("SQL", "NoSQL");
17
```

10 Exercise 5

- Solution included in *exercise_5.sql*

11 Searching Within a Range of Values

- Returns results between *lesser value* and *greater value*
- **Syntax:** SELECT *columns* FROM *table name* WHERE *column name* BETWEEN *lesser value* AND *greater value*;

Examples:

```
SELECT * FROM movies WHERE release_year BETWEEN 2000 AND 2010;


SELECT name, description FROM products WHERE price BETWEEN 9.99
AND 19.99;


SELECT name, appointment_date FROM appointments WHERE
appointment_date BETWEEN "2015-01-01" AND "2015-01-07";

```

12 Exercise 6

- Solution included in *exercise_6.sql*

13 Finding Data that Matches a Pattern

- LIKE operator
  - Is used inside of *WHERE* clause to match a pattern
  - **Syntax:** SELECT *columns* FROM *table name* WHERE *column name* LIKE *pattern*;
  - Can be used to make search case insensitive

```
SELECT title FROM books WHERE title LIKE "Harry Potter";
// returns items like 'Harry potter', 'harry potter'


```

- LIKE operator with wild card %
  - Works to match zero or more unspecified characters

- works the same as '' in regex

```
SELECT title FROM books WHERE title LIKE "Harry Potter%Fire";
// returns items like 'Harry Potter and Dragon Fire', 'Harry Potter and Fire', 'Harry Potter Rising Fire'

SELECT title FROM movies WHERE title LIKE "Alien%";
// Returns items like 'Alien attack', 'Alien', "Alienate"


SELECT * FROM contacts WHERE first_name LIKE "%drew";
// Returns items like 'tigerdrew', 'mountaindrew', 'morning drew', 'andrew'


SELECT * FROM books WHERE title LIKE "%Brief History%";
// Returns items like 'Canadian Brief History Channel', 'Brief History'

```

14 Exercise 7

- Solution included in *exercise_7.sql*

15 Filtering Out or Finding Missing Information

- Using IS NULL
  - Is used in WHERE
  - Retrieve rows with information missing.
  - **Syntax:** SELECT *columns* FROM *table name* WHERE *column name* IS NULL;
- Using IS NOT NULL
  - **Syntax:** SELECT *columns* FROM *table name* WHERE *column name* IS NOT NULL;

Example:

```
SELECT address FROM records WHERE address IS NOT NULL;

```