# 📕 SQL for Data Analyst 102

| | |
|---|---|
| ↗ Parent item | 📽️OnDemand Class |
| Status | Success |

## 💡Table of Contents💡



---

# How to use WHERE clause?

## Use WHERE to filter data

เราใช้ `WHERE` เวลาที่เราต้องการจะ filter เฉพาะข้อมูลแถวที่เราต้องการจาก Tables

99.99% ของ query ที่เราเขียนในชีวิตจริงส่วนใหญ่ต้องเขียน `WHERE` เสมอเลย ยิ่งเราเขียน query ได้ specific มากเท่าไหร่ ยิ่งช่วยให้ analysis เราตอบโจทย์ได้แม่นยำขึ้นเท่านั้น

```
SELECT * FROM customers WHERE country = 'USA';
```

## Common WHERE usage

ตัวอย่างด้านล่างคือตัวการเขียน `WHERE` ที่เราใช้กันบ่อยๆเวลาเขียน `SQL` query

🌵 `LIKE` คือการเขียน pattern matching ปกติเราจะใช้ `LIKE` กับ wildcard คือ `%` หรือ `_`

- `%` ใช้ match any character กี่ตัวก็ได้
- `_` ใช้ match single character ตัวเดียว

ตัวอย่างเช่น `WHERE country LIKE 'U%'` จะฟิลเตอร์เฉพาะประเทศที่ขึ้นต้นด้วยตัว U ทั้งหมด

หรือ `WHERE firstname LIKE 'J_hn'` จะฟิลเตอร์ชื่อ firstname ลูกค้าขึ้นต้นด้วยตัว J ตามด้วยตัวอักษรอะไรก็ได้หนึ่งตัวและปิดท้ายด้วย hn เช่น `John` `Jahn` `Jihn` `Jehn` เป็นต้น

🌵 ใน SQLite ตัว `LIKE` operator จะเป็นแบบ case insensitive ไม่สนตัวพิมเล็กใหญ่ เวลาเขียน `LIKE 'J_hn'` สามารถ match ได้ทั้ง `JOHN` `john` หรือ `JohN` ไม่แตกต่างกัน

```
SELECT * FROM customers
WHERE country = 'USA';

SELECT * FROM customers
WHERE country = 'USA' AND state = 'CA';

SELECT * FROM customers
WHERE country = 'USA' OR country = 'United Kingdom';

SELECT * FROM customers
WHERE country IN ('USA', 'United Kingdom');

SELECT * FROM customers
WHERE country NOT IN ('USA', 'United Kingdom');

SELECT * FROM customers
WHERE email LIKE '%@gmail.com';

SELECT * FROM customers
WHERE email NOT LIKE '%gmail.com';

SELECT * FROM customers
WHERE company IS NULL;

SELECT * FROM customers
WHERE company IS NOT NULL;
```

```
SELECT * FROM customers
WHERE customerid BETWEEN 10 AND 15;
```

---

# Filter Data 1

วิธีใช้ `AND` และ `OR` และ `NOT`

```
SELECT * FROM customers
WHERE country = 'USA' AND state = 'CA';
-- AND คือ และ   USA และ CA เท่านั้น

SELECT * FROM customers
WHERE country = 'USA' OR state = 'CA';
-- OR คือ หรือ   USA หรือ CA ก็ได้

SELECT * FROM customers
WHERE NOT (country = 'USA' OR state = 'CA');
-- NOT คือ ไม่เอา  ในที่นี้คือ ไม่เอา USA และ CA
```

ถ้าเรากลัวว่าจะเขียนชื่อประเทศผิดเพราะบางคำเป็นตัวพิมพ์ใหญ่ ให้เราใช้ `LOWER`

```
SELECT * FROM customers
WHERE LOWER(country) = 'united kingdom';
```

---

# Filter Data 2

วิธีใช้ `IN` มีค่าเหมือน OR  และ `NOT IN` คือไม่อยู่ใน

```
SELECT * FROM customers
WHERE country = 'Brazil' OR country = 'Germany' OR country =

--เขียนให้สั้นลงได้โดยใช้คำสั่ง IN กับ NOT IN

SELECT * FROM customers
WHERE country IN ('Brazil','Germany','Norway');
```

```
SELECT * FROM customers
WHERE country NOT IN ('Brazil','Germany','Norway');
```

วิธีใช้ `BETWEEN` ในการกำหนดเงื่อนไขตัวเลขเป็นช่วงได้

```
SELECT * FROM customers
WHERE customerid >= 5 AND customerid <= 10;

--เขียนให้สั้นลงได้โดยใช้คำสั่ง BETWEEN

SELECT * FROM customers
WHERE customerid BETWEEN 5 AND 10;

SELECT invoicedate FROM invoices
WHERE Invoicedate BETWEEN '2009-01-01 00:00:00' AND '2009-02-
```

วิธีจัดการกับค่าว่าง(null) ด้วย `IS NULL` และ `IS NOT NULL`

```
SELECT * FROM customers
WHERE company IS NULL;

SELECT * FROM customers
WHERE company IS NOT NULL;
```

---

# Filter Data 3

`LIKE` คือคำสั่งหาตัวที่เหมือนกันกับตัวที่ต้องการ เช่น คนที่ใช้ ‘@gmail.com’

`%` คือ จะเป็นตัวอะไรก็ได้กี่ตัวก็ได้ขึ้นอยู่กับตำแหน่งที่ใส่

`_` คือ จะเป็นตัวอะไรก็ได้ 1 ตัว

```
-- pattern matching
SELECT firstname,lastname,country,email
FROM customers
WHERE email LIKE '%@gmail.com'; -- wildcard %
-- หาคนที่ใช้อีเมลล์ที่ลงท้ายด้วย @gmail.com
```

```
SELECT firstname,lastname,country,email,phone
FROM customers
WHERE email NOT LIKE '%@hotmail.com';
-- หาคนที่ไม่ใช้อีเมลล์ที่ลงท้ายด้วย @hotmail.com

SELECT firstname,lastname,country,email,phone
FROM customers
WHERE phone LIKE '%99%';
-- หาเบอร์โทรศัพท์ที่ มี 99 อยู่ในตำแหน่งไหนก็ได้

SELECT firstname,lastname,country,email,phone
FROM customers
WHERE firstname LIKE 'Leoni_';
-- หาชื่อ Leoni_ คำสุดท้ายจะลงท้ายด้วยอะไรก็ได้
```

---

# Coalesce

วิธีจัดการกับค่าว่าง ด้วย `COALESCE` และ `CASE WHEN`

`COALESCE` คือคำสั่งจัดการกับค่าว่าง ด้วยการแทนค่าลงไป

`CASE WHEN` คือคำสั่งเงื่อนไข เรากำหนดว่าถ้าเป็นค่าว่าง และถ้าไม่ว่าง

```
SELECT
    company,
    COALESCE(company,'End Customer') AS 'Company Clean'
FROM customers;

SELECT
    company,
    CASE WHEN company IS NULL THEN 'End Customer'
        ELSE 'Coporate'
    END AS 'Company Clean 2'
FROM customers;
```

---

# JOIN Data using WHERE

การ JOIN TABLE โดยใช้ `WHERE` ทำได้เหมือนกับ INNER JOIN

> 💡 Primary Key (PK) = Foreign Key (FK) เท่านั้น

```
SELECT * FROM artists,albums
WHERE artists.artistid = albums.artistid;

SELECT
    artists.artistid,
    artists.name AS artist_name,
    albums.title AS album_name
FROM artists,albums
WHERE artists.artistid = albums.artistid  --PK = FK
    AND artists.artistid IN (8,100,120);

SELECT
    artists.artistid,
    artists.name AS artist_name,
    albums.title AS album_name,
    tracks.name AS song_name
FROM artists,albums,tracks
WHERE artists.artistid = albums.artistid -- PK = FK
    AND albums.albumid = tracks.albumid
    AND artists.artistid IN (1,50,100);
```

---

# Aggregate Function

Aggregate functions คือฟังก์ชันสถิติเบื้องต้นไว้สรุปผลข้อมูล ฟังก์ชันที่เราเป็นประจำใน standard SQL จะมีอยู่ 5 functions คือ

- `COUNT` นับจำนวนทั้งหมด
- `AVG` หาค่าเฉลี่ย
- `SUM` ผลบวกรวมทั้งหมด
- `MIN` หาค่าน้อยสุด

- `MAX` หาค่ามากสุด

📣 สิ่งที่ทุกคนต้องรู้คือ aggregate functions ไม่สนใจค่า `NULL` ในคอลัมน์นั้นๆ เช่น

`SELECT COUNT(company) FROM customers;`

Query นี้เราต้องการนับจำนวน company ใน customers table

ถ้าข้อมูลใน customers table มีทั้งหมด 59 rows แต่เรา `COUNT(company)` ได้แค่ 10 rows แปลว่าคอลัมน์ company มีค่า NULL ทั้งหมด 49 rows (คิดจาก 59 - 10)

```
SELECT
    ROUND(AVG(milliseconds),2) AS avg_mill,
    SUM(milliseconds) AS sum_mill,
    MAX(milliseconds) AS max_mill,
    MIN(milliseconds) AS min_mill,
    COUNT(milliseconds) AS count_mill
FROM tracks;
```

---

# Count Distinct

`DISTINCT` คือ นับค่าแบบไม่ซ้ำกัน  ถ้า COUNT ปกติจะนับค่าแบบซ้ำกัน

```
SELECT DISTINCT country FROM customers;
-- แสดงทุกประเทศ แบบไม่ซ้ำกัน

SELECT COUNT(DISTINCT country),COUNT(*) FROM customers;
-- นับจำนวนประเทศที่ไม่ซ้ำกัน , และนับจำนวนทั้งหมด
```

`COUNT(*)` คือ การนับจำนวน row ทั้งหมด

---

# Group By

เราใช้ `GROUP BY` กับ `Aggregate Functions` เพื่อสรุปข้อมูลแบ่งตามกลุ่มที่เราต้องการ ตัวอย่าง เช่น การนับจำนวนลูกค้าในแต่ละประเทศ

```
SELECT
    country,
    COUNT(*) AS count_country
```

```
FROM customers
GROUP BY country;
-- นับจำนวนลูกค้าในแต่ละประเทศ

SELECT
    genres.name,
    COUNT(*) AS count_songs
FROM genres,tracks
WHERE genres.genreid = tracks.genreid
GROUP BY genres.name;
-- JOIN 2 ตาราง แล้วนับจำนวนประเภทของเพลง ว่าแต่ละประเภท มีกี่เพลง
```

`COUNT(*)` คือ การนับจำนวน row ทั้งหมด

> 💡 Tip - คอลัมน์ไหนที่อยู่ `GROUP BY` ให้เราเขียนคอลัมน์นั้นใน `SELECT` ด้วย 😄

---

# Having

`HAVING` เขียนเหมือนกับ `WHERE`

แต่เราใช้ `HAVING` ในการกรองข้อมูลที่ผ่านการ `GROUP BY` มาแล้วนะครับ 😄

อธิบายง่ายๆคือการกรองกลุ่มที่เราต้องการนั่นเอง

```
SELECT
    genres.name,
    COUNT(*) AS count_songs
FROM genres,tracks
WHERE genres.genreid = tracks.genreid AND genres.name NOT LIK
GROUP BY genres.name
HAVING COUNT(*) >= 100;
```

| | Name | count_songs |
|---|---|---|
| 1 | Alternative & Punk | 332 |
| 2 | Jazz | 130 |
| 3 | Latin | 579 |
| 4 | Metal | 374 |

`WHERE` ใช้ในการ Filter Table

`HAVING` ใช้ในการ Filter กลุ่มที่ผ่านการ `GROUP BY` แล้ว

---

# Order By & Limit

`ORDER BY` ใช้เพื่อ sort data เรียงข้อมูลจากน้อยไปมาก (default, ascending order) หรือถ้าอยากเรียงจากมากไปน้อยให้ใส่ `DESC` ต่อท้ายชื่อคอลัมน์

```
SELECT
    genres.name,
    COUNT(*)
FROM genres,tracks
WHERE genres.genreid = tracks.genreid
GROUP BY genres.name
ORDER BY COUNT(*) DESC
LIMIT 5;
```

| | Name | COUNT(*) |
|---|---|---|
| 1 | Rock | 1297 |
| 2 | Latin | 579 |
| 3 | Metal | 374 |
| 4 | Alternative & Punk | 332 |
| 5 | Jazz | 130 |

---