

# Filtering numbers

INTERMEDIATE SQL

SQL

Jasmin Ludolf

Data Science Content Developer,  
DataCamp

# WHERE

- WHERE filtering clause



# WHERE

```
WHERE color = 'green'
```



# WHERE with comparison operators

```
SELECT title  
FROM films  
WHERE release_year > 1960;
```

```
|title          |  
|-----|  
|Judgment at Nuremberg|  
|Pocketful of Miracles|  
|The Hustler       |  
|The Misfits       |  
...  
...
```

# Comparison operators

```
SELECT title  
FROM films  
WHERE release_year < 1960;
```

```
|title  
|-----|  
|Intolerance:Love's Struggle Throughout the Ages|  
|Over the Hill to the Poorhouse|  
|The Big Parade|  
|Metropolis|  
...|
```

# Comparison operators

```
SELECT title  
FROM films  
WHERE release_year <= 1960;
```

```
|title  
|-----|  
|Intolerance:Love's Struggle Throughout the Ages|  
|Over the Hill to the Poorhouse|  
|The Big Parade|  
|Metropolis|  
...|
```

# Comparison operators

```
SELECT title  
FROM films  
WHERE release_year = 1960;
```

```
|title      |  
|-----|  
|Elmer Gantry |  
|Psycho      |  
|The Apartment|
```

# Comparison operators

```
SELECT title  
FROM films  
WHERE release_year <> 1960;
```

```
|title  
|-----|  
|Intolerance:Love's Struggle Throughout the Ages|  
|Over the Hill to the Poorhouse|  
|The Big Parade|  
|Metropolis|  
...|
```

# Comparison operators

- `>` Greater than or after
- `<` Less than or before
- `=` Equal to
- `>=` Greater than or equal to
- `<=` Less than or equal to
- `<>` Not equal to

# WHERE with strings

- Use single-quotes around strings we want to filter

```
SELECT title  
FROM films  
WHERE country = 'Japan';
```

title
-----
Seven Samurai
Tora! Tora! Tora!
Akira
Madadayo
Street Fighter
...

# Order of execution

-- Written code:

```
SELECT item  
FROM coats  
WHERE color = 'green'  
LIMIT 5;
```

- Order of execution:

- **FROM** coats
- **WHERE** color = 'green'
- **SELECT** item
- **LIMIT** 5;

# **Let's practice!**

**INTERMEDIATE SQL**

# Multiple criteria

INTERMEDIATE SQL

SQL

Jasmin Ludolf

Data Science Content Developer,  
DataCamp

# Multiple criteria



# Multiple criteria



# Multiple criteria



# Multiple criteria

- OR , AND , BETWEEN

```
SELECT *
FROM coats
WHERE color = 'yellow' OR length = 'short';
```

```
SELECT *
FROM coats
WHERE color = 'yellow' AND length = 'short';
```

```
SELECT *
FROM coats
WHERE buttons BETWEEN 1 AND 5;
```

# OR operator

- Use OR when you need to satisfy at least one condition



# OR operator

- Correct:

```
SELECT title  
FROM films  
WHERE release_year = 1994  
      OR release_year = 2000;
```

```
|title  
|-----|  
|3 Ninjas Kick Back |  
|A Low Down Dirty Shame |  
|Ace Ventura:Pet Detective|  
...
```

- Invalid:

```
SELECT title  
FROM films  
WHERE release_year = 1994 OR 2000;
```

argument of OR must be type boolean,  
not type integer  
LINE 3: WHERE release\_year = 1994  
 OR 2000;  
 ^

# AND operator

- Use `AND` if we need to satisfy all criteria
- Correct:

```
SELECT title  
FROM films  
WHERE release_year > 1994  
    AND release_year < 2000;
```

```
|title  
|-----|  
|Ace Ventura:When Nature Calls|  
|Apollo 13|  
|Assassins|  
|Babe|  
|...|
```

- Invalid:

```
SELECT title  
FROM films  
WHERE release_year > 1994 AND < 2000;
```

```
syntax error at or near "[removed]  
1994 AND < 2000;  
^
```

# AND, OR

- Filter films released in 1994 or 1995, and certified PG or R
- Enclose individual clauses in parentheses

```
SELECT title
FROM films
WHERE (release_year = 1994 OR release_year = 1995)
      AND (certification = 'PG' OR certification = 'R');
```

title	
3 Ninjas Kick Back	
A Low Down Dirty Shame	
Baby's Day Out	
Beverly Hills Cop III	
...	

# BETWEEN, AND

```
SELECT title  
FROM films  
WHERE release_year >= 1994  
AND release_year <= 2000;
```

```
SELECT title  
FROM films  
WHERE release_year  
BETWEEN 1994 AND 2000;
```

title	
-----	
3 Ninjas Kick Back	
A Low Down Dirty Shame	
Ace Ventura:Pet Detective	
Baby's Day Out	
...	

title	
-----	
3 Ninjas Kick Back	
A Low Down Dirty Shame	
Ace Ventura:Pet Detective	
Baby's Day Out	
...	

# BETWEEN, AND, OR

```
SELECT title  
FROM films  
WHERE release_year  
BETWEEN 1994 AND 2000 AND country='UK';
```

title
Four Weddings and a Funeral
The Hudsucker Proxy
Dead Man Walking
GoldenEye
...

# **Let's practice!**

**INTERMEDIATE SQL**

# Filtering text

INTERMEDIATE SQL

SQL

Jasmin Ludolf

Data Science Content Developer,  
DataCamp

# Filtering text

- WHERE can also filter text

```
SELECT title  
FROM films  
WHERE country = 'Japan';
```

title
-----
Seven Samurai
Tora! Tora! Tora!
Akira
Madadayo
Street Fighter

# Filtering text

- Filter a pattern rather than specific text
- `LIKE`
- `NOT LIKE`
- `IN`

# LIKE

- Used to search for a pattern in a field

% match zero, one, or many characters

\_ match a single character

```
SELECT name  
FROM people  
WHERE name LIKE 'Ade%';
```

```
| name      |  
|-----|  
| Adel Karam |  
| Adelaide Kane |  
| Aden Young |
```

```
SELECT name  
FROM people  
WHERE name LIKE 'Ev_';
```

```
| name      |  
|-----|  
| Eve      |
```

- Ev\_ Mendes

# NOT LIKE

```
SELECT name  
FROM people;
```

name
-----
50 Cent
A. Michael Baldwin
A. Raven Cruz
A.J. Buckley
A.J. DeLucia
...

```
SELECT name  
FROM people  
WHERE name NOT LIKE 'A.%';
```

name
-----
50 Cent
Aaliyah
Aaron Ashmore
Aaron Hann
...

# Wildcard position

```
SELECT name  
FROM people  
WHERE name LIKE '%r';
```

name
-----
A.J. Langer
Aaron Schneider
Aaron Seltzer
Abigail Spencer
...

```
SELECT name  
FROM people  
WHERE name LIKE '_t%';
```

name
-----
Aitana Sánchez-Gijón
Anthony 'Critic' Campos
Anthony Bell
Anthony Burrell
...

# WHERE, OR

```
SELECT title  
FROM films  
WHERE release_year = 1920  
OR release_year = 1930  
OR release_year = 1940;
```

title
Over the Hill to the Poorhouse
Hell's Angels
Boom Town
...

# WHERE, IN

```
SELECT title  
FROM films  
WHERE release_year IN (1920, 1930, 1940);
```

```
|title  
|-----|  
|Over the Hill to the Poorhouse|  
|Hell's Angels|  
|Boom Town|  
...  
...
```

# WHERE, IN

```
SELECT title  
FROM films  
WHERE country IN ('Germany', 'France');
```

title	
Metropolis	
Pandora's Box	
The Train	
...	

# **Let's practice!**

**INTERMEDIATE SQL**

# NULL values

INTERMEDIATE SQL

SQL

Jasmin Ludolf

Data Science Content Developer,  
DataCamp

# Missing values

- `COUNT(field_name)` includes only non-missing values
- `COUNT(*)` includes missing values

null

- Missing values:
  - Human error
  - Information not available
  - Unknown

# null

```
SELECT COUNT(*) AS count_records  
FROM people;
```

```
|count_records|  
|-----|  
|8397|
```

```
SELECT *  
FROM people;
```

id	name	birthdate	deathdate
1	50 Cent	1975-07-06	null
2	A. Michael Baldwin	1963-04-04	null
3	A. Raven Cruz	null	null
...			

# IS NULL

```
SELECT name  
FROM people  
WHERE birthdate IS NULL;
```

name
A. Raven Cruz
A.J. DeLucia
Aaron Hann
...

# IS NOT NULL

```
SELECT COUNT(*) AS no_birthdates  
FROM people  
WHERE birthdate IS NULL;
```

```
|no_birthdates|  
|-----|  
|2245|
```

```
SELECT COUNT(name) AS count_birthdates  
FROM people  
WHERE birthdate IS NOT NULL;
```

```
|count_birthdates|  
|-----|  
|6152|
```

# COUNT() vs IS NOT NULL

**SELECT**

```
COUNT(certification)  
AS count_certification  
FROM films;
```

```
|count_certification|  
|-----|  
|4666|
```

**SELECT**

```
COUNT(certification)  
AS count_certification  
FROM films  
WHERE certification IS NOT NULL;
```

```
|count_certification|  
|-----|  
|4666|
```

# NULL put simply

- `NULL` values are missing values
- Very common
- Use `IS NULL` or `IS NOT NULL` to:
  - Identify missing values
  - Select missing values
  - Exclude missing values

# **Let's practice!**

**INTERMEDIATE SQL**