Introduction to data driven decision making

DATA-DRIVEN DECISION MAKING IN SQL

Irene Ortner
Data Scientist at Applied Statistics

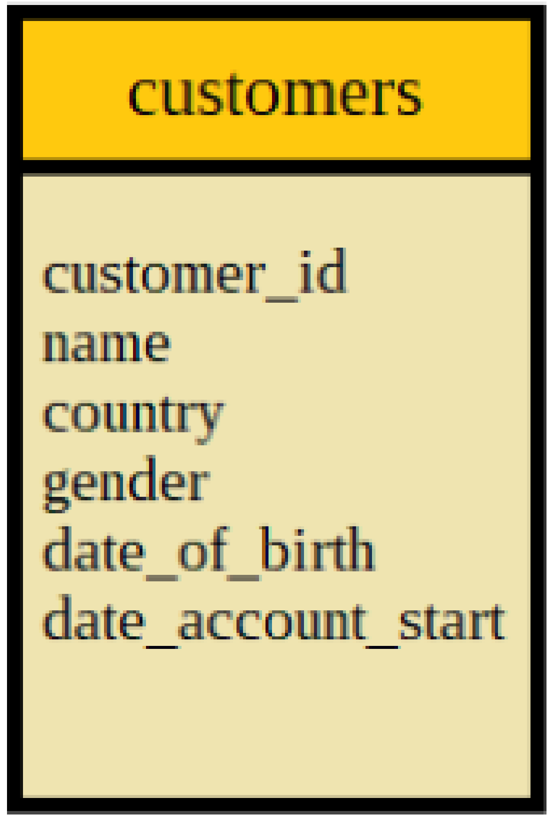
Aim of this course

- A short review of SQL know-how
- Apply your SQL know-how to extract business insights from data
- Learn about new SQL statements to summarize data
  - OLAP extensions were developed specifically for business intelligence
  - Examples are CUBE, ROLLUP and GROUPING SETS

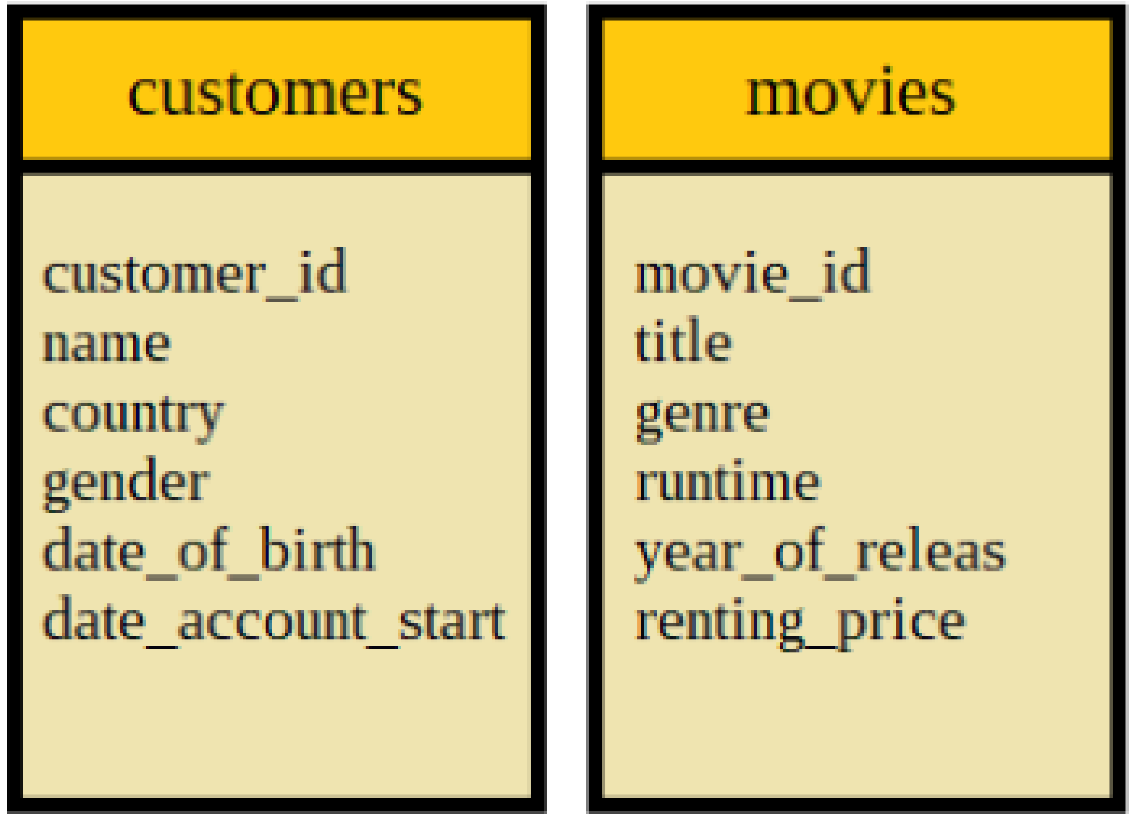
MovieNow: an online movie rental company

- Platform to stream movies
- Additional information for each movie: genre, main actors, etc.
- Customer information
- Customers can give a rating after watching a movie

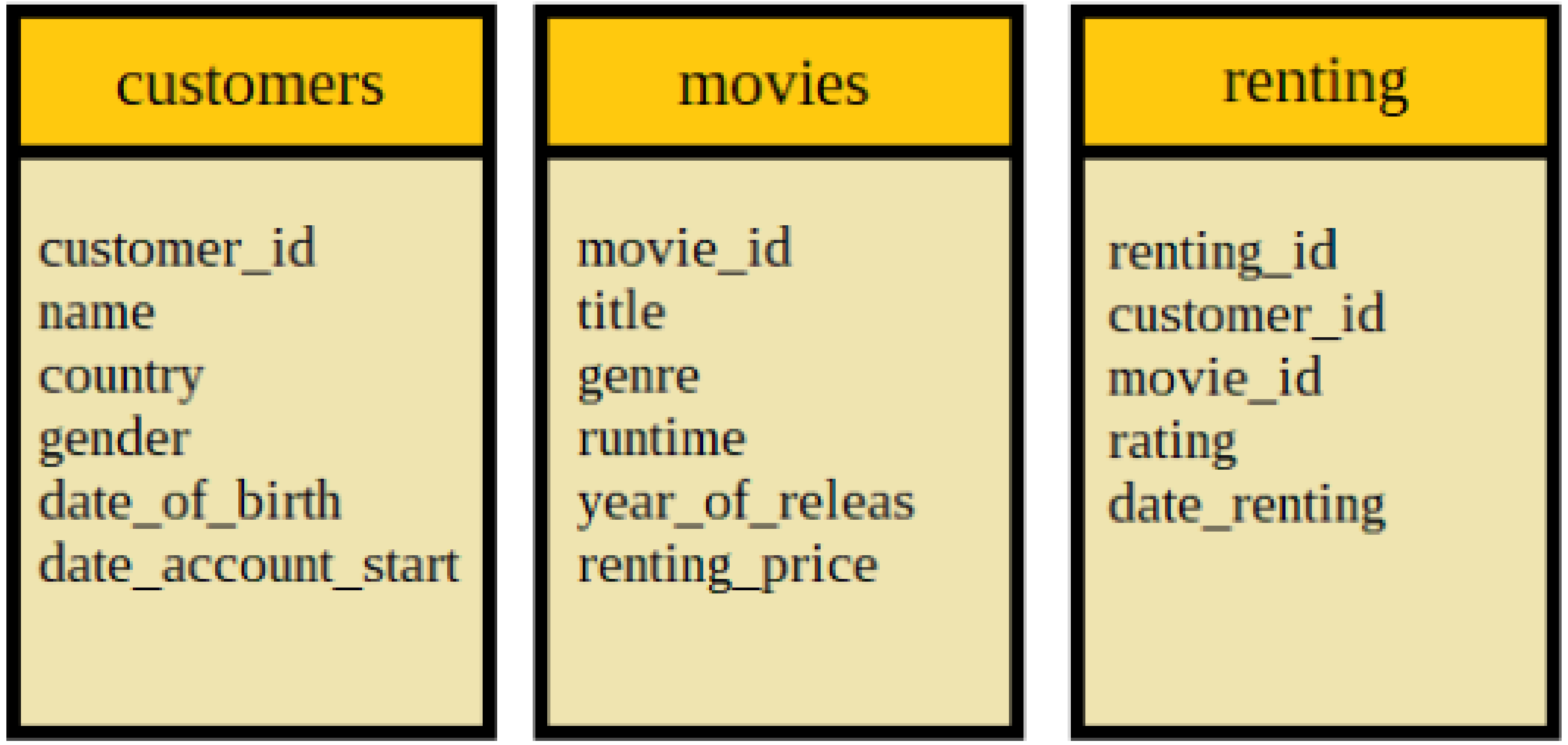
MovieNow data structure



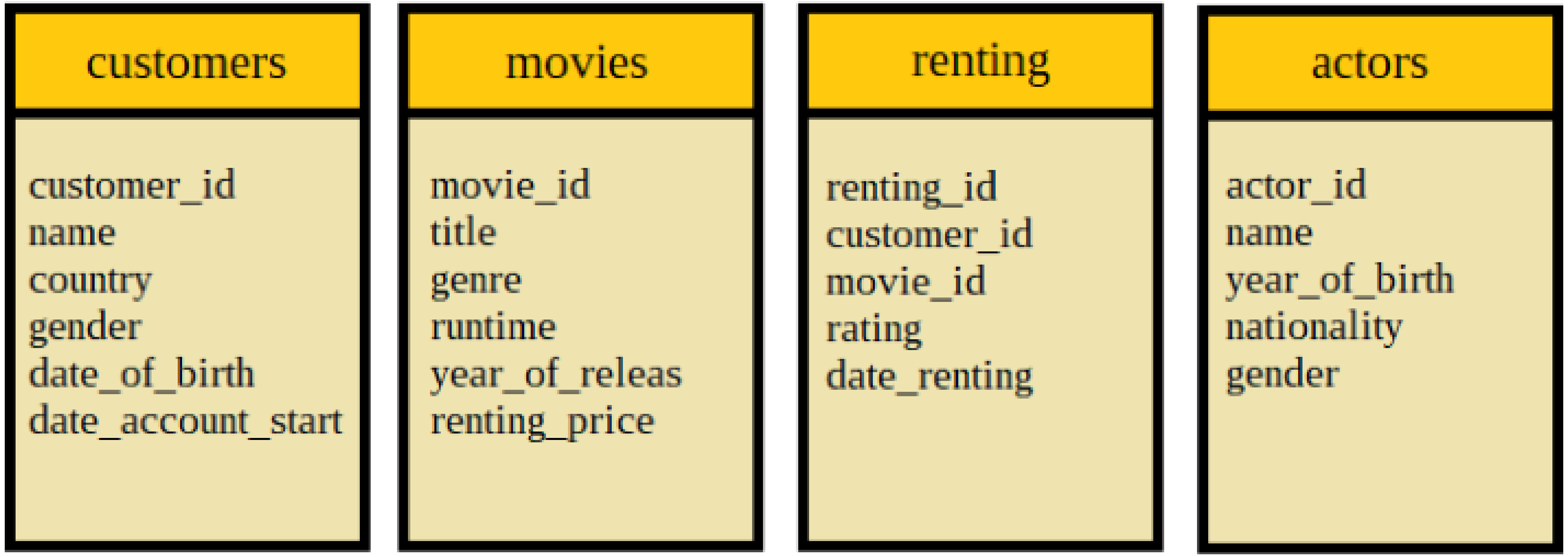
MovieNow data structure



MovieNow data structure

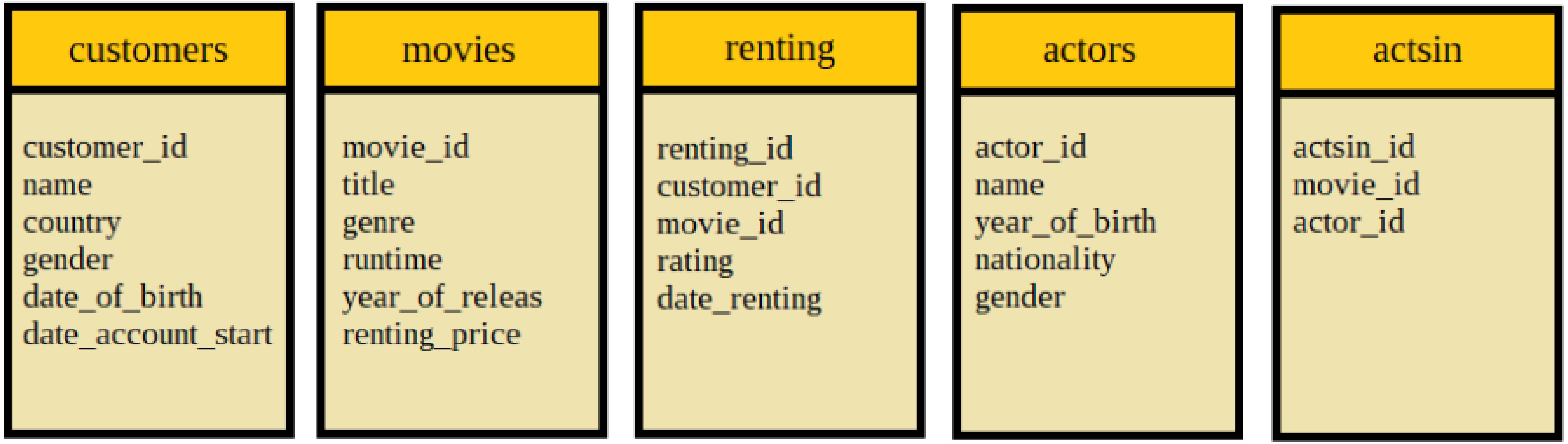


MovieNow data structure



customers	movies	renting	actors
customer_id	movie_id	renting_id	actor_id
name	title	customer_id	name
country	genre	movie_id	year_of_birth
gender	runtime	rating	nationality
date_of_birth	year_of_releas	date_renting	gender
date_account_start	renting_price		

MovieNow data structure



Objectives of data driven decision making

- Information for operational decisions
  - Popularity of actors to decide which movies to invest in.
  - Revenue of the last months to estimate budget for short term investments.
- Information for strategic decisions
  - Success across countries to decide on market extensions.
  - Longterm development of revenue for long term investments.

KPIs: Key Performance Indicators

Extract information from the data which is relevant to measure the success of MovieNow.

- Total number of rentals: revenue
- The average rating of all movies: customer satisfaction
- Number of active customers: customer engagement

Let's get started!

DATA-DRIVEN DECISION MAKING IN SQL

Filtering and ordering

DATA-DRIVEN DECISION MAKING IN SQL

SQL

Tim Verdonck
Professor Statistics and Data Science

datacamp

WHERE

Select all customers from Italy:

```sql
SELECT *
FROM customers
WHERE country = 'Italy';
```

```
|customer_id | name                | country | gender | date_of_birth | date_account_start|
|------------|-------------------|---------|--------|---------------|-------------------|
| 53         | Teresio Panicucci | Italy   | male   | 1999-07-21    | 2018-11-06        |
| 54         | Demetrio Palermo  | Italy   | male   | 1997-10-10    | 2018-10-17        |
| 55         | Facino Milano     | Italy   | male   | 1973-05-23    | 2018-01-02        |
```

Operators in the WHERE clause

- Comparison operators:
  - Equal `=`
  - Not equal `<>`
  - Less than `<`
  - Less than or equal to `<=`
  - Greater than `>`
  - Greater than or equal to `>=`
- `BETWEEN` operator
- `IN` operator
- `IS NULL` and `IS NOT NULL` operators

Example comparison operators

Select all columns from `movies` where the genre is not Drama.

```
SELECT *
FROM movies
WHERE genre <> 'Drama';
```

Select all columns from `movies` where the price for renting is larger equal 2.

```
SELECT *
FROM movies
WHERE renting_price >= 2;
```

Example: BETWEEN operator

Select all columns of `customers` where the date when the account was created is between 2018-01-01 and 2018-09-30.

```
SELECT *
FROM customers
WHERE date_account_start BETWEEN '2018-01-01' AND '2018-09-30';
```

Example: IN operator

Select all actors with nationality USA or Australia.

```
SELECT *
FROM actors
WHERE nationality IN ('USA', 'Australia')
```

Example: NULL operator

Select all columns from `renting` where `rating` is `NULL`.

```
SELECT *
FROM renting
WHERE rating IS NULL
```

Select all columns from `renting` where `rating` is not `NULL`.

```
SELECT *
FROM renting
WHERE rating IS NOT NULL
```

Boolean operators AND

Select customer name and the date when they created their account for customers who are from **Italy** *AND* who **created an account between 2018-01-01 and 2018-09-30**.

```sql
SELECT name, date_account_start
FROM customers
WHERE country = 'Italy'
AND date_account_start BETWEEN '2018-01-01' AND '2018-09-30';
```

```
| name                | date_account_start |
|---------------------|--------------------|
| Facino Milano       | 2018-01-02         |
| Mario Lettiere      | 2018-01-30         |
| Rocco Buccho        | 2018-02-27         |
| Cristoforo Mancini  | 2018-01-12         |
```

Boolean operators OR

Select customer name and the date when they created their account for customers who are from **Italy** _OR_ who **created an account between 2018-01-01 and 2018-09-30**.

```
SELECT name, date_account_start
FROM customers
WHERE country = 'Italy'
OR date_account_start BETWEEN '2018-01-01' AND '2018-09-30';
```

```
| name                   | country | date_account_start |
|------------------------|-------- |--------------------|
| Rowanne Couperus       | Belgium | 2018-08-26         |
| Annelous Sneep         | Belgium | 2018-05-12         |
| Jaëla van den Dolder   | Belgium | 2018-02-08         |
| ...                    | ...     | ...                |
```

ORDER BY

Order the results of a query by rating.

```
SELECT *
FROM renting
WHERE rating IS NOT NULL
ORDER BY rating;
```

```
| renting_id | customer_id | movie_id | rating | date_renting |
|------------|-------------|----------|--------|--------------|
| 552        | 28          | 56       | 1      | 2017-03-27   |
| 558        | 41          | 19       | 3      | 2019-01-13   |
| 444        | 120         | 59       | 3      | 2018-08-10   |
| 200        | 86          | 46       | 3      | 2018-08-26   |
| 234        | 104         | 28       | 4      | 2018-10-04   |
```

ORDER BY ... DESC

Order the results of a query by rating in descending order.

```
SELECT *
FROM renting
WHERE rating IS NOT NULL
ORDER BY rating DESC;
```

```
| renting_id | customer_id | movie_id | rating | date_renting |
|------------|-------------|----------|--------|--------------|
| 243        | 7           | 5        | 10     | 2019-01-11   |
| 18         | 36          | 39       | 10     | 2019-03-20   |
| 396        | 7           | 40       | 10     | 2018-09-11   |
| 487        | 61          | 48       | 10     | 2017-08-14   |
| 476        | 78          | 42       | 10     | 2018-07-04   |
```

Let's practice!

DATA-DRIVEN DECISION MAKING IN SQL

Aggregations - summarizing data

DATA-DRIVEN DECISION MAKING IN SQL

SQL

Bart Baesens
Professor Data Science and Analytics

datacamp

Overview aggregations

```sql
SELECT AVG(renting_price)
FROM movies;
```

Overview aggregations

```
SELECT AVG(renting_price)
FROM movies;
```

Some aggregate functions in SQL

- `AVG()`
- `SUM()`
- `COUNT()`
- `MIN()`
- `MAX()`

Aggregation with NULL values

```
SELECT COUNT(*)
FROM actors;
```

- Result: 145

```
SELECT COUNT(name)
FROM actors;
```

- Result: 145

```
SELECT COUNT(year_of_birth)
FROM actors;
```

- Result: 143

DISTINCT

```sql
SELECT DISTINCT country
FROM customers;
```

```
| country       |
|---------------|
| Spain         |
| Great Britain |
| Austria       |
| Poland        |
| ............. |
```

```sql
SELECT COUNT(DISTINCT country)
FROM customers;
```

- Result: 11

DISTINCT with `NULL` values

```sql
SELECT DISTINCT rating
FROM renting
ORDER BY rating;
```

```
| rating |
|--------|
| 1      |
| ...... |
| 10     |
| null   |
```

Give an alias to column names

```
SELECT AVG(renting_price) AS average_price,
         COUNT(DISTINCT genre) AS number_genres
FROM movies;
```

```
| average_price | number_genres |
|---------------|---------------|
| 2.21          | 8             |
```

- Helps to understand the results when column names are self-explaining.

Let's practice!

DATA-DRIVEN DECISION MAKING IN SQL