Querying a database

INTERMEDIATE SQL

Jasmin Ludolf

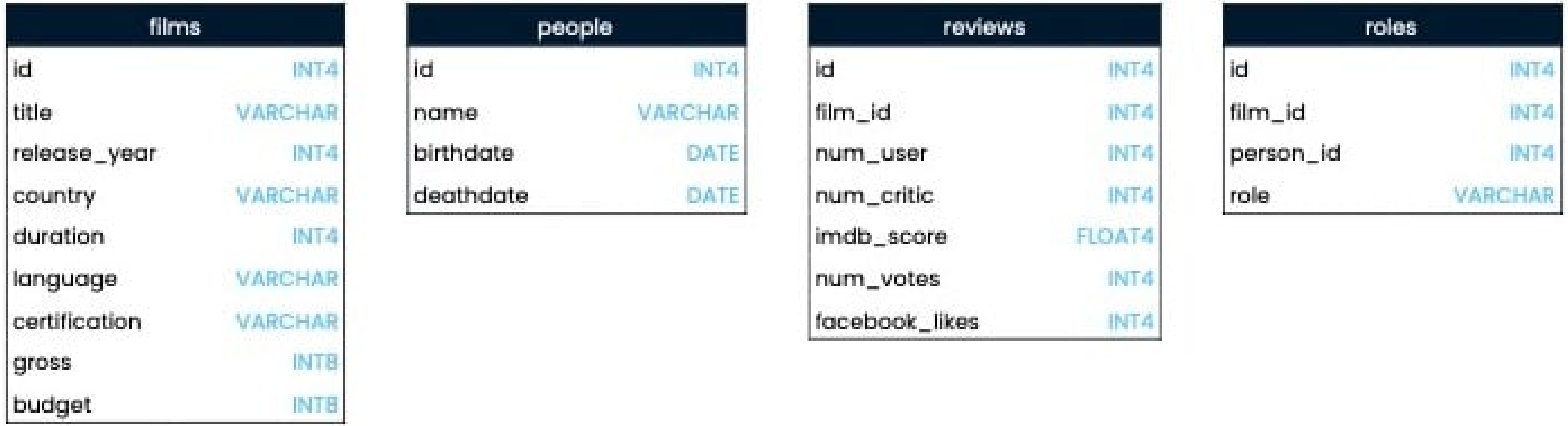
Data Science Content Developer, DataCamp

Course roadmap

- Querying databases
- Count and view specified records
- Understand query execution and style
- Filtering
- Aggregate functions
- Sorting and grouping

PostgreSQL

Our films database



films	
id	INT4
title	VARCHAR
release_year	INT4
country	VARCHAR
duration	INT4
language	VARCHAR
certification	VARCHAR
gross	INT8
budget	INT8

people	
id	INT4
name	VARCHAR
birthdate	DATE
deathdate	DATE

reviews	
id	INT4
film_id	INT4
num_user	INT4
num_critic	INT4
imdb_score	FLOAT4
num_votes	INT4
facebook_likes	INT4

roles	
id	INT4
film_id	INT4
person_id	INT4
role	VARCHAR

COUNT()

- `COUNT()`
- Counts the number of records with a value in a field
- Use an alias for clarity

```sql
SELECT COUNT(birthdate) AS count_birthdates
FROM people;
```

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

COUNT() multiple fields

```
SELECT COUNT(name) AS count_names, COUNT(birthdate) AS count_birthdates
FROM people;
```

```
|count_names|count_birthdates|
|-----------|----------------|
|6397       |6152            |
```

Using * with COUNT()

- `COUNT(field_name)` counts values in a field
- `COUNT(*)` counts records in a table
- `*` represents all fields

```sql
SELECT COUNT(*) AS total_records
FROM people;
```

```
|total_records|
|-------------|
|8397         |
```

DISTINCT

- `DISTINCT` removes duplicates to return only unique values

```
SELECT language
FROM films;
```

```
|language |
|---------|
|Danish   |
|Danish   |
|Greek    |
|Greek    |
|Greek    |
```

- Which languages are in our `films` table?

```
SELECT DISTINCT language
FROM films;
```

```
|language |
|---------|
|Danish   |
|Greek    |
```

COUNT() with DISTINCT

- Combine `COUNT()` with `DISTINCT` to count unique values

```
SELECT COUNT(DISTINCT birthdate) AS count_distinct_birthdates
FROM people;
```

```
|count_distinct_birthdates|
|-------------------------|
|5398                     |
```

- `COUNT()` includes duplicates
- `DISTINCT` excludes duplicates

Let's practice!

INTERMEDIATE SQL

Query execution

INTERMEDIATE SQL

Jasmin Ludolf
Data Science Content Developer, DataCamp

Order of execution

- SQL is not processed in its written order

```
-- Order of execution
SELECT name
FROM people
LIMIT 10;
```

- `LIMIT` limits how many results we return
- Good to know processing order for debugging and aliasing
- Aliases are declared in the `SELECT` statement

Debugging SQL

```sql
SELECT nme
FROM people;
```

```
field "nme" does not exist
LINE 1: SELECT nme
               ^
HINT:  Perhaps you meant to reference the field "people.name".
```

- Misspelling
- Incorrect capitalization
- Incorrect or missing punctuation

Comma errors

- Look out for comma errors!

```
SELECT title, country duration
FROM films;
```

```
syntax error at or near "duration"
LINE 1: SELECT title, country duration
                          ^
```

Keyword errors

```sql
SELCT title, country, duration
FROM films;
```

```
syntax error at or near "SELCT"
LINE 1: SELCT title, country, duration
        ^
```

Final note on errors

Most common errors:

- Misspelling
- Incorrect capitalization
- Incorrect or missing punctuation, especially commas

Learn by making mistakes

Let's practice!

INTERMEDIATE SQL

SQL style

INTERMEDIATE SQL

Jasmin Ludolf
Data Science Content Developer, DataCamp

SQL formatting

- Formatting is not required
- But lack of formatting can cause issues

```sql
select title, release_year, country from films limit 3
```

```
|title                                   |release_year|country|
|----------------------------------------|------------|-------|
|Intolerance: Love's Struggle Throughout the Ages|1916        |USA    |
|Over the Hill to the Poorhouse          |1920        |USA    |
|The Big Parade                          |1925        |USA    |
```

Best practices

```
SELECT title, release_year, country
FROM films
LIMIT 3;
```

```
|title                                           |release_year|country|
|------------------------------------------------|------------|-------|
|Intolerance: Love's Struggle Throughout the Ages|1916        |USA    |
|Over the Hill to the Poorhouse                  |1920        |USA    |
|The Big Parade                                  |1925        |USA    |
```

- Capitalize keywords
- Add new lines

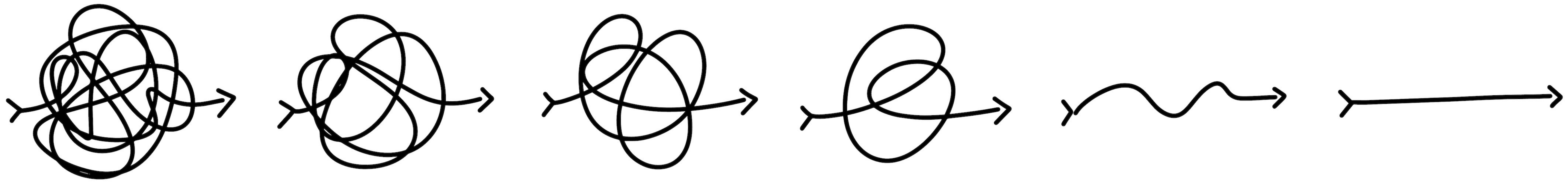
Style guides

```
SELECT
    title,
    release_year,
    country
FROM films
LIMIT 3;
```

```
|title                                   |release_year|country|
|----------------------------------------|------------|-------|
|Intolerance: Love's Struggle Throughout the Ages|1916        |USA    |
|Over the Hill to the Poorhouse          |1920        |USA    |
|The Big Parade                          |1925        |USA    |
```

Style guides

Holywell's style guide: https://www.sqlstyle.guide/



Write clear and readable code

Semicolon

```
SELECT title, release_year, country
FROM films
LIMIT 3;
```

- Best practice
- Easier to translate between SQL flavors
- Indicates the end of a query

Dealing with non-standard field names

- `release year` instead of `release_year`
- Put non-standard field names in double-quotes

```
SELECT title, "release year", country
FROM films
LIMIT 3;
```

Why do we format?

- Easier collaboration
- Clean and readable
- Looks professional
- Easier to understand
- Easier to debug

Let's practice!

INTERMEDIATE SQL