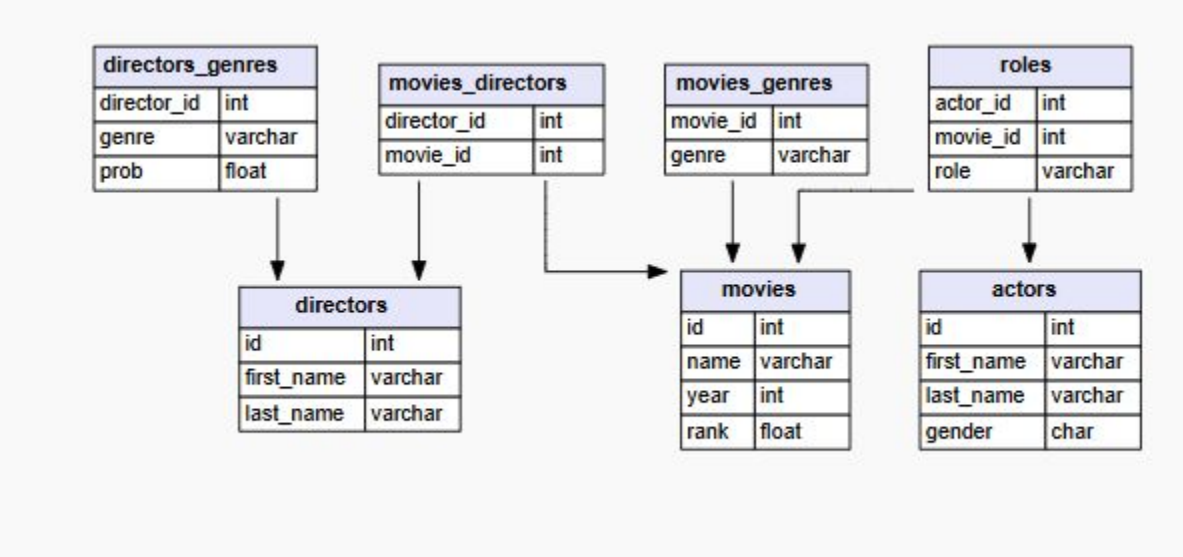
Databases for data analytics

https://github.com/evidencebp/databases-course/

SQL Select

IMDB dataset



Source - https://relational.fel.cvut.cz/dataset/IMDb

Our dataset - https://github.com/evidencebp/databases-course/tree/main/IMDB/Data

Sql

- SEQUEL: A Structured English Query Language, 1974
- C was created in 1972, C++ in 1985, Python in 1991, Java in 1995
- Based on set theory, making it elegant and powerful for analytics

Select - The Superman of data analytics

```
select *
from imdb_ijs.movies
where name like '%Superman%'
order by year;
```

See https://github.com/evidencebp/databases-course/blob/main/Examples/2_Select.txt

Select statement structure (simplified)

```
SELECT
    [ALL | DISTINCT | DISTINCTROW ]
     select_expr [, select_expr] ...
     [FROM table_references
     [WHERE where_condition]
    [GROUP BY {col_name | expr | position}
    [HAVING where_condition]
        [ORDER BY {col_name | expr | position}
      [ASC | DESC], ..]
    [LIMIT {[offset,] row_count | row_count OFFSET offset}]
```

Select - table reference (simplified)

```
table_reference: {    table_factor  | joined_table}

table_factor: {
    tbl_name [ [[AS] alias]
  | table_subquery [AS] alias
  | ( table_references )
}
joined_table: {
    table_reference {[INNER | CROSS] JOIN | STRAIGHT_JOIN} table_factor [join_specification]
  | table_reference {LEFT|RIGHT} [OUTER] JOIN table_reference join_specification
  | table_reference NATURAL [INNER | {LEFT|RIGHT} [OUTER]] JOIN table_factor
}
join_specification: {    ON search_condition  | USING (join_column_list)}
```

In class exercises

- Actors named Marilyn
- Directors named Hitchcock
- Actors whose first and last name start in the same letter
- A row per director and the related genres
- Sherlock Holmes with production periods

Exercises

- All the movies from the eighties
- All movies that have ‘star’, insensitive to case, in their name
- All movies whose name is longer than 80 characters