E03 – SQL Review (Part 2)

Business Intelligence

Exercise

Winter Term 2025/2026

Agenda

- Introduction
  - Introduction
  - Notes on software
- Exercise
  - Database description
  - Tasks
- Credits and materials

Introduction

- The main goal of the “SQL Review” exercise is to refresh your knowledge SQL and make an gentle introduction into a practical world of SQL.

- Despite this set of exercises is dedicated to only refresh existing SQL knowledge, they also could be very useful for the ones who works with the SQL for the first time.

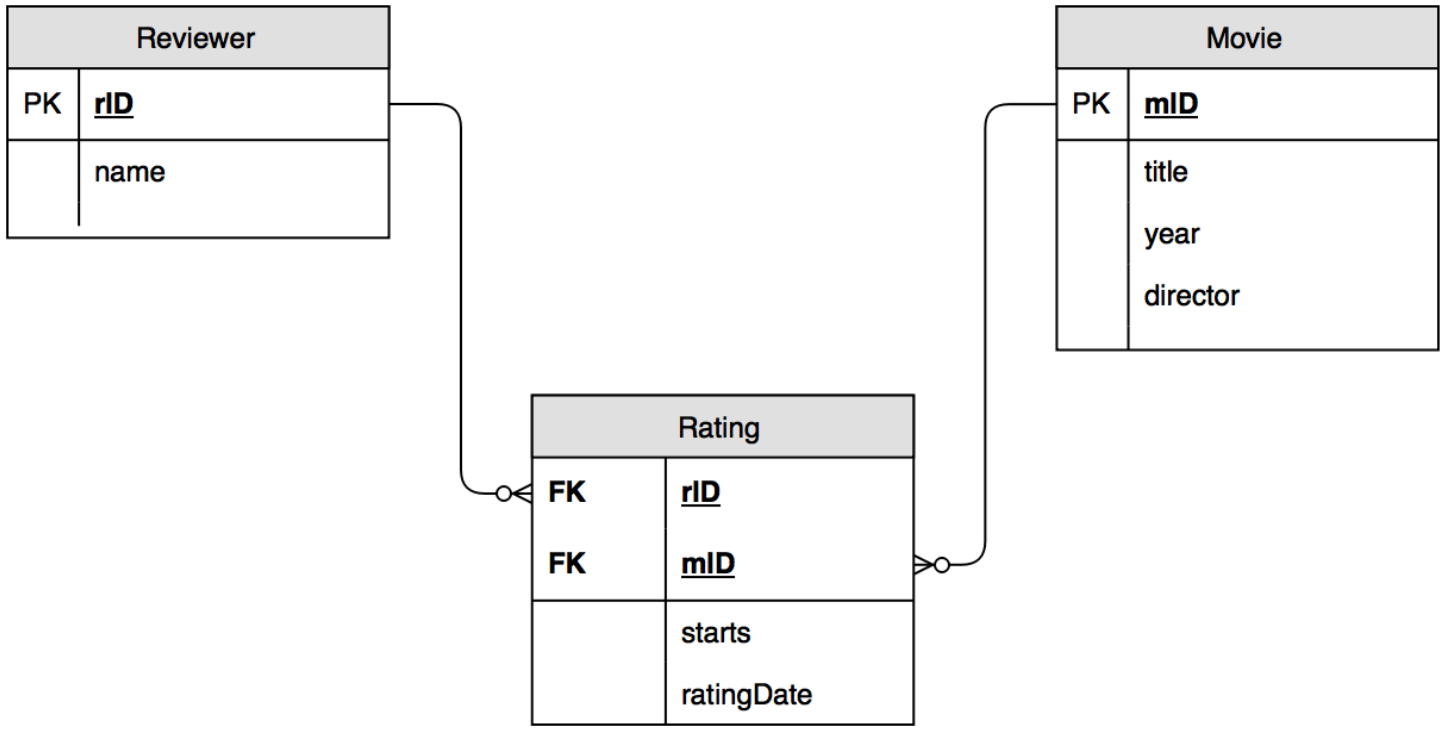
Notes on software

See the materials from the previous exercise **E02**

Exercise

Description - Movie Rating Database

- Assume that you have started a new **movie-rating website**, and you have been collecting data on reviewers' ratings for various movies. There are not much data for a moment, but you can still try out interesting queries in order to get insights about most popular movies, directors, etc..



Database Description (I/III) – Table “Movie”

- Table name:
  - **Movie**
- Columns:
  - **m_id, title, release_year, director**
- Description:
  - **There is a movie with ID number m_id, a title, a release year, and a director.**
- Table Data
  - **select * from movie;**

	m_id	title	release_year	director
1	101	Gone with the Wind	1,939	Victor Fleming
2	102	Star Wars	1,977	George Lucas
3	103	The Sound of Music	1,965	Robert Wise
4	104	E.T.	1,982	Steven Spielberg
5	105	Titanic	1,997	James Cameron
6	106	Snow White	1,937	[NULL]
7	107	Avatar	2,009	James Cameron
8	108	Raiders of the Lost Ark	1,981	Steven Spielberg

Database Description (II/III) – Table “Reviewer”

- Table name:
  - **Reviewer**
- Columns:
  - **r_id, name**
- Description:
  - **The reviewer with ID number r_id has a certain name.**
- Table Data
  - **select * from reviewer;**

	r_id	reviewer_name
1	201	Sarah Martinez
2	202	Daniel Lewis
3	203	Brittany Harris
4	204	Mike Anderson
5	205	Chris Jackson
6	206	Elizabeth Thomas
7	207	James Cameron
8	208	Ashley White

Database Description (III/III) – Table “Rating”

- Table name:
  - **Rating**
- Columns:
  - **r_id, m_id, stars, rating_date**
- Description:
  - **The reviewer rID gave the movie mID a number of stars rating (1-5) on a certain rating date.**
- Table Data
  - **select * from rating;**

	r_id	m_id	stars	rating_date
1	201	101	2	2011-01-22
2	201	101	4	2011-01-27
3	202	106	4	[NULL]
4	203	103	2	2011-01-20
5	203	108	4	2011-01-12
6	203	108	2	2011-01-30
7	204	101	3	2011-01-09
8	205	103	3	2011-01-27
9	205	104	2	2011-01-22
10	205	108	4	[NULL]
11	206	107	3	2011-01-15
12	206	106	5	2011-01-19
13	207	107	5	2011-01-20
14	208	104	3	2011-01-02

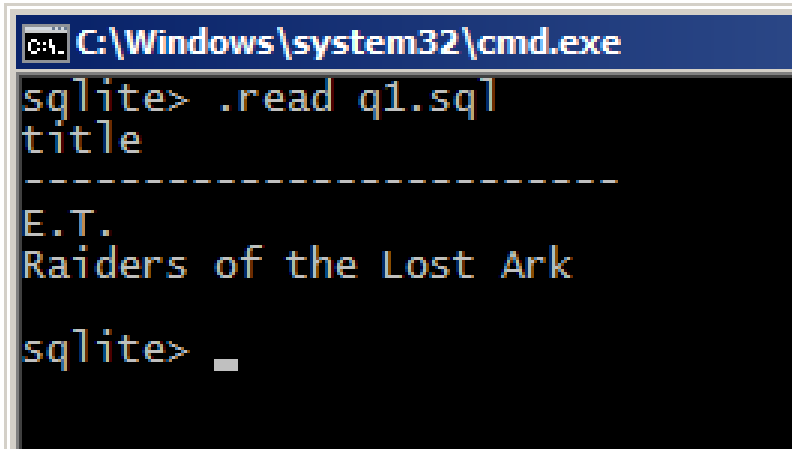
Tasks

Task 01

Task:

Find the titles of all movies directed by Steven Spielberg.

Expected output:



Note: In order to see the correct solution of the first task write (or copy/paste) following command into SQLite's command prompt (or see "**q1.sql"** file)

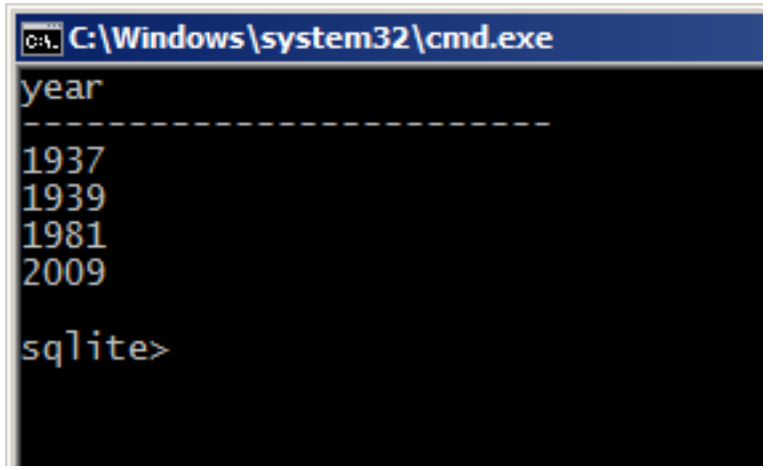
```
.read q1.sql
```

Task 02

Task:

Find all years that have a movie that received a rating of 4 or 5, and sort them in increasing order.

Expected output:

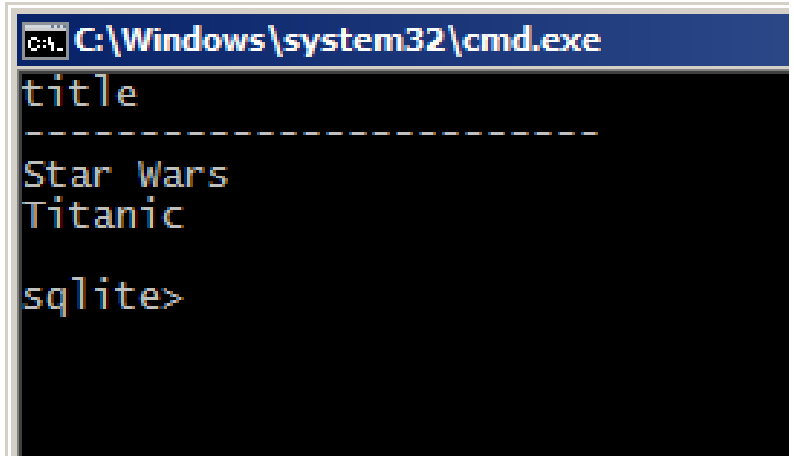


Task 03

Task:

Find the titles of all movies that have no ratings.

Expected output:

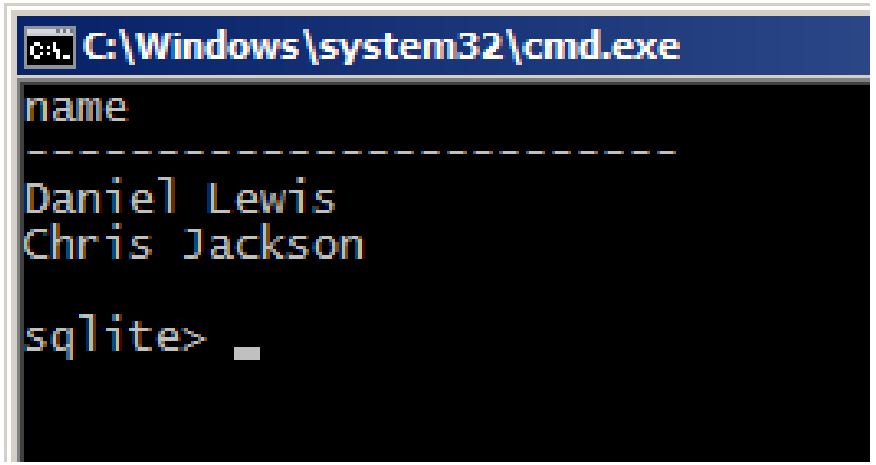


Task 04

Task:

Some reviewers didn’t provide a date with their rating. Find the names of all reviewers who have ratings with a NULL value for the date.

Expected output:

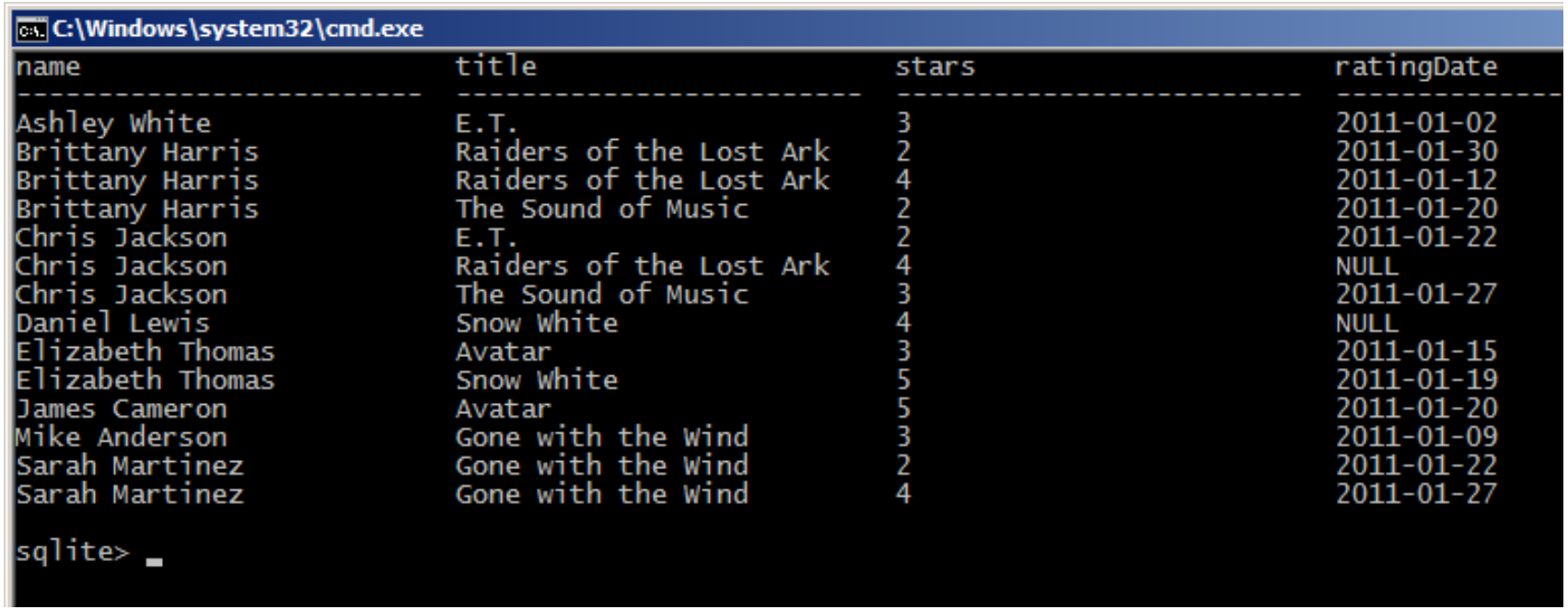


Task 05

Task:

Write a query to return the ratings data in a more readable format: reviewer name, movie title, stars, and ratingDate. Also, sort the data, first by reviewer name, then by movie title, and lastly by number of stars.

Expected output:

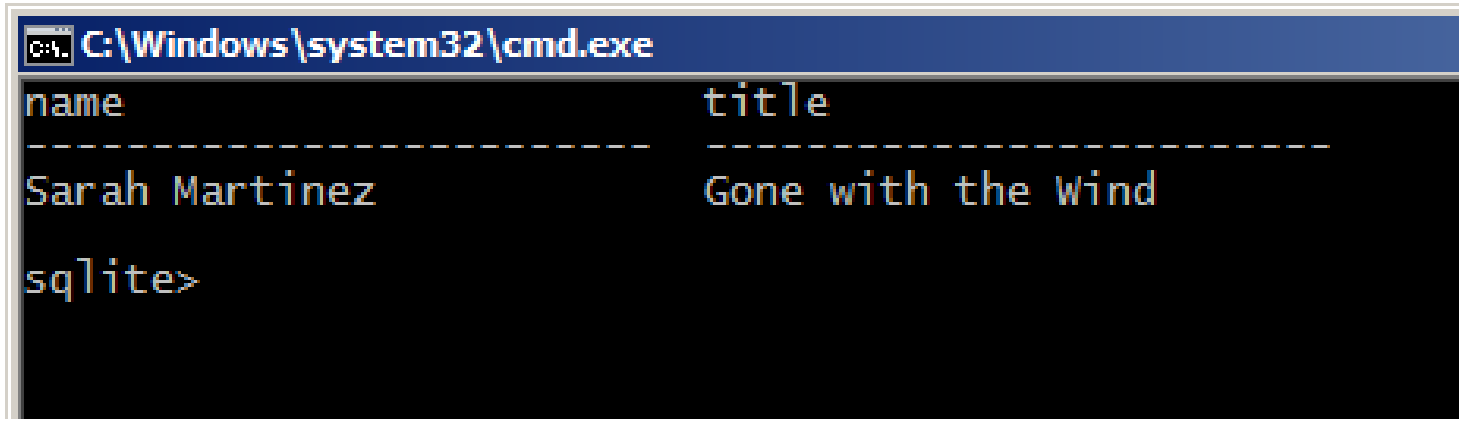


Task 06

Task:

For all cases where the same reviewer rated the same movie twice and gave it a higher rating the second time, return the reviewer's name and the title of the movie.

Expected output:

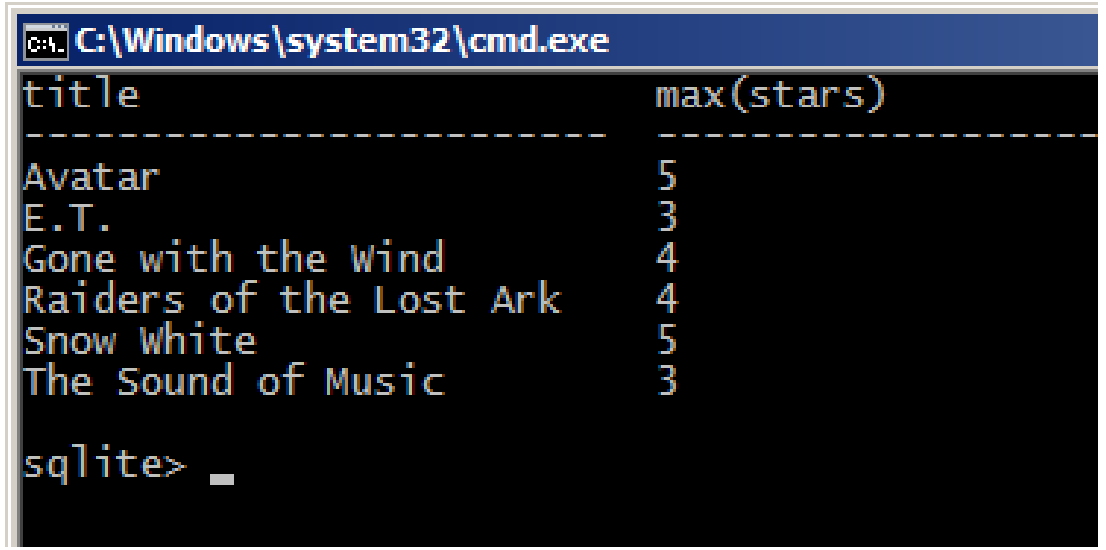


Task 07

Task:

For each movie that has at least one rating, find the highest number of stars that movie received. Return the movie title and number of stars. Sort by movie title.

Expected output:

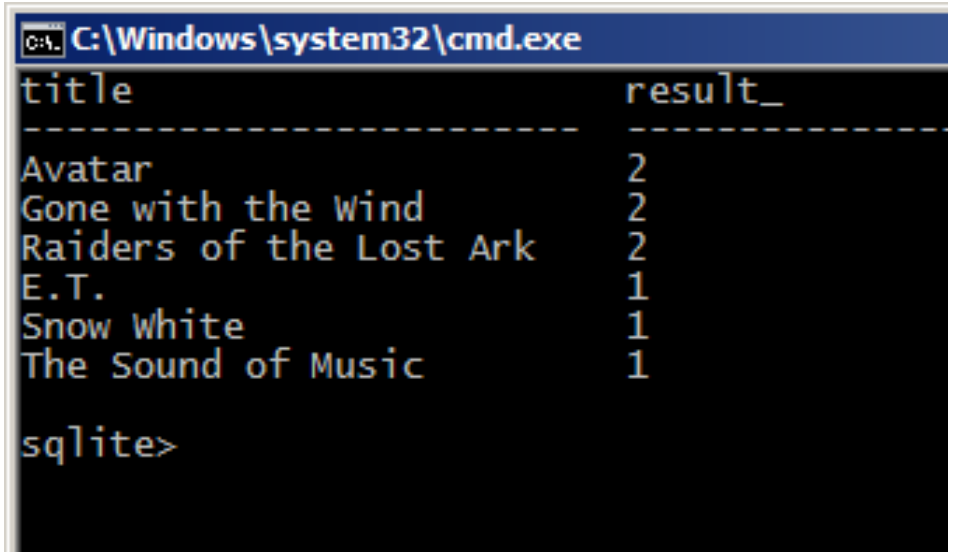


Task 08

Task:

For each movie, return the title and the ‘rating spread’, that is, the difference between highest and lowest ratings given to that movie. Sort by rating spread from highest to lowest, then by movie title.

Expected output:

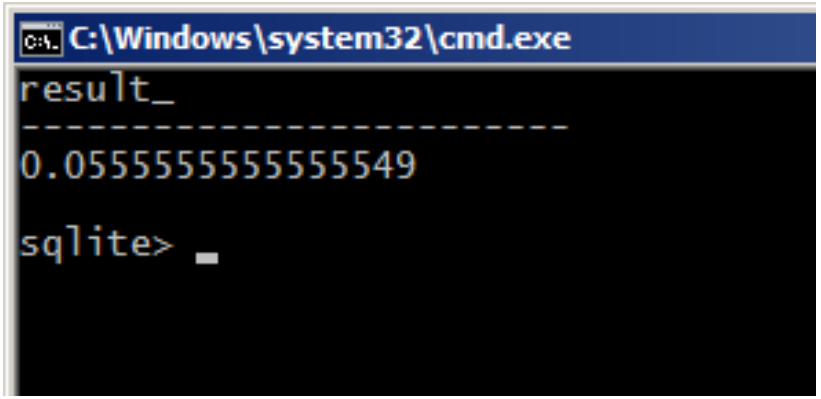


Task 09

Task:

Find the difference between the average rating of movies released before 1980 and the average rating of movies released after 1980. (Make sure to calculate the average rating for each movie, then the average of those averages for movies before 1980 and movies after. Don't just calculate the overall average rating before and after 1980).

Expected output:



Credits and Materials

- *Get started with SQL: Plan and design a database* by Thomas Nield
- Tutorial "Beginner's Guide to Data Modelling" via http://www.databaseanswers.org/tutorial4_data_modelling/index.htm
- “Industry Data Models” via http://www.databaseanswers.org/data_models/
- MOOC course on databases Introduction to Databases by Jennifer Widom.
- “SQL Exercises, Practice, Solution” via http://www.w3resource.com/sql-exercises/
- "SQLBolt - Learn SQL with simple, interactive exercises" via https://sqlbolt.com/
- SQL tasks with different difficulty levels can be found on HackeRank

Submission

- Use StudIP to upload your solution (PDF report with SQL queries)

- You should upload PDF report with verified SQL queries
  - Further information on the SQL verification system is announced separately

- Name convention for your submission file (without extension)
  - **E03**_FIRSTNAME LASTNAME

- Submission deadline (it is a “soft” deadline)
  - 10 days after this exercise starts
  - Some exercises could take a bit more time and could be submitted later
  - NOTE: to receive feedback, you should first submit your progress