Querying Data with SQL

Table of Contents

Dataset Used: https://www.kaggle.com/datasets/ashirwadsangwan/imdb-dataset

Blog Link: https://medium.com/@ghazal3501/how-i-mastered-sql-in-7-days-a-data-engineers-journey-2153382d025e

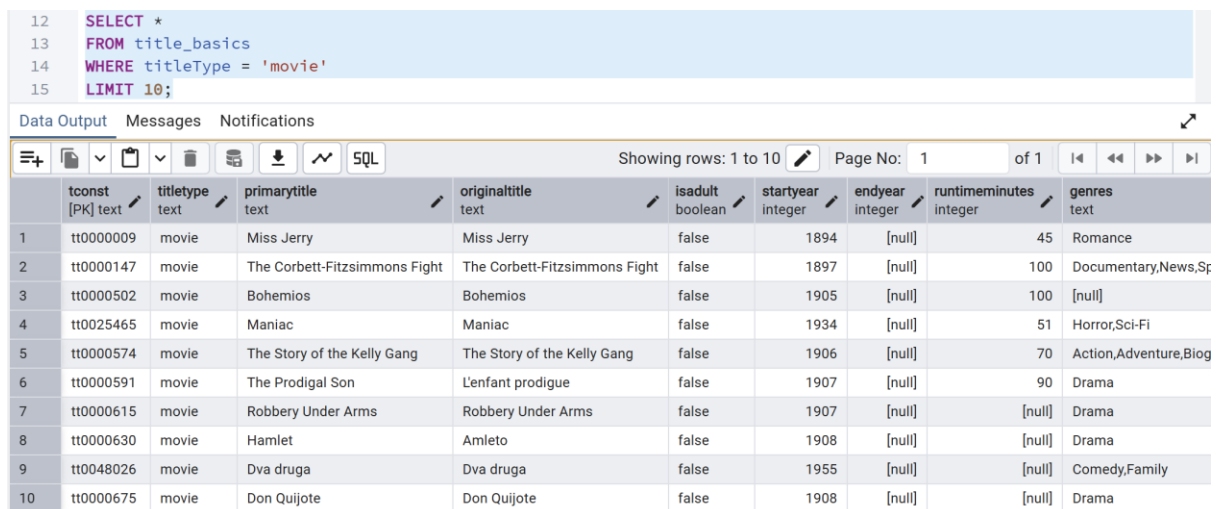**GitHub Repo:** https://github.com/ghazal-ashar/7-day-sql-recap/

Foundation

Query 1: Simple SELECT - Retrieve all movies

Purpose: Get a quick look at movie records in the dataset

Concepts Used: SELECT, WHERE, LIMIT

```
SELECT *
FROM title_basics
WHERE titleType = 'movie'
LIMIT 10;
```

Expected Output: 10 sample movies with all columns

```
12   SELECT *
13   FROM title_basics
14   WHERE titleType = 'movie'
15   LIMIT 10;
```

Data Output   Messages   Notifications

Showing rows: 1 to 10   Page No: 1   of 1

	tconst [PK] text	titletype text	primarytitle text	originaltitle text	isadult boolean	startyear integer	endyear integer	runtimeminutes integer	genres text
1	tt0000009	movie	Miss Jerry	Miss Jerry	false	1894	[null]	45	Romance
2	tt0000147	movie	The Corbett-Fitzsimmons Fight	The Corbett-Fitzsimmons Fight	false	1897	[null]	100	Documentary,News,Sp
3	tt0000502	movie	Bohemios	Bohemios	false	1905	[null]	100	[null]
4	tt0025465	movie	Maniac	Maniac	false	1934	[null]	51	Horror,Sci-Fi
5	tt0000574	movie	The Story of the Kelly Gang	The Story of the Kelly Gang	false	1906	[null]	70	Action,Adventure,Biog
6	tt0000591	movie	The Prodigal Son	L'enfant prodigue	false	1907	[null]	90	Drama
7	tt0000615	movie	Robbery Under Arms	Robbery Under Arms	false	1907	[null]	[null]	Drama
8	tt0000630	movie	Hamlet	Amleto	false	1908	[null]	[null]	Drama
9	tt0048026	movie	Dva druga	Dva druga	false	1955	[null]	[null]	Comedy,Family
10	tt0000675	movie	Don Quijote	Don Quijote	false	1908	[null]	[null]	Drama

Business Insight:

- Provides an initial sense of the dataset's breadth and structure.
- Useful for checking if attributes like runtime, year, and genres are consistently populated.
- Serves as a launchpad for deeper filtering and aggregation work.

Query 2: Column Selection - Movie title, year, runtime

Purpose: Focus only on essential descriptive columns

Concepts Used: SELECT specific columns, WHERE

```
SELECT primaryTitle, startYear, runtimeMinutes
FROM title_basics
WHERE titleType = 'movie'
LIMIT 10;
```

Expected Output: Movie name, release year, runtime

```
25    SELECT primaryTitle, startYear, runtimeMinutes
26    FROM title_basics
27    WHERE titleType = 'movie'
28    LIMIT 10;
```
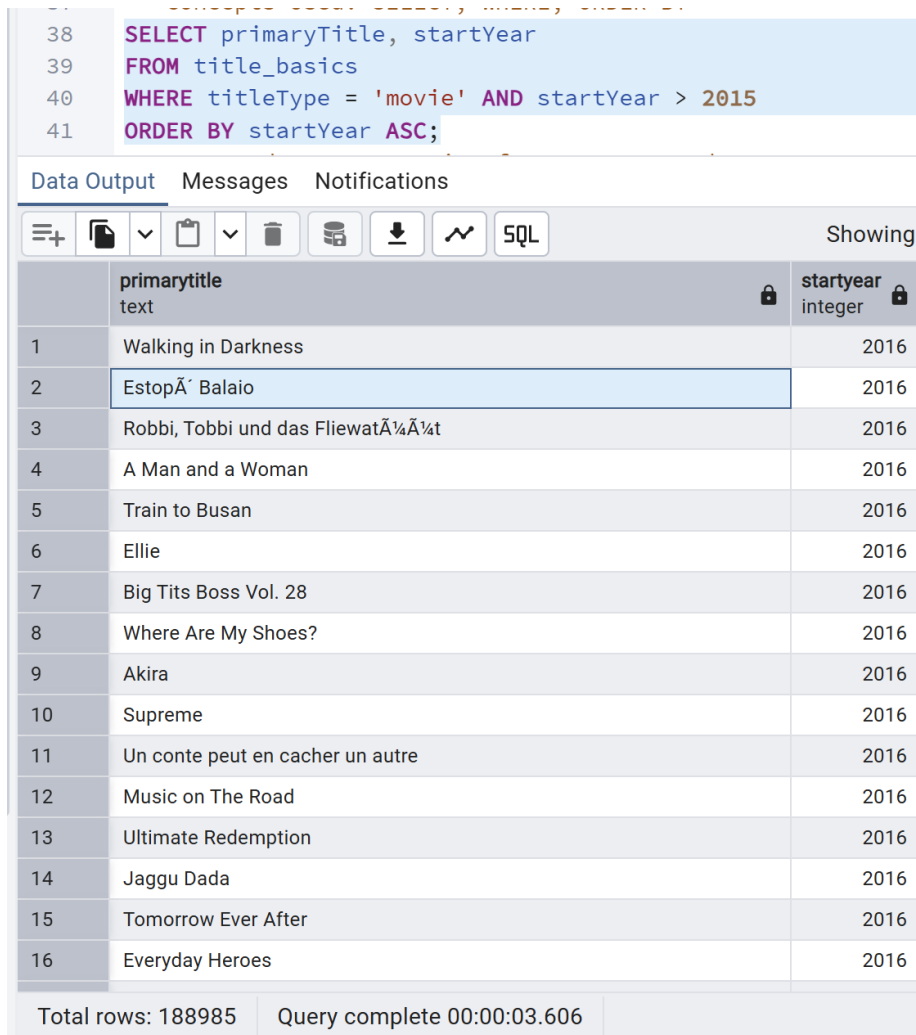
Data Output   Messages   Notifications

	primarytitle text	startyear integer	runtimeminutes integer
1	Miss Jerry	1894	45
2	The Corbett-Fitzsimmons Fight	1897	100
3	Bohemios	1905	100
4	Maniac	1934	51
5	The Story of the Kelly Gang	1906	70
6	The Prodigal Son	1907	90
7	Robbery Under Arms	1907	[null]
8	Hamlet	1908	[null]
9	Dva druga	1955	[null]
10	Don Quijote	1908	[null]

Business Insight:

- Allows analysts to zero in on core attributes without noise from ancillary fields.
- Helps streamline reporting and dashboard design.
- Immediately highlights missing runtimes or release years, which can distort trend analysis.

Query 3: WHERE Filtering - Movies released after 2015

Purpose: Identify recent movies

Concepts Used: SELECT, WHERE, ORDER BY

```
SELECT primaryTitle, startYear

FROM title_basics

WHERE titleType = 'movie' AND startYear > 2015

ORDER BY startYear ASC;
```

Expected Output: Movies from 2016 onwards

	primarytitle text	startyear integer
1	Walking in Darkness	2016
2	EstopÃ´ Balaio	2016
3	Robbi, Tobbi und das FliewatÃ¼Ã¼t	2016
4	A Man and a Woman	2016
5	Train to Busan	2016
6	Ellie	2016
7	Big Tits Boss Vol. 28	2016
8	Where Are My Shoes?	2016
9	Akira	2016
10	Supreme	2016
11	Un conte peut en cacher un autre	2016
12	Music on The Road	2016
13	Ultimate Redemption	2016
14	Jaggu Dada	2016
15	Tomorrow Ever After	2016
16	Everyday Heroes	2016

Total rows: 188985   Query complete 00:00:03.606

Business Insight:

- Establishes a cut of "modern" content for contemporary trend analysis.
- Useful in studying audience shifts, streaming behaviors, or franchise reboots post-2015.
- Helps business teams identify fresh IPs or competitive market entries.

Query 4: Multiple Conditions - Long action movies after 2010

Purpose: Identify action films with long runtimes

Concepts Used: SELECT, WHERE with multiple AND conditions, LIKE

```
SELECT primaryTitle, startYear, runtimeMinutes, genres
FROM title_basics
WHERE titleType = 'movie'
  AND startYear > 2010
  AND runtimeMinutes > 120
  AND genres LIKE '%Action%';
```

Expected Output: Action movies post-2010 with runtime > 120 min

```
51    SELECT primaryTitle, startYear, runtimeMinutes, genres
52    FROM title_basics
53    WHERE titleType = 'movie'
54      AND startYear > 2010
55      AND runtimeMinutes > 120
56      AND genres LIKE '%Action%';
```

Data Output   Messages   Notifications

Showing rows: 1 to 1000   Page No:

	primarytitle text	startyear integer	runtimeminutes integer	genres text
1	Aman Ke Farishtey	2016	137	Action
2	John Carter	2012	132	Action,Adventure,Sci-Fi
3	The Flash	2023	144	Action,Adventure,Fantasy
4	Shazam!	2019	132	Action,Adventure,Comedy
5	Red Tails	2012	125	Action,Drama,History
6	Man of Steel	2013	143	Action,Adventure,Sci-Fi
7	Godzilla	2014	123	Action,Adventure,Sci-Fi
8	Jaddi Sardar	2019	150	Action,Drama,Romance
9	Krrish 3	2013	152	Action,Adventure,Sci-Fi
10	Shershaah	2021	135	Action,Biography,Drama
11	Thrissur Pooram	2019	156	Action,Drama,Thriller
12	Hero	2019	121	Action,Crime,Drama
13	Finding 'Ohana	2021	123	Action,Adventure,Comedy
14	Sivappu Manjal Pachai	2019	140	Action,Drama,Thriller
15	Etharkkum Thunindhavan	2022	151	Action,Crime,Drama
16	The Dark Knight: The Ballad of the N Word	2018	129	Action,Comedy,Crime

Total rows: 3221   Query complete 00:00:02.250

Business Insight:

- Surfaces big-budget "blockbuster-style" films for performance tracking.
- Highlights where studios invest heavily in runtime and spectacle.
- Critical for analyzing ROI of high-investment productions.

Query 5: ORDER BY - Sort movies by release year

Purpose: Show the newest movies first

Concepts Used: SELECT, WHERE, ORDER BY DESC

```
SELECT primaryTitle, startYear

FROM title_basics

WHERE titleType = 'movie'

  AND startYear IS NOT NULL

ORDER BY startYear DESC

LIMIT 10;
```
Expected Output: Top 10 most recent movies

```
66    SELECT primaryTitle, startYear
67    FROM title_basics
68    WHERE titleType = 'movie'
69      AND startYear IS NOT NULL
70    ORDER BY startYear DESC
71    LIMIT 10;
```

Data Output   Messages   Notifications

	primarytitle text	startyear integer
1	Flipside	2032
2	Avatar 5	2031
3	Totally	2030
4	The End of Us	2030
5	Orion	2030
6	Legacy: The Forest of Death Part 3	2029
7	The Time Machine	2029
8	The Nightmares of Terrence Bergman: A Hollywood Story	2029
9	Avatar 4	2029
10	Coco 2	2029

Business Insight:

- Quick validation of dataset freshness.
- Helps content teams see which latest titles are tracked in IMDb.
- Useful for monitoring competitive pipelines and launch cycles.

Aggregation

Query 6: COUNT with GROUP BY - Movies per genre

Purpose: Count how many movies exist per genre

Concepts Used: COUNT, GROUP BY, ORDER BY

```sql
SELECT genres, COUNT(*) AS movie_count

FROM title_basics

WHERE titleType = 'movie'

AND genres IS NOT NULL

GROUP BY genres

ORDER BY movie_count DESC;
```

Expected Output: Genre vs total movies

Business Insight:

- Reveals dominant genres within the catalogue.
- Provides strategic signals on saturation vs opportunity for niche genres.
- Foundation metric for content portfolio benchmarking.

Query 7: SUM with GROUP BY - Total votes per genre

Purpose: Aggregate audience engagement by genre

Concepts Used: JOIN, SUM, GROUP BY

SELECT b.genres, SUM(r.numVotes) AS total_votes

FROM title_basics b

JOIN title_ratings r ON b.tconst = r.tconst

WHERE b.titleType = 'movie'

GROUP BY b.genres

ORDER BY total_votes DESC;

Expected Output: Genre vs total votes

```
100   SELECT b.genres, SUM(r.numVotes) AS total_votes
101   FROM title_basics b
102   JOIN title_ratings r ON b.tconst = r.tconst
103   WHERE b.titleType = 'movie'
104   GROUP BY b.genres
105   ORDER BY total_votes DESC;
```

Data Output   Messages   Notifications

	genres text	total_votes bigint
1	Action,Adventure,Sci-Fi	57225151
2	Drama	45522605
3	Comedy,Drama,Romance	38766940
4	Adventure,Animation,Comedy	37189762
5	Comedy	33635493
6	Action,Crime,Drama	33608882
7	Comedy,Drama	32802869
8	Action,Adventure,Fantasy	32284959
9	Drama,Romance	30339889
10	Action,Adventure,Comedy	28207256
11	Crime,Drama,Thriller	27788302
12	Comedy,Romance	24651097
13	Crime,Drama,Mystery	24155527
14	Action,Crime,Thriller	21887826
15	Crime,Drama	21571797
16	Action,Adventure,Drama	21184500

Total rows: 1268   Query complete 00:00:02.906

Business Insight:

- Captures audience demand and participation intensity per genre.
- Signals where audience loyalty and fan communities concentrate.
- Guides promotional spend and content acquisition decisions.

Query 8: AVG with GROUP BY - Average rating per genre

Purpose: Identify genre quality based on audience ratings

Concepts Used: JOIN, AVG, GROUP BY

```
SELECT b.genres, AVG(r.averageRating) AS avg_rating

FROM title_basics b

JOIN title_ratings r ON b.tconst = r.tconst

WHERE b.titleType = 'movie'

GROUP BY b.genres

ORDER BY avg_rating DESC;
```

Expected Output: Genre vs average rating

```
115    SELECT b.genres, AVG(r.averageRating) AS avg_rating
116    FROM title_basics b
117    JOIN title_ratings r ON b.tconst = r.tconst
118    WHERE b.titleType = 'movie'
119    GROUP BY b.genres
120    ORDER BY avg_rating DESC;
```

Data Output   Messages   Notifications

Showing r

	genres text	avg_rating double precision
1	Documentary,Fantasy,Thriller	9.7
2	Documentary,News,Reality-TV	9.3
3	Adult,Reality-TV	9.3
4	Documentary,Reality-TV,Romance	9.1
5	Comedy,Drama,Talk-Show	8.8
6	Documentary,Music,Sci-Fi	8.7
7	Documentary,Musical,Thriller	8.7
8	Animation,Music,Sci-Fi	8.7
9	Documentary,Drama,Game-Show	8.6
10	Adventure,Thriller,War	8.5
11	Adventure,Animation,Sport	8.4
12	Family,Music,Sport	8.4
13	Action,Biography,Fantasy	8.4
14	History,Mystery	8.399999999999999
15	Action,Documentary,Mystery	8.3
16	Biography,Comedy,War	8.3

Total rows: 1268   Query complete 00:00:02.699

Business Insight:

- Identifies genres that deliver quality consistently, not just quantity.
- Enables differentiation between commercially popular vs critically acclaimed segments.
- Helps prioritize prestige genres for awards campaigns or branding.

Query 9: HAVING - Genres with >500 movies

Purpose: Filter popular genres

Concepts Used: GROUP BY, HAVING, COUNT

```
SELECT genres, COUNT(*) AS total_movies

FROM title_basics

WHERE titleType = 'movie'

GROUP BY genres

HAVING COUNT(*) > 500

ORDER BY total_movies DESC;
```

Expected Output: Only genres with 500+ movies

```
130    SELECT genres, COUNT(*) AS total_movies
131    FROM title_basics
132    WHERE titleType = 'movie'
133    GROUP BY genres
134    HAVING COUNT(*) > 500
135    ORDER BY total_movies DESC;
```

Data Output   Messages   Notifications

	genres text	total_movies bigint
1	Drama	133241
2	Documentary	110580
3	[null]	77414
4	Comedy	49823
5	Horror	19982
6	Thriller	16988
7	Action	16516
8	Comedy,Drama	14525
9	Drama,Romance	13974
10	Romance	8363
11	Comedy,Romance	7892
12	Adult	6414
13	Crime,Drama	6247
14	Comedy,Drama,Romance	5937
15	Crime	5545
16	Drama,Thriller	5152

Total rows: 119   Query complete 00:00:01.824

Business Insight:

- Narrows focus to statistically significant genres.
- Prevents strategic bias from outlier categories with thin data.
- Ensures that market analysis is based on genres with real commercial weight.

Query 10: MAX/MIN with GROUP BY - Yearly extremes

Purpose: Find highest and lowest rated movies per year

Concepts Used: JOIN, GROUP BY, MAX, MIN

```sql
SELECT b.startYear,
       MAX(r.averageRating) AS highest_rating,
       MIN(r.averageRating) AS lowest_rating
FROM title_basics b
JOIN title_ratings r ON b.tconst = r.tconst
WHERE b.titleType = 'movie'
GROUP BY b.startYear
ORDER BY b.startYear DESC;
```

Expected Output: Year vs top & bottom movie ratings

```sql
145    SELECT b.startYear,
146          MAX(r.averageRating) AS highest_rating,
147          MIN(r.averageRating) AS lowest_rating
148    FROM title_basics b
149    JOIN title_ratings r ON b.tconst = r.tconst
150    WHERE b.titleType = 'movie'
151    AND StartYear is NOT NULL
152    GROUP BY b.startYear
153    ORDER BY b.startYear DESC;
```

Data Output | Messages | Notifications

	startyear integer	highest_rating double precision	lowest_rating double precision
1	2025	10	1
2	2024	9.9	1
3	2023	10	1
4	2022	10	1
5	2021	10	1
6	2020	9.8	1
7	2019	10	1
8	2018	9.8	1
9	2017	10	1
10	2016	9.7	1.1
11	2015	9.8	1
12	2014	9.8	1
13	2013	10	1
14	2012	9.8	1
15	2011	9.8	1

Total rows: 131   Query complete 00:00:02.817

Business Insight:

- Surfaces volatility in audience perception year over year.
- Can flag years with strong hits but also notable flops.
- Provides a balanced view of both upside and risk in content pipelines.

Multi-Table JOINs

Query 11: INNER JOIN - Movies with ratings

Purpose: Combine movie details with ratings

Concepts Used: INNER JOIN, ORDER BY

```
SELECT b.primaryTitle, r.averageRating, r.numVotes
FROM title_basics b
INNER JOIN title_ratings r ON b.tconst = r.tconst
WHERE b.titleType = 'movie'
ORDER BY r.averageRating DESC
LIMIT 10;
```

Expected Output: Top-rated movies with votes

```
165    SELECT b.primaryTitle, r.averageRating, r.numVotes
166    FROM title_basics b
167    INNER JOIN title_ratings r ON b.tconst = r.tconst
168    WHERE b.titleType = 'movie'
169    ORDER BY r.averageRating DESC
170    LIMIT 10;
```

Data Output   Messages   Notifications

	primarytitle text	averagerating double precision	numvotes integer
1	MamÃ¡ quiero ser futbolista profesional	10	5
2	Hua Se	10	9
3	Hard Heat	10	6
4	The Next Right Thing	10	5
5	Ivan Semyonov. Pervyy potseluy	10	7
6	King of Hearts - The Chris Barnard Story	10	8
7	I leoni della Folgore	10	9
8	Limpa ja unemaailm	10	6
9	Billy Knight	10	7
10	Ivan del mare	10	7

Business Insight:

- Links production data with performance feedback.
- Useful for quickly validating dataset joins and rating coverage.
- Powers "best of all time" reports for critical acclaim.

Query 12: LEFT JOIN - Movies including unrated

Purpose: Ensure all movies are shown, even unrated ones

Concepts Used: LEFT JOIN, ORDER BY NULLS LAST

```
SELECT b.primaryTitle, r.averageRating, r.numVotes
FROM title_basics b
LEFT JOIN title_ratings r ON b.tconst = r.tconst
WHERE b.titleType = 'movie'
ORDER BY r.averageRating NULLS LAST
LIMIT 10;
```

Expected Output: Movies with or without ratings

```
180   SELECT b.primaryTitle, r.averageRating, r.numVotes
181   FROM title_basics b
182   LEFT JOIN title_ratings r ON b.tconst = r.tconst
183   WHERE b.titleType = 'movie'
184   ORDER BY r.averageRating NULLS LAST
185   LIMIT 10;
```

Data Output   Messages   Notifications

	primarytitle text	averagerating double precision	numvotes integer
1	Our President's Money	1	12
2	Girls' Fright Out	1	5
3	Un momento en la vida del tiempo	1	8
4	Zirkus Renz	1	14
5	Stand Up Face the Fear	1	38
6	2025 - The World enslaved by a Virus	1	2813
7	Tavan	1	18
8	Belli, carucci e pettinati	1	12
9	Couch Destruction: Angel Release	1	18
10	Ghafel	1	13

Business Insight:

- Exposes content gaps where user engagement has not yet been captured.
- Critical for understanding early lifecycle of new titles.
- Prevents skewed reporting that omits unrated films.

Query 13: RIGHT JOIN - Ratings and corresponding movies

Purpose: Include all rating records, even if unmatched

Concepts Used: RIGHT JOIN, ORDER BY

```
SELECT b.primaryTitle, r.averageRating

FROM title_basics b

RIGHT JOIN title_ratings r ON b.tconst = r.tconst

ORDER BY r.averageRating DESC

LIMIT 10;
```

Expected Output: Ratings aligned with movie titles

```
195    SELECT b.primaryTitle, r.averageRating
196    FROM title_basics b
197    RIGHT JOIN title_ratings r ON b.tconst = r.tconst
198    ORDER BY r.averageRating DESC
199    LIMIT 10;
```

Data Output   Messages   Notifications

Showing rows: 1 to 10

	primarytitle text	averagerating double precision
1	Renegades 2	10
2	The Night Before the Morning After	10
3	The Concert	10
4	Planes, Trains, and Kitty Balloons	10
5	"Harry & Meghan" Pushes Race-Baiting Victimhood (Meghan Markle Netflix Series Review)	10
6	Episode #1.18	10
7	Ahamasmi Yodha	10
8	A Challenging Journey	10
9	Lakshya Ki Raah	10
10	Let's Objectify Some Women with Dayna Pereira	10

Business Insight:

- Validates referential integrity between ratings and titles.
- Ensures no orphan ratings exist without a movie reference.
- Strengthens trust in the dataset for downstream reporting.

Query 14: Multiple JOINs - Movie, rating, actor

Purpose: Link movies to actors alongside ratings

Concepts Used: Multiple JOINs (INNER), ORDER BY

```sql
SELECT b.primaryTitle, r.averageRating, n.primaryName AS actor

FROM title_basics b

JOIN title_ratings r ON b.tconst = r.tconst

JOIN title_principals p ON b.tconst = p.tconst

JOIN name_basics n ON p.nconst = n.nconst

WHERE b.titleType = 'movie'

  AND p.category = 'actor'

ORDER BY r.averageRating DESC

LIMIT 10;
```

Expected Output: Top-rated movies with actor names

```sql
111    SELECT b.primaryTitle, r.averageRating, n.primaryName AS actor
112    FROM title_basics b
113    JOIN title_ratings r ON b.tconst = r.tconst
114    JOIN title_principals p ON b.tconst = p.tconst
115    JOIN name_basics n ON p.nconst = n.nconst
116    WHERE b.titleType = 'movie'
117      AND p.category = 'actor'
118    ORDER BY r.averageRating DESC
119    LIMIT 10;
```

Data Output | Messages | Notifications

	primarytitle text	averagerating double precision	actor text
2	Limpa ja unemaailm	10	Hendrik Laanemaa
3	Limpa ja unemaailm	10	Hendrik Laanemaa
4	Limpa ja unemaailm	10	Hendrik Laanemaa
5	Limpa ja unemaailm	10	Joosep Toom
6	Limpa ja unemaailm	10	Joosep Toom
7	Limpa ja unemaailm	10	Matthias Loona
8	Limpa ja unemaailm	10	Uku Viispert
9	Limpa ja unemaailm	10	Saamuel Sasi
10	Ivan del mare	10	Ivan Matteo Pederbelli

Business Insight:

- Connects content quality with on-screen talent.
- Enables star-power analysis—actors correlated with high ratings.
- Valuable for casting, talent acquisition, and negotiation insights.

Nested Queries & CTEs

Query 15a: Subquery - Above average movies

Purpose: Find movies rated higher than global average

Concepts Used: Subquery, JOIN, AVG

```
SELECT primaryTitle, averageRating

FROM title_basics b

JOIN title_ratings r ON b.tconst = r.tconst

WHERE r.averageRating > (

    SELECT AVG(averageRating)

    FROM title_ratings

)

ORDER BY r.averageRating DESC

LIMIT 10;
```

Expected Output: List of standout movies above mean rating

```
1   SELECT primaryTitle, averageRating
2   FROM title_basics b
3   JOIN title_ratings r ON b.tconst = r.tconst
4   WHERE r.averageRating > (
5       SELECT AVG(averageRating)
6       FROM title_ratings
7   )
8   ORDER BY r.averageRating DESC
9   LIMIT 10;
```

Data Output | Messages | Notifications

	primarytitle text	averagerating double precision
1	The Unbroken Circle: A Tribute to Mother Maybelle Carter	10
2	Heroes	10
3	Heroes	10
4	Renegades 2	10
5	Innocent Bi-Standers	10
6	Doli: The Fragments of My Chilhood	10
7	Super Video Vixens 7: Christy Canyon	10
8	The Outlands	10
9	Lana & Theresa Having It Off	10
10	God Comes as a Child	10

Business Insight:

- Identifies "winners" that outperform the market baseline.
- Focuses attention on exceptional quality rather than volume.
- Supports awards targeting and critical brand positioning.

Query 15b: CTE - Top 5 movies per year

Purpose: Rank best movies yearly

Concepts Used: CTE, ROW_NUMBER, PARTITION BY

```
WITH yearly_ranked AS (

    SELECT b.startYear, b.primaryTitle, r.averageRating,

           ROW_NUMBER() OVER (PARTITION BY b.startYear ORDER BY r.averageRating
DESC) AS rank_in_year

    FROM title_basics b

    JOIN title_ratings r ON b.tconst = r.tconst

    WHERE b.titleType = 'movie'

)

SELECT * FROM yearly_ranked

WHERE rank_in_year <= 5

ORDER BY startYear DESC, averageRating DESC;
```

Expected Output: Top 5 movies each year

```
1    WITH yearly_ranked AS (
2        SELECT b.startYear, b.primaryTitle, r.averageRating,
3            ROW_NUMBER() OVER (PARTITION BY b.startYear ORDER BY r.averageRating DESC) AS rank_in_year
4        FROM title_basics b
5        JOIN title_ratings r ON b.tconst = r.tconst
6        WHERE b.titleType = 'movie'
7    )
8    SELECT * FROM yearly_ranked
9    WHERE rank_in_year <= 5
10   AND StartYear is NOT NULL
11   ORDER BY startYear DESC, averageRating DESC;
```

Data Output   Messages   Notifications

Showing rows: 1 to 630   Page No: 1   of 1

startyear integer	primarytitle text	averagerating double precision	rank_in_year bigint
2025	Revelations	10	1
2025	Ivan Semyonov. Pervyy potseluy	10	5
2025	Billy Knight	10	4
2025	Secrets	10	3
2025	The Next Right Thing	10	2
2024	Dawn Dusk	9.9	1
2024	Survivors Speak: Domestic Abuse and the Legal System	9.8	5
2024	Blackwoods	9.8	4
2024	Indie Deal	9.8	3
2024	TÃ³xico Narcisista	9.8	2
2023	Limpa ja unemaailm	10	1
2023	Magnum 87	9.8	5

Total rows: 630   Query complete 00:00:02.891

Business Insight:

- Structures data into yearly competitive landscapes.
- Useful for understanding how quality distribution evolves annually.
- Supports investor decks and strategic planning by spotlighting top content leaders.

Integration

Query 16: Complex yearly report

Purpose: Generate annual stats with top genres

Concepts Used: CTEs, Aggregation, RANK, JOIN

```
WITH yearly_stats AS (

    SELECT b.startYear,

            COUNT(*) AS total_movies,

            AVG(r.averageRating) AS avg_rating

    FROM title_basics b

    JOIN title_ratings r ON b.tconst = r.tconst

    WHERE b.titleType = 'movie'

    GROUP BY b.startYear

),

genre_stats AS (

    SELECT b.startYear, b.genres, AVG(r.averageRating) AS genre_avg_rating,

            RANK() OVER (PARTITION BY b.startYear ORDER BY AVG(r.averageRating)
DESC) AS genre_rank

    FROM title_basics b

    JOIN title_ratings r ON b.tconst = r.tconst

    WHERE b.titleType = 'movie'

    GROUP BY b.startYear, b.genres

)

SELECT y.startYear, y.total_movies, y.avg_rating, g.genres AS top_genre,
g.genre_avg_rating

FROM yearly_stats y

JOIN genre_stats g ON y.startYear = g.startYear

WHERE g.genre_rank = 1

ORDER BY y.startYear DESC;
```

Expected Output: Yearly movie counts, avg rating, and best genre

	startyear integer	total_movies bigint	avg_rating double precision	top_genre text	genre_avg_rating double precision
1	2025	4659	6.744644773556561	Documentary,Drama,Sport	9.7
2	2025	4659	6.744644773556561	Documentary,News	9.7
3	2024	10651	6.470603699183172	History,Romance	9.8
4	2023	11387	6.344146834109078	Comedy,Romance,Thriller	9.6
5	2023	11387	6.344146834109078	Drama,Music,Sci-Fi	9.6
6	2022	11348	6.3203295734931215	Documentary,News,Reality-TV	9.7
7	2021	9667	6.21358229026584	Adventure,Comedy,Music	9.8
8	2020	8973	6.141223671013037	Biography,Documentary,Reality-TV	9.6
9	2019	10989	6.16542906542907	Adult,Thriller	9.1
10	2018	10895	6.15836622303808	Adventure,Music	9.7
11	2017	10720	6.2369402985074665	Romance,Thriller,War	9.4
12	2016	10186	6.250107991360689	Crime,Documentary,News	9
13	2015	9739	6.229592360611965	Comedy,Documentary,Fantasy	9.3
14	2014	9497	6.269537748762773	Adult,Reality-TV	9.3
15	2013	8809	6.245385401294127	Documentary,Horror,Mystery	9.3
16	2013	8809	6.245385401294127	Action,History	9.3
17	2012	8422	6.268606031821418	Adventure,Biography,Crime	9
18	2011	7989	6.267668043559901	Family,Musical	8.9
19	2010	7383	6.229351212244345	Biography,Comedy,Romance	8.9
20	2010	7383	6.229351212244345	Fantasy,Mystery,Thriller	8.9
21	2009	7120	6.2438061797752855	Comedy,Documentary,Sport	9
22	2008	6373	6.225090224384126	Drama,History,Music	9

Total rows: 146   Query complete 00:00:05.693

- **Business Insight:**
- Consolidates multiple KPIs (volume, quality, genre leadership) into one annual report.
- Supports long-term strategic trend analysis across genres and years.
- Valuable for greenlighting, investment roadmaps, and forecasting content direction.