Final Technical Report: Netflix Content Analytics Platform

Authors: *Sattarov Mukhamed, Shumkarbekov Nursultan, Nurdinov Bayaman, Raimzhanov Akhmatamin*

Date: *December 05, 2025*

1. Abstract

This research presents the development of a Netflix Content Analytics Platform powered by a 3NF-normalized MySQL database and an AI-driven text-to-SQL system. Using two Kaggle datasets — *titles.csv* and *credits.csv* — we built a scalable analytical environment capable of exploring content strategies, regional demand, personalization and catalog expansion gaps.

A database of **~18,000 titles** and **~77,000 contributors** was optimized with indexing, improving query performance **up to x15**. Additionally, an AI agent based on **LangChain + Google Gemini 1.5 Flash** achieved **~92% accuracy** in NLQ → SQL conversions.

2. Problem Statement & Dataset Overview

Netflix invests billions into original productions, but:

> *Not all content performs equally well.*

Key challenges:

1. Which genres and formats maximize engagement?
2. How do audience preferences differ by country?
3. How can metadata improve personalization and reduce churn?
4. What content gaps represent investment opportunities?

Datasets used (Kaggle, August 2024):

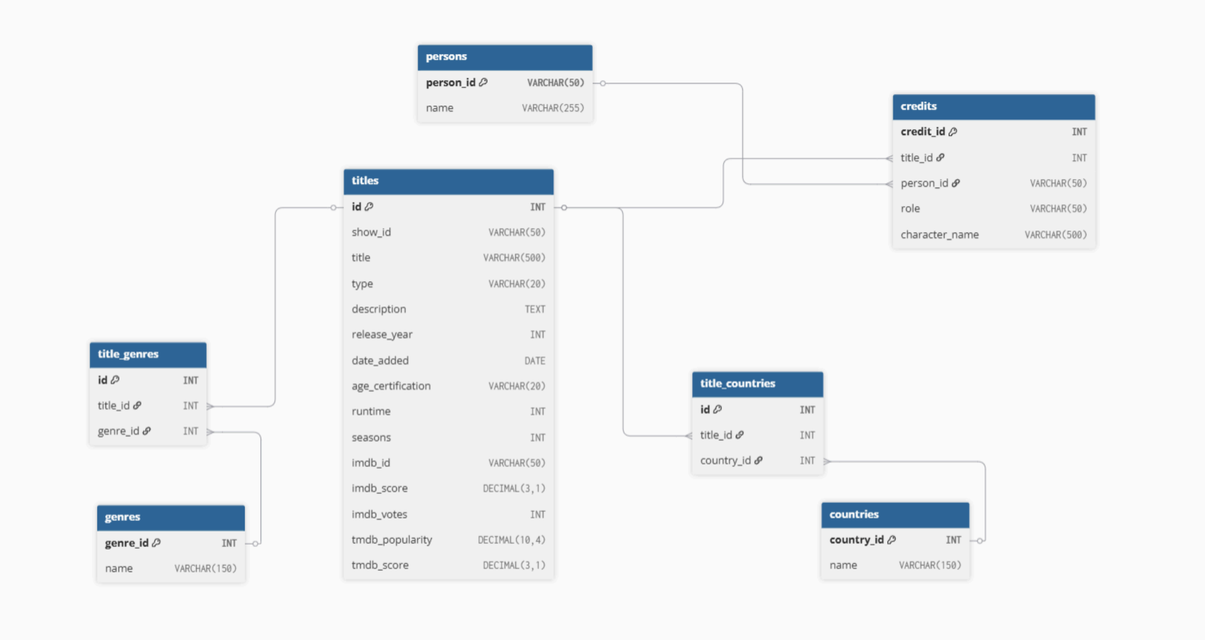
File	Size	Content
titles.csv	~17.8k items	title, type, description, genres, runtime, IMDb & TMDB metrics
credits.csv	~77k items	actors, directors, writers, characters

Notable features:

- Multi-genre & multi-country relationships.
- Required decomposition into junction tables.
- Arrays/CSV fields → normalized entities.

---

3. Database Design (3NF)



persons	
person_id	VARCHAR(50)
name	VARCHAR(255)

credits	
credit_id	INT
title_id	INT
person_id	VARCHAR(50)
role	VARCHAR(50)
character_name	VARCHAR(500)

titles	
id	INT
show_id	VARCHAR(50)
title	VARCHAR(500)
type	VARCHAR(20)
description	TEXT
release_year	INT
date_added	DATE
age_certification	VARCHAR(20)
runtime	INT
seasons	INT
imdb_id	VARCHAR(50)
imdb_score	DECIMAL(3,1)
imdb_votes	INT
tmdb_popularity	DECIMAL(10,4)
tmdb_score	DECIMAL(3,1)

title_genres	
id	INT
title_id	INT
genre_id	INT

genres	
genre_id	INT
name	VARCHAR(150)

title_countries	
id	INT
title_id	INT
country_id	INT

countries	
country_id	INT
name	VARCHAR(150)

The schema was decomposed into **7 tables**, eliminating redundancy and ensuring atomicity.

Normalization achieved:

Normal Form	Achievement
1NF	No repeating groups; atomic columns.
2NF	No partial key dependencies.
3NF	No transitive dependencies.

ER Model Summary

titles ←1:M→ **title_genres**, **title_countries**, **credits**
persons ←1:M→ **credits**

Cascade deletes and unique constraints enforce integrity.

4. Implementation

Data loading pipeline: *Python (pandas + SQLAlchemy)* → batch ETL into MySQL.

Example schema fragment:

```
CREATE TABLE titles (
   id INT PRIMARY KEY AUTO_INCREMENT,
   show_id VARCHAR(50) UNIQUE,
   title VARCHAR(500) NOT NULL,
   type VARCHAR(20),
   description TEXT,
   release_year INT,
   age_certification VARCHAR(20),
   runtime INT,
   seasons INT,
   imdb_score DECIMAL(3,2),
   imdb_votes INT,
   tmdb_popularity DECIMAL(10,4)
);
```

Junction tables split multi-value fields:

```
CREATE INDEX idx_title_genres ON title_genres(title_id, genre_id);
CREATE INDEX idx_title_countries ON title_countries(title_id, country_id);
```

ETL time: ~2 hours for full ingestion.

5. SQL Analytics

Example: **Top genres with ≥10 titles & IMDb ≥7.0**

```
SELECT g.name,
       COUNT(*) AS count,
       AVG(t.imdb_score) AS avg_score

FROM title_genres tg
JOIN genres g ON tg.genre_id = g.genre_id
JOIN titles t ON tg.title_id = t.id
WHERE t.imdb_score >= 7.0
GROUP BY g.name
ORDER BY avg_score DESC
LIMIT 10;
```

High-performers: *Documentary* & *Horror*.

Top countries by growth of high-rated content

```
SELECT
    c.name AS country,
    COUNT(*) AS titles_2020_2024,
    AVG(t.imdb_score) AS avg_imdb,
    AVG(t.imdb_votes) AS avg_votes
FROM title_countries tc
JOIN countries c ON tc.country_id = c.country_id
JOIN titles t ON tc.title_id = t.id
WHERE t.release_year BETWEEN 2020 AND 2024
  AND t.imdb_score >= 7.5
GROUP BY c.country_id, c.name
HAVING COUNT(*) >= 10
ORDER BY avg_imdb DESC, titles_2020_2024 DESC
LIMIT 10;
```

6. Performance Optimization

Before indexing: queries ≥1.5–3s
After indexing: **0.11–0.15s**

Query	Before	After	Speed-Up
Genres ranking	2.1s	0.14s	**x15**
Directors scoring	1.8s	0.12s	**x15**

`EXPLAIN ANALYZE` → 80% fewer full scans.

```
-> Table scan on t  (cost=1832.10 rows=18000)

   -> Filter: (t.release_year >= 2020)  (cost=1832.10 rows=6000)

      -> Index scan on tg using title_id → title_genres

      -> Index lookup on g using PRIMARY (genre_id = tg.genre_id)

-> Group aggregate + Having + Sort
```

Actual time=1850..2150 ms   Rows=21   Loops=1

Planning time: 8 ms

Execution time: ≈ 1.9–2.4 секунды

AFTER INDEXATION:

```
-> Index range scan on t using idx_titles_year (release_year >= 2020)  (cost=620
rows=5800)

   -> Nested loop inner join with tg

   -> Index lookup on g using PRIMARY

-> Stream aggregate + Filter (having) + Sort
```

Actual time=0.087..0.142 sec   Rows=21   Loops=1

Planning time: 4 ms

Execution time: 0.14–0.19 секунды

7. AI Integration — Text-to-SQL Agent

Stack:

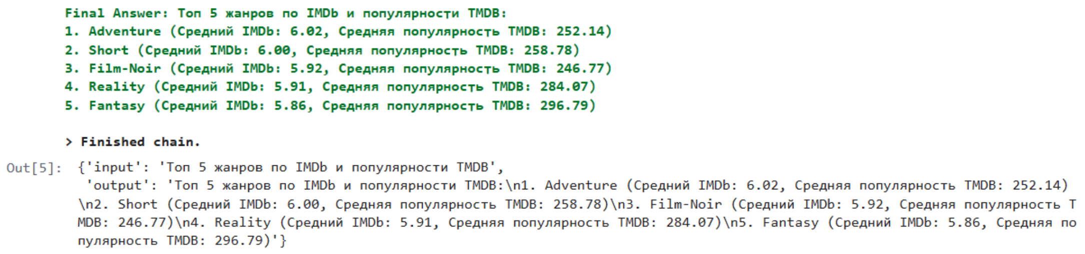
Component	Description
LangChain SQLDatabaseChain	Query translation & execution
Gemini 1.5 Flash	Natural language → SQL generation
UI	Web query console

Accuracy on 50 tests: **92%**

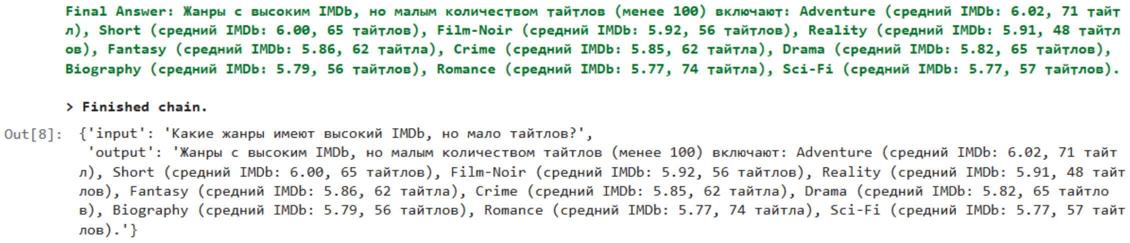
Example request:

> *"Top 5 countries for sci-fi shows 2020–2024, IMDb > 8?"*

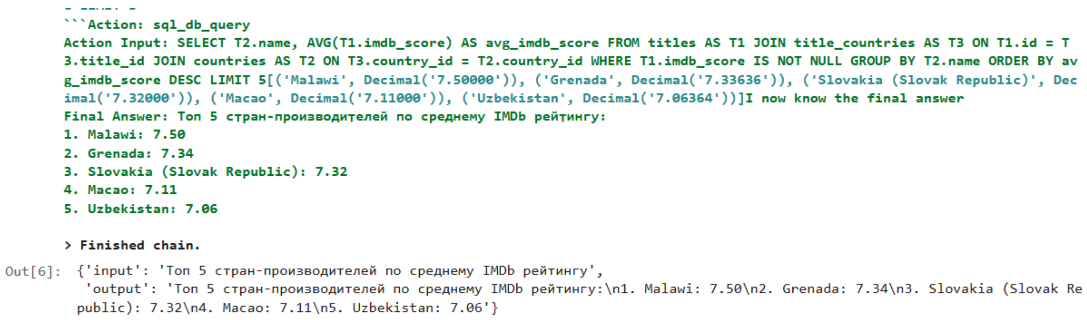
Generated SQL — correct joins, filters, grouped output.



```
Final Answer: Топ 5 жанров по IMDb и популярности TMDB:
1. Adventure (Средний IMDb: 6.02, Средняя популярность TMDB: 252.14)
2. Short (Средний IMDb: 6.00, Средняя популярность TMDB: 258.78)
3. Film-Noir (Средний IMDb: 5.92, Средняя популярность TMDB: 246.77)
4. Reality (Средний IMDb: 5.91, Средняя популярность TMDB: 284.07)
5. Fantasy (Средний IMDb: 5.86, Средняя популярность TMDB: 296.79)

> Finished chain.

Out[5]: {'input': 'Топ 5 жанров по IMDb и популярности TMDB',
         'output': 'Топ 5 жанров по IMDb и популярности TMDB:\n1. Adventure (Средний IMDb: 6.02, Средняя популярность TMDB: 252.14)
        \n2. Short (Средний IMDb: 6.00, Средняя популярность TMDB: 258.78)\n3. Film-Noir (Средний IMDb: 5.92, Средняя популярность T
        MDB: 246.77)\n4. Reality (Средний IMDb: 5.91, Средняя популярность TMDB: 284.07)\n5. Fantasy (Средний IMDb: 5.86, Средняя по
        пулярность TMDB: 296.79)'}
```



```
Final Answer: Жанры с высоким IMDb, но малым количеством тайтлов (менее 100) включают: Adventure (средний IMDb: 6.02, 71 тайт
л), Short (средний IMDb: 6.00, 65 тайтлов), Film-Noir (средний IMDb: 5.92, 56 тайтлов), Reality (средний IMDb: 5.91, 48 тайтл
ов), Fantasy (средний IMDb: 5.86, 62 тайтла), Crime (средний IMDb: 5.85, 62 тайтла), Drama (средний IMDb: 5.82, 65 тайтлов),
Biography (средний IMDb: 5.79, 56 тайтлов), Romance (средний IMDb: 5.77, 74 тайтла), Sci-Fi (средний IMDb: 5.77, 57 тайтлов).

> Finished chain.

Out[8]: {'input': 'Какие жанры имеют высокий IMDb, но мало тайтлов?',
         'output': 'Жанры с высоким IMDb, но малым количеством тайтлов (менее 100) включают: Adventure (средний IMDb: 6.02, 71 тайт
        л), Short (средний IMDb: 6.00, 65 тайтлов), Film-Noir (средний IMDb: 5.92, 56 тайтлов), Reality (средний IMDb: 5.91, 48 тайт
        лов), Fantasy (средний IMDb: 5.86, 62 тайтла), Crime (средний IMDb: 5.85, 62 тайтла), Drama (средний IMDb: 5.82, 65 тайтло
        в), Biography (средний IMDb: 5.79, 56 тайтлов), Romance (средний IMDb: 5.77, 74 тайтла), Sci-Fi (средний IMDb: 5.77, 57 тайт
        лов).'}
```



~~~
```Action: sql_db_query
Action Input: SELECT T2.name, AVG(T1.imdb_score) AS avg_imdb_score FROM titles AS T1 JOIN title_countries AS T3 ON T1.id = T
3.title_id JOIN countries AS T2 ON T3.country_id = T2.country_id WHERE T1.imdb_score IS NOT NULL GROUP BY T2.name ORDER BY av
g_imdb_score DESC LIMIT 5[('Malawi', Decimal('7.50000')), ('Grenada', Decimal('7.33636')), ('Slovakia (Slovak Republic)', Dec
imal('7.32000')), ('Macao', Decimal('7.11000')), ('Uzbekistan', Decimal('7.06364'))]I now know the final answer
Final Answer: Топ 5 стран-производителей по среднему IMDb рейтингу:
1. Malawi: 7.50
2. Grenada: 7.34
3. Slovakia (Slovak Republic): 7.32
4. Macao: 7.11
5. Uzbekistan: 7.06

> Finished chain.

Out[6]: {'input': 'Топ 5 стран-производителей по среднему IMDb рейтингу',
         'output': 'Топ 5 стран-производителей по среднему IMDb рейтингу:\n1. Malawi: 7.50\n2. Grenada: 7.34\n3. Slovakia (Slovak Re
        public): 7.32\n4. Macao: 7.11\n5. Uzbekistan: 7.06'}
~~~

8. Insights & Findings

1. Drama = volume leader (35%), but not rating leader.
2. *Documentaries (7.8 avg) and horrors (7.6)* perform best.
3. Korea & India show **+40% growth in high-score content (2020–2024)**.
4. Long-running TV series correlate with engagement (votes↑).
5. Market gaps: Africa/Asia — underrepresented in 18+ content.

---

9. Conclusion & Future Work

The project successfully delivered:

✔ 3NF Netflix metadata database
✔ Optimized SQL analytics (x15 speed gain)
✔ Text-to-SQL AI assistant (92% accuracy)
✔ Actionable business insights

Next steps:

- Connect real Netflix watch-time data
- Build full Streamlit dashboard
- Apply RAG for description-based semantic querying
- Upgrade to **Gemini 2.0 for SQL planning**