STUDIO SHODWE

CINEMA *Movie*

2025 MOVIE PRESENTATION

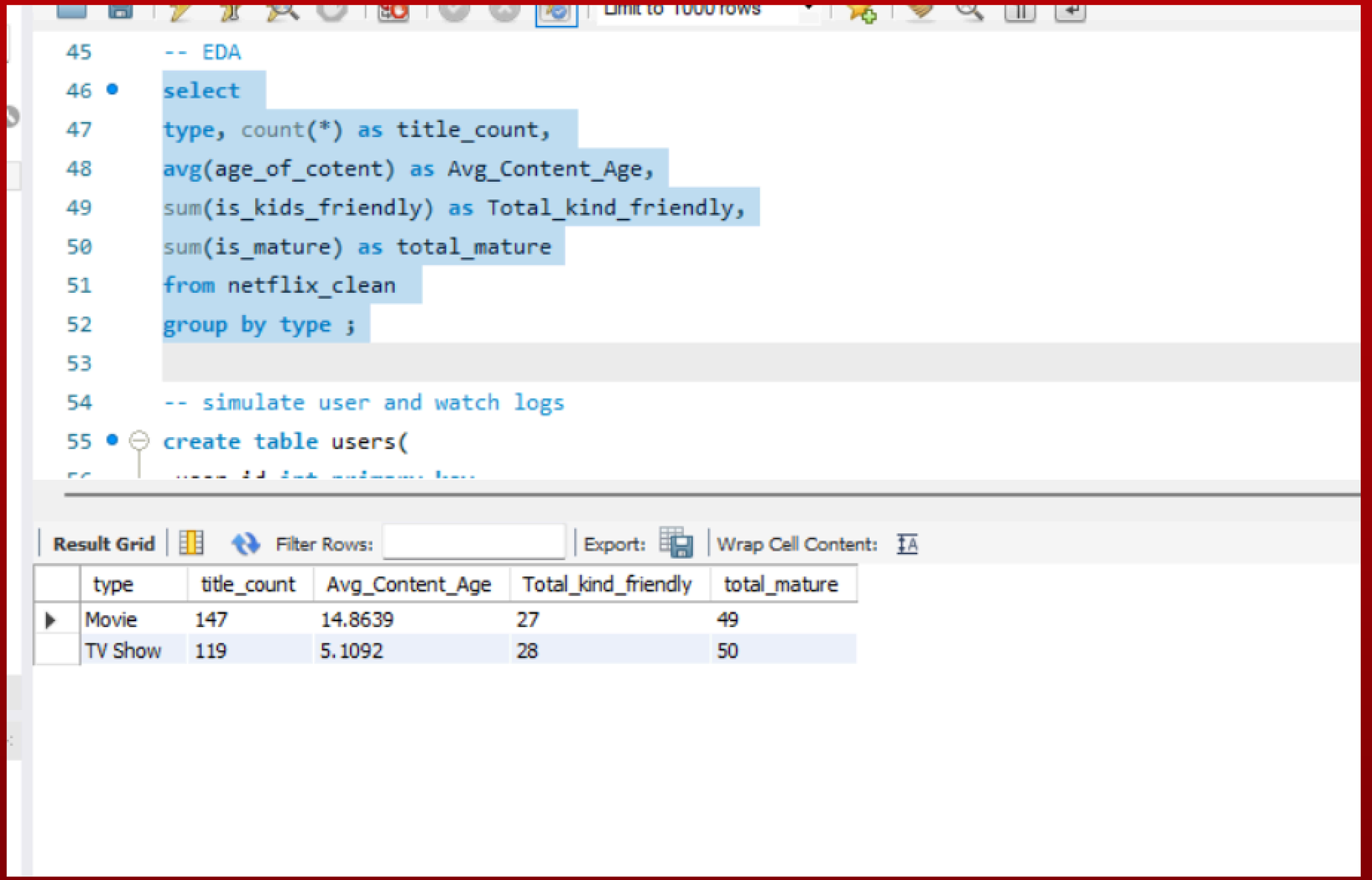
NEXT SLIDE

```sql
-- cleaning country column (replacing the null value with 'Not Gien')
select coalesce(country, 'Not Given') as country
from netflix1;

-- features engineering and cleaning in the views
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

country
United States
France
United States
Brazil
United States
United Kingdom
United States
India
Germany
India
India
India
Pakistan
United States
United States
United States
United States
Pakistan
Pakistan
United States
United States
United States


45 -- EDA
46 select
47 type, count(*) as title_count,
48 avg(age_of_cotent) as Avg_Content_Age,
49 sum(is_kids_friendly) as Total_kind_friendly,
50 sum(is_mature) as total_mature
51 from netflix_clean
52 group by type ;
53
54 -- simulate user and watch logs
55 create table users(
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
type
title_count
Avg_Content_Age
Total_kind_friendly
total_mature
Movie
147
14.8639
27
49
TV Show
119
5.1092
28
50

```
92
93 •  select * from watch_log;
94
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

user_id	show_id	watch_min	linked	watch_date
266	s1	53	0	2024-02-15
265	s1	115	1	2024-02-08
259	s1	64	1	2024-03-24
256	s1	99	1	2024-01-12
253	s1	65	1	2024-03-03
251	s1	96	1	2024-03-10
236	s1	36	1	2024-01-04
235	s1	34	1	2024-02-13
232	s1	34	0	2024-03-11
230	s1	80	1	2024-02-23
226	s1	86	1	2024-03-06
221	s1	89	1	2024-02-29
218	s1	113	1	2024-02-14
216	s1	76	1	2024-03-18
211	s1	33	1	2024-01-04
207	s1	55	1	2024-03-06
193	s1	92	1	2024-02-07
192	s1	45	1	2024-02-16
189	s1	97	1	2024-03-21
188	s1	100	1	2024-03-16

```sql
select thumbnail_show,
count(*) as views , round(avg(watch_min),2) as avg_watch_min,
round(stddev(watch_min),2) as sd_watch_min
from ab_test
group by thumbnail_show;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

thumbnail_show	views	avg_watch_min	sd_watch_min
generic_thumb	16724	74.31	26
mature_thumb	3622	74.92	26.15

```sql
-- most & least watched titles
SELECT c.title,
       c.type,
       ROUND(SUM(w.watch_min),2) AS total_watch_min,
       COUNT(*)                  AS views,
       ROUND(AVG(w.watch_min),2) AS avg_watch_min
FROM netflix_clean c
JOIN watch_log w ON c.show_id = w.show_id
GROUP BY c.title, c.type
ORDER BY total_watch_min DESC
LIMIT 15;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

title	type	total_watch_min	views	avg_watch_min
Angamaly Diaries	Movie	22071	298	74.06
Omo Ghetto: the Saga	Movie	21474	298	72.06
Poseidon	Movie	15682	206	76.13
Jaws	Movie	15670	206	76.07
Birth of the Dragon	Movie	15646	206	75.95
Clear and Present Danger	Movie	15472	206	75.11
The Stronghold	Movie	15396	206	74.74
Jaws 2	Movie	15276	202	75.62
Jaws: The Revenge	Movie	15150	200	75.75
Post Mortem: No One Dies in Skarnes	TV Show	15122	196	77.15
Love in a Puff	Movie	15105	196	77.07

```sql
-- bottom 10
SELECT c.title,
       c.type,
       ROUND(SUM(w.watch_min), 2) AS total_watch_min,
       COUNT(*)                   AS views,
       ROUND(AVG(w.watch_min), 2) AS avg_watch_min
FROM netflix_clean c
JOIN watch_log w ON c.show_id = w.show_id
GROUP BY c.title, c.type
ORDER BY total_watch_min ASC
LIMIT 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

title	type	total_watch_min	views	avg_watch_min
The Nutty Professor	Movie	6270	92	68.15
The Smart Money Woman	TV Show	6374	92	69.28
Show Dogs	Movie	6415	92	69.73
Dancing Angels	TV Show	6481	92	70.45
Hotel Del Luna	TV Show	6492	92	70.57
Motu Patlu in the Game of Zones	Movie	6526	92	70.93

```sql
--  USER ENGAGEMENT BY SEGMENT
-- average watch time per age bucket
SELECT CASE WHEN age < 25 THEN '18-24'
            WHEN age < 35 THEN '25-34'
            WHEN age < 45 THEN '35-44'
            ELSE '45+' END AS age_bucket,
       ROUND(AVG(watch_min),2) AS avg_watch_min,
       COUNT(*)                AS total_views
FROM users u
JOIN watch_log w ON u.user_id = w.user_id
GROUP BY age_bucket
ORDER BY age_bucket;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

age_bucket	avg_watch_min	total_views
18-24	73.55	3182
25-34	75.43	4030
35-44	74.03	3674
45+	74.43	9460

```sql
-- GENRE POPULARITY  (simple LIKE filter)
SELECT g.genre,
       COUNT(*) AS watch_count,
       ROUND(AVG(w.watch_min),2) AS avg_watch_min
FROM (
      SELECT 'COMEDIES' AS genre UNION ALL
      SELECT 'DRAMAS' UNION ALL
      SELECT 'ACTION' UNION ALL
      SELECT 'HORROR' UNION ALL
      SELECT 'DOCUMENTARIES'
) g
JOIN netflix_clean c ON INSTR(UPPER(c.listed_in), g.genre) > 0
JOIN watch_log w ON c.show_id = w.show_id
GROUP BY g.genre
ORDER BY watch_count DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

genre	watch_count	avg_watch_min
COMEDIES	8606	74.39
DRAMAS	8460	74.73
ACTION	6344	74.50
DOCUMENTARIES	2584	73.10
HORROR	902	74.34

```sql
-- DIRECTOR & COUNTRY INSIGHTS
-- top directors by total watch time
SELECT director,
       ROUND(SUM(w.watch_min),2) AS total_watch_min,
       COUNT(*)                  AS views
FROM netflix_clean c
JOIN watch_log w ON c.show_id = w.show_id
WHERE director <> 'Not Given'
GROUP BY director
ORDER BY total_watch_min DESC
LIMIT 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

director	total_watch_min	views
Toshiya Shinohara	30712	408
Delhiprasad Deenadayalan	28688	392
Masahiko Murata	23015	306
Antoine Fuqua	22638	310
Wolfgang Petersen	22589	308
Lijo Jose Pellissery	22071	298
JJC Skillz, Funke Akindele	21474	298
Suhas Kadav	20215	276
Chapman Way, Maclain Way	20088	272
Lasse HallstrÃ¶m	15962	208

```sql
-- top countries
SELECT country,
       COUNT(*) AS titles,
       ROUND(AVG(age_of_cotent),1) AS avg_age
FROM netflix_clean
WHERE country <> 'Not Given'
GROUP BY country
ORDER BY titles DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

country	titles	avg_age
Pakistan	124	5.1
United States	68	18.2
India	12	13.5
Japan	12	19.2
United Kingdom	7	22.3
France	4	8.3
Germany	4	11.5
South Africa	3	9.3
China	2	14.0
Nigeria	2	6.0
Brazil	1	4.0
Spain	1	6.0
Philippines	1	5.0
Australia	1	24.0
Argentina	1	11.0
Canada	1	7.0

```sql
201
202     -- churn rate
203 •   SELECT status, COUNT(*) AS users
204     FROM user_stats
205     GROUP BY status;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	status	users
▶	Active	266

```sql
-- TIME-BASED TRENDS
-- watch activity by calendar month
SELECT MONTH(watch_date) AS month_num,
       MONTHNAME(watch_date) AS month_name,
       COUNT(*) AS views,
       ROUND(AVG(watch_min),2) AS avg_watch_min
FROM watch_log
GROUP BY month_num, month_name
ORDER BY month_num;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

month_num	month_name	views	avg_watch_min
1	January	7028	74.20
2	February	6536	74.42
3	March	6782	74.64

```sql
-- day-of-week pattern
SELECT DAYNAME(watch_date) AS day_name,
       COUNT(*) AS views,
       ROUND(AVG(watch_min),2) AS avg_watch_min
FROM watch_log
GROUP BY day_name
ORDER BY FIELD(day_name,'Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday');
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

day_name	views	avg_watch_min
Monday	2884	74.00
Tuesday	2985	74.79
Wednesday	2965	74.97
Thursday	2993	73.50
Friday	2908	74.99
Saturday	2921	73.89
Sunday	2690	74.82

```sql
-- checking the user, title pair exists
SELECT COUNT(*) AS total_pairs
FROM users u
JOIN netflix_clean c ON INSTR(UPPER(c.listed_in), UPPER(u.genre_pref)) > 0;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_pairs
10173

```
244     -- checking watchlog existing
245 •   SELECT COUNT(*) AS total_watches FROM watch_log;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_watches
20346

THANK YOU!

and See You Next!

2025 MOVIE PRESENTATION

NEXT SLIDE