SQL Queries:

1)
```
SELECT
  Title,
  COUNT(DISTINCT Week) AS weeks_in_top10
FROM netflixglobal
GROUP BY Title
ORDER BY weeks_in_top10 DESC;
```

2)
```
SELECT
  yt.Title,
  CAST(yt."View Count" AS BIGINT) AS views,
  CAST(yt."Like Count" AS BIGINT) AS likes,
  COUNT(DISTINCT nr.Week) AS weeks_in_top10,
  SUM(CAST(REPLACE(REPLACE(nr."Hours Viewed", '"', ''), ',', '') AS BIGINT)) AS
total_hours
FROM youtube yt
JOIN netflixglobal nr
  ON TRIM(yt.Title) = TRIM(nr.Title)
GROUP BY
  yt.Title,
  yt."View Count",
  yt."Like Count"
ORDER BY
  weeks_in_top10 DESC;
```

3)
```
SELECT
  r.Region,
  i.Genre,
  COUNT(DISTINCT r.Week) AS total_weeks_in_top10
FROM netflixregion r
JOIN imdb i
  ON TRIM(LOWER(r.Title)) = TRIM(LOWER(REPLACE(i.Title, '"', '')))
GROUP BY r.Region, i.Genre
ORDER BY r.Region, total_weeks_in_top10 DESC;
```

4)
```
-- Shows with drop from week 1 to week 2
WITH ranked_weeks AS (
  SELECT
    Title,
    Week,
    RANK() OVER (PARTITION BY Title ORDER BY Week ASC) AS week_number,
```

```sql
    CAST(Rank AS INT) AS rank_position
  FROM netflixregion
),
week_change AS (
  SELECT
    Title,
    MAX(CASE WHEN week_number = 1 THEN rank_position END) AS week1_rank,
    MAX(CASE WHEN week_number = 2 THEN rank_position END) AS week2_rank
  FROM ranked_weeks
  GROUP BY Title
)
SELECT
  Title,
  week1_rank,
  week2_rank,
  week2_rank - week1_rank AS rank_drop
FROM week_change
WHERE week2_rank IS NOT NULL
ORDER BY rank_drop DESC;
```

5)

```sql
WITH buzz AS (
  SELECT
    LOWER(TRIM(title)) AS clean_title,
    MAX(CAST(REPLACE("View Count", '"', '') AS BIGINT)) AS trailer_views
  FROM youtube
  GROUP BY LOWER(TRIM(title))
),

interest AS (
  SELECT
    LOWER(REPLACE(TRIM(title), '"', '')) AS clean_title,
    MAX(CAST("Interest Score" AS INTEGER)) AS search_interest
  FROM googletrends
  GROUP BY LOWER(REPLACE(TRIM(title), '"', ''))
),

watchtime AS (
  SELECT
    LOWER(TRIM(title)) AS clean_title,
    SUM(
      CAST(REPLACE(REPLACE("Hours Viewed", '"', ''), ',', '') AS BIGINT)
    ) AS total_watchtime
  FROM netflixglobal
  GROUP BY LOWER(TRIM(title))
)

SELECT
  COALESCE(b.clean_title, i.clean_title, w.clean_title) AS title,
  COALESCE(b.trailer_views, 0) AS trailer_views,
  COALESCE(i.search_interest, 0) AS search_interest,
  COALESCE(w.total_watchtime, 0) AS total_watchtime
FROM buzz b
LEFT JOIN interest i ON b.clean_title = i.clean_title
LEFT JOIN watchtime w ON b.clean_title = w.clean_title
WHERE COALESCE(b.trailer_views, 0) > 1000000
  AND COALESCE(i.search_interest, 0) > 50
  AND COALESCE(w.total_watchtime, 0) > 0
ORDER BY trailer_views DESC
LIMIT 50;
```

6)

```
WITH imdb_clean AS (
  SELECT
    LOWER(TRIM(title)) AS clean_title,
    genre,
    actors,
    TRY_CAST(NULLIF(imdbrating, '') AS DOUBLE) AS rating
  FROM imdb
),

youtube_clean AS (
  SELECT
    LOWER(TRIM(title)) AS clean_title,
    MAX(CAST(REPLACE("View Count", '"', '') AS BIGINT)) AS trailer_views
  FROM youtube
  GROUP BY LOWER(TRIM(title))
),

reddit_clean AS (
  SELECT
    LOWER(TRIM(title)) AS clean_title,
    AVG(TRY_CAST("Sentiment Polarity" AS DOUBLE)) AS avg_polarity,
    AVG(TRY_CAST("Sentiment Subjectivity" AS DOUBLE)) AS avg_subjectivity
  FROM reddit
  GROUP BY LOWER(TRIM(title))
),

netflix_avg_rank AS (
  SELECT
```

```
    LOWER(TRIM(title)) AS clean_title,
    AVG(TRY_CAST(rank AS DOUBLE)) AS avg_netflix_rank
  FROM netflixregion
  WHERE CAST(rank AS VARCHAR) <> ''
  GROUP BY LOWER(TRIM(title))
)

SELECT
  n.clean_title AS title,
  i.genre,
  i.actors,
  i.rating,
  y.trailer_views,
  r.avg_polarity,
  r.avg_subjectivity,
  n.avg_netflix_rank
FROM netflix_avg_rank n
LEFT JOIN imdb_clean i ON n.clean_title = i.clean_title
LEFT JOIN youtube_clean y ON n.clean_title = y.clean_title
LEFT JOIN reddit_clean r ON n.clean_title = r.clean_title
WHERE n.avg_netflix_rank IS NOT NULL
ORDER BY n.avg_netflix_rank ASC
LIMIT 100;
```