GameQuest – SQL Analysis Report

1. Project Overview

GameQuest is a standalone SQL project focused on exploring and analyzing a video-game dataset. The goal was to practice writing analytical SQL queries such as ranking, aggregation, trend analysis, and performance classification using pure SQL

2. Dataset Summary

Table: **games**

Columns: **id, title, genre, platform, rating, release_year**

The dataset was manually created for learning purposes.

3. Tools Used

SQL (MySQL)

GitHub for version control and sharing

4. Analysis Performed

- Genre and platform exploration

- Top-rated games identification

- Ranking per platform

- Yearly release trends and cumulative totals

- Platform performance classification

- Percentile ranking

5. Key SQL Queries

1. Find the average rating and total number of games for each platform.

```
SELECT platform, round(AVG(rating),1) AS AvgRating,
COUNT(*) AS TotalGames FROM games GROUP BY platform;
```

platform	AvgRating	TotalGames
Nintendo	4.8	5
PC	4.5	133
PlayStation	4.7	30
Xbox	4.7	3

2. Find the average rating and game count for each genre, sorted from highest to lowest average rating.

```
SELECT genre, round(AVG(rating),1) AS AvgRating, COUNT(*)
AS TotalGames

FROM games GROUP BY genre ORDER BY 2 DESC;
```

genre	AvgRating	TotalGames
MMORPG	4.9	1
Adventure	4.8	8
Open World	4.8	5
Roguelike	4.8	1
Stealth	4.7	5
Simulation	4.7	2
Puzzle	4.7	3
RPG	4.6	28
Horror	4.6	10
Racing	4.6	4

3. Show how many games were released in each decade.

```
SELECT (release_year DIV 10) * 10 AS Decade, COUNT(*) AS
TotalGames FROM Games GROUP BY Decade ORDER BY decade;
```

Decade	TotalGames
1990	20
2000	41
2010	55
2020	55

4. List platforms that have more than 1 game.

```
select platform,count(*)TotalGames

from games GROUP BY 1 having count(*)>1;
```

platform	TotalGames
Nintendo	5
PC	133
PlayStation	30
Xbox	3

5. For every game, show whether its rating is above or below the overall average rating.

```
SELECT title As Title,rating as Rating,
   CASE
       WHEN rating > (SELECT AVG(rating) FROM games)
        THEN 'Above Average'
       ELSE 'Below Average'
 END AS Rating_Vs_Avg
FROM games;
```

Title	Rating	Rating_Vs_Avg
Chrono Trigger	4.9	Above Average
Command & Conquer	4.6	Above Average
Quake	4.7	Above Average
Diablo	4.8	Above Average
Tomb Raider	4.6	Above Average
Crash Bandicoot	4.5	Below Average
Pokemon Red	4.4	Below Average
Final Fantasy VII	4.9	Above Average
GoldenEye 007	4.8	Above Average

6. Return only the top-rated game(s) for each platform.

```
SELECT Title,Platform,Rating,Release_Year,Rnk
FROM ( SELECT *,
      RANK() OVER (PARTITION BY platform ORDER BY
      rating DESC) AS Rnk
 FROM games ) ranked
WHERE Rnk = 1;
```

Title	Platform	Rating	Release_Year	Rnk
The Legend of Zelda: Ocarina of Time	Nintendo	5.0	1998	1
Baldur's Gate 3	PC	5.0	2023	1
Half-Life	PC	5.0	1998	1
Elden Ring	PC	5.0	2022	1
Half-Life 2	PC	5.0	2004	1
Resident Evil 4	PlayStation	4.9	2005	1
Bloodborne	PlayStation	4.9	2015	1
Shadow of the Colossus	PlayStation	4.9	2005	1
God of War (2018)	PlayStation	4.9	2018	1

7. Rank games within each platform based on rating (highest first).

```
SELECT *,

       RANK () OVER (PARTITION BY platform ORDER BY rating
DESC) AS rnk

FROM games;
```

id	title	genre	platform	rating	release_year	rnk
14	The Legend of Zelda: ...	Adventure	Nintendo	5.0	1998	1
1	Chrono Trigger	RPG	Nintendo	4.9	1995	2
91	Breath of the Wild	Adventure	Nintendo	4.9	2017	2
9	GoldenEye 007	Shooter	Nintendo	4.8	1997	4
7	Pokemon Red	RPG	Nintendo	4.4	1996	5
110	Baldur's Gate 3	RPG	PC	5.0	2023	1
11	Half-Life	Shooter	PC	5.0	1998	1
107	Elden Ring	RPG	PC	5.0	2022	1
37	Half-Life 2	Shooter	PC	5.0	2004	1

Result 9

8. Find games whose rating is higher than the average rating of their own Platform.

```
SELECT * FROM

 ( SELECT g.*,

        AVG(rating) OVER (PARTITION BY platform) AS
avg_platform_rating

    FROM Games g ) t

WHERE t.rating > t.avg_platform_rating;
```

id	title	genre	platform	rating	release_year	avg_platform_rating
1	Chrono Trigger	RPG	Nintendo	4.9	1995	4.80000
14	The Legend of Zelda: ...	Adventure	Nintendo	5.0	1998	4.80000
91	Breath of the Wild	Adventure	Nintendo	4.9	2017	4.80000
2	Command & Conquer	Strategy	PC	4.6	1995	4.45940
3	Quake	Shooter	PC	4.7	1996	4.45940
4	Diablo	RPG	PC	4.8	1996	4.45940
5	Tomb Raider	Adventure	PC	4.6	1996	4.45940
153	Star Wars Jedi: Fallen ...	Action	PC	4.5	2019	4.45940
10	Age of Empires	Strategy	PC	4.7	1997	4.45940

Result 10

9. Show yearly release counts and a running total of games released over time.

```
SELECT *, SUM(releasecount) OVER (ORDER BY
release_year) AS Running_total

FROM ( SELECT Release_year,

      COUNT(*) AS Releasecount

FROM games

GROUP BY release_year

) t;
```

Release_year	Releasecount	Running_total
1995	2	2
1996	5	7
1997	3	10
1998	5	15
1999	5	20
2000	4	24
2001	5	29
2002	4	33
2003	3	36

10. Classify each platform into performance tiers based on its average rating (Elite / Strong / Average).

```
SELECT platform, AVG(rating) AS avg_rating,

    CASE WHEN AVG(rating) > 4.5 THEN 'Elite'

        WHEN AVG(rating) > 4 THEN 'Strong'

        ELSE 'Average'  END AS Performance

FROM games

GROUP BY platform;
```

platform	avg_rating	Performance
Nintendo	4.80000	Elite
PC	4.45940	Strong
PlayStation	4.68667	Elite
Xbox	4.73333	Elite

10. Percentile Ranking of Games

```
SELECT

    title,

    rating,

    concat( ROUND(PERCENT_RANK() OVER (ORDER BY rating) *
100, 1),'%') AS percentile_rank_percentage

FROM Games;
```

title	rating	percentile_rank_percentage
Paladins	3.9	5.3%
Watch Dogs: Legion	3.9	5.3%
Dragon's Dogma 2	4.0	6.5%
Fall Guys	4.0	6.5%
Dead by Daylight	4.0	6.5%
Palworld	4.1	8.2%
EA Sports FC 25	4.1	8.2%
Ashes Cricket 2021	4.1	8.2%
Fortnite Mobile	4.1	8.2%

9. Conclusion

This project demonstrates SQL-based exploratory and analytical techniques such as ranking, aggregation, and trend analysis. It serves as a foundational portfolio project for junior data analyst roles.

10. Author

Name: M Kishore Kumar

GitHub: https://github.com/KishoreM-06/SQL-PROJECTS

LinkedIn: https://www.linkedin.com/in/mkishorekumar/