In-Depth Analysis of Engagement Patterns & Player Behavior in Steam Games Using Business Intelligence Architecture

Joshua Alexander R. Laxa[1], Aljirah Brendl Y. Resurreccion[2], and Andrei G. Tamse[3]
College of Computer Studies, De La Salle University
[1]joshua_laxa@dlsu.edu.ph ,, [2]aljirah_resurreccion@dlsu.edu.ph , [3]andrei_tamse@dlsu.edu.ph

ABSTRACT

The original data that is in a CSV file can't be handled by Excel properly. Working on large data sets is more efficient when using RDBMS or NoSQL as it is designed to handle large data sets. ETL tools also allow people to manipulate and clean large data. This paper provides an in-depth report of the team process from building a data warehouse to generating meaningful reports and evaluating its performance. Star schema was utilized as the team interpreted that the relationship of the data in the schema is one-to-one. Star schema is influenced by informed decision due to its simplicity which will create OLAP reports more efficiently.

Through this analysis, we aim to reveal critical insights into player engagement, illustrating how genre, publisher, and playtime influence player behavior. Our findings will not only contribute to a better understanding of gaming dynamics on the Steam platform but also provide actionable intelligence for stakeholders in the gaming industry to enhance player retention and engagement. Ultimately, this paper will demonstrate both the technical processes involved in our analysis, including the ETL pipeline and OLAP framework, and the comprehensive insights gained from the dataset.

The analysis underscores significant relationships between game characteristics, such as game mode, genre, and platform, and player engagement metrics, including peak concurrent users , playtime, and reviews. Multiplayer features and platform support emerge as particularly influential in driving engagement, while price and genre critically shape user satisfaction and feedback. These insights provide game developers, publishers, and analysts with a foundation for data-driven decision-making, enabling optimization of platform support and pricing strategies, ultimately aimed at improving player retention, satisfaction, and market competitiveness.

Keywords

Data Set, Data Warehousing, ETL Processes, OLAP Analytics, and Query Optimization

1. Introduction

Our dataset comprises reports of Steam games, featuring information such as playtime, genre, publisher, and more. This paper aims to analyze the engagement patterns and behaviors of players, showcasing the power of Business Intelligence Architecture (BI architecture) techniques, which serve as the foundation for our analysis.

This study uses a data warehouse and ETL pipeline (Extract, Transform, Load) processes to facilitate our analysis and present OLAP (Online Analytical Processing) reports. These reports demonstrate how BIA techniques create a meaningful report and optimize the analytical capabilities of businesses.

We will provide an in-depth overview of each step taken to achieve these reports, focusing on data cleaning and the employment of a star schema, which is considered the most effective schema in MySQL for our analysis. This focus allows us to streamline the data used in our study. OLAP visualizations are performed by creating a simple web application for easy and dynamic filtering of the OLAP reports, enabling more efficient querying, while query optimization techniques ensure that our processes remain both effective and efficient.

2. Data Warehouse

The team chose a star schema for the data warehouse, with dimension tables representing key data categories (e.g., developers, publishers, operating systems, categories, and genres) and a central fact table to store primary data points for Steam games. Because the team decided that fields with potential multiple values for a game do not need to be separated, many-to-many relationships were avoided. This schema offers a clear relational structure, enhancing data integrity and simplifying analytical queries.

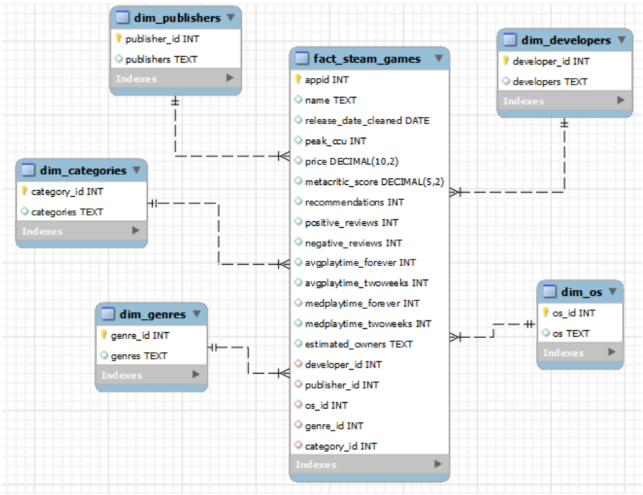

Figure 1. Star Schema Designed for the OLAP Applications

However, as the dataset was explored, the team identified some data fields that would not be required for immediate analysis or reporting. Given the size of the dataset and the potential for extensive processing demands, the team decided to prioritize the data warehouse's Online Analytical Processing (OLAP) potential first. This approach would allow faster retrieval of relevant data and improve performance when analyzing large data sets, such as those containing game sales metrics, user reviews, and playtime statistics.

By excluding unnecessary fields from the ETL process, the team can focus on efficiently loading, transforming, and querying only the most valuable data. This decision not only conserves time and resources during ETL but also minimizes data storage costs and processing loads within the warehouse.

3. ETL Script

Before loading data into a warehouse, it is essential to ensure its quality through thorough data cleansing, a critical part of the ETL (Extract, Transform, Load) process. The ETL process is essential in BI architecture, managing and preparing data to build a data warehouse optimized for analysis. Without proper ETL handling, raw data can become inconsistent and inaccurate, making it nearly unusable. By addressing issues such as inconsistent column names, null values, and other discrepancies, ETL enables data to be cleaned, transformed, and optimized before entering the warehouse, supporting the generation of accurate, meaningful analytics.

In this project, the team used Python and RapidMiner as ETL tools to process data from a Kaggle dataset on Steam games. The initial data cleaning was performed using Python, specifically the pandas library, to handle the dataset's transformation and address issues in the CSV files, such as improperly assigned columns and null values. For example, the team noticed two column names were concatenated, resulting in an empty column on the far right. By splitting the concatenated "DiscountDLC count" column and renaming it into two separate columns ("Discount" and "DLC count"), this issue was resolved. Using the isna().sum() function, the team identified that columns such as 'reviews', 'notes,' and 'metacritic url' contained over 90% null values, while 'website' and 'support url' had over 50%. The team chose to drop these columns while addressing remaining columns with lower percentages of null values.

A helpful guide by Alkabani (2023) on Kaggle for the Steam games dataset highlighted key discrepancies in the dataset and outlined effective techniques for handling null values, guiding the team's data-cleansing process and making it more efficient. Once the initial data was cleaned, it was loaded into RapidMiner for the main ETL process. RapidMiner's visualization capabilities and graphical interface facilitated handling relational structures, making it possible to link foreign keys in the fact table to create a cohesive star schema from the raw dataset. About half of the total project time was spent on ETL due to challenges such as implementing surrogate keys, splitting and linking data across tables, and resolving performance issues like interface lags and occasional crashes.

The loading of cleaned and structured data into the data warehouse was completed by using key RapidMiner operators, such as **Select Attributes**, **Generate ID**, **Join**, and **Aggregate**:

- **Select Attributes**: Filtered out unnecessary data, focusing only on relevant columns.
- **Generate ID** and **Aggregate**: Enabled surrogate key implementation by grouping similar rows.
- **Join**: This operator was critical in establishing relationships, making it possible to reference data within the fact table.
- **MySQL Integration**: RapidMiner's smooth integration with MySQL allowed monitoring data transparency and ensuring accuracy during loading.

Each ETL step was carried out rigorously to ensure that the data warehouse was well-designed and optimized for querying, given the large volume of data in the project.

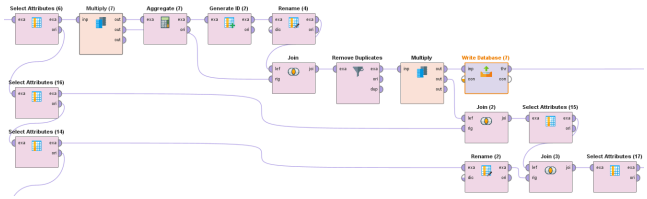

Figure 2. ETL Pipeline for Surrogate Key Creation and Fact Table Referencing

4. OLAP Application

An OLAP is used for complex data analysis that enables users to handle large amounts of data in multiple dimensions. This organized data allows for fast query, processing, and interactive analysis through the use of its operations such as roll-up, drill down, slicing,and dicing, and pivoting. These operations help users and businesses gain insights on summarized and granular levels of data helping them to make informed decisions.

In this project, the data warehouse schema was designed to a star schema allowing for faster and more efficient query processes, and a much more direct and simpler query. The goal was to create an OLAP application that displays interactive and insightful reports on *"The Average Peak CCU and Playtime for Multiplayer vs Single Player Games"*, *"Average Playtime, Metacritic Score, and Total Positive Reviews by Genre"*, *"Average Playtime by Year, OS, and Multiplayer Mode"*, and lastly *"Average Reviews, Median Playtime, and Metacritic Score by Publisher and Price Range"*.

4.1 Report and Chart Types

- **Roll-up operation:** The roll-up operation was used to get the average peak concurrent users and average playing time for two game modes instead of calculating it for each game.

```sql
SELECT
    CASE
        WHEN dc.categories LIKE '%{cat1}%' THEN '{cat1}'
        WHEN dc.categories LIKE '%{cat2}%' THEN '{cat2}'
    END AS game_mode,
    AVG(fsg.peak_ccu) AS avg_peak_ccu,
    AVG(fsg.avgplaytime_forever) AS avg_playtime
FROM Fact_Steam_Games fsg
JOIN
    Dim_Categories dc ON fsg.category_id = dc.category_id
WHERE
    dc.categories LIKE '%{cat1}%' OR dc.categories LIKE '%{cat2}%'
GROUP BY
    game_mode;
```

Figure 3. SQL Query for the Average Peak CCU and Playtime for Multiplayer and Single-player Games

The CASE statement in the query checks if the input belongs to the two categories which is "Single-Player" and "Multiplayer". The game_mode is assigned to the CASE to use it to aggregate the data in order to summarize the data, which corresponds to a roll-up operation.

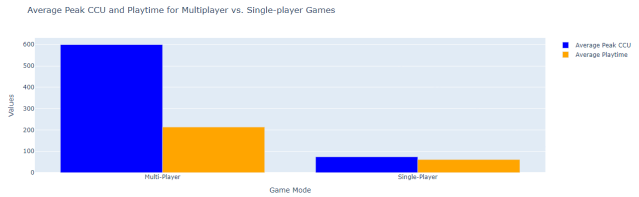

Figure 3.1 Roll-up for Average Peak CCU and Playtime for Multiplayer and Single-player Games

The choice of graph used is the grouped bar chart, this chart was used as it allows a side by side comparison of their ave_peak_ccu and ave_playtime between the two game modes.Each color corresponds to a metric, blue being the ave_peak_ccu and orange being the ave_playtime. The grouped layout also makes it visually straightforward easy to compare on which gamemode has a higher metric

- **Drill Down operation:** The drill down operation was used to get the average playtime, metacritic score, and total positive reviews by genre.

```sql
list_genres = ['Accounting', 'Action', 'Adventure', 'Casual', 'Indie', 'RPG',
               'Racing', 'Sports', 'Simulation', 'Strategy']

drill_down_query = """
SELECT
    dg.genres AS genre,
    AVG(fsg.avgplaytime_forever) AS avg_playtime_forever,
    AVG(fsg.metacritic_score) AS avg_meta_critic_score,
    COUNT(fsg.positive_reviews) AS total_reviews
FROM
    Fact_Steam_Games fsg
JOIN
    Dim_Genres dg ON fsg.genre_id = dg.genre_id
"""

if genre == 'ALL':
    genre_list = "', '".join(list_genres)
    drill_down_query += f" WHERE dg.genres IN ('{genre_list}')"
else:
    drill_down_query += f" WHERE dg.genres = '{genre}'"

drill_down_query += """
GROUP BY
    dg.genres
ORDER BY
    avg_playtime_forever DESC;
```

Figure 4. SQL Query for the Average Playtime, Metacritic Score, and Total Positive Reviews By Genre

The query allows users to explore more detailed data by drilling down on a specific metric.In this case the metric used was genre, to select one or view all genres to see how they each compare in terms of playtime, critic score, and reviews.

Additionally, an OLAP operation "Aggregate" was also used in this query. The use of AVG (average), COUNT, GROUP BY allows for calculation of metrics by genre, and ORDER BY to sort the data was used in order to properly display the report.

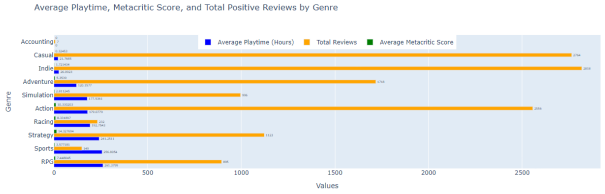

Figure 4.1 Drill Down for the Average Playtime, Metacritic Score, and Total Positive Reviews By Genre

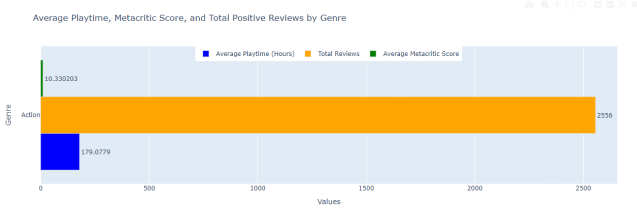

Figure 4.2 Drill Down for the Average Playtime, Metacritic Score, and Total Positive Reviews By Genre

The first image displays all genres and all their metrics, while the second image displays when one genre is chosen. The graph used is a horizontal grouped bar chart, it was used to allow easier visual comparison across genres. Users can discern quickly which genres are better received, higher rating, and more reviews. The horizontal approach was used since each genre had three metrics, and in order to properly display and compare each genre the horizontal bar was needed.

- **Slice and Dice operation:** The slice and dice operation was used to get average playtime by year, operating system, and mode; specifically multiplayer mode. .

```sql
SELECT
    do.os AS operating_system,
    YEAR(fsg.release_date_cleaned) AS release_year,
    CASE
        WHEN INSTR(dc.categories, 'Multi-player') > 0 THEN 'Yes'
        ELSE 'No'
    END AS multiplayer_mode,
    AVG(fsg.avgplaytime_forever) AS avg_playtime
FROM
    Fact_Steam_Games fsg
JOIN
    Dim_OS do ON fsg.os_id = do.os_id
JOIN
    Dim_Categories dc ON fsg.category_id = dc.category_id
WHERE
    {platforms_condition}
    AND YEAR(fsg.release_date_cleaned) IN ({years})
    {multiplayer_condition}
GROUP BY
    operating_system, release_year, multiplayer_mode
ORDER BY
    release_year;
```

Figure 5. SQL Query for the Average Playtime By Year, OS, and Mode

The query analyzes the player trends overtime (year) in regards to different operating systems, and the presence of a multiplayer mode in game. It allows the user to filter results based on specific metrics, and allows the user, or businesses, or developers to gain insight on which platform or game features are better received by players.

Additionally, OLAP operation "Aggregate" was also used in this query. The use of AVG (Average) to calculate the playtime, and GROUP BY in order to aggregate the data based on the three filter metrics.

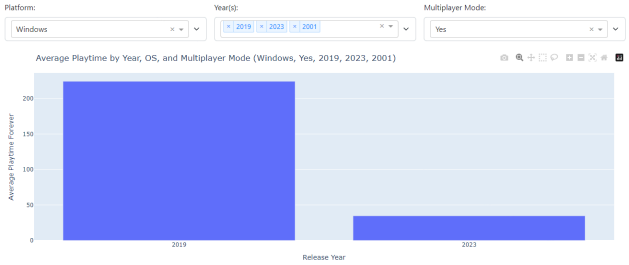

Figure 5.1 SQL Query for the Average Playtime By Year, OS, and Mode

The graph used is a bar chart graph, it allows a straightforward visualization of trends broken down by the year, operating software and the presence of multiplayer mode. The graph is dynamically interactive allowing users to filter the results based on the metrics that are selected.

- **Pivot operation:** The pivot operation was used to get average reviews, median play time, and metacritic score by publisher and price range.

```
pivot_query = f"""
SELECT
    dp.publishers,
    CASE
        WHEN fsg.price BETWEEN {price_range[0]} AND {price_range[1]} THEN 'Within Price Range'
        ELSE 'Outside Price Range'
    END AS price_range,
    AVG(fsg.positive_reviews) AS avg_positive_reviews,
    AVG(fsg.negative_reviews) AS avg_negative_reviews,
    AVG(fsg.medplaytime_forever) AS avg_median_playtime,
    AVG(fsg.metacritic_score) AS avg_metacritic_score
FROM
    Fact_Steam_Games fsg
JOIN
    Dim_Publishers dp ON fsg.publisher_id = dp.publisher_id
WHERE
    fsg.price BETWEEN {price_range[0]} AND {price_range[1]}
    AND fsg.metacritic_score BETWEEN {metacritic_score_range[0]} AND {metacritic_score_range[1]}
GROUP BY
    dp.publishers, price_range
ORDER BY
    avg_positive_reviews DESC
LIMIT {top_publishers};
"""
```

Figure 6. SQL Query for the Average Reviews, Median Play Time, and Metacritic Score by Publisher and Price Range.

The query provides insights into how publishers perform through playtime, reviews, and metacritic scores based on pricing and score parameters. It compares publishers by averaging metrics, users can identify which publishers have a better overall review, better receivability by players, and better critic scores. By limiting data to a specific price and metacritic range, users can focus on specific values and ranges.

Furthermore, the data is re-oriented, changing rows into columns and vice versa, to allow users to compare data across different categories. The GROUP BY clause coupled with conditional labels such as "Within Price Range" and "Outside Price Range" helps with the pivoted views across each publisher, price range and other types of reviews and scores.

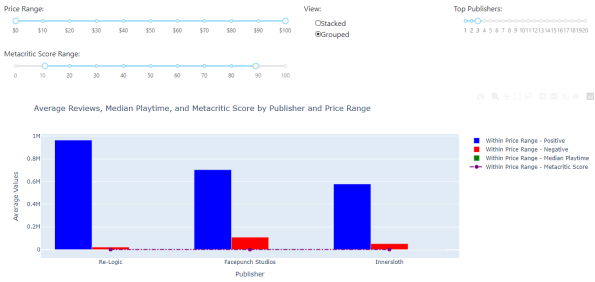

Figure 6.1 Pivot for the Average Reviews, Median Play Time, and Metacritic Score by Publisher and Price Range.

The graph used was a mixture of bar graphs and scatter plots. Blue bars represent the total positive reviews, red bars for the negative reviews, and the green bars for the median playtime. These bars are grouped to compare between metrics of score, price range, and top publishers. The metacritic score uses a scatterplot with a line in order to not overlap with the bar graphs.

5. Query Processing and Optimization

Query Optimization is the process of enhancing query execution efficiency by minimizing resource usage and reducing execution time, making it especially crucial for OLAP (Online Analytical Processing) systems. In cases like the star schema for the Steam Games dataset, optimized queries ensure smooth performance when dealing with extensive data volumes, enabling faster data retrieval that is essential for reporting, analytical processing, and improving the overall user experience (Dremio, 2024). To optimize the queries, the team implemented strategies such as adding indices and testing on an alternative device. However, further analysis revealed that the addition of indices led to a slight performance drawback compared to the original configuration, resulting in a less efficient execution plan than anticipated.

In conclusion, the team determined that the original schema, without additional indices, proved to be the most optimized version for this particular dataset and workload.

6. Results and Analysis

Functional Testing

We evaluated the effectiveness of the OLAP application through functional testing and performance testing. Functional testing involved manual verification of OLAP reports within the database to confirm their accuracy and reliability, ensuring that each report met the expected standards and provided accurate insights. This validation process included a series of test cases designed to assess the correctness of the ETL process, where we verified that data was accurately extracted from source systems, transformed according to defined rules, and correctly loaded into the data warehouse. We conducted test cases that compared source and target data to ensure that all records were accounted for and that data integrity was maintained throughout the ETL workflow.

Additionally, we validated the correctness of our OLAP operations by executing predefined queries against the data warehouse and cross-referencing the results with expected outcomes. This involved creating specific scenarios to test various dimensions and measures, ensuring that the aggregations,

calculations, and transformations performed by the OLAP application were precise. By systematically documenting any discrepancies and addressing them promptly, we ensured that our OLAP application not only functioned as intended but also delivered valuable and trustworthy insights for decision-making.

Performance Testing

Performance testing was conducted using Python to measure query execution times, assessing the speed of data retrieval processes. By running three iterations of each query, the team was able to gather a robust set of performance data, which included execution times across varying input sizes and configurations. These evaluations enabled the team to analyze the application's efficiency and identify potential optimizations to improve response times, thereby enhancing user satisfaction with faster, more responsive results. Additionally, the testing process involved examining the hardware specifications, including CPU and memory resources, as well as the size and structure of the input data.

By comparing the execution times of the original queries against those optimized through strategies such as indexing and query restructuring, we could pinpoint specific areas where performance bottlenecks occurred. This comprehensive approach not only informed our understanding of how database design impacts query speed but also highlighted the importance of tailoring optimization strategies to the unique requirements of our applications.

6.1 Functional Testing

Functional testing is conducted to verify that the software behaves as specified and performs its intended functions accurately. For this project, the team focused on testing four key OLAP operations: roll-up, drill-down, slice and dice, and pivot. The objective of this functional testing was to confirm that the application produces consistent results across both MySQL Workbench and the graphical representations. By comparing outputs from these two sources, we ensured that the application calculations are accurate and align with the underlying data in MySQL Workbench.

Roll-up Operation:

For this operation, the team focused on the 'categories' field within the Dim_Categories table to classify data based on game modes, specifically 'Multi-player' and 'Single-player' games. By using conditional logic in the query, the team could categorize each entry under the appropriate game mode. Key metrics were then calculated for each mode, including the average peak concurrent users (peak_ccu) and the average playtime (avgplaytime_forever).

```
-- Comparison of average peak CCU and playtime between multiplayer and single-player games --
SELECT
    CASE
        WHEN dc.categories LIKE '%Multi-player%' THEN 'Multiplayer'
        WHEN dc.categories LIKE '%Single-player%' THEN 'Single-player'
    END AS game_mode,
    AVG(fsg.peak_ccu) AS avg_peak_ccu,
    AVG(fsg.avgplaytime_forever) AS avg_playtime
FROM Fact_Steam_Games fsg
JOIN Dim_Categories dc ON fsg.category_id = dc.category_id
WHERE dc.categories LIKE '%Multi-player%' OR dc.categories LIKE '%Single-player%'
GROUP BY game_mode;
```

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

game_mode	avg_peak_ccu	avg_playtime
Multiplayer	599.3826	213.1650
Single-player	73.8436	61.0043

Figure 7.1. SQL Query for the Roll-up Operation Testing

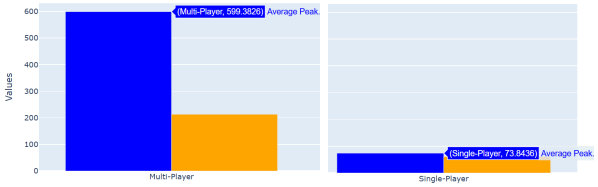

Figure 7.2. Visual Graph for the Roll-up Operation Testing

The graph accurately displayed 599.3826 (blue bar) as its *average peak CCU* and 213.1650 *average playtime* (yellow bar) for the *multi-player* category. As for the single-player, it also shows in the graph that it has 73.8436 *average peak CCU* and 61.0043 *average playtime.*

This proves that the graph accurately reflected the expected data, thereby validating the effectiveness of the roll-up operation. This alignment confirms that the roll-up function in our application is working as intended and can reliably aggregate data across specified fields.

Insight: There is a clear contrast in average peak concurrent users (CCU) and playtime between multiplayer and single-player games. The roll-up operation highlights that multiplayer games tend to have a higher peak CCU due to their social and interactive nature, whereas single-player games may show varied playtime patterns depending on their genre and depth.

Implication: This suggests that games with multiplayer features have a broader appeal in terms of player engagement and CCU, making them a more attractive option for publishers focused on maximizing user engagement.

Drill-down operation:

For this operation, the team chose the 'genres' field from the Dim_Genres table to categorize and analyze data across selected game genres, such as 'Accounting,' 'Action,' 'Adventure,' 'Casual,' 'Indie,' 'RPG,' 'Racing,' 'Sports,' 'Simulation,' and 'Strategy.' Key metrics were aggregated, including the average playtime (calculated from avgplaytime_forever) and the average Metacritic score for each genre, allowing insight into typical engagement and quality ratings per genre. Additionally, the positive_reviews field was counted to obtain the total number of reviews, providing a measure of genre popularity.

```sql
SELECT dg.genres AS genre,
    AVG(fsg.avgplaytime_forever) AS avg_playtime_forever,
    AVG(fsg.metacritic_score) AS avg_meta_critic_score,
    COUNT(fsg.positive_reviews) AS total_reviews
FROM Fact_Steam_Games fsg
JOIN Dim_Genres dg ON fsg.genre_id = dg.genre_id
WHERE dg.genres IN ('Accounting', 'Action', 'Adventure', 'Casual', 'Indie', 'RPG', 'Racing', 'Sports', 'Simulation', 'Strategy')
GROUP BY dg.genres
ORDER BY avg_playtime_forever;
```

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

genre	avg_playtime_forever	avg_meta_critic_score	total_reviews
Accounting	0.0000	0.000000	7
Casual	21.7685	0.32453	2764
Indie	26.0923	1.721434	2818
Adventure	120.1577	5.353900	1718
Simulation	177.5361	2.811245	996
Action	179.0779	10.330203	2556
Racing	192.7543	8.331897	232
Strategy	241.2511	14.327694	1123
Sports	256.8054	3.577181	149
RPG	261.3799	7.448045	895

Figure 8.1. SQL Query for the Drill-down Operation Testing

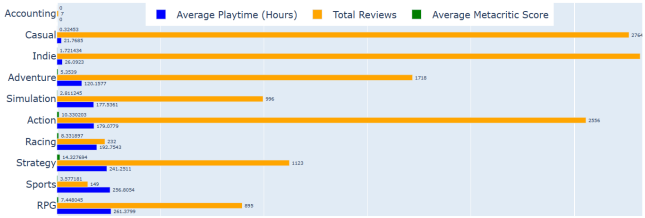

Figure 8.2. Visual Graph for the Drill-down Operation Testing

As shown, the values displayed in the visual graph precisely match those in the MySQL Workbench results. This alignment demonstrates that the graph accurately represents the expected data, validating the effectiveness of the drill-down operation. This consistency confirms that the drill-down function in our application performs as intended and can reliably aggregate data across the specified fields.

Insight: By examining playtime, Metacritic scores, and positive reviews by genre, the analysis reveals that certain genres consistently achieve higher engagement and ratings. For example, genres like RPGs or adventure games might have higher playtime, while more niche genres may have lower but more targeted appeal.

Implication**:** Developers and publishers can use this insight to understand which genres are better received critically and by users, guiding future investments and development focus toward genres with higher user satisfaction and engagement.

Slice and Dice:

For this operation, the team used the os field from the Dim_OS table to filter data for games available on the Windows operating system. Focusing on release years 2018, 2019, and 2020 (can be adjusted in application), the team categorized each game based on whether it supports multiplayer mode, using conditional logic to label each entry accordingly. Key metrics, such as the average playtime (avgplaytime_forever), were then calculated for each year, providing insights into user engagement trends over time for single-player versus multiplayer modes on Windows.

```sql
SELECT do.os AS operating_system,
    YEAR(fsg.release_date_cleaned) AS release_year,
    CASE WHEN INSTR(dc.categories, 'Multi-player') > 0 THEN 'Yes'
        ELSE 'No'
    END AS multiplayer_mode,
    AVG(fsg.avgplaytime_forever) AS avg_playtime
FROM Fact_Steam_Games fsg
JOIN Dim_OS do ON fsg.os_id = do.os_id
JOIN Dim_Categories dc ON fsg.category_id = dc.category_id
WHERE do.os IN ('windows')
    AND YEAR(fsg.release_date_cleaned) IN (2018, 2019, 2020)
GROUP BY operating_system, release_year, multiplayer_mode
ORDER BY release_year;
```

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

operating_system	release_year	multiplayer_mode	avg_playtime
windows	2018	No	75.4680
windows	2018	Yes	225.6699
windows	2019	No	94.3993
windows	2019	Yes	224.3023
windows	2020	No	59.2451
windows	2020	Yes	252.2217

Figure 9.1. SQL Query for the Slice and Dice Operation Testing

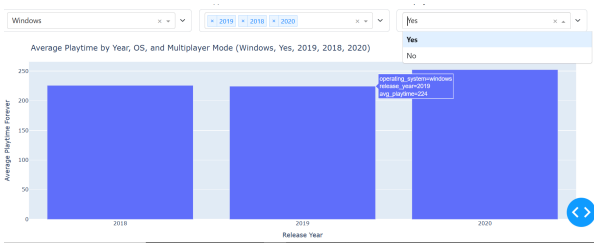

Figure 9.1. Visual Graph for the Slice and Dice Operation Testing (Yes)

The data displayed in the graph perfectly matches the correct values from the MySQL Workbench results, accurately reflecting the conditional logic for "Yes" in multiplayer mode.

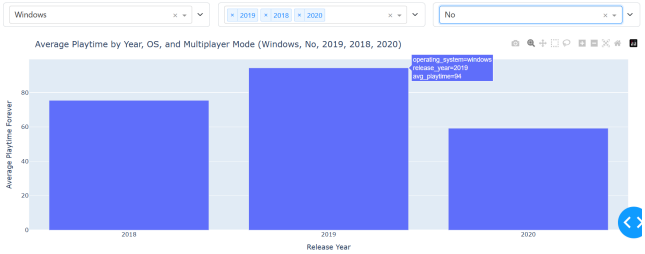

Figure 9.2. Visual Graph for the Slice and Dice Operation Testing (No)

Similarly, entries labeled with "No" for multiplayer mode also align with the expected results from the workbench.

This proves that the graph accurately reflected the expected data, thereby validating the effectiveness of the slice and dice operation. This alignment confirms that the slice and dice function in our application is working as intended and can reliably aggregate data across specified fields.

Insight: The slice and dice operation demonstrates trends in playtime across different years, operating systems, and game modes, with noticeable changes in engagement over time and variations in popularity based on platform and multiplayer availability. Multiplayer games on Windows, for example, might show higher average playtime, reflecting user preferences for more accessible or feature-rich platforms.

Implication: This insight helps stakeholders identify which OS and game modes are gaining or losing popularity, guiding future decisions about platform support, feature prioritization, and potential OS-specific optimizations.

Pivot Operation:

In this setup, the team provided an interactive visualization where users can dynamically adjust the filters for price and Metacritic score. The price range can be adjusted from $0 to $100, while the Metacritic score can vary from 0 to 100, allowing flexible exploration of the data.

Users can also toggle between specific metrics, such as positive reviews, negative reviews, median playtime, and Metacritic score, to isolate only the metrics they want to view in the chart.

Additionally, the visualization allows users to adjust the number of top publishers displayed, from as few as 1 to as many as 20.

```sql
SELECT dp.publishers,
    CASE WHEN fsg.price BETWEEN 10 AND 50 THEN 'Within Price Range'
        ELSE 'Outside Price Range'
    END AS price_range,
    AVG(fsg.positive_reviews) AS avg_positive_reviews,
    AVG(fsg.negative_reviews) AS avg_negative_reviews,
    AVG(fsg.medplaytime_forever) AS avg_median_playtime,
    AVG(fsg.metacritic_score) AS avg_metacritic_score
FROM Fact_Steam_Games fsg
JOIN Dim_Publishers dp ON fsg.publisher_id = dp.publisher_id
WHERE fsg.price BETWEEN 10 AND 50
    AND fsg.metacritic_score BETWEEN 50 AND 90
GROUP BY dp.publishers, price_range
ORDER BY avg_positive_reviews DESC
LIMIT 3;
```

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

publishers	price_range	avg_positive_reviews	avg_negative_reviews	avg_median_playtime	avg_metacritic_score
Facepunch Studios	Within Price Range	703687.0000	108223.0000	3035.0000	69.000000
ConcernedApe	Within Price Range	475785.0000	8983.0000	1458.0000	89.000000
Studio Wildcard	Within Price Range	461567.0000	98701.0000	1062.0000	70.000000

Figure 10.1. SQL Query for the Pivot Operation Testing

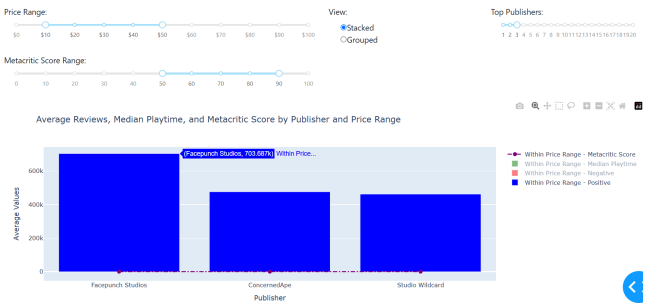

Figure 10.2. Visual Graph for the Pivot Operation Testing (Positive)

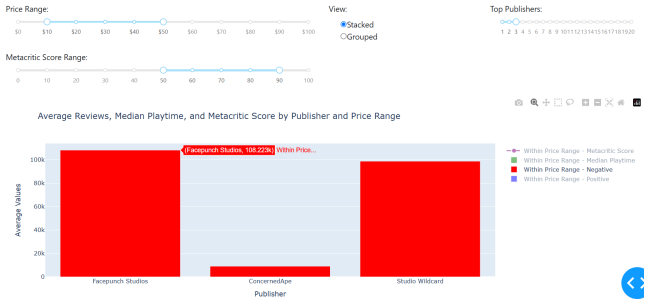

Figure 10.3. Visual Graph for the Pivot Operation Testing (Negative)

Figure 10.4. Visual Graph for the Pivot Operation Testing (Median Playtime)

Figure 10.5. Visual Graph for the Pivot Operation Testing (Metacritic Score)

The pivot graph results were accurate and precise, with no discrepancies in values between the workbench output and the graph. This consistency validates the success of the pivot operation, ensuring reliable data representation across both platforms.

Insight: The pivot operation reveals performance metrics reviews, playtime, and Metacritic scores across publishers and different price ranges. It highlights which publishers receive better feedback and engagement within specific price ranges, showing how price impacts user perception and playtime.

Implication: By understanding these patterns, publishers can refine their pricing strategies and evaluate competitor performance. This may lead to more targeted pricing models that consider both critical reception and user feedback.

6.2 Performance Testing

The team conducted 3 performance testing for each queries used for the OLAP operations: Performance of Queries using Indices,

Performance of Queries using Star, and Performance of Queries in various Hardware.

These are tested to see which of the following are the best and have the most optimized performance.

6.2.1 Performance of Queries using Indices

Roll-up Operation Performance Testing:

```
Run 1: 0.34104 seconds
Run 2: 0.32910 seconds
Run 3: 0.33503 seconds
Average execution time: 0.33506 seconds
```

After running it 3 times, it resulted in an average of 0.33506 seconds.

Drill-down Operation Performance Testing:

```
Run 1: 0.19304 seconds
Run 2: 0.20117 seconds
Run 3: 0.20320 seconds
Average execution time: 0.19914 seconds
```

After running it 3 times, it resulted in an average of 0.19914 seconds.

Slice and Dice Operation Performance Testing:

```
Run 1: 0.27340 seconds
Run 2: 0.20596 seconds
Run 3: 0.20896 seconds
Average execution time: 0.22944 seconds
```

After running it 3 times, it resulted in an average of 0.22944 seconds.

Pivot Operation Performance Testing:

```
Run 1: 0.06904 seconds
Run 2: 0.07004 seconds
Run 3: 0.07404 seconds
Average execution time: 0.07104 seconds
```

After running it 3 times, it resulted in an average of 0.07104 seconds.

6.2.2 Performance of Queries using Star

Roll-up Operation Performance Testing:

```
Run 1: 0.26106 seconds
Run 2: 0.26601 seconds
Run 3: 0.27221 seconds
Average execution time: 0.26643 seconds
```

After running it 3 times, it resulted in an average of 0.26643 seconds.

Drill-down Operation Performance Testing:

```
Run 1: 0.15904 seconds
Run 2: 0.16204 seconds
Run 3: 0.16300 seconds
Average execution time: 0.16136 seconds
```

After running it 3 times, it resulted in an average of 0.16136 seconds.

Slice and Dice Operation Performance Testing:

```
Run 1: 0.18056 seconds
Run 2: 0.17185 seconds
Run 3: 0.17353 seconds
Average execution time: 0.17531 seconds
```

After running it 3 times, it resulted in an average of 0.17531 seconds.

Pivot Operation Performance Testing:

```
Run 1: 0.07600 seconds
Run 2: 0.07303 seconds
Run 3: 0.07297 seconds
Average execution time: 0.07400 seconds
```

After running it 3 times, it resulted in an average of 0.07400 seconds.

6.2.3 Performance of Queries in various Hardware

To examine how hardware setup impacts query performance, we tested our OLAP application on two different devices with varying specifications. Device 1 has a more powerful Ryzen 5 processor and 16 GB of RAM, while Device 2 features an Intel i5 processor with only 8 GB of RAM.

Our team found that Device 2, despite having lower specifications, still runs the OLAP application slightly slower than Device 1. However, this difference in performance is not very noticeable during regular use. The improved specifications of Device 1, such as its faster CPU and more RAM, contribute to its efficiency, but the overall user experience remains satisfactory on both devices. This highlights that even with lower hardware specifications, the system can still perform adequately for our needs.

Upon further investigation, the team observed that adding indices unexpectedly resulted in a less optimized query execution plan compared to the original version, which did not use any additional indices. Instead of enhancing performance, the indices introduced additional complexity, causing the database engine to select suboptimal paths for data retrieval. This unexpected outcome highlighted that, in this particular case, the absence of indices allowed for a more efficient execution plan, as the database engine could rely on simpler, more direct scanning methods.

7. Conclusion

This study explored the design and development of an OLAP application that analyzes the Steam Game Dataset, focusing on generating reports regarding player engagement and behavior across different operating systems, publishers, genres, and game

modes. The team also designed and developed a star data warehouse schema, which allowed for faster and more efficient querying and analysis through the ETL process. To create a well-structured, organized, and optimized warehouse, Python and RapidMiner were utilized in the pre-processing, cleaning, and structuring of data.

OLAP operations such as roll-up, drill-down, slice and dice, and pivot were successfully implemented to generate reports on metrics like average playtime, peak concurrent users, Metacritic scores, and review counts. Our findings emphasized the importance of a well-structured data warehouse, offering valuable insights that can inform analysts on how to enhance player engagement and retention.

These insights reveal significant relationships between game characteristics and player engagement metrics, emphasizing the importance of multiplayer options and platform support for enhancing user satisfaction. While the study provides a solid framework for analyzing player behavior, limitations such as dataset biases and the analysis scope should be acknowledged. Future research could explore additional datasets or other dimensions of player engagement.

In conclusion, this project highlights the technical processes of developing an OLAP application and data warehouse, as well as the critical role of data analytics in the gaming industry. By leveraging these analyses, stakeholders can better understand player dynamics, leading to improved game design and increased player satisfaction.

8. References

[1] Alkabani A. (2023, January 18). *Games data cleaning, analysis , and visualization*. Kaggle. https://www.kaggle.com/code/abdalrhmanalkabani/games-data-cleaning-analysis-and-visualization#dealing-with-null-values-in-name-column

[2] RapidMiner. (n.d.). *How to resolve the join operator problem?*. RapidMiner Community. https://community.rapidminer.com/discussion/60631/how-to-resolve-the-join-operator-problem

[3] *Query optimization*. Dremio. (2024, July 2). https://www.dremio.com/wiki/query-optimization/#:~:text=Query%20Optimization%20offers%20numerous%20advantages,and%20maintaining%20application%20response%20times.

[4] Huez, A. (2024, April 9). Reporting : How to choose the right graph for your data. https://www.toucantoco.com/en/blog/reporting-how-to-choose-the-right-graph-for-your-data#:~:text=Bar%20and%20line%20chart&text=What%20is%20amazing%20about%20the,a%20set%20of%20continuous%20data .

9. Declarations

We, the undersigned, declare that this paper titled "In-Depth Analysis of Engagement Patterns & Player Behavior of Games in Steam Using a Data Warehouse," authored by Andrei G. Tamse, Joshua Laxa, and Aljirah Resurreccion, is our original work. We affirm that this study adheres to ethical research standards and has been conducted with integrity. All sources and references used in this research have been properly acknowledged. We also acknowledge the assistance of AI tools, specifically ChatGPT, in generating ideas and supporting our research process. We take full responsibility for the content and findings presented in this paper.

9.1 Contribution

Laxa, Joshua - ETL Pipeline, OLAP, OLAP Web application, Functional Testing, Performance Testing, Query Optimization

Resurreccion, Aljirah - Data Warehouse Schema Formulation, OLAP, OLAP Web application, Functional Testing, Conclusion

Tamse, Andrei - ETL Pipeline, OLAP, Query Timer, Abstract, Introduction, and Result and analysis