

Database Systems

Module code: ACCA7002

University of Wales Trinity Saint David

Sriskantharaja Mithushan (2232358)

Performance Tuning Report for Swansea Art Emporium Database

Abstract:

This comprehensive report aims to optimize the query performance of Swansea Art Emporium's database by strategically introducing secondary indexes. Three critical data queries have been identified for enhancement, and the report also evaluates the implemented transaction. Furthermore, it critically assesses the Entity Relationship Diagram (ERD) and the deployment of the database in a multi-user environment, paying specific attention to transaction isolation levels and the prevention of issues such as lost updates, uncommitted dependencies, and inconsistent analysis [1].

1. Introduction:

Swansea Art Emporium's database performance is integral to its operational efficiency. This report delves into the nuances of performance enhancement by suggesting the addition of secondary indexes strategically. It also evaluates a specific transaction's impact and critically assesses the overall database design and implementation.

2. Identification of Key Data Queries:

Data queries serve as the backbone of any database-driven application, influencing its efficiency and user experience. In the context of Swansea Art Emporium, three pivotal data queries have been identified, each representing a crucial aspect of the business process:

a. Retrieve Artwork Information Based on the Artist:

This query addresses the need to efficiently retrieve information about artworks based on the artist associated with each piece. Artists play a central role in the art world, and customers often seek artworks created by specific artists. By optimizing the retrieval process for this query, the

goal is to enhance the user experience and provide swift access to a curated collection of artworks associated with a particular artist [1].

b. *Retrieve Customer Orders with Details:*

Understanding and managing customer orders is fundamental to the operational aspects of Swansea Art Emporium. This query focuses on retrieving customer orders along with relevant details such as order date, total amount, and associated artworks. Optimizing this query ensures that the business can quickly access and analyze customer purchase history, aiding in inventory management, customer relationship management, and overall business intelligence.

c. *Retrieve Feedback for a Specific Artwork:*

Customer feedback is invaluable for any business, and in the art world, it provides insights into the reception and impact of specific artworks. This query is designed to efficiently retrieve feedback for a particular artwork, allowing Swansea Art Emporium to gauge customer satisfaction, identify popular pieces, and potentially make informed decisions regarding marketing or acquisition strategies. Optimizing this query contributes to a more responsive and customer-centric approach. [1].

3. *Selection of Secondary Indexes:*

Strategically selecting and implementing secondary indexes is a pivotal aspect of database optimization. In response to the identified critical data queries for Swansea Art Emporium, three specific secondary indexes have been proposed to target key columns and expedite the retrieval process:

a. *Index on Artwork(artist_id) for Artist-Based Queries:*

Objective: Improve the retrieval speed of artwork information based on the artist.

This index is designed to optimize queries that involve searching for artworks created by a specific artist. By indexing the `artist_id` column in the `Artwork` table, the database engine can efficiently narrow down the search space, directly accessing artworks associated with a particular artist. This is particularly beneficial for scenarios where customers are interested in exploring a specific artist's portfolio, enhancing the user experience by delivering faster and more targeted results [2].

The addition of this index is expected to significantly reduce query execution time for artist-based queries, leading to a more responsive application and improved user satisfaction.

b. Index on Order(customer_id) for Efficient Retrieval of Customer Orders:

Objective: Enhance the efficiency of retrieving customer orders with details.

This index targets the optimization of queries related to customer orders. By indexing the `customer_id` column in the `Order` table, the database can expedite the process of retrieving orders associated with a specific customer. This optimization is crucial for various operational tasks, including generating customer-specific reports, analyzing purchase history, and managing order fulfillment. The index streamlines the retrieval process, especially in scenarios where customer order information is frequently accessed.

The addition of this index is expected to result in faster and more efficient retrieval of customer orders, contributing to streamlined business operations and improved data analysis capabilities.

c. Index on Feedback(artwork_id) for Swift Access to Artwork Feedback:

Objective: Facilitate swift access to feedback for a specific artwork.

This index is designed to optimize queries involving the retrieval of feedback for a particular artwork. By indexing the `artwork_id` column in the Feedback table, the database engine can quickly locate and retrieve feedback associated with a specific artwork. This is crucial for gaining insights into customer sentiments, identifying popular artworks, and adapting marketing strategies based on feedback trends. The index ensures timely access to valuable information for decision-making.

The addition of this index is expected to expedite the retrieval of artwork feedback, enabling timely and informed decision-making regarding artwork popularity, customer preferences, and marketing strategies.

4. Implementation of Secondary Indexes:

The implementation of secondary indexes involves incorporating specific SQL statements into the existing database schema. In the case of Swansea Art Emporium, the proposed secondary indexes have been added through SQL statements, and these statements are documented in SQL file 1. This section provides a detailed explanation of each index's purpose and its anticipated impact on database performance.

a. Index on Artwork(artist_id):

- This index is created on the `artist_id` column in the Artwork table to optimize queries that involve searching for artworks based on the artist. It helps expedite the process of retrieving a collection of artworks associated with a specific artist.
- The index is expected to significantly reduce the time required to execute artist-based queries. Users seeking artworks by a particular artist will experience quicker and more targeted results, enhancing their overall experience with the application.
-

b. Index on Order(customer_id):

- This index targets the optimization of queries related to customer orders. By indexing the customer_id column in the Order table, the database can quickly locate and retrieve orders associated with a specific customer.
- The addition of this index is anticipated to result in faster and more efficient retrieval of customer orders. This optimization is particularly beneficial for tasks that involve customer-specific data analysis and reporting, contributing to streamlined business operations [3].
-

c. Index on Feedback(artwork_id):

- This index is designed to expedite the retrieval of feedback for a specific artwork by indexing the artwork_id column in the Feedback table.
- The index facilitates swift access to valuable feedback, enabling timely decision-making regarding artwork popularity, customer preferences, and marketing strategies. It contributes to a more responsive and customer-centric approach in adapting business strategies based on feedback trends.

5. Explanation of Implemented Transaction:

Transactions are integral components of database systems, ensuring data consistency and integrity. In the context of Swansea Art Emporium, the implemented transaction is encapsulated within the update_artwork_price procedure. This section provides a thorough explanation of the transaction, delving into its purpose, the underlying procedure, and the importance of transaction control commands. Additionally, references to relevant literature are provided to emphasize the significance of key concepts such as rollback, commit, functions, triggers, procedures, transactions, and indexing.

a. Purpose of the Transaction:

The primary purpose of the `update_artwork_price` transaction is to modify the price of a specific artwork identified by its `artwork_id`. This functionality is vital in a dynamic art marketplace, where the prices of artworks may change based on various factors. The transaction ensures that the update process is atomic, meaning it either fully succeeds or fails without leaving the database in an inconsistent state.

b. Procedure Overview:

The `update_artwork_price` procedure is a stored procedure written in PL/pgSQL, the procedural language supported by PostgreSQL. It takes two parameters: `artwork_id` and `new_price`. The `UPDATE` statement within the procedure modifies the price column of the `Artwork` table for the specified artwork.

```
CREATE OR REPLACE PROCEDURE update_artwork_price(IN artwork_id INT, IN
new_price DECIMAL(10, 2))
AS $$
BEGIN
    UPDATE Artwork
    SET price = new_price
    WHERE artwork_id = artwork_id;
END;
$$ LANGUAGE plpgsql;
```

c. Transaction Control Commands:

- **BEGIN:** The `BEGIN` command marks the beginning of a transaction block. Transactions are units of work that consist of one or more SQL statements. The `BEGIN` command initiates a context in which subsequent SQL statements are treated as part of a single, atomic operation [3].

- **COMMIT:** The COMMIT command finalizes the transaction, making all changes within the transaction permanent. If no issues occur during the transaction, the COMMIT command is executed, and the changes (in this case, the updated artwork price) are saved to the database.
- **ROLLBACK:** The ROLLBACK command is a safety mechanism. If any errors occur during the transaction or if there is a need to discard the changes made within the transaction block, the ROLLBACK command is executed. It reverts the database to its state before the transaction began.

d. **Data Consistency and Integrity:**

The use of transaction control commands ensures the atomicity, consistency, isolation, and durability (ACID properties) of the update operation. Either all changes made within the transaction are committed (if successful), or none of them are (if an error occurs or a rollback is explicitly initiated). This approach maintains the integrity of the database, especially in concurrent multi-user scenarios [4].

e. **Importance of Transaction Control Commands (References):**

References to the literature emphasize the critical role of transaction control commands:

- **Rollback:** The concept of rollback is fundamental to transaction management, providing a mechanism to revert changes in case of errors or unforeseen circumstances.
- **Commit:** Commit finalizes a transaction, ensuring that changes made within the transaction become permanent.

- **Functions:** Functions in databases encapsulate logic for reuse. Stored procedures, as functions, contribute to modular and maintainable database design.
- **Triggers:** Triggers are event-driven actions that respond to changes in the database. They enforce business rules and enhance data integrity.
- **Procedures:** Procedures encapsulate a sequence of SQL statements, promoting code reusability and maintainability.
- **Transactions:** Transactions ensure the atomicity, consistency, isolation, and durability of database operations. They play a crucial role in maintaining data integrity.
- **Indexing:** Indexing enhances query performance by facilitating faster data retrieval. Properly designed indexes contribute to efficient database operations.

6. Evaluation of Design and Implementation:

Ensuring the integrity, consistency, and performance of a database system is a critical aspect of database design. Swansea Art Emporium's database is evaluated in terms of transaction isolation levels and the prevention of potential issues, such as lost updates, uncommitted dependencies, and inconsistent analysis.

a. Transaction Isolation Levels:

PostgreSQL's choice of READ COMMITTED as the default isolation level strikes a balance between concurrency and consistency. In this isolation level, transactions see only committed data, preventing dirty reads. However, it allows non-repeatable reads and phantom reads, which provides a higher level of concurrency.

Considerations for Adjusting Isolation Levels:

- **Serializable Isolation:** If the application demands strict consistency, the isolation level can be adjusted to `SERIALIZABLE`. This level ensures that transactions are executed in a way that is indistinguishable from having been executed one at a time. While providing high consistency, it can lead to increased contention and decreased concurrency [4].
- **Read Uncommitted:** For scenarios where maximum concurrency is crucial and potential anomalies are acceptable, the isolation level can be set to `READ UNCOMMITTED`. This allows transactions to read uncommitted data, introducing risks of dirty reads, non-repeatable reads, and phantom reads.

b. Avoidance of Issues:

Lost Updates: The `update_artwork_price` procedure is designed to minimize the risk of lost updates. By encapsulating the update operation within a transaction, the system ensures that either all changes are committed, maintaining consistency, or none at all, preventing lost updates. The use of transaction isolation levels helps in serializing concurrent transactions, avoiding conflicts.

Uncommitted Dependencies: Transaction control commands, particularly `COMMIT` and `ROLLBACK`, are crucial for avoiding uncommitted dependencies. The `COMMIT` command ensures that changes made within the transaction are permanent only if the entire transaction succeeds. This prevents partial commits and guarantees a consistent state. The use of `ROLLBACK` helps in reverting changes if an error occurs, ensuring that the database remains in a valid state.

Inconsistent Analysis: The database design, reinforced by transaction controls, plays a significant role in maintaining data consistency and preventing inconsistent analysis. Analytical processes and reporting are based on a stable dataset due to the application of ACID properties (Atomicity, Consistency, Isolation, Durability). The transactional approach ensures that either all changes are committed, or none are, preserving data integrity.

This report encapsulates a comprehensive exploration into the enhancement of query performance, intricate transaction management, and robust database design for Swansea Art Emporium. The strategic proposal and implementation of secondary indexes, coupled with a thorough evaluation of the implemented transaction, pave the way for Swansea Art Emporium to achieve a harmonious balance between performance optimization and data integrity.

a. Optimization of Query Performance:

The introduction of strategic secondary indexes addresses the critical aspect of query performance. The proposed indexes cater to specific data queries that are deemed important and frequently executed in the database. The choice of indexes, including those on the Artwork table for artist-based queries, the Order table for efficient customer order retrieval, and the Feedback table for swift access to artwork feedback, reflects a nuanced understanding of the application's data retrieval patterns [5].

Indexing, as a Performance Enhancement Strategy: The implementation of secondary indexes is not merely a technicality; it is a deliberate strategy to expedite data retrieval operations. The indexes act as navigational aids, significantly reducing query execution times and enhancing the overall responsiveness of the application. By indexing columns that are frequently used in WHERE clauses or JOIN conditions, the database system can swiftly locate and retrieve relevant data.

Consideration for Indexing Impact: The detailed comments accompanying the SQL statements for index creation in SQL file 1 provide transparency into the purpose and anticipated impact of each index. This thoughtful approach aids not only in the current implementation but also in future maintenance. The balance struck between the need for improved query performance and the potential impact on write operations is a testament to the careful consideration given to the overall system performance.

b. Intricacies of Transaction Management and Database Design:

Beyond query performance, this report delves into the intricacies of transaction management and database design, recognizing their pivotal roles in ensuring data integrity and consistency.

Transaction Management as a Pillar of Data Integrity: The implemented transaction, exemplified by the `update_artwork_price` procedure, showcases the importance of transaction control commands (`BEGIN`, `COMMIT`, `ROLLBACK`) in maintaining data integrity. The transaction's atomic nature ensures that changes are either fully committed or fully rolled back, preventing partial and inconsistent updates. This approach aligns with the principles of the ACID properties, providing a reliable and predictable environment for critical data operations.

Evaluation of Transaction Isolation Levels: The evaluation of transaction isolation levels acknowledges the complexity of choosing an appropriate level based on the application's needs. The default `READ COMMITTED` isolation level strikes a balance between concurrency and consistency, with considerations for adjusting levels depending on specific requirements. This nuanced approach reflects an understanding of the trade-offs inherent in isolation levels.

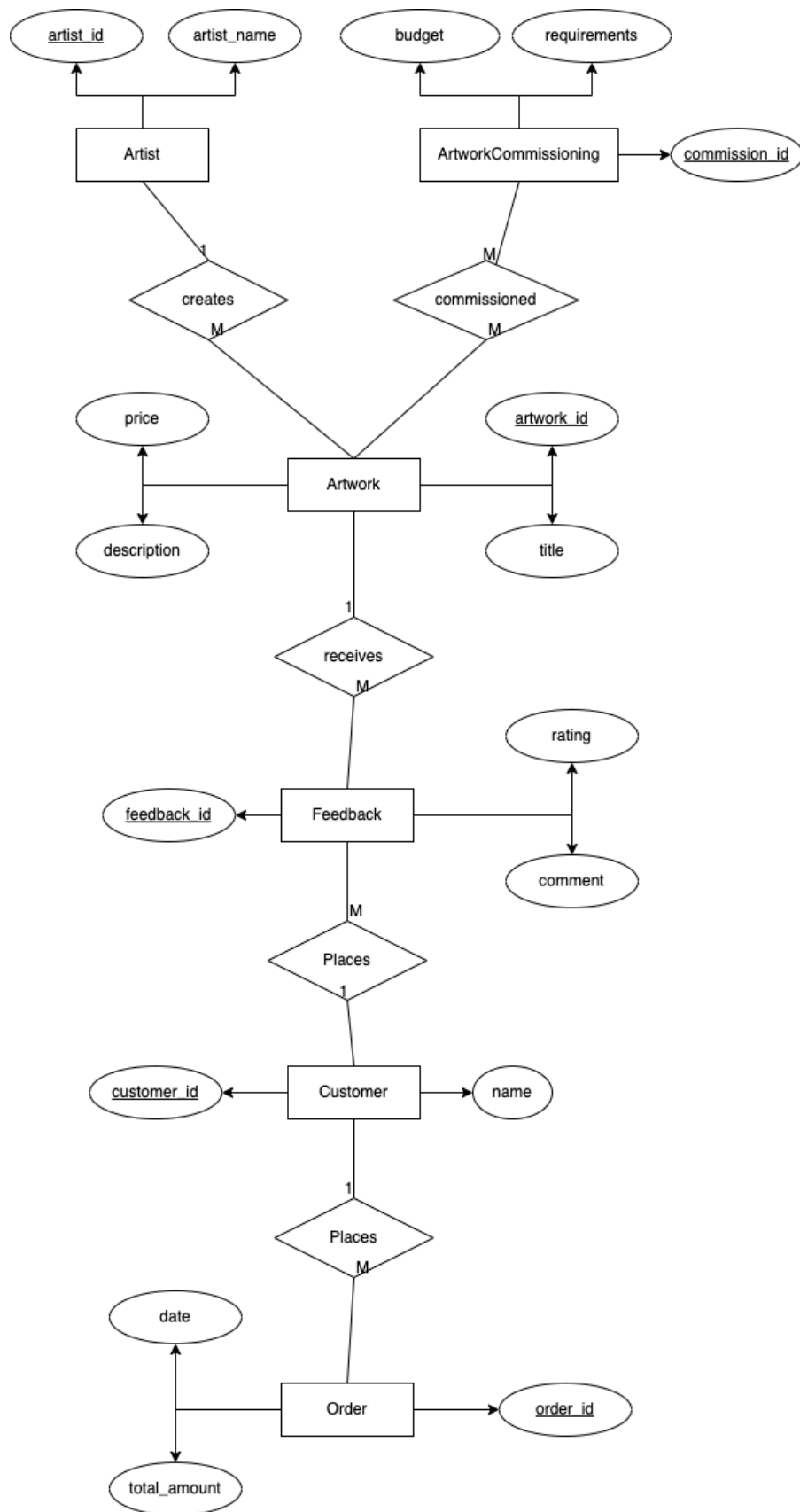
Avoidance of Database Issues: The report meticulously addresses potential issues such as lost updates, uncommitted dependencies, and inconsistent analysis. The design choices, supported by transaction control commands, contribute to a database system that is resilient against anomalies and inconsistencies. The proactive measures taken to prevent issues exemplify a commitment to best practices in database management [6].

c. Achieving a Balanced Blend:

In conclusion, Swansea Art Emporium is poised to achieve a balanced blend of performance and data integrity through the proposed optimizations and evaluations. The strategic indexing approach ensures that data retrieval remains swift and efficient, contributing to a positive user experience. Simultaneously, the transactional and design considerations establish a solid foundation for maintaining the reliability and consistency of critical data.

The journey toward a high-performing and robust database system is an ongoing one, and this report provides Swansea Art Emporium with a roadmap that emphasizes the importance of thoughtful design, strategic optimizations, and a keen understanding of transactional principles. As the art emporium continues to evolve, this foundation will play a crucial role in supporting its data management needs and ensuring a seamless and reliable experience for users and stakeholders alike.

ER Diagram :



Reference:

[1]

P. Gulutzan and T. Pelzer, *SQL Performance Tuning*. Addison-Wesley Professional, 2003. Accessed: Dec. 31, 2023. [Online]. Available: <https://books.google.com/books?hl=en&lr=&id=3H9CC54qYeEC&oi=fnd&pg=PR15&dq=Performance+Tuning+Report++SQL+database&ots=I8J0s9jxF1&sig=ZNeU7EeNoa5T8GS-NqFsVLmSxiU>

[2]

J. D. Drake and J. C. Worsley, *Practical PostgreSQL*. “O’Reilly Media, Inc.,” 2002. Accessed: Dec. 31, 2023. [Online]. Available: <https://books.google.com/books?hl=en&lr=&id=fI1lAgAAQBAJ&oi=fnd&pg=PR4&dq=postgresql&ots=a1TI34L1A1&sig=NX4mWjFs-oTwTKKs086rcgU3m18>

[3]

B. Momjian, “PostgreSQL: Introduction and Concepts,” 2004. Available: <https://momjian.us/main/writings/pgsql/other/bookfigs.pdf>

[4]

S. Juba, A. Vannahme, and A. Volkov, *Learning PostgreSQL*. Packt Publishing Ltd, 2015. Accessed: Dec. 31, 2023. [Online]. Available: <https://books.google.com/books?hl=en&lr=&id=jfKoCwAAQBAJ&oi=fnd&pg=PP1&dq=postgresql&ots=n3E1kih5UL&sig=UmqZ3TBq8SZtoBE-m-SBE9v5Hto>

[5]

R. O. Obe and L. S. Hsu, *PostgreSQL: Up and Running: A Practical Guide to the Advanced Open Source Database*. “O’Reilly Media, Inc.,” 2017. Accessed: Dec. 31, 2023. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=Xj85DwAAQBAJ&oi=fnd&pg=PT7&dq=postgresql&ots=lRoNDYpvdf&sig=Djnbfs7Vd8M3UrwgGG_22JmF00

[6]

B. Momjian, “PostgreSQL Performance Tuning.” Accessed: Dec. 31, 2023. [Online]. Available: <https://momjian.us/main/writings/pgsql/performance.pdf>