Database Optimization

Research document

Contents

INTRODUCTION 2

METHODOLOGY 2

RESEARCH RESULTS 3

How can indexing be used to improve MySQL performance? 3

What role do stored procedures and views play in MySQL optimization, and how can these features be used to improve query performance? 4

How can queries be optimized to improve execution time and improve performance? 5

How can table structures be best optimized to improve database performance? 6

INTRODUCTION

The purpose of this research is to look into database optimization with the purpose of creating a better database structure and better querying in order to serve as a quality assurance guideline for the products of our group project. This research will be done using DAF's current database structure however it will also focus on how it can potentially be improved in the future.

METHODOLOGY

Main research question

What are the most effective techniques for optimizing MySQL database performance?

Sub-questions

- How can indexing be used to improve MySQL performance?
    - Best Good and bad practices, Design patterns, Literature study
- What role do stored procedures and views play in MySQL optimization, and how can these features be used to improve query performance?
    - Data analysis, Benchmark test, Best Good and bad practices
- How can queries be optimized to improve execution time and improve performance?
    - Benchmark test, Best Good and bad practices
- How can table structures be best optimized to improve database performance?
    - Benchmark test, Best Good and bad practices, Data analysis

RESEARCH RESULTS

How can indexing be used to improve MySQL performance?

Indexing is a technique used to optimize database performance by enabling faster data retrieval. In MySQL, an index is a data structure that allows for efficient data lookup based on specific columns or combinations of columns. When a query is executed, the index is used to quickly locate the relevant data instead of scanning the entire table.

- Indexing can improve MySQL performance by ensuring faster query execution - By using an index to find the relevant data, the database can avoid full table scans and retrieve only the necessary data.
- Indexes can be used to speed up queries that sort results in a certain way such as queries with ORDER BY and GROUP BY clauses.
- Indexing improves concurrency in a database – by indexing, we can ensure that MySQL can handle multiple queries simultaneously without locking the entire table, which can improve concurrency and reduce wait times for queries.
- Join query performance is also improved by indexing as it allows MySQL to match rows from different tables faster.

The following command can be used to create indexes in MySQL:

```
CREATE INDEX index_name ON table_name (column_name);
```

Even though indexing is a very powerful tool it is not a good idea to use indexes on all table columns as excessive indexing can slow down query performance. It is important to take into account which are the desired/most used columns that queries will need and then define a plan for indexing a table using that information.

What role do stored procedures and views play in MySQL optimization, and how can these features be used to improve query performance?

Stored procedures

A stored procedure is a group of SQL statements that has been created and stored into the database. It allows for the use of input parameters in order to improve flexibility. Stored procedures are precompiled and can be executed on demand. This allows for an increase in performance by reducing network traffic between the client and the server and reducing the time it takes to parse and compile queries. Furthermore, stored procedures can be used to more easily perform tasks that would usually require multiple queries.

A stored procedure can be created using the CREATE PROCEDURE keyword and follows the following syntax:

```
CREATE PROCEDURE procedure_name ( parameter_1, parameter_2, parameter_3, … )

BEGIN

sql_statement

END;
```

In order to execute a stored procedure the CALL keyword is used followed by the name of the stored procedure like this: `CALL procedure_name();`. This will automatically execute the SQL statement stored in the procedure.

Views

Views are virtual tables based on the result of a SELECT statement query. Their main purpose is to provide an interface to underlying data. When a view is created it acts as a table that is made of the columns from the select statement that was used to create the view. This allows for the execution of simpler queries that only need to target one table(or view).

Views can be created using the following SQL statement syntax:

```
CREATE VIEW view_name AS

SELECT column1, column2, ...

FROM table_name

WHERE condition;
```

This creates the view and allows for future SQL select statements to target that view just like a normal table:

```
SELECT * FROM view_name;
```

How can queries be optimized to improve execution time and improve performance?

A lot of factors take part in query execution time and therefore they should all be accounted for to ensure the fastest data querying.

Some ways to optimize query performance include:

- Only selecting the columns that are needed – If not all columns are needed SELECT * is not the best choice. Instead, it is better to manually input only columns that are wanted.
- Using joins efficiently – It is important to use the proper JOIN command when joining two tables to exclude unnecessary data. This will help query performance as needless data is not being returned by the query. It is also important to avoid using joins if it can be helped as it is an inherently performance-heavy command. Furthermore, joining queries should be done using the ON condition and only on indexed columns.
- Using subqueries properly – Subqueries should be used only if needed and they should also be properly optimized themselves as they are a performance-heavy element of SQL querying. Preferably, use JOIN when possible.
- Minimizing data retrieval – functions such as COUNT(), MAX(), SUM() can be used to minimize the data that is retrieved instead of retrieving all rows. For example the MAX() function combined with a timestamp row can be used to only get the latest row in a table.
- It is better to avoid using UNION and DISTINCT if they are not needed as they introduce extra sorting into the query processes. If using the UNION command can't be avoided it is better to use UNION ALL as it does not remove duplicates and therefore reducing the amount of processing needed.
- Stored Procedures are a great idea for queries that are going to be executed a lot as they are precompiled and cached which improves execution time.

How can table structures be best optimized to improve database performance?

Database normalization is the process of organizing data in a database in a way that reduces redundancy and dependency and improves query efficiency it is usually achieved by reducing or normalizing large tables into smaller tables and then connecting them using relationships. The typical process of database normalization involves a series of steps called normal forms. There are 5 normal forms however the most used are the first 3 normal forms.

1. The First Normal Form (1NF) states that each table cell in a database table should contain only a single value and each row has to be unique. This can be achieved by using a primary key which should be unique for each row. This ensures that the second statement is correct as each row is unique at least based on it's Primary Key value.
2. The Second Normal Form (2NF) states that each non-key attribute must depend on the entire primary key rather than just a part of it. In order to put a table in Second Normal Form all non-key columns should be moved to a new table and the two tables should be connected with each other on the parts of their primary keys that are identical.
3. The Third Normal Form (3NF) states that each non-key column must not be dependent on another non-key column and instead it should depend on the whole primary key and nothing else.
4. Fourth Normal Form (4NF) states that a table should not have multivalued dependencies unless they are multivalued dependencies on the key.
5. Fifth Normal Form (5NF) states that a table should not be able to be described as a join between two other tables. This means that a single table should not be able to be separated into different tables without the loss of data.

SOURCES

1. *MySQL performance tuning tips to optimize database*. The Official Cloudways Blog. (2021, July 7). Retrieved March 27, 2023, from https://www.cloudways.com/blog/mysql-performance-tuning/
2. Alex The Analyst. (2021, March 16). *Advanced SQL Tutorial | Stored Procedures + Use Cases* [Video]. YouTube. https://www.youtube.com/watch?v=NrBJmtD0kEw
3. Upadhyay, N. (2021, September 7). *Learn MySQL: The Basics of MySQL Stored Procedures*. SQL Shack - Articles About Database Auditing, Server Performance, Data Recovery, and More. https://www.sqlshack.com/learn-mysql-the-basics-of-mysql-stored-procedures/
4. *MySQL :: MySQL Documentation*. (n.d.). https://dev.mysql.com/doc/
5. Decomplexify. (2021, November 21). *Learn Database Normalization - 1NF, 2NF, 3NF, 4NF, 5NF* [Video]. YouTube. https://www.youtube.com/watch?v=GFQaEYEc8_8
6. Peterson, R. (2023, February 11). *What is Normalization in DBMS (SQL)? 1NF, 2NF, 3NF Example* Guru99. https://www.guru99.com/database-normalization.html