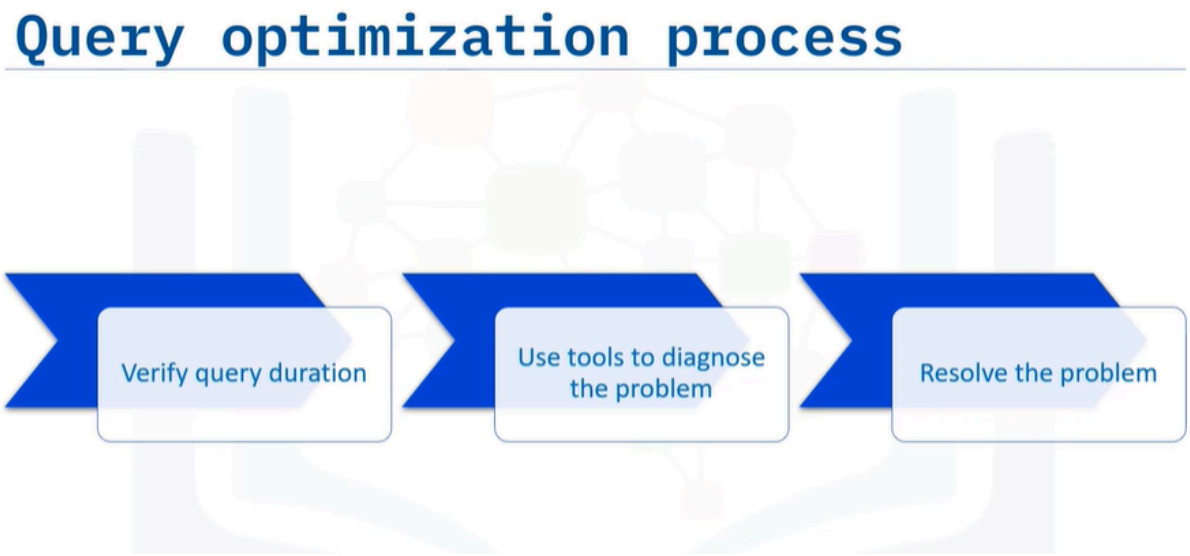
Optimizing Databases

Sure! Let's talk about the query optimization process in simple terms.

Query optimization is like finding the fastest route to your destination when driving. Imagine you want to get to a friend's house, but there are many roads you could take. Some roads might be busy, while others are clear. In the same way, when you ask a database to retrieve information (like asking for a list of your favorite movies), the database has to decide the best way to get that information. It looks at different paths (or methods) it can take to get the data and chooses the quickest one.

To help with this, databases use something called a "query execution plan." Think of it as a map that shows the steps the database will take to find the information you need. Just like you might check traffic conditions before you leave, database administrators can use tools to see how long a query takes and make adjustments if it's running too slowly. This way, they can ensure that the database runs efficiently and provides results quickly.



- If you're receiving feedback that it's taking too long to get the results of a query There's a process you can follow to evaluate and resolve the problem

Begin by verifying how long the query takes Is it an intermittent issue, a permanent problem, or user perception? Once you have verified the issue You can use RDBMS or third-party tools to diagnose the problem Based on the diagnosis, you can take steps to resolve the problem.

Determine query duration

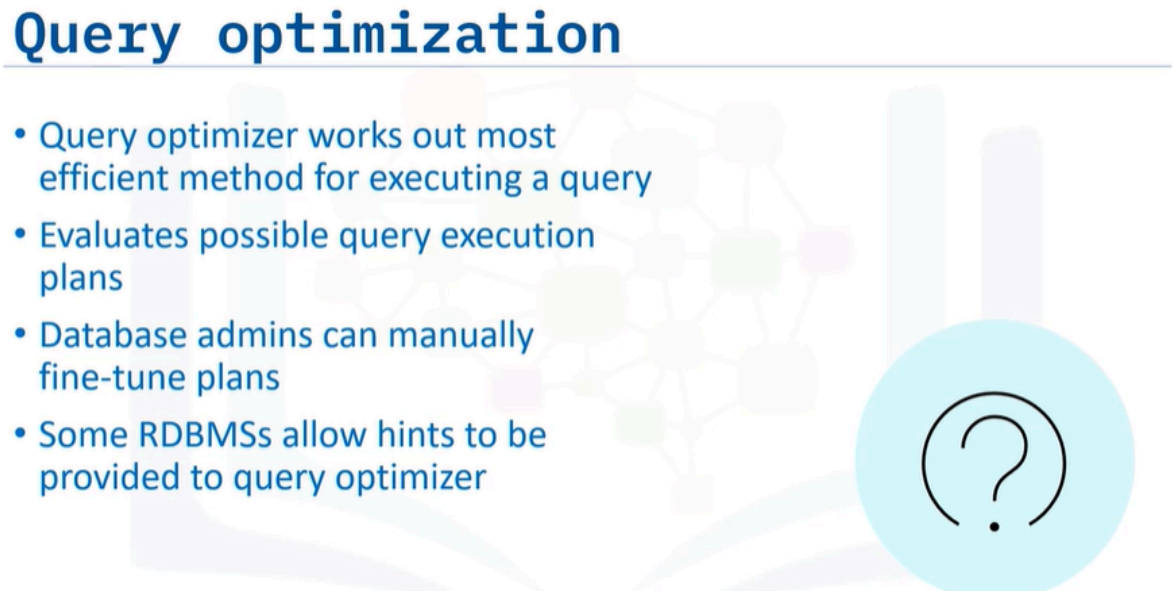
- Specialized logs
  - MySQL slow query log
- Benchmarking tools
  - `\timing`
  - shell time command

Determine query duration

Does duration include network latency?

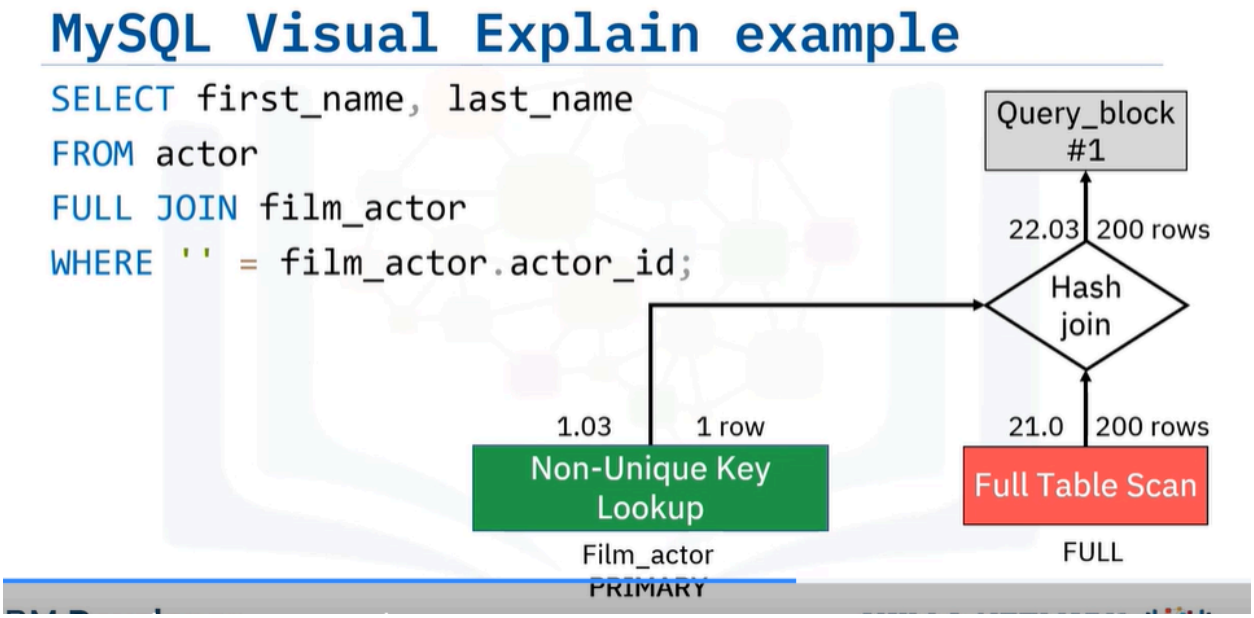
- You can use several mechanisms and tools to determine the actual time a query is taking These tools include specialized logs like the MySQL slow query log You should also benchmark expected query performance yourself Using tools like timing in PostgreSQL Or using the shell time command when executing a query When determining the duration of a query You must recognize what the tool you are using is measuring Is it measuring only server-side computational effort? Or does it include network latency as the command and results travel between client and server? For example, in PostgreSQL The time that timing returns includes the network latency If you're connecting to a remote server.

- A query execution plan or query plan Is the series of steps used to access data when running SQL statements An RDBMS will often provide several methods for returning the detail of a query execution plan Some offer tools that create a graphical representation of their query plans While others allow a distinct mode to be set on the connection Which causes the RDBMS to return a text-based description of their query plans And lastly, some RDBMSs allow you to obtain a query plan By using an EXPLAIN statement to query a virtual database table after running the query .

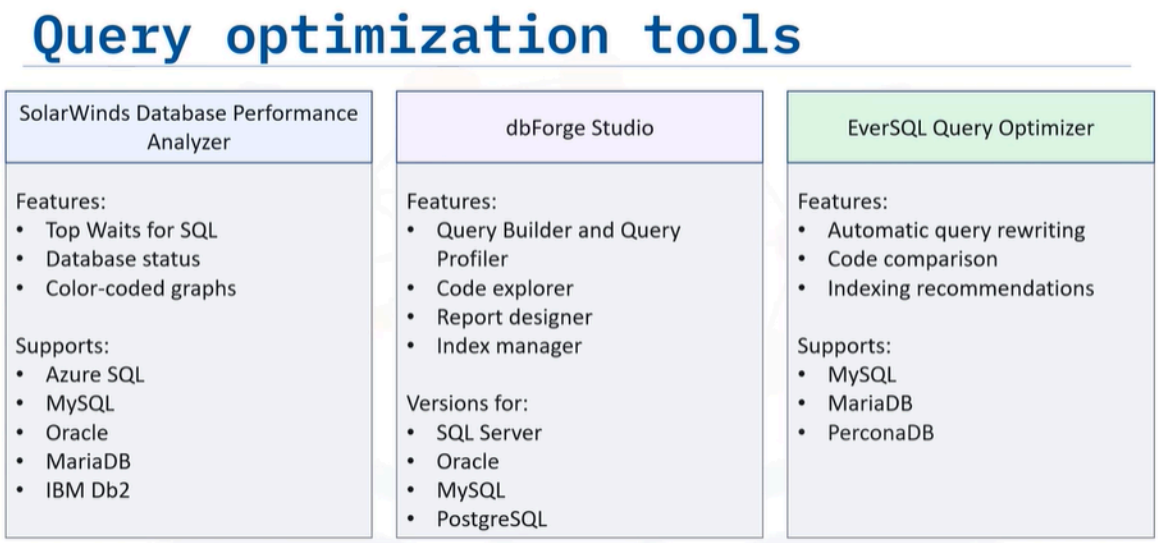


- Most RDBMSs have a query optimization feature That uses a query optimizer tool to calculate the most efficient method For executing a query by evaluating all the available query execution plans When a query gets submitted to the database The query optimizer evaluates the various possible query execution plans And returns what it determines to be the best choice However, query optimizers can be fallible So database admins will sometimes need to manually inspect and fine-tune the plans produced by the query optimizer To get optimum query performance Some RDBMSs allow you to provide hints to the query optimizer A hint is an additional component to the SQL statement That informs the database engine about how it wants it to execute a query Such as instructing the database engine to use an index when executing the query Even though the query optimizer might have decided not to All flavors of RDBMSs, such as DB2, MySQL, and PostgreSQL.

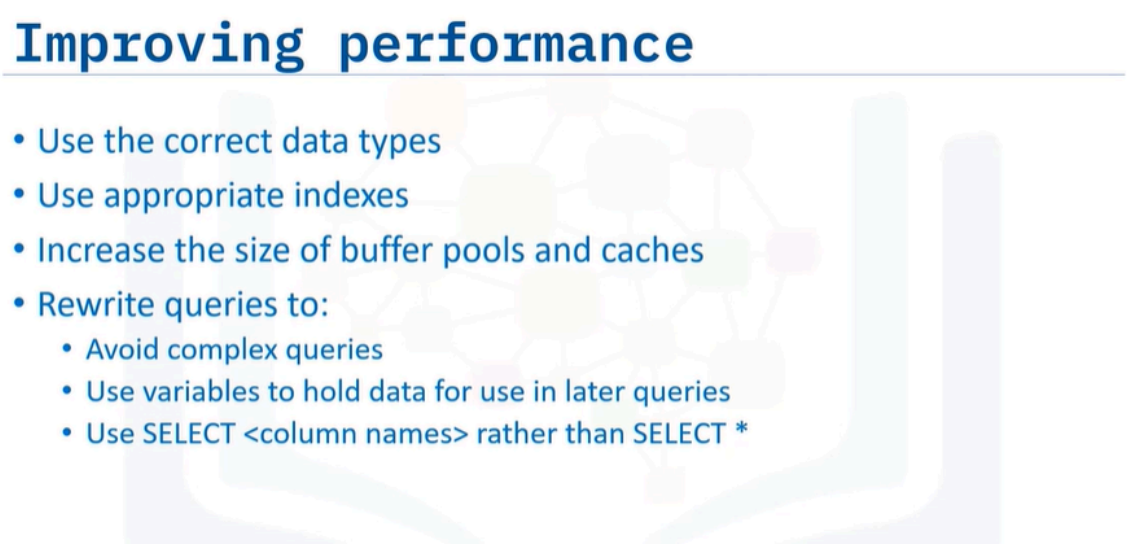
- Have an EXPLAIN statement that you can use to show a text-based representation Of the details of a query execution plan for a statement Including the processes that occur and in what order they occur An EXPLAIN statement can be a good way to swiftly cure slow-running queries Some RDBMSs also provide a graphical version of the EXPLAIN statement For example, DB2's Visual Explain uses information from a number of sources To enable you to view the access plan for explained SQL or XQuery statements as a graph You can use the information available from the access plan graph To tune your queries for better performance And for PostgreSQL systems, the pgAdmin utility provides a graphical EXPLAIN plan feature Although this is not an entire substitute for EXPLAIN or EXPLAIN ANALYZE text plans It does offer a fast and simple method for viewing plans for additional analysis For MySQL systems, the MySQL Workbench provides a Visual Explain plan That produces and presents a visual representation of the MySQL EXPLAIN statement MySQL Workbench provides all of the EXPLAIN formats for executed queries Including the standard format, the raw extended JSON format, and the Visual Query plan This is an example of MySQL Workbench Visual EXPLAIN query plan for a given SQL statement.



- The order of execution in a Visual Explain diagram is bottom to top and left to right The diagram shows an overview of the query as it is processed In Visual Explain diagrams, standard boxes represent tables, and diamonds represent joins The text in Visual Explain diagrams is also important to understand Standard text below boxes identifies the table name Bold text below boxes shows the key or index used The number to the top right of a box or diamond shows the number of rows produced And the number to the top left of a box or diamond shows the relative cost In addition to the tools supplied in the RDBMS.



- the relative cost In addition to the tools supplied in the RDBMS There are many third-party query optimization tools to choose from SolarWinds Database Performance Analyzer, which features TopWeights for SQL, which displays network state and performance Database status, including wait time, tuning, CPU, memory, and disk statistics And color-coded graphs for each information category SolarWinds Database Performance Analyzer supports Azure SQL, MySQL, Oracle, MariaDB, and IBM DB2.
- dbForge Studio, which features Query Builder and Query Profiler Which are query optimization tools for tuning queries and investigating query performance issues Code Explorer for inspecting or writing query code Report Designer for sending performance issue feedback to your team And Index Manager for resolving index fragmentation.
- dbForge Studio has versions for SQL Server, Oracle, MySQL, and PostgreSQL And finally, we have EverSQL Query Optimizer, which features Automatic query rewriting, code comparison and change notes after query rewriting And indexing recommendations for improving query speed EverSQL Query Optimizer supports MySQL, MariaDB, and PerconaDB.



- Query plans and explain can help identify the cause behind a performance issue But how can you resolve it? Understanding your data and how it behaves is key for making good query design decisions Including using the correct data types, when and where to use indexes And how to size your buffer pools and caches Some query performance tools will also help you understand how to write efficient queries Techniques you can use include Avoiding complex queries that contain many subqueries, especially when using joins Using variables to hold data for use in later queries And being specific about the data you select from tables Select named columns, rather than using SELECT asterisk.

Summary

In this video, you learned that:

- Determine query duration, use tools to diagnose the problem, and finally resolve the problem
- Query execution plans show details of the steps used to access data when running query statements
- EXPLAIN statements and visual explain tools show query execution plans
- Query optimization tools help DBAs determine the most efficient method for executing a query
- Query performance can be improved by using good query design