

MySQL Community Edition 8.2

CSCI 6370 Database Management
Department of Computer Science
University of Georgia

Spring 2024

Presentation By

Aravind Mandiga
Hari Priya Muppidi
Karishma Hegde
Lahari Basavaraju
Sanket Veer



What is MySQL?

- MySQL is an open-source Relational Database Management System (RDBMS) that allows users to organize, manage, and retrieve data efficiently.
- Originally developed by a Swedish company, MySQL AB, initially released in 1995.
- Crucial RDBMS in the industry as it's powerful and reliable.
- Adheres to the SQL (Structured Query Language) standard.
- Popularity and usage statistics:
 - **Community Support:** MySQL has a large and active community of users, developers, and contributors who contribute to its growth and improvement.
 - **Web Applications:** It is extensively used in web development, being a common choice for database management in conjunction with popular web development stacks like LAMP (Linux, Apache, MySQL, PHP/Python/Perl).
 - **Embedded Systems:** MySQL is often utilized in embedded systems and software applications where a compact and reliable database solution is required.
 - **Enterprise Solutions:** Many enterprises leverage MySQL for managing data in their applications, benefiting from its performance and scalability features.

MySQL Architecture

- Client-Server Architecture.
- Two main components: **MySQL Server** and **Client**
- **Storage Engine**: In charge of storage, retrieval, and management of data in the database.

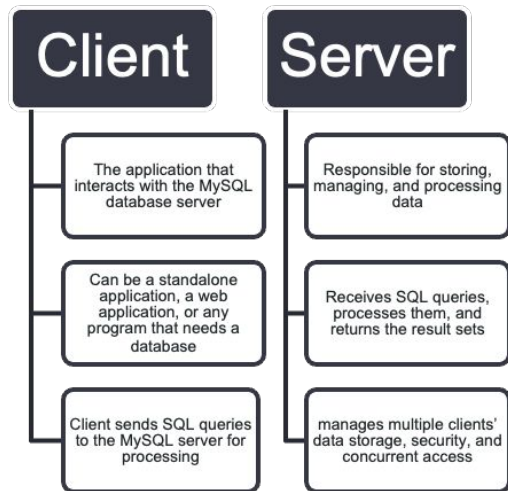


fig 1: client & server

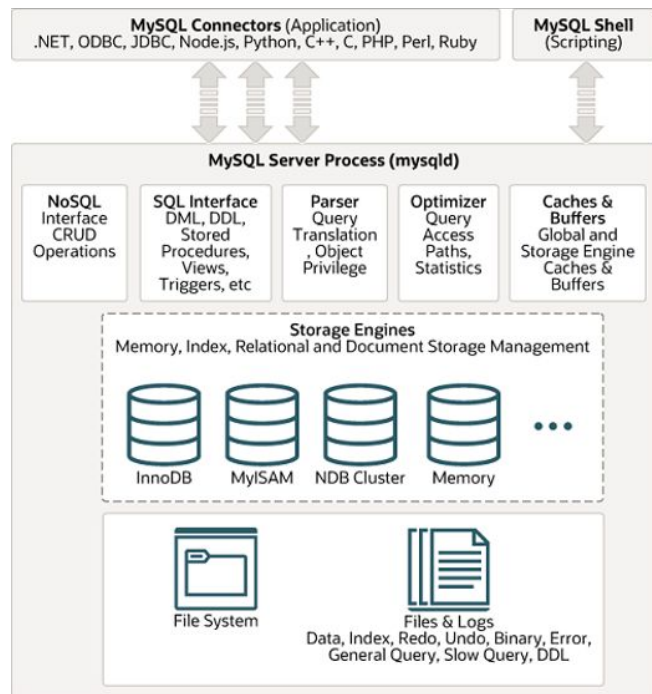


fig 2: MySQL Architecture

Introduction to MySQL Community Edition 8.2

- This version attempts to give customers a better and more effective database administration experience by improving the overall performance, security, and scalability of the MySQL database.
- Key Features and Improvements in 8.2
 - Improvements in performance, security, and scalability.
 - Includes support for Common Table Expressions (CTEs), simplifying complex queries and enhancing query readability.
 - Introduces support for atomic Data Definition Language (DDL) statements.
 - Changes in syntax or functionality compared to previous versions.

```
BEGIN;  
ALTER TABLE employees ADD COLUMN new_column INT;  
COMMIT;
```

fig 3: example for Atomic DDL

SQL

- Structured Query Language (SQL), is a programming language designed for managing and manipulating data stored in relational databases.
- **SQL Query:** a request for information from a database.
- **MySQL Installation:**
 - MySQL Community Server and Workbench (suitable versions) or MySQL Installer.
 - Configure the MySQL Server.
- **SQL Constraints:** Specify rules for data in a table.
 - NOT NULL – Ensures that a column cannot have a NULL value.
 - UNIQUE – Ensures that all values in a column are different.
 - PRIMARY KEY – Uniquely identifies each row in a table.
 - FOREIGN KEY – Establishes a link between tables.
 - CHECK – Ensures that the values in a column satisfies a specific condition.
 - DEFAULT – Sets a default value for a column if no value is specified.
 - CREATE INDEX – Used to create and retrieve data from the database very quickly.

Types of SQL Commands

- Data Definition Language (DDL)
 - Create database and table structure
 - Commands: CREATE, ALTER, DROP, TRUNCATE, COMMENT, RENAME
- Data Manipulation Language (DML)
 - Record/Row related operations
 - Commands: SELECT, INSERT, DELETE, UPDATE)
- Data Control Language (DCL)
 - Manipulates permissions or access rights to the tables
 - Commands: GRANT & REVOKE
- Transactional Control Language
 - Control transactions
 - Commands: COMMIT, ROLLBACK, SAVEPOINT

Basic SQL Statements & Operators

- CREATE DATABASE <database_name>;
- USE <database_name>;
- SHOW DATABASES; or SHOW TABLES;
- DESCRIBE <table_name>; OR DESC <table_name>;
- Aggregate Functions
- LIKE, WILDCARDS, ALIASES, CASE

ALTER	UPDATE
DDL Command	DML Command
It is used to add or delete or change the name of a column	It is used to change/modify the values(s) in particular column(s)
ALTER TABLE <table_name> ADD/MODIFY/CHANGE <column_name> <datatype>(size)	UPDATE <table_name> SET <column_name> = <value> WHERE <column_name> = <value>

DROP	DELETE
DDL Command	DML Command
It is used to delete the table or database permanently.	It is used to delete a particular record(s) from the table.
DROP TABLE <table_name>	DELETE FROM <table_name> WHERE <condition>

Advanced SQL Techniques

- Joins
- Subqueries
- Views
- Indexing
- Stored Procedures
- Triggers

Join

- A JOIN clause is used to combine rows from two or more tables, based on a related column between them.
- Types of JOINS:
 - INNER JOIN
 - CROSS JOIN
 - LEFT JOIN
 - RIGHT JOIN
 - SELF JOIN
 - FULL JOIN

INNER JOIN & CROSS JOIN

INNER JOIN

The `INNER JOIN` keyword selects records that have matching values in both tables.

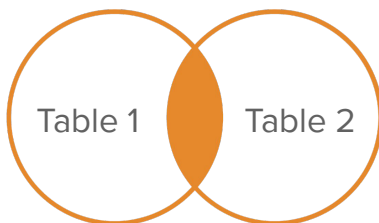


fig 4: inner join

CROSS JOIN

- A `CROSS JOIN` produces a Cartesian product of the two tables it joins. It joins every row of the first table with every row of the second table.
- If the first table has X rows and the second table has Y rows, the result set will have $X * Y$ rows, assuming there are no `WHERE` clause or filters applied.

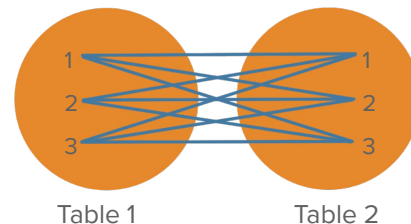


fig 5: cross join

LEFT JOIN & RIGHT JOIN

LEFT JOIN

The LEFT JOIN keyword returns all records from the left table (table1), and the matching records (if any) from the right table (table2).

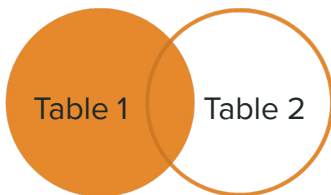


fig 6: left join

RIGHT JOIN

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records (if any) from the left table (table1).

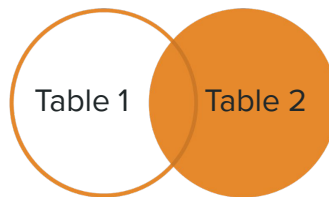


fig 7: right join

SELF JOIN & FULL JOIN

SELF JOIN

A self join is a regular join, but the table is joined with itself.

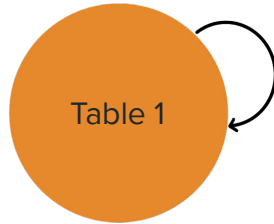


fig 8: self join

FULL JOIN

- A FULL JOIN (FULL OUTER JOIN) returns all records when there is a match in either left (table1) or right (table2) table records.
- IT combines the results of both LEFT JOIN and RIGHT JOIN.
- Rows from both tables that do not meet the join condition are also included in the result set, with NULLs in place of columns from the table where there is no match.

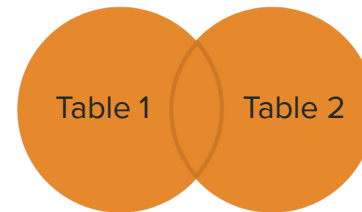


fig 9: full join

Subqueries

- Also known as inner queries or nested queries
- Allows operations using multiple select statements across different levels of query depth.
- They can be used in various contexts including SELECT, INSERT, UPDATE, and DELETE statements, as well as in the FROM, WHERE, and HAVING clauses of a query.

Subquery in a WHERE Clause

Scenario:

To find the names of all employees who work in the 'Sales' department.

```
SELECT EmployeeName, DepartmentID
FROM Employees
WHERE DepartmentID IN
    (SELECT DepartmentID
     FROM Departments
     WHERE DepartmentName = 'Sales');
```

Subquery in a FROM Clause

Scenario:

We want to find the Total Amount of Sales in a particular day.

```
SELECT SaleDate, TotalAmount
FROM (
    SELECT SaleDate, SUM(Amount) AS
TotalAmount
    FROM Sales
    GROUP BY SaleDate
) AS DailySales;
```

Views

- A virtual table resulting from a predefined SQL query.
- Stored query accessible as a virtual table composed of the result set of a database query.
- Does not physically store data – Dynamically retrieves data from one or more tables at the time of access.

```
CREATE VIEW view_name AS  
SELECT column1, column2, ...  
FROM table_name  
WHERE condition;
```


Views: Example

Scenario:


- We have a table called Employees.
- To create a view that shows only the employees from the 'Sales' department

```
CREATE VIEW SalesEmployees AS  
SELECT EmployeeID, EmployeeName,  
Department  
FROM Employees  
WHERE Department = 'Sales';
```

- To retrieve the data from the 'SalesEmployees' view

```
SELECT * FROM SalesEmployees;
```

Indexing

- Used to find values within a specific column more quickly.
- It is a data structure (B-Tree, R-Tree, Hash Index) that improves the speed of operations on a database table.
-  **NOTE:** MySQL normally searches sequentially through a column. The longer the column, the more expensive the operation is.
- Indexes reduce the efficiency of INSERT and UPDATE operations on a table.
- Indexes can be created on one or more columns, providing the basis for both rapid random lookups and efficient ordering of access to records

Indexing Syntax

- We can create indexes on existing tables using CREATE INDEX statement or ALTER... ADD command.
- Single-Column Index
 - CREATE INDEX *index_name* ON *table_name*(*column_name*);
 - ALTER TABLE *table_name* ADD INDEX *index_name* (*column_name*);
- Multi-Column Index
 - CREATE INDEX *index_name* ON *table_name*(*column_list*);
 - ALTER TABLE *table_name* ADD INDEX *index_name* (*column_list*);
- We can see created indexes using SHOW INDEXES FROM *table_name* statement.

Stored Procedures

- A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.
- You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.
- To create a stored procedure, use CREATE PROCEDURE command.

```
CREATE PROCEDURE procedure_name (IN parameter DATATYPE(LENGTH), OUT  
return_variable DATATYPE(LENGTH))  
BEGIN  
<SQL Code>  
END
```

- Executing the Stored Procedure:
`<procedure_name>(<parameter_list>, @return_variable)`

Creating and Executing Stored Procedures

```
mysql> delimiter //
mysql> CREATE PROCEDURE citycount (IN country CHAR(3), OUT cities INT)
      BEGIN
          SELECT COUNT(*) INTO cities FROM world.city
          WHERE CountryCode = country;
      END//
mysql> delimiter ;
mysql> CALL citycount('JPN', @cities); -- cities in Japan
mysql> SELECT @cities;
```

Triggers

- A trigger is a response to an event.
- In MySQL, a trigger is a special stored procedure that is executed automatically (without being called explicitly like regular stored procedures) whenever an event is performed.
- These events include statements like INSERT, UPDATE and DELETE etc.
- Applications:
 - Perform checks of values to be inserted into a table
 - Perform calculations on values involved in an update.
- A trigger can be set to activate either before or after the trigger event.

CREATE and DROP Triggers

- Create Trigger

```
CREATE TRIGGER trigger_name
    {BEFORE | AFTER} {INSERT | UPDATE |
    DELETE}
    ON table_name
    FOR EACH ROW
    BEGIN
        -- Trigger body (SQL statements)
    END;
```

- Drop Trigger

```
DROP TRIGGER [IF EXISTS]
[schema_name.]trigger_name;
```

Security in MySQL

- Data Control Language (DCL)
 - GRANT

```
GRANT ALL ON db1.* TO 'jeffrey'@'localhost';  
GRANT 'role1', 'role2' TO 'user1'@'localhost', 'user2'@'localhost';  
GRANT SELECT ON world.* TO 'role3';
```

- REVOKE

```
REVOKE INSERT ON *.* FROM 'jeffrey'@'localhost';  
REVOKE 'role1', 'role2' FROM 'user1'@'localhost', 'user2'@'localhost';  
REVOKE SELECT ON world.* FROM 'role3';
```


D
C
L

D
E
M
O

GRANT

Pluggable Authentication

- Pluggable Authentication

- Used to verify a client when they connect to the MySQL Server
- The plugin is invoked by the server to return a status indicating whether the user provided the correct password and is permitted to connect.

```
CREATE USER 'jeffrey'@'localhost' IDENTIFIED BY 'password';
```

- Plugins using well-known protocols and cryptographic methods like LDAP and SHA-2 are available.
- The default authentication plugin is stored in the system variable *default_authentication_plugin*.
- Two-Factor Authentication & Three-Factor Authentication

```
CREATE USER 'mateo'@'localhost' IDENTIFIED BY 'password'  
AND IDENTIFIED WITH authentication_ldap_simple  
AND IDENTIFIED WITH authentication_webauthn;
```

Performance Optimization

- Optimizing at the Database Level
 - Table Structures & Column data-types
 - Indexing for optimizing queries
 - Appropriate Storage Engines (Eg: InnoDB, MyISAM)
 - Appropriate Locking Strategies
 - Cache Memory Sizing
- Optimizing at Hardware Level
 - Bottlenecks faced at seek time, main memory bandwidth, etc.

References

- MySQL Architecture – <https://www.mysqltutorial.org/mysql-administration/mysql-architecture/>
- JOINS – <https://www.w3schools.com/>
- GRANT – <https://dev.mysql.com/doc/refman/8.0/en/grant.html>
- REVOKE – <https://dev.mysql.com/doc/refman/8.0/en/revoke.html>
- Triggers – https://www.tutorialspoint.com/plsql/plsql_triggers.htm



Thank You