1. Introduction to MySQL (MySQL का परिचय)

1.1 MySQL क्या है? (What is MySQL?)

MySQL एक **ओपन-सोर्स रिलेशनल डेटाबेस मैनेजमेंट सिस्टम (RDBMS)** है, जिसे Oracle Corporation द्वारा मैनेज किया जाता है। यह डेटा को **स्टोर, मैनेज और रिट्रीव** करने के लिए SQL (Structured Query Language) का उपयोग करता है।

MySQL की विशेषताएँ:

- **ओपन-सोर्स:** कोई भी इसे मुफ्त में इस्तेमाल कर सकता है।
- **हाई परफॉर्मेंस:** बड़ी मात्रा में डेटा को जल्दी प्रोसेस करता है।
- **स्केलेबल:** छोटे से बड़े प्रोजेक्ट्स तक आसानी से उपयोग किया जा सकता है।
- **सेक्योरिटी:** डेटा को सुरक्षित रखने के लिए मजबूत सिक्योरिटी फीचर्स देता है।
- **क्रॉस-प्लेटफॉर्म सपोर्ट:** MySQL Windows, Linux, और MacOS सभी ऑपरेटिंग सिस्टम्स पर चलता है।
- **ACID Compliance:** MySQL ट्रांज़ेक्शन सेफ्टी और डेटा कंसिस्टेंसी के लिए ACID प्रॉपर्टीज़ को सपोर्ट करता है।

1.2 MySQL कहाँ उपयोग किया जाता है? (Where is MySQL Used?)

- **वेबसाइट्स (Websites)** – Facebook, Twitter, YouTube जैसी वेबसाइटें MySQL का उपयोग करती हैं।
- **वेब एप्लिकेशन (Web Applications)** – CMS (WordPress, Joomla, Drupal) MySQL पर निर्भर करते हैं।
- **डेटा एनालिटिक्स (Data Analytics)** – बड़ी कंपनियाँ डेटा एनालिसिस के लिए MySQL का उपयोग करती हैं।
- **मोबाइल एप्लिकेशन (Mobile Applications)** – कई मोबाइल ऐप्स बैकएंड में MySQL का उपयोग करते हैं।
- **E-Commerce Platforms** – Shopify, Magento, WooCommerce जैसे ऑनलाइन स्टोर्स डेटा मैनेजमेंट के लिए MySQL का उपयोग करते हैं।
- **Banking और Financial Systems** – MySQL सुरक्षित डेटा स्टोरेज और ट्रांज़ेक्शन प्रोसेसिंग के लिए बैंकों में उपयोग किया जाता है।

1.3 MySQL vs Other Databases (MySQL अन्य डेटाबेस से कैसे अलग है?)

Feature	MySQL	PostgreSQL	MongoDB
Type	RDBMS	RDBMS	NoSQL

Speed	High	Moderate	Very High
Structure	Structured	Structured	Unstructured
Use Case	Web, Apps	Advanced Apps	Big Data
Joins Support	Yes	Yes	No
Transactions	Yes	Yes	Limited
Scalability	Moderate	High	Very High
Flexibility	Schema-Based	Schema-Based	Schema-Less

1.4 MySQL के महत्वपूर्ण संस्करण (Versions of MySQL)

MySQL के कई संस्करण उपलब्ध हैं, जिनमें प्रत्येक में नई विशेषताएँ और सुधार जोड़े गए हैं।

Version	Release Year	Key Features
MySQL 3.x	1995	**शुरुआती स्टेबल रिलीज**
MySQL 4.x	2001	Query Caching, Improved Joins
MySQL 5.x	2005	Stored Procedures, Triggers, Views
MySQL 6.x	**रद्द कर दिया**	N/A
MySQL 7.x	**उपलब्ध नहीं**	N/A

MySQL 8.x	2018	JSON Support, CTEs, Window Functions

1.5 MySQL कैसे काम करता है? (How Does MySQL Work?)

MySQL डेटा को **टेबल्स (Tables)** के रूप में स्टोर करता है और SQL कमांड्स की मदद से डेटा को प्रोसेस करता है। इसका वर्कफ्लो कुछ इस प्रकार है:

1. **Client Request:** यूज़र या एप्लिकेशन MySQL सर्वर को डेटा के लिए अनुरोध करता है।
2. **Query Execution:** MySQL इंजन SQL क्वेरी को प्रोसेस करता है।
3. **Storage Engine:** डेटा स्टोरेज इंजन क्वेरी को एक्सीक्यूट करता है और परिणाम देता है।
4. **Response to Client:** प्रोसेस्ड डेटा क्लाइंट को वापस भेज दिया जाता है।

---

2. Installing and Setting Up MySQL (MySQL को इंस्टॉल और सेटअप करना)

2.1 MySQL को कैसे डाउनलोड करें? (How to Download MySQL?)

MySQL को Oracle की आधिकारिक वेबसाइट से डाउनलोड किया जा सकता है।

डाउनलोड करने के स्टेप्स:

1. **वेबसाइट खोलें:** https://dev.mysql.com/downloads/
2. अपने ऑपरेटिंग सिस्टम (Windows, Mac, Linux) के अनुसार MySQL Server डाउनलोड करें।
3. डाउनलोड करने के बाद, इंस्टॉलर फ़ाइल को रन करें।

2.2 Windows पर MySQL इंस्टॉल करना (Installing MySQL on Windows)

1. **MySQL Installer चलाएँ** और Server + Workbench ऑप्शन को सेलेक्ट करें।
2. Installation Type चुनें (Developer Default, Custom, आदि)।
3. **Root Password सेट करें** – यह MySQL के एडमिन लॉगिन के लिए आवश्यक होता है।
4. **Default Port (3306) सेट करें** – MySQL डिफ़ॉल्ट रूप से 3306 पोर्ट का उपयोग करता है।
5. **Installation को पूरा करें** और MySQL सर्वर को स्टार्ट करें।

2.3 Linux पर MySQL इंस्टॉल करना (Installing MySQL on Linux)

Linux (Ubuntu/Debian) पर MySQL इंस्टॉल करने के लिए टर्मिनल में निम्नलिखित कमांड्स चलाएँ:

```
sudo apt update
sudo apt install mysql-server
sudo systemctl start mysql
sudo mysql_secure_installation
```

महत्वपूर्ण स्टेप: `mysql_secure_installation` **कमांड का उपयोग करके सुरक्षा सेटिंग्स को कॉन्फ़िगर करें।**

2.4 MacOS पर MySQL इंस्टॉल करना (Installing MySQL on MacOS)

MacOS में MySQL Homebrew के ज़रिए इंस्टॉल किया जा सकता है:

```
brew install mysql
brew services start mysql
```

इसके बाद, MySQL सर्वर स्टार्ट हो जाएगा।

2.5 MySQL Workbench का परिचय (Introduction to MySQL Workbench)

MySQL Workbench एक GUI टूल है जो **डेटाबेस को मैनेज और क्वेरी रन** करने के लिए उपयोग किया जाता है।

Workbench में कनेक्ट करने के स्टेप्स:

1. MySQL Workbench खोलें।
2. New Connection पर क्लिक करें और Root User Credentials डालें।
3. **Connect पर क्लिक करें** और अपने डेटाबेस को मैनेज करें।

3. Basic SQL Queries in MySQL (बेसिक SQL क्वेरीज in MySQL)

3.1 SQL क्या है? (What is SQL?)

SQL (Structured Query Language) एक स्टैंडर्ड लैंग्वेज है जिसका उपयोग डेटाबेस में **डेटा को स्टोर, रिट्रीव, अपडेट और डिलीट** करने के लिए किया जाता है। MySQL में डेटा को मैनेज करने के लिए SQL क्वेरीज का उपयोग किया जाता है।

3.2 Common SQL Commands (आम SQL कमांड्स)

Command	**विवरण**

`SELECT`	**डेटा को रिट्रीव करने के लिए**
`INSERT`	**नया डेटा जोड़ने के लिए**
`UPDATE`	**मौजूदा डेटा को बदलने के लिए**
`DELETE`	**डेटा को हटाने के लिए**
`CREATE TABLE`	**नई टेबल बनाने के लिए**
`DROP TABLE`	**टेबल को हटाने के लिए**
`ALTER TABLE`	**टेबल स्ट्रक्चर को बदलने के लिए**

3.3 SELECT Query (डेटा को क्वेरी करना)

`SELECT` **कमांड का उपयोग डेटाबेस से डेटा को रिट्रीव करने के लिए किया जाता है।**

```
SELECT * FROM employees;
```

यह क्वेरी `employees` **टेबल से सभी रिकॉर्ड्स को लाएगी।**

फिल्टर करने के लिए:

SELECT name, salary FROM employees WHERE salary > 50000;

यह क्वेरी उन कर्मचारियों के नाम और सैलरी दिखाएगी जिनकी सैलरी 50000 से ज्यादा है।

3.4 INSERT Query (डेटा जोड़ना)

नए डेटा को जोड़ने के लिए `INSERT INTO` का उपयोग किया जाता है।

INSERT INTO employees (name, age, salary) VALUES ('Rahul', 30, 60000);

यह `employees` टेबल में एक नया कर्मचारी जोड़ देगा।

3.5 UPDATE Query (डेटा अपडेट करना)

किसी रिकॉर्ड को अपडेट करने के लिए `UPDATE` का उपयोग किया जाता है।

UPDATE employees SET salary = 70000 WHERE name = 'Rahul';

यह कमांड Rahul की सैलरी को 70000 में अपडेट कर देगा।

3.6 DELETE Query (डेटा हटाना)

डेटा हटाने के लिए `DELETE` का उपयोग किया जाता है।

```
DELETE FROM employees WHERE name = 'Rahul';
```

यह `employees` टेबल से Rahul का डेटा हटा देगा।

---

4. Advanced SQL Queries in MySQL (एडवांस्ड SQL क्वेरीज in MySQL)

4.1 Joins in MySQL (MySQL में Joins)

Joins का उपयोग एक से अधिक टेबल्स से डेटा निकालने के लिए किया जाता है।

Types of Joins (Joins के प्रकार)

Join Type	विवरण
`INNER JOIN`	केवल वही रिकॉर्ड्स लाता है जहाँ दोनों टेबल्स में मैचिंग डेटा हो।
`LEFT JOIN`	पहली टेबल से सभी रिकॉर्ड्स लाता है, भले ही दूसरी टेबल में मैच न हो।
`RIGHT JOIN`	दूसरी टेबल से सभी रिकॉर्ड्स लाता है, भले ही पहली टेबल में मैच न हो।

`FULL JOIN`	**दोनों टेबल्स से सभी रिकॉर्ड्स लाता है, भले ही कोई मैच न हो।**

Example of INNER JOIN:

SELECT employees.name, departments.department_name

FROM employees

INNER JOIN departments ON employees.department_id = departments.id;

इसका प्रभाव: यह क्वेरी `employees` और `departments` टेबल्स को जोड़कर कर्मचारियों के नाम और उनके डिपार्टमेंट दिखाएगी।

4.2 Subqueries in MySQL (MySQL में सबक्वेरीज)

Subquery एक **क्वेरी के अंदर दूसरी क्वेरी** होती है, जो किसी मुख्य क्वेरी के लिए डेटा प्रदान करती है।

Example:

SELECT name, salary FROM employees

WHERE salary > (SELECT AVG(salary) FROM employees);

इसका प्रभाव: यह क्वेरी उन कर्मचारियों को दिखाएगी जिनकी सैलरी औसत सैलरी से अधिक है।

4.3 Group By and Having Clause (ग्रुपिंग और फ़िल्टरिंग)

Group By: यह डेटा को एक निश्चित कॉलम के आधार पर ग्रुप करता है।

```
SELECT department, COUNT(*) FROM employees
GROUP BY department;
```

इसका प्रभाव: यह क्वेरी हर डिपार्टमेंट में कर्मचारियों की संख्या दिखाएगी।

Having Clause: यह GROUP BY के बाद डेटा को फ़िल्टर करता है।

```
SELECT department, AVG(salary) FROM employees
GROUP BY department
HAVING AVG(salary) > 50000;
```

इसका प्रभाव: यह क्वेरी केवल उन्हीं डिपार्टमेंट्स को दिखाएगी जिनकी औसत सैलरी 50000 से अधिक है।

4.4 Stored Procedures in MySQL (MySQL में स्टोर्ड प्रोसीजर्स)

Stored Procedure SQL स्टेटमेंट्स का एक समूह होता है जिसे बार-बार एक्सीक्यूट किया जा सकता है।

Stored Procedure बनाने का तरीका:

```
DELIMITER //

CREATE PROCEDURE GetEmployees()

BEGIN

    SELECT * FROM employees;

END //

DELIMITER ;
```

Stored Procedure को कॉल करने का तरीका:

```
CALL GetEmployees();
```

इसका प्रभाव: यह प्रोसीजर `employees` टेबल से सभी रिकॉर्ड्स को रिट्रीव करेगा।

5. Indexes and Performance Optimization in MySQL (MySQL में इंडेक्स और परफॉर्मेंस ऑप्टिमाइज़ेशन)

5.1 Indexes in MySQL (MySQL में इंडेक्स क्या है?)

इंडेक्स एक **डाटा स्ट्रक्चर** है जो MySQL को क्वेरीज को तेज़ी से एक्सीक्यूट करने में मदद करता है। यह **टेबल्स में डेटा को व्यवस्थित** करता है जिससे सर्चिंग और फेचिंग फास्ट हो जाती है।

Indexes के प्रकार (Types of Indexes)

Index Type	विवरण
Primary Index	Primary Key वाला इंडेक्स, प्रत्येक टेबल में एक ही होता है।
Unique Index	ऐसे इंडेक्स जो डुप्लिकेट वैल्यू की अनुमति नहीं देते।
Composite Index	एक से अधिक कॉलम पर आधारित इंडेक्स।
Full-Text Index	टेक्स्ट सर्चिंग के लिए उपयोग किया जाता है।
Clustered Index	डेटा को फिजिकली ऑर्डर करता है।

Example of Creating an Index:

```
CREATE INDEX idx_employee_name ON employees(name);
```

इसका प्रभाव: `employees` टेबल के `name` कॉलम पर एक इंडेक्स बना देगा जिससे सर्चिंग फास्ट होगी।

Indexed Query Performance Comparison:

```
EXPLAIN SELECT * FROM employees WHERE name = 'Rahul';
```

EXPLAIN का उपयोग करके यह देखा जा सकता है कि इंडेक्स क्वेरी को कितना तेज करता है।

5.2 How to Optimize Queries in MySQL (MySQL में क्वेरीज ऑप्टिमाइज़ कैसे करें?)

1. Proper Indexing (सही इंडेक्सिंग करें)

- इंडेक्स का उपयोग करें लेकिन ज़रूरत से ज़्यादा इंडेक्स न बनाएं।
- `WHERE`, `JOIN`, और `ORDER BY` में उपयोग होने वाले कॉलम्स को इंडेक्स करें।

2. SELECT * Avoid करें (सभी कॉलम्स को न चुनें)

```
SELECT name, age FROM employees WHERE department = 'IT';
```

क्यों?

- `SELECT *` पूरे टेबल के सभी कॉलम्स लाता है, जिससे परफॉर्मेंस स्लो होती है।

3. Joins को ऑप्टिमाइज़ करें

```
SELECT e.name, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.id;
```

- हमेशा इंडेक्स किए गए कॉलम्स पर जॉइन करें।
- `INNER JOIN` का उपयोग करें जब भी संभव हो।

4. Query Caching (क्वेरी कैशिंग)

```
SET GLOBAL query_cache_size = 1000000;
SET GLOBAL query_cache_type = ON;
```

- बार-बार उपयोग होने वाली क्वेरीज को कैश करना परफॉर्मेंस सुधारता है।

5. Use EXPLAIN for Query Analysis (EXPLAIN का उपयोग करें)

```
EXPLAIN SELECT * FROM employees WHERE department = 'IT';
```

- यह दिखाता है कि MySQL क्वेरी को कैसे प्रोसेस कर रहा है और कहाँ सुधार की जरूरत है।

5.3 Partitioning in MySQL (MySQL में डेटा पार्टिशनिंग)

Partitioning का उपयोग बड़े टेबल्स को छोटे-छोटे भागों में बांटने के लिए किया जाता है जिससे परफॉर्मेंस बेहतर होती है।

```
CREATE TABLE employees (
    id INT NOT NULL,
    name VARCHAR(50),
    department VARCHAR(50),
    salary INT,
    PRIMARY KEY (id, department)
)
PARTITION BY RANGE (salary) (
    PARTITION p1 VALUES LESS THAN (30000),
    PARTITION p2 VALUES LESS THAN (60000),
    PARTITION p3 VALUES LESS THAN (100000)
);
```

इसका प्रभाव:

- सैलरी के आधार पर डेटा को **तीन अलग-अलग पार्टिशन्स** में स्टोर करेगा।
- सर्च और फेचिंग स्पीड तेज होगी।

5.4 Performance Testing Tools (MySQL परफॉर्मेंस टेस्टिंग टूल्स)

Tool Name	Use Case
MySQL Slow Query Log	स्लो क्वेरीज को मॉनिटर करता है
MySQLTuner	डेटाबेस परफॉर्मेंस अनालिसिस करता है
Percona Toolkit	इंडेक्स और क्वेरी ऑप्टिमाइज़ेशन में मदद करता है
EXPLAIN & ANALYZE	क्वेरी के एग्ज़ीक्यूशन प्लान को दिखाता है

6. Transactions and Concurrency Control in MySQL (MySQL में ट्रांज़ेक्शन और कंकरेन्सी कंट्रोल)

6.1 Transactions in MySQL (MySQL में ट्रांज़ेक्शन क्या है?)

ट्रांज़ेक्शन एक **ग्रोअप ऑपरेशन्स** है जो यह सुनिश्चित करता है कि या तो सभी SQL स्टेटमेंट्स सफल हों या कोई भी लागू न हो।

ACID Properties (ACID गुण)

MySQL में ट्रांज़ेक्शन को ACID (Atomicity, Consistency, Isolation, Durability) प्रॉपर्टीज़ का पालन करना चाहिए:

- **Atomicity (एटॉमिकिटी):** सभी ऑपरेशन्स पूरे होने चाहिए या एक भी लागू न हो।
- **Consistency (कंसिस्टेंसी):** डेटा हमेशा वैध स्टेट में रहना चाहिए।
- **Isolation (आइसोलेशन):** एक ट्रांज़ेक्शन को दूसरे ट्रांज़ेक्शन से प्रभावित नहीं होना चाहिए।
- **Durability (ड्युरेबिलिटी):** एक बार कमिट होने के बाद डेटा हमेशा के लिए सेव रहेगा।

6.2 Using Transactions in MySQL (MySQL में ट्रांज़ेक्शन का उपयोग करना)

ट्रांज़ेक्शन को `START TRANSACTION`, `COMMIT`, और `ROLLBACK` का उपयोग करके नियंत्रित किया जाता है।

Example:

```
START TRANSACTION;
UPDATE accounts SET balance = balance - 500 WHERE name = 'Amit';
UPDATE accounts SET balance = balance + 500 WHERE name = 'Rahul';
COMMIT;
```

इसका प्रभाव:

- **अगर दोनों `UPDATE` स्टेटमेंट सफल होते हैं, तो बदलाव सेव हो जाएगा।**
- **अगर कोई भी स्टेटमेंट फेल होता है, तो डेटा पिछले स्टेट में ही रहेगा।**

ROLLBACK Example:

```
START TRANSACTION;
UPDATE accounts SET balance = balance - 1000 WHERE name = 'Amit';
ROLLBACK;
```

इसका प्रभाव: यह बदलाव को वापस कर देगा, जैसे कुछ हुआ ही नहीं था।

6.3 Isolation Levels in MySQL (MySQL में आइसोलेशन लेवल्स)

MySQL में विभिन्न आइसोलेशन लेवल्स होते हैं, जो ट्रांज़ेक्शन को Concurrency Issues से बचाते हैं।

Isolation Level	Issue Prevented
READ UNCOMMITTED	कोई रोक नहीं (डर्टी रीड)
READ COMMITTED	डर्टी रीड नहीं होगा
REPEATABLE READ	डर्टी रीड + नॉन-रीपीटेबल रीड रोका जाता है
SERIALIZABLE	सभी प्रकार की रेस कंडीशन्स को रोका जाता है

Example: Setting Isolation Level

```
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
```

6.4 Deadlocks in MySQL (MySQL में डेडलॉक्स)

जब दो ट्रांज़ेक्शन एक-दूसरे के संसाधनों को लॉक कर देते हैं और आगे नहीं बढ़ पाते, तब Deadlock की स्थिति उत्पन्न होती है।

Deadlock Example:

```
-- Transaction 1
START TRANSACTION;
UPDATE employees SET salary = salary + 1000 WHERE id = 1;
UPDATE employees SET salary = salary + 500 WHERE id = 2;
COMMIT;

-- Transaction 2
START TRANSACTION;
UPDATE employees SET salary = salary + 500 WHERE id = 2;
UPDATE employees SET salary = salary + 1000 WHERE id = 1;
COMMIT;
```

इस स्थिति में दोनों ट्रांज़ेक्शन एक-दूसरे को ब्लॉक कर देंगे।

Deadlock से बचने के उपाय:

- **ट्रांज़ेक्शन को छोटे रखें।**
- **हमेशा एक ही ऑर्डर में टेबल्स को अपडेट करें।**
- **टाइमआउट सेट करें:**

```
SET innodb_lock_wait_timeout = 5;
```

---

7. Stored Procedures and Triggers in MySQL (MySQL में स्टोर्ड प्रोसीजर्स और ट्रिगर्स)

7.1 Stored Procedures in MySQL (MySQL में स्टोर्ड प्रोसीजर्स क्या हैं?)

Stored Procedure एक **प्रोग्रामेबल SQL स्क्रिप्ट** होती है, जिसे **बार-बार इस्तेमाल किए जाने वाले ऑपरेशन्स** के लिए स्टोर किया जाता है। इससे **कोड रिपीटेशन कम होता है और परफॉर्मेंस बेहतर होती है।**

Stored Procedure के फायदे:

- **कोड रीयूज़ेबिलिटी** – एक बार लिखकर कई बार उपयोग किया जा सकता है।
- **परफॉर्मेंस बेहतर होती है** – डेटा प्रोसेसिंग तेज़ होती है।
- **सिक्योरिटी** – यूज़र्स को केवल स्टोर्ड प्रोसीजर एक्सेस देने से डेटा सुरक्षित रहता है।

Stored Procedure बनाने का सिंटैक्स:

```
DELIMITER //
CREATE PROCEDURE GetEmployees()
BEGIN
    SELECT * FROM employees;
END //
DELIMITER ;
```

Stored Procedure को कॉल करना:

```
CALL GetEmployees();
```

इसका प्रभाव: `employees` टेबल से सभी रिकॉर्ड्स को रिट्रीव करेगा।

7.2 Stored Procedure with Parameters (पैरामीटर्स के साथ स्टोर्ड प्रोसीजर)

Stored Procedure में **IN, OUT, और INOUT** पैरामीटर्स का उपयोग किया जाता है।

Example: IN Parameter का उपयोग

```
DELIMITER //
CREATE PROCEDURE GetEmployeeByDept(IN dept_name VARCHAR(50))
BEGIN
   SELECT * FROM employees WHERE department = dept_name;
END //
DELIMITER ;
```

Stored Procedure को कॉल करना:

```
CALL GetEmployeeByDept('IT');
```

इसका प्रभाव: यह क्वेरी **IT डिपार्टमेंट के सभी कर्मचारियों** को दिखाएगी।

7.3 Triggers in MySQL (MySQL में ट्रिगर्स क्या हैं?)

Trigger एक **ऑटोमेटिक SQL प्रोसेस** होता है, जो किसी टेबल में **INSERT, UPDATE, या DELETE** होने पर अपने आप एक्सीक्यूट हो जाता है।

Triggers के प्रकार:

Trigger Type	Description
BEFORE INSERT	**डेटा इन्सर्ट होने से पहले ट्रिगर होता है**
AFTER INSERT	**डेटा इन्सर्ट होने के बाद ट्रिगर होता है**
BEFORE UPDATE	**डेटा अपडेट होने से पहले ट्रिगर होता है**
AFTER UPDATE	**डेटा अपडेट होने के बाद ट्रिगर होता है**
BEFORE DELETE	**डेटा डिलीट होने से पहले ट्रिगर होता है**
AFTER DELETE	**डेटा डिलीट होने के बाद ट्रिगर होता है**

7.4 Creating a Trigger in MySQL (MySQL में ट्रिगर बनाना)

Example: AFTER INSERT Trigger

```
DELIMITER //
CREATE TRIGGER after_employee_insert
AFTER INSERT ON employees
FOR EACH ROW
BEGIN
   INSERT INTO audit_log (action, employee_id, timestamp)
   VALUES ('INSERT', NEW.id, NOW());
END //
DELIMITER ;
```

इसका प्रभाव: जब भी `employees` टेबल में नया डेटा जोड़ा जाएगा, तब audit_log टेबल में ऑटोमेटिक एंट्री होगी।

7.5 Deleting a Stored Procedure or Trigger (स्टोर्ड प्रोसीजर या ट्रिगर को हटाना)

Stored Procedure हटाने के लिए:

```
DROP PROCEDURE IF EXISTS GetEmployees;
```

Trigger हटाने के लिए:

```
DROP TRIGGER IF EXISTS after_employee_insert;
```

---

8. Backup and Restore in MySQL (MySQL में बैकअप और रिस्टोर)

8.1 MySQL Backup क्या है? (What is MySQL Backup?)

बैकअप का मतलब MySQL डेटाबेस का **कॉपी** बनाना है ताकि किसी भी डेटा लॉस, हार्डवेयर फेलियर या गलत ऑपरेशन के कारण डेटा को आसानी से रिकवर किया जा सके।

MySQL में बैकअप के प्रकार:

Backup Type	विवरण
Logical Backup	`mysqldump` कमांड का उपयोग कर डेटा को SQL स्क्रिप्ट के रूप में सेव करना
Physical Backup	डायरेक्ट डेटाबेस फाइल्स को कॉपी करना
Incremental Backup	केवल बदले हुए डेटा का बैकअप लेना
Full Backup	पूरे डेटाबेस का बैकअप लेना

8.2 MySQL Backup लेने के तरीके (Methods to Take Backup in MySQL)

1. mysqldump के साथ बैकअप लेना

`mysqldump` एक कमांड-लाइन टूल है जो MySQL डेटाबेस का **लॉजिकल बैकअप** बनाता है।

सिंटैक्स:

```
mysqldump -u root -p database_name > backup.sql
```

Example:

mysqldump -u root -p mydatabase > mydatabase_backup.sql

इसका प्रभाव: यह `mydatabase` का बैकअप `mydatabase_backup.sql` फाइल में सेव करेगा।

---

2. Multiple Databases का Backup लेना

अगर आपको एक से अधिक डेटाबेस का बैकअप लेना हो:

mysqldump -u root -p --databases database1 database2 > backup.sql

इसका प्रभाव: यह `database1` और `database2` का बैकअप एक ही फाइल में सेव करेगा।

---

3. सभी डेटाबेस का Backup लेना

अगर पूरे MySQL सर्वर का बैकअप चाहिए:

mysqldump -u root -p --all-databases > all_databases_backup.sql

इसका प्रभाव: यह MySQL सर्वर के सभी डेटाबेस का बैकअप सेव करेगा।

---

4. Physical Backup लेना (डेटाबेस फाइल्स कॉपी करके)

Linux/Unix में डेटाबेस फाइल्स का डायरेक्ट बैकअप लेने के लिए:

cp -r /var/lib/mysql /backup/mysql_backup

ध्यान दें: यह बैकअप लेने के लिए MySQL Server को पहले रोकना जरूरी है।

sudo systemctl stop mysql

8.3 MySQL में Backup Restore कैसे करें? (How to Restore Backup in MySQL)

1. mysqldump से बैकअप रिस्टोर करना

mysql -u root -p database_name < backup.sql

Example:

mysql -u root -p mydatabase < mydatabase_backup.sql

इसका प्रभाव: `mydatabase_backup.sql` **से** `mydatabase` **को रिस्टोर कर देगा।**

---

2. सभी डेटाबेस को Restore करना

mysql -u root -p < all_databases_backup.sql

इसका प्रभाव: यह सभी डेटाबेस को `all_databases_backup.sql` **से रिस्टोर करेगा।**

8.4 Automating MySQL Backup (MySQL Backup को ऑटोमेट करना)

अगर आप **रोजाना बैकअप** लेना चाहते हैं, तो `cron job` का उपयोग कर सकते हैं।

Linux में ऑटोमेटिक बैकअप के लिए:

crontab -e

और इसमें यह लाइन जोड़ें:

0 2 * * * mysqldump -u root -p mydatabase > /backup/mydatabase_$(date +\%F).sql

इसका प्रभाव:

- **यह हर दिन रात 2:00 बजे** `mydatabase` **का बैकअप** `/backup/` **फोल्डर में सेव करेगा।**
- **बैकअप फाइल का नाम** `mydatabase_YYYY-MM-DD.sql` **होगा।**

9. User Management and Security in MySQL (MySQL में यूज़र मैनेजमेंट और सिक्योरिटी)

9.1 MySQL में यूज़र मैनेजमेंट क्या है? (What is User Management in MySQL?)

MySQL में **यूज़र मैनेजमेंट** का उपयोग यह नियंत्रित करने के लिए किया जाता है कि कौन डेटाबेस को एक्सेस कर सकता है और क्या कर सकता है। इसमें **यूज़र क्रिएशन, परमिशन असाइन करना, और सिक्योरिटी सेटिंग्स** शामिल होती हैं।

9.2 Creating a New User (नया यूज़र बनाना)

सिंटैक्स:

```
CREATE USER 'username'@'host' IDENTIFIED BY 'password';
```

Example:

```
CREATE USER 'rahul'@'localhost' IDENTIFIED BY 'Rahul@123';
```

इसका प्रभाव:

- `rahul` नाम का एक नया यूज़र `localhost` पर बनाया जाएगा।
- इस यूज़र को डिफ़ॉल्ट रूप से कोई विशेष परमिशन नहीं होगी।

9.3 Assigning Privileges to Users (यूज़र्स को परमिशन देना)

सिंटैक्स:

```
GRANT privileges ON database.table TO 'username'@'host';
```

Example:

```
GRANT ALL PRIVILEGES ON mydatabase.* TO 'rahul'@'localhost';
```

इसका प्रभाव:

- `rahul` यूज़र को `mydatabase` के सभी टेबल्स पर **सभी प्रकार की एक्सेस** मिल जाएगी।

Specific Privileges:

Privilege	विवरण
SELECT	डेटा देखने की अनुमति
INSERT	नया डेटा जोड़ने की अनुमति
UPDATE	मौजूदा डेटा बदलने की अनुमति
DELETE	डेटा हटाने की अनुमति
CREATE	नई टेबल्स बनाने की अनुमति
DROP	टेबल्स हटाने की अनुमति
GRANT OPTION	दूसरों को परमिशन देने की अनुमति

Example: केवल SELECT और INSERT की अनुमति देना

GRANT SELECT, INSERT ON mydatabase.* TO 'rahul'@'localhost';

9.4 Checking and Revoking Privileges (परमिशन चेक और हटाना)

चेक करने के लिए:

```
SHOW GRANTS FOR 'rahul'@'localhost';
```

परमिशन हटाने के लिए:

```
REVOKE INSERT ON mydatabase.* FROM 'rahul'@'localhost';
```

इसका प्रभाव: `rahul` यूज़र अब `INSERT` ऑपरेशन नहीं कर पाएगा।

9.5 Deleting a User (यूज़र हटाना)

सिंटैक्स:

```
DROP USER 'username'@'host';
```

Example:

```
DROP USER 'rahul'@'localhost';
```

इसका प्रभाव: `rahul` यूज़र MySQL सर्वर से पूरी तरह से हटा दिया जाएगा।

9.6 MySQL Security Best Practices (MySQL में सिक्योरिटी के सर्वोत्तम तरीके)

1. Strong Passwords का उपयोग करें

```
ALTER USER 'rahul'@'localhost' IDENTIFIED BY 'NewStrong@123';
```

2. Root User के लिए Remote Access Disable करें

```
UPDATE mysql.user SET Host='localhost' WHERE User='root';
FLUSH PRIVILEGES;
```

3. Anonymous Users हटाएँ

```
DELETE FROM mysql.user WHERE User='';
FLUSH PRIVILEGES;
```

4. Firewall और SSL/TLS उपयोग करें

```
ALTER USER 'rahul'@'%' REQUIRE SSL;
```

इसका प्रभाव: `rahul` यूज़र केवल SSL कनेक्शन के माध्यम से MySQL को एक्सेस कर सकता है।

10. MySQL Performance Tuning and Optimization (MySQL परफॉर्मेंस ट्यूनिंग और ऑप्टिमाइज़ेशन)

10.1 MySQL परफॉर्मेंस ट्यूनिंग क्या है? (What is MySQL Performance Tuning?)

परफॉर्मेंस ट्यूनिंग का उद्देश्य **डेटाबेस क्वेरीज को तेज़ बनाना, रिसोर्स उपयोग को ऑप्टिमाइज़ करना और सिस्टम लोड को कम करना** है। MySQL की परफॉर्मेंस को बेहतर बनाने के लिए कई ट्यूनिंग तकनीकों का उपयोग किया जाता है।

10.2 Indexing for Performance (इंडेक्सिंग से परफॉर्मेंस सुधारना)

इंडेक्स का उपयोग डेटा रिट्रीवल स्पीड बढ़ाने के लिए किया जाता है।

Index बनाने का सिंटैक्स:

```
CREATE INDEX idx_name ON employees(name);
```

इसका प्रभाव: यह `employees` टेबल के `name` कॉलम पर इंडेक्स बनाएगा जिससे सर्चिंग फास्ट होगी।

Index का उपयोग करने वाली क्वेरी:

```
EXPLAIN SELECT * FROM employees WHERE name = 'Rahul';
```

EXPLAIN कमांड यह दिखाएगा कि MySQL क्वेरी को कैसे प्रोसेस कर रहा है।

10.3 Query Optimization (क्वेरी ऑप्टिमाइज़ेशन)

1. SELECT * से बचें

```
SELECT name, age FROM employees WHERE department = 'IT';
```

क्यों?

- `SELECT *` पूरे टेबल को स्कैन करता है, जिससे परफॉर्मेंस स्लो होती है।

2. Proper Joins का उपयोग करें

```
SELECT e.name, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.id;
```

- **Indexes के साथ Joins का उपयोग करें।**
- `INNER JOIN` **का उपयोग जब भी संभव हो।**

3. Query Caching Enable करें

```
SET GLOBAL query_cache_size = 1000000;
SET GLOBAL query_cache_type = ON;
```

- **बार-बार उपयोग होने वाली क्वेरीज को कैश करना परफॉर्मेंस सुधारता है।**

10.4 Partitioning for Large Datasets (बड़े डेटा के लिए पार्टिशनिंग)

Partitioning का उपयोग बड़े टेबल्स को छोटे-छोटे भागों में बांटने के लिए किया जाता है जिससे परफॉर्मेंस बेहतर होती है।

```
CREATE TABLE employees (
    id INT NOT NULL,
    name VARCHAR(50),
    department VARCHAR(50),
    salary INT,
    PRIMARY KEY (id, department)
)
PARTITION BY RANGE (salary) (
    PARTITION p1 VALUES LESS THAN (30000),
    PARTITION p2 VALUES LESS THAN (60000),
    PARTITION p3 VALUES LESS THAN (100000)
);
```

इसका प्रभाव:

- **सैलरी के आधार पर डेटा को तीन अलग-अलग पार्टिशन्स में स्टोर करेगा।**
- **सर्च और फेचिंग स्पीड तेज होगी।**

10.5 Performance Monitoring Tools (परफॉर्मेंस मॉनिटरिंग टूल्स)

Tool Name	Use Case
MySQL Slow Query Log	**स्लो क्वेरीज को मॉनिटर करता है**
MySQLTuner	**डेटाबेस परफॉर्मेंस अनालिसिस करता है**
Percona Toolkit	**इंडेक्स और क्वेरी ऑप्टिमाइज़ेशन में मदद करता है**
EXPLAIN & ANALYZE	**क्वेरी के एग्ज़ीक्यूशन प्लान को दिखाता है**

---

11. MySQL Replication and High Availability (MySQL में रेप्लिकेशन और हाई अवेलेबिलिटी)

11.1 MySQL Replication क्या है? (What is MySQL Replication?)

MySQL Replication एक **प्रक्रिया** है, जिसमें डेटा को **एक डेटाबेस सर्वर से दूसरे डेटाबेस सर्वर पर कॉपी किया जाता है।** यह **डेटा बैकअप, लोड बैलेंसिंग, और फॉल्ट टॉलरेंस** के लिए उपयोगी है।

Replication के फायदे:

- **डेटा बैकअप** - अगर एक सर्वर फेल हो जाए, तो दूसरा डेटा सेव रखता है।
- **लोड बैलेंसिंग** - एक से अधिक सर्वर पर क्वेरीज डिस्ट्रीब्यूट होती हैं।
- **रीड परफॉर्मेंस सुधारना** - अलग-अलग सर्वर से डेटा रीड किया जा सकता है।

11.2 Types of MySQL Replication (MySQL Replication के प्रकार)

Replication Type	विवरण
Master-Slave Replication	एक मास्टर सर्वर डेटा अपडेट करता है, और स्लेव सर्वर कॉपी करता है
Master-Master Replication	दोनों सर्वर डेटा अपडेट और कॉपी कर सकते हैं
Group Replication	एक से अधिक नोड्स एक साथ काम कर सकते हैं
Semi-Synchronous Replication	ट्रांज़ैक्शंस को मास्टर और स्लेव दोनों को कन्फर्म करना पड़ता है

11.3 Master-Slave Replication सेटअप कैसे करें? (Setting Up Master-Slave Replication)

Step 1: Master Server को कॉन्फ़िगर करें

1. `my.cnf` फाइल को एडिट करें:

```
[mysqld]
log-bin=mysql-bin
server-id=1
binlog-do-db=mydatabase
```

2. MySQL को रीस्टार्ट करें:

```
sudo systemctl restart mysql
```

3. Master Server में एक यूज़र बनाएं:

```
CREATE USER 'replica'@'%' IDENTIFIED BY 'password';
GRANT REPLICATION SLAVE ON *.* TO 'replica'@'%';
FLUSH PRIVILEGES;
```

4. **Master Status चेक करें:**

```
SHOW MASTER STATUS;
```

Step 2: Slave Server को कॉन्फ़िगर करें

1. `my.cnf` **फाइल एडिट करें:**

```
[mysqld]
server-id=2
relay-log=relay-log-bin
```

2. **Slave Server पर Master को कनेक्ट करें:**

```
CHANGE MASTER TO MASTER_HOST='master_ip',
MASTER_USER='replica', MASTER_PASSWORD='password',
MASTER_LOG_FILE='mysql-bin.000001', MASTER_LOG_POS=12345;
START SLAVE;
```

3. **Slave Status चेक करें:**

```
SHOW SLAVE STATUS\G;
```

इसका प्रभाव: अब मास्टर का डेटा स्लेव सर्वर पर ऑटोमेटिक सिंक होगा।

11.4 High Availability in MySQL (MySQL में हाई अवेलेबिलिटी)

High Availability का मतलब है कि डेटाबेस **हमेशा उपलब्ध** रहे, भले ही कोई हार्डवेयर या नेटवर्क फेल हो जाए।

High Availability के समाधान:

Solution	विवरण
MySQL Replication	डेटा को मल्टीपल सर्वर पर कॉपी करना
MySQL Cluster	सभी नोड्स में डेटा सिंक्रोनाइज़ रहता है
Load Balancing	ट्रैफिक को बैलेंस करने के लिए MySQL Proxy का उपयोग
Failover Mechanism	स्वचालित स्विचिंग जब कोई सर्वर डाउन हो

11.5 MySQL Cluster क्या है? (What is MySQL Cluster?)

MySQL Cluster एक **डिस्ट्रीब्यूटेड डेटाबेस सिस्टम** है जो **100% अपटाइम** सुनिश्चित करता है। इसमें **मल्टीपल डेटा नोड्स होते हैं** जो डेटा को सिंक्रोनाइज़ रखते हैं।

MySQL Cluster के फायदे:

- **नो सिंगल पॉइंट ऑफ फेलियर** – डेटा मल्टीपल नोड्स में स्टोर रहता है।
- **हाई स्पीड परफॉर्मेंस** – इन-मेमोरी स्टोरेज का उपयोग करता है।
- **स्केलेबिलिटी** – बड़ी वेबसाइट्स और ऐप्स के लिए आदर्श।

---

12. MySQL with NoSQL Features (JSON, Document Store) (MySQL में NoSQL फीचर्स)

12.1 MySQL में NoSQL का परिचय (Introduction to NoSQL in MySQL)

MySQL पारंपरिक रूप से एक **रिलेशनल डेटाबेस** है, लेकिन MySQL 5.7 और 8.0 के बाद इसमें **NoSQL फीचर्स** भी जोड़े गए हैं। यह फीचर्स JSON डेटा स्टोरेज और Document Store सपोर्ट प्रदान करते हैं।

NoSQL फीचर्स के फायदे:

- **JSON डेटा स्टोरेज:** डेटा को JSON फ़ॉर्मेट में स्टोर और प्रोसेस कर सकते हैं।
- **फ्लेक्सिबल स्कीमा:** डेटा को बिना फिक्स्ड टेबल स्ट्रक्चर के स्टोर किया जा सकता है।
- **हाई परफॉर्मेंस:** NoSQL क्वेरीज फास्ट होती हैं और बड़े डेटा के लिए ऑप्टिमाइज़ की जाती हैं।

12.2 JSON Data Type in MySQL (MySQL में JSON डेटा टाइप)

MySQL JSON को एक डेटा टाइप के रूप में सपोर्ट करता है, जिससे हम Semi-Structured Data स्टोर कर सकते हैं।

JSON डेटा टाइप के फायदे:

- **डेटा को हाइरार्किकल (Nested) संरचना में स्टोर किया जा सकता है।**
- **तेज़ सर्चिंग और क्वेरी ऑप्टिमाइज़ेशन।**
- **JSON फ़ॉर्मेट REST API और वेब एप्लिकेशन के लिए उपयोगी है।**

JSON कॉलम के साथ टेबल बनाना:

```
CREATE TABLE employees (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    details JSON
);
```

इसका प्रभाव: `details` कॉलम JSON डेटा को स्टोर करने में सक्षम होगा।

JSON डेटा इंसर्ट करना:

```
INSERT INTO employees (name, details)
VALUES ('Rahul', '{"age": 30, "department": "IT", "skills": ["SQL", "Python"]}');
```

12.3 JSON Functions in MySQL (MySQL में JSON फंक्शंस)

MySQL JSON डेटा को मैनेज करने के लिए कई फंक्शंस प्रदान करता है।

Function	विवरण
JSON_EXTRACT()	JSON से किसी वैल्यू को निकालने के लिए
JSON_SET()	JSON डेटा को अपडेट करने के लिए
JSON_REMOVE()	JSON डेटा से किसी फील्ड को हटाने के लिए
JSON_ARRAY()	JSON में एक ऐरे बनाने के लिए
JSON_OBJECT()	JSON में एक ऑब्जेक्ट बनाने के लिए

Example: JSON डेटा से एक वैल्यू निकालना

```
SELECT JSON_EXTRACT(details, '$.department') AS department FROM employees;
```

इसका प्रभाव: यह `employees` टेबल से department की वैल्यू को JSON फॉर्मेट से निकालकर दिखाएगा।

12.4 MySQL Document Store (MySQL में डॉक्यूमेंट स्टोर)

MySQL 8.0 में Document Store फीचर जोड़ा गया, जो एक NoSQL डेटाबेस की तरह काम करता है। यह X DevAPI का उपयोग करता है और MongoDB जैसे NoSQL सिस्टम्स की तरह कार्य करता है।

Document Store के फायदे:

- डेटा को **ट्रेडिशनल टेबल्स के बिना** स्टोर किया जा सकता है।
- JSON डॉक्यूमेंट्स को क्वेरी और अपडेट किया जा सकता है।
- हाई स्केलेबिलिटी और फ्लेक्सिबल डेटा मॉडल।

Document Store के लिए MySQL Shell में काम करना:

```
mysqlsh --mysqlx -u root -p
```

Document Store में कलेक्शन बनाना:

```
var db = session.getSchema("test");
var collection = db.createCollection("employees");
```

JSON डॉक्यूमेंट इंसर्ट करना:

```
collection.add({"name": "Rahul", "age": 30, "department": "IT"}).execute();
```

इसका प्रभाव: यह JSON डेटा को MySQL Document Store में स्टोर करेगा।

12.5 JSON और रिलेशनल डेटा का मिश्रण (Mixing JSON and Relational Data)

MySQL JSON को रिलेशनल डेटा के साथ मिलाकर उपयोग करने की अनुमति देता है।

Example:

```
SELECT name, JSON_EXTRACT(details, '$.skills[0]') AS first_skill
FROM employees;
```

इसका प्रभाव: यह JSON डेटा से पहला `skill` निकालकर दिखाएगा।

13. MySQL Best Practices and Real-World Use Cases (MySQL के सर्वोत्तम अभ्यास और वास्तविक जीवन के उपयोग)

13.1 MySQL Best Practices (MySQL के सर्वोत्तम अभ्यास)

MySQL का सही उपयोग करने के लिए कुछ बेहतरीन रणनीतियाँ अपनाई जानी चाहिए। यह **डेटाबेस परफॉर्मेंस, सुरक्षा और स्केलेबिलिटी** को बढ़ाने में मदद करती हैं।

1. सही डेटा टाइप चुनें

- VARCHAR का उपयोग केवल छोटे स्ट्रिंग्स के लिए करें।
- **TEXT और BLOB** का उपयोग बड़े डेटा (जैसे इमेज, डॉक्यूमेंट) के लिए करें।
- **INT, BIGINT, और DECIMAL** का सही उपयोग करें।

2. Indexing का सही उपयोग करें

- हर Primary Key के लिए Index होना चाहिए।
- Frequently Used Columns पर Index बनाएं।
- Unused Indexes को हटा दें।

3. Large Queries को Optimize करें

- `SELECT *` से बचें और केवल आवश्यक कॉलम चुनें।
- Joins और Subqueries को Optimize करें।
- `EXPLAIN` का उपयोग करें यह देखने के लिए कि क्वेरी कैसे चल रही है।

4. Connection Management पर ध्यान दें

- Persistent Connections का उपयोग करें।
- Connection Pooling सेट करें।
- Unused Connections को जल्दी बंद करें।

5. सुरक्षा पर ध्यान दें

- `root` यूज़र का उपयोग न करें।
- प्रत्येक एप्लिकेशन के लिए एक अलग यूज़र बनाएं।
- Strong Password और Firewall सेट करें।

13.2 Real-World Use Cases of MySQL (MySQL के वास्तविक जीवन के उपयोग)

MySQL विभिन्न उद्योगों में व्यापक रूप से उपयोग किया जाता है। नीचे कुछ प्रमुख उपयोग दिए गए हैं:

1. E-Commerce Platforms (ई-कॉमर्स प्लेटफॉर्म्स)

- Amazon, Flipkart, Shopify जैसी कंपनियाँ MySQL का उपयोग करती हैं।

- उपयोग:
    - प्रोडक्ट कैटलॉग मैनेजमेंट
    - ऑर्डर प्रोसेसिंग और ट्रांज़ैक्शन मैनेजमेंट
    - यूज़र डेटा स्टोरेज

2. Social Media Applications (सोशल मीडिया एप्लिकेशन)

- Facebook, Twitter, Instagram जैसी सोशल मीडिया साइट्स डेटा स्टोर करने के लिए MySQL का उपयोग करती हैं।
- उपयोग:
    - यूज़र प्रोफाइल और डेटा स्टोरेज
    - फॉलोअर्स/फ्रेंड लिस्ट मैनेजमेंट
    - चैट और मैसेज स्टोरेज

3. Banking and Financial Applications (बैंकिंग और वित्तीय एप्लिकेशन)

- SBI, HDFC, ICICI जैसी बैंकिंग कंपनियाँ MySQL को अपने Core Banking सिस्टम में उपयोग करती हैं।
- उपयोग:
    - ट्रांज़ैक्शन डेटा स्टोरेज और सिक्योरिटी
    - ग्राहक खाता प्रबंधन
    - ऑनलाइन बैंकिंग डेटा हैंडलिंग

4. Content Management Systems (CMS - कंटेंट मैनेजमेंट सिस्टम)

- WordPress, Joomla, Drupal जैसे CMS MySQL का उपयोग करते हैं।
- उपयोग:
    - वेबपेज और ब्लॉग पोस्ट स्टोरेज
    - यूज़र प्रोफाइल और कमेंट मैनेजमेंट
    - SEO डेटा और साइट ट्रैकिंग

5. Healthcare and Hospital Management (स्वास्थ्य और अस्पताल प्रबंधन)

- अस्पतालों में Electronic Medical Records (EMR) स्टोर करने के लिए MySQL का उपयोग किया जाता है।
- उपयोग:
    - मरीजों की मेडिकल हिस्ट्री और रिपोर्ट स्टोरेज
    - डॉक्टर अपॉइंटमेंट मैनेजमेंट
    - फार्मेसी और दवा वितरण प्रणाली

14. MySQL Interview Questions and Answers (MySQL इंटरव्यू प्रश्न और उत्तर)

14.1 Basic MySQL Interview Questions (बेसिक MySQL इंटरव्यू प्रश्न)

Q1: MySQL क्या है?

उत्तर: MySQL एक **ओपन-सोर्स रिलेशनल डेटाबेस मैनेजमेंट सिस्टम (RDBMS)** है, जो डेटा को **टेबल्स में स्टोर** करता है और SQL (Structured Query Language) का उपयोग करता है।

Q2: MySQL और SQL में क्या अंतर है?

उत्तर:

MySQL	SQL
MySQL एक RDBMS सॉफ्टवेयर है।	SQL एक क्वेरी लैंग्वेज है।
यह डेटा को स्टोर और मैनेज करने के लिए SQL का उपयोग करता है।	इसका उपयोग डेटाबेस से डेटा एक्सेस करने के लिए किया जाता है।

Q3: MySQL में Primary Key और Foreign Key में क्या अंतर है?

उत्तर:

Primary Key	Foreign Key
यह टेबल में एक यूनिक आइडेंटिफायर होता है।	यह दूसरी टेबल के Primary Key को रेफर करता है।
एक टेबल में केवल एक Primary Key हो सकती है।	एक टेबल में एक से अधिक Foreign Keys हो सकते हैं।

14.2 Intermediate MySQL Interview Questions

(मध्यम स्तर के MySQL इंटरव्यू प्रश्न)

Q4: MySQL में Index क्या होता है और यह परफॉर्मेंस कैसे सुधारता है?

उत्तर: Index एक **डेटा स्ट्रक्चर** है, जो क्वेरीज को तेजी से एक्सीक्यूट करने में मदद करता है। यह **डेटाबेस सर्च ऑपरेशन्स को ऑप्टिमाइज़** करता है।

Example:

```
CREATE INDEX idx_name ON employees(name);
```

Q5: MySQL में Stored Procedure क्या होता है?

उत्तर: Stored Procedure एक **सेट ऑफ SQL स्टेटमेंट्स** होता है, जिसे बार-बार उपयोग किया जा सकता है।

Example:

```
DELIMITER //
CREATE PROCEDURE GetEmployees()
BEGIN
   SELECT * FROM employees;
END //
DELIMITER ;
```

Q6: MySQL में Joins क्या होते हैं?

उत्तर: Joins का उपयोग **एक से अधिक टेबल्स** से डेटा फेच करने के लिए किया जाता है।

Join Type	विवरण

INNER JOIN	**केवल मैचिंग रिकॉर्ड्स को रिट्रीव करता है।**
LEFT JOIN	**बाईं टेबल के सभी रिकॉर्ड्स और दाईं टेबल के मैचिंग रिकॉर्ड्स लाता है।**
RIGHT JOIN	**दाईं टेबल के सभी रिकॉर्ड्स और बाईं टेबल के मैचिंग रिकॉर्ड्स लाता है।**
FULL JOIN	**दोनों टेबल्स के सभी रिकॉर्ड्स को रिट्रीव करता है।**

Example of INNER JOIN:

```
SELECT employees.name, departments.department_name
FROM employees
INNER JOIN departments ON employees.department_id = departments.id;
```

14.3 Advanced MySQL Interview Questions (एडवांस MySQL इंटरव्यू प्रश्न)

Q7: ACID Properties क्या होती हैं?

उत्तर: ACID का मतलब Atomicity, Consistency, Isolation, Durability है।

- **Atomicity:** सभी ऑपरेशन पूरे होने चाहिए या कोई भी नहीं।
- **Consistency:** डेटा हमेशा वैध स्टेट में रहना चाहिए।
- **Isolation:** एक ट्रांज़ैक्शन दूसरे को प्रभावित नहीं करता।
- **Durability:** कमिट हुआ डेटा स्थायी होता है।

Q8: MySQL में Replication क्या होता है?

उत्तर: Replication का उपयोग **डेटाबेस डेटा को एक से अधिक सर्वर में कॉपी करने** के लिए किया जाता है।

Replication Type	विवरण
Master-Slave Replication	मास्टर सर्वर डेटा अपडेट करता है और स्लेव इसे कॉपी करता है।
Master-Master Replication	दोनों सर्वर डेटा अपडेट और कॉपी कर सकते हैं।

Q9: MySQL में Partitioning क्या होती है?

उत्तर: Partitioning एक तरीका है जिससे **टेबल डेटा को छोटे-छोटे भागों में बांटा जाता है, जिससे परफॉर्मेंस बेहतर होती है।**

Example:

```
CREATE TABLE employees (
    id INT NOT NULL,
    name VARCHAR(50),
    salary INT,
    PRIMARY KEY (id, name)
)
PARTITION BY RANGE (salary) (
    PARTITION p1 VALUES LESS THAN (30000),
    PARTITION p2 VALUES LESS THAN (60000)
);
```

Q10: MySQL Performance Optimization के लिए कौन-कौन से तरीके होते हैं?

उत्तर:

- **Indexing का सही उपयोग करें।**
- **Joins को ऑप्टिमाइज़ करें।**
- **Query Caching इनेबल करें।**
- **Partitioning का उपयोग करें।**
- **Large Queries को Optimize करें।**

15. Conclusion and Final Thoughts on MySQL (MySQL का निष्कर्ष और अंतिम विचार)

15.1 MySQL का महत्व (Importance of MySQL)

MySQL एक पावरफुल, स्केलेबल और सिक्योर रिलेशनल डेटाबेस मैनेजमेंट सिस्टम है, जिसे दुनिया भर की बड़ी और छोटी कंपनियाँ उपयोग करती हैं।

मुख्य विशेषताएँ:

✓ **ओपन-सोर्स** और उपयोग में आसान। ✓ **हाई परफॉर्मेंस** और बड़ी मात्रा में डेटा को संभालने की क्षमता। ✓ **ACID कंप्लायंट ट्रांजैक्शन्स** जो डेटा कंसिस्टेंसी बनाए रखती हैं। ✓ **JSON और Document Store सपोर्ट** जिससे NoSQL फीचर्स भी मिलते हैं। ✓

Replication और High Availability फीचर, जो इसे बड़े प्रोजेक्ट्स के लिए उपयुक्त बनाता है।

15.2 MySQL सीखने के फायदे (Benefits of Learning MySQL)

🎯 **डेटाबेस मैनेजमेंट स्किल्स** – SQL क्वेरीज लिखने और डेटाबेस को ऑप्टिमाइज़ करने की क्षमता। 🎯 **वेब डेवलपमेंट** – WordPress, Magento, Joomla जैसे प्लेटफॉर्म्स में MySQL का उपयोग। 🎯 **डेटा एनालिटिक्स और बिजनेस इंटेलिजेंस** – MySQL का उपयोग बड़े डेटा एनालिसिस में किया जाता है। 🎯 **बैंकिंग और फाइनेंस सेक्टर - सिक्योर और ट्रांजैक्शन-इंटेंसिव एप्लिकेशन में उपयोग।**

15.3 MySQL का भविष्य (Future of MySQL)

MySQL लगातार विकसित हो रहा है और नई-नई तकनीकों को अपनाकर और भी शक्तिशाली बनता जा रहा है।

🔹 **MySQL 8.0** में JSON सपोर्ट, Window Functions, और CTEs जैसी एडवांस्ड सुविधाएँ शामिल की गई हैं। 🔹 **Cloud-Based MySQL Solutions** – AWS RDS, Google Cloud SQL और Azure Database for MySQL जैसी क्लाउड सेवाएँ तेजी से लोकप्रिय हो रही हैं।

🔹 **AI & Machine Learning के साथ Integration - MySQL डेटा साइंस और ML मॉडल्स के लिए भी उपयोग हो रहा है।**

---

16. MySQL in Cloud Computing (MySQL का क्लाउड कंप्यूटिंग में उपयोग)

16.1 MySQL और क्लाउड कंप्यूटिंग का परिचय (Introduction to MySQL in Cloud Computing)

आज के समय में, MySQL को Cloud Platforms पर होस्ट करने का चलन बढ़ रहा है। Cloud Computing में MySQL को AWS RDS, Google Cloud SQL, और Azure Database for MySQL जैसे प्लेटफार्म्स पर होस्ट किया जाता है।

Cloud-Based MySQL Solutions के फायदे:

✓ **ऑटोमैटिक बैकअप और रिकवरी** – डेटा लॉस से बचाव। ✓ **हाई स्केलेबिलिटी** – जरूरत के अनुसार संसाधनों को बढ़ाया या घटाया जा सकता है। ✓ **हाई अवेलेबिलिटी** – 99.99% अपटाइम और मल्टी-रीजन डेटा सेंटर सपोर्ट। ✓ **लो मेंटेनेंस - डेटाबेस सर्वर को मैन्युअली मैनेज करने की जरूरत नहीं।**

16.2 AWS RDS for MySQL (Amazon RDS पर MySQL)

AWS RDS (Relational Database Service) MySQL के लिए एक Fully Managed Cloud Solution है।

AWS RDS पर MySQL सेटअप करने के स्टेप्स:

1️⃣ **AWS Management Console** पर लॉग इन करें। 2️⃣ **RDS Service** खोलें और "Create Database" पर क्लिक करें। 3️⃣ **Database Engine** में "MySQL" चुनें। 4️⃣ **Instance Type** और स्टोरेज आवश्यकताओं को सेट करें। 5️⃣ **Backup और High Availability** के विकल्पों को कॉन्फ़िगर करें। 6️⃣ **Database Credentials सेट करें और "Create Database" पर क्लिक करें।**

AWS RDS से कनेक्ट करने के लिए:

mysql -h your-db-instance.rds.amazonaws.com -u admin -p

इसका प्रभाव: आपका AWS RDS इंस्टेंस `mysql` क्लाइंट से कनेक्ट हो जाएगा।

16.3 Google Cloud SQL for MySQL (Google Cloud पर MySQL सेटअप करना)

Google Cloud SQL एक Fully Managed MySQL सर्विस है, जो ऑटो स्केलिंग और हाई परफॉर्मेंस प्रदान करती है।

Google Cloud SQL में MySQL सेटअप करने के स्टेप्स:

1️⃣ **Google Cloud Console** खोलें और "Cloud SQL" में जाएं। 2️⃣ **Create Instance** पर क्लिक करें और "MySQL" चुनें। 3️⃣ **Machine Type और Storage** सेट करें। 4️⃣ **Authorization और Networking ऑप्शन** सेट करें। 5️⃣ "Create" पर क्लिक करें और डेटाबेस इंस्टेंस बनाएं।

Cloud SQL से कनेक्ट करने के लिए:

gcloud sql connect your-instance-name --user=root

16.4 Azure Database for MySQL (Microsoft Azure पर MySQL होस्टिंग)

Azure Database for MySQL एक Highly Available, Fully Managed MySQL सर्विस है, जो Enterprises के लिए एक बेहतरीन समाधान है।

Azure पर MySQL सेटअप करने के स्टेप्स:

1️⃣ **Azure Portal** खोलें और "Create a Resource" पर क्लिक करें। 2️⃣ "Azure Database for MySQL" सर्च करें और चुनें। 3️⃣ **Database Tier और Storage** विकल्प सेट करें। 4️⃣ **Firewall और Networking सेटअप करें** ताकि आपकी एप्लिकेशन इसे एक्सेस कर सके। 5️⃣ **Create पर क्लिक करें और डेटाबेस इंस्टेंस बनाएं।**

Azure MySQL से कनेक्ट करने के लिए:

mysql -h your-server-name.mysql.database.azure.com -u admin -p

---

16.5 Cloud-Based MySQL का भविष्य (Future of Cloud MySQL)

✅ **Serverless MySQL** – AWS Aurora Serverless और Google Cloud SQL Serverless जैसी सेवाएँ अधिक लोकप्रिय हो रही हैं। ✅ **AI-Powered Database**

Optimization – क्लाउड AI-बेस्ड डेटाबेस ट्यूनिंग प्रदान कर रहा है। ✅ **Multi-Cloud Database Management – कंपनियाँ AWS, Google Cloud, और Azure को एक साथ उपयोग कर रही हैं।**

17. MySQL and Python Integration (MySQL और Python का इंटीग्रेशन)

17.1 Python और MySQL का परिचय (Introduction to Python and MySQL)

Python **एक पॉपुलर प्रोग्रामिंग लैंग्वेज है** और MySQL **एक रिलेशनल डेटाबेस।** जब इन दोनों को जोड़ा जाता है, तो यह डेटा स्टोरेज और प्रोसेसिंग को ऑटोमेट करने में मदद करता है।

Python और MySQL इंटीग्रेशन के फायदे:

✓ डेटा कोड के माध्यम से मैनेज करना आसान ✓ वेब एप्लिकेशन और डेटा साइंस में उपयोग ✓ बड़े पैमाने पर ऑटोमेशन संभव ✓ डेटाबेस से डायनामिक डेटा एक्सेस

17.2 MySQL Connector for Python (Python में MySQL कनेक्टर का उपयोग)

Python और MySQL को जोड़ने के लिए `mysql-connector-python` लाइब्रेरी का उपयोग किया जाता है।

इंस्टॉलेशन:

```
pip install mysql-connector-python
```

17.3 Connecting to MySQL from Python (Python से MySQL को कनेक्ट करना)

```
import mysql.connector


# MySQL से कनेक्ट करें
conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="password",
    database="test_db"
)
```

 # कर्सर ऑब्जेक्ट बनाएं

cursor = conn.cursor()

print("MySQL Connection Successful!")

इसका प्रभाव:

- **यह स्क्रिप्ट MySQL से कनेक्शन स्थापित करेगी।**
- `cursor()` **ऑब्जेक्ट डेटा एक्सेस और मैनेज करने के लिए उपयोग होगा।**

17.4 Creating and Managing Tables (टेबल बनाना और मैनेज करना)

cursor.execute("CREATE TABLE IF NOT EXISTS employees (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(100), age INT)")

print("Table Created Successfully!")

इसका प्रभाव: `employees` **नाम की टेबल बन जाएगी।**

17.5 Inserting Data into MySQL using Python (Python से डेटा इंसर्ट करना)

query = "INSERT INTO employees (name, age) VALUES (%s, %s)"

data = ("Rahul", 28)

```
cursor.execute(query, data)

conn.commit()

print("Data Inserted Successfully!")
```

इसका प्रभाव: यह स्क्रिप्ट `employees` टेबल में नया डेटा जोड़ेगी।

17.6 Fetching Data from MySQL using Python (Python से MySQL डेटा रिट्रीव करना)

```
cursor.execute("SELECT * FROM employees")

for row in cursor.fetchall():

    print(row)
```

इसका प्रभाव: यह स्क्रिप्ट `employees` टेबल से सभी डेटा को प्रिंट करेगी।

17.7 Updating and Deleting Data in MySQL (Python से डेटा अपडेट और डिलीट करना)

```
# डेटा अपडेट करना

update_query = "UPDATE employees SET age = 30 WHERE name = 'Rahul'"

cursor.execute(update_query)

conn.commit()
```

 # डेटा डिलीट करना

```
delete_query = "DELETE FROM employees WHERE name = 'Rahul'"
cursor.execute(delete_query)
conn.commit()
```

इसका प्रभाव:

- `UPDATE` क्वेरी **Rahul की उम्र 30 सेट** कर देगी।
- `DELETE` क्वेरी **Rahul को टेबल से हटा** देगी।

17.8 Closing the MySQL Connection (MySQL कनेक्शन बंद करना)

```
cursor.close()
conn.close()
print("MySQL Connection Closed.")
```

इसका प्रभाव: यह डेटाबेस से कनेक्शन सुरक्षित रूप से बंद कर देगा।

18. MySQL for Big Data (बड़े डेटा के लिए MySQL)

18.1 Big Data और MySQL का परिचय (Introduction to Big Data and MySQL)

Big Data बड़े पैमाने पर डेटा को स्टोर, प्रोसेस और विश्लेषण करने की प्रक्रिया है। MySQL, Big Data समाधान के रूप में **डेटा मैनेजमेंट, स्केलेबिलिटी और हाई परफॉर्मेंस** प्रदान करता है।

Big Data में MySQL के उपयोग के फायदे:

✓ रिलेशनल डेटा को हैंडल करने की क्षमता ✓ SQL क्वेरी सपोर्ट के साथ डेटा प्रोसेसिंग ✓ स्केलेबल और हाई परफॉर्मेंस स्टोरेज ✓ Cloud-Based Big Data समाधान

18.2 MySQL और Hadoop का इंटीग्रेशन (Integrating MySQL with Hadoop)

Hadoop एक Big Data Framework है जो डिस्ट्रिब्यूटेड स्टोरेज और प्रोसेसिंग प्रदान करता है। MySQL को Hadoop के साथ ETL (Extract, Transform, Load) प्रक्रिया में उपयोग किया जा सकता है।

MySQL से Hadoop में डेटा माइग्रेट करने के लिए Sqoop का उपयोग:

```
sqoop import --connect jdbc:mysql://localhost/test_db \
--username root --password password \
```

--table employees --target-dir /hadoop/employees

इसका प्रभाव: यह MySQL से डेटा को Hadoop HDFS में माइग्रेट करेगा।

18.3 MySQL और Apache Spark (MySQL with Apache Spark)

Apache Spark एक फास्ट डेटा प्रोसेसिंग इंजन है, जिसे MySQL डेटा को एनालाइज़ करने के लिए उपयोग किया जाता है।

MySQL से Spark में डेटा लोड करना:

```
from pyspark.sql import SparkSession


spark = SparkSession.builder.appName("MySQL_Spark_Integration").getOrCreate()

df = spark.read.format("jdbc").option("url", "jdbc:mysql://localhost:3306/test_db") \
    .option("dbtable", "employees").option("user", "root").option("password", "password").load()

df.show()
```

इसका प्रभाव: यह `employees` टेबल से डेटा लोड करके Spark में प्रोसेस करेगा।

18.4 MySQL Sharding for Big Data (बड़े डेटा के लिए MySQL शार्डिंग)

Sharding एक तकनीक है, जिसमें बड़े डेटा को अलग-अलग सर्वर्स पर विभाजित किया जाता है ताकि परफॉर्मेंस में सुधार हो।

Sharding के फायदे:

✓ **हाई स्केलेबिलिटी** – डेटा को छोटे-छोटे भागों में विभाजित किया जाता है। ✓ **लोड बैलेंसिंग** – सभी सर्वर पर ट्रैफिक समान रूप से बंटता है। ✓ **फास्ट क्वेरी प्रोसेसिंग** - कम लोड वाले सर्वर से डेटा जल्दी एक्सेस होता है।

18.5 NoSQL और MySQL का संयोजन (Combining MySQL with NoSQL for Big Data)

Big Data में MySQL और NoSQL (जैसे MongoDB, Cassandra) को एक साथ उपयोग किया जा सकता है।

MySQL + NoSQL का उपयोग कब करें?

- **रिलेशनल डेटा** – MySQL
- **अनस्ट्रक्चर्ड डेटा (जैसे JSON, XML)** – NoSQL
- **रीड-हेवी एप्लिकेशन** – NoSQL
- **ट्रांज़ैक्शनल डेटा** – MySQL

19. Stored Functions in MySQL (MySQL में स्टोर्ड फंक्शंस)

19.1 MySQL में स्टोर्ड फंक्शन क्या होता है? (What is a Stored Function in MySQL?)

Stored Function एक **यूज़र-डिफाइंड SQL फंक्शन** होता है, जिसे बार-बार उपयोग किया जा सकता है। यह एक वैल्यू रिटर्न करता है और SQL क्वेरीज के भीतर उपयोग किया जा सकता है।

Stored Function के फायदे:

✓ **कोड रीयूज़ेबिलिटी** – एक बार लिखकर कई बार उपयोग किया जा सकता है। ✓ **परफॉर्मेंस सुधार** – क्वेरी ऑप्टिमाइज़ेशन में मदद करता है। ✓ **डेटा प्रोसेसिंग ऑटोमेशन** - बार-बार उपयोग होने वाले लॉजिक को फंक्शन में स्टोर किया जा सकता है।

19.2 Creating a Stored Function (MySQL में स्टोर्ड फंक्शन बनाना)

सिंटैक्स:

```
DELIMITER //
CREATE FUNCTION function_name(param1 DATATYPE, param2 DATATYPE)
RETURNS RETURN_DATATYPE
DETERMINISTIC
BEGIN
    -- Function Logic Here
    RETURN value;
END //
DELIMITER ;
```

19.3 Example: Calculating Bonus (बोनस कैलकुलेशन के लिए फंक्शन बनाना)

```
DELIMITER //
CREATE FUNCTION CalculateBonus(salary INT)
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE bonus INT;
    SET bonus = salary * 0.10;
    RETURN bonus;
END //
```

DELIMITER ;

इसका प्रभाव: यह फंक्शन **कर्मचारियों के सैलरी के आधार पर बोनस** कैलकुलेट करेगा।

Stored Function को कॉल करना:

SELECT name, salary, CalculateBonus(salary) AS bonus FROM employees;

19.4 Modifying and Deleting Stored Functions (स्टोर्ड फंक्शंस को अपडेट और डिलीट करना)

Existing Function को अपडेट करना:

MySQL में किसी स्टोर्ड फंक्शन को **डायरेक्टली मॉडिफाई नहीं किया जा सकता**। पहले इसे हटाना होगा और फिर दोबारा बनाना होगा।

DROP FUNCTION IF EXISTS CalculateBonus;

इसके बाद आप नया फंक्शन क्रिएट कर सकते हैं।

19.5 Using Stored Functions in Queries (स्टोर्ड फंक्शन का उपयोग क्वेरीज में)

SELECT employee_id, name, salary, CalculateBonus(salary) FROM employees WHERE salary > 50000;

इसका प्रभाव: यह उन कर्मचारियों की सैलरी और बोनस दिखाएगा जिनकी सैलरी 50,000 से अधिक है।

19.6 Restrictions on Stored Functions (स्टोर्ड फंक्शन की सीमाएँ)

❌ **Stored Function में ट्रांज़ैक्शन्स (`COMMIT`, `ROLLBACK`) का उपयोग नहीं किया जा सकता।** ❌ **Stored Function को `INSERT`, `UPDATE`, `DELETE` ऑपरेशन्स के लिए उपयोग नहीं किया जाना चाहिए।** ❌ **Stored Procedures की तुलना में यह कुछ मामलों में धीमा हो सकता है।**

20. MySQL and Data Visualization (MySQL और डेटा विज़ुअलाइज़ेशन)

20.1 डेटा विज़ुअलाइज़ेशन क्या है? (What is Data Visualization?)

डेटा विज़ुअलाइज़ेशन वह प्रक्रिया है जिसमें डेटा को **ग्राफ, चार्ट, और डैशबोर्ड** के रूप में प्रस्तुत किया जाता है, जिससे डेटा को समझना और निर्णय लेना आसान हो जाता है। MySQL डेटा को विज़ुअलाइज़ करने के लिए कई टूल्स और तकनीकों का उपयोग किया जाता है।

डेटा विज़ुअलाइज़ेशन के फायदे:

✓ **डेटा को आसानी से समझना** – संख्यात्मक डेटा को ग्राफ और चार्ट के रूप में देख सकते हैं। ✓ **बेहतर निर्णय लेना** – ट्रेंड्स और पैटर्न्स को जल्दी पहचाना जा सकता है। ✓ **डेटा एनालिसिस को आसान बनाना - रिपोर्टिंग और बिजनेस इंटेलिजेंस में उपयोगी।**

20.2 MySQL के लिए लोकप्रिय डेटा विज़ुअलाइज़ेशन टूल्स (Popular Data Visualization Tools for MySQL)

टूल	मुख्य विशेषताएँ
Tableau	इंटरएक्टिव डैशबोर्ड और ड्रैग-एंड-ड्रॉप इंटरफ़ेस
Power BI	Microsoft द्वारा डेवलप्ड, बिजनेस एनालिटिक्स के लिए उपयोगी
Google Data Studio	फ्री टूल, जिसे गूगल प्रोडक्ट्स के साथ इंटीग्रेट किया जा सकता है
Grafana	ओपन-सोर्स टूल, लाइव डेटा विज़ुअलाइज़ेशन के लिए उपयुक्त

Metabase	**आसान SQL क्वेरी इंटरफ़ेस और ऑटोमेटेड ग्राफ्स**

20.3 Tableau के साथ MySQL को कनेक्ट करना (Connecting MySQL with Tableau)

Tableau एक शक्तिशाली डेटा विज़ुअलाइज़ेशन टूल है। इसे MySQL के साथ कनेक्ट करने के लिए: 1️⃣ **Tableau खोलें** और "Connect to Data" ऑप्शन चुनें। 2️⃣ **MySQL Connector** का चयन करें। 3️⃣ **Host, Username, Password, और Database Name** डालें। 4️⃣ "Connect" पर क्लिक करें और अपने डेटा को चार्ट्स में कन्वर्ट करें।

20.4 Power BI के साथ MySQL को कनेक्ट करना (Connecting MySQL with Power BI)

1️⃣ **Power BI Desktop खोलें**। 2️⃣ "Get Data" पर जाएं और "MySQL Database" चुनें। 3️⃣ **Server Name और Database Name** दर्ज करें। 4️⃣ डेटा लोड करें और विज़ुअलाइज़ेशन बनाएं।

20.5 SQL Queries से डेटा विज़ुअलाइज़ेशन (Data Visualization using SQL Queries)

अगर आप बिना किसी टूल के MySQL डेटा का विश्लेषण करना चाहते हैं, तो SQL क्वेरी का उपयोग करें:

```
SELECT department, COUNT(*) AS total_employees
FROM employees
GROUP BY department;
```

इसका प्रभाव: यह हर डिपार्टमेंट में कर्मचारियों की संख्या दिखाएगा, जिसे बाद में ग्राफ में कन्वर्ट किया जा सकता है।

20.6 Google Data Studio के साथ MySQL डेटा विज़ुअलाइज़ करना

Google Data Studio एक फ्री ऑनलाइन टूल है जो MySQL डेटा को रिपोर्ट और डैशबोर्ड में बदलता है। 1️⃣ **Google Data Studio खोलें** और "Create New Report" पर क्लिक करें। 2️⃣ "MySQL Connector" जोड़ें और डेटाबेस डिटेल्स डालें। 3️⃣ डेटा को चार्ट्स और ग्राफ्स में विज़ुअलाइज़ करें।

20.7 लाइव डेटा विज़ुअलाइज़ेशन के लिए Grafana का उपयोग

Grafana एक ओपन-सोर्स टूल है जो लाइव MySQL डेटा को मॉनिटरिंग डैशबोर्ड में बदल सकता है। 1️⃣ **Grafana इंस्टॉल करें** और "Data Sources" में जाएं। 2️⃣ "MySQL" को जोड़ें और डेटाबेस डिटेल्स डालें। 3️⃣ **रियल-टाइम डेटा ग्राफ्स और डैशबोर्ड बनाएं।**
