**Batch -** T7

**Assignment No. -** 8

**Title -** Configuration of MySQL for distributed databases

**Student Name -** Sharaneshwar Bharat Punjal

**Student PRN -** 23520011

1. Do the installation and configuration of Oracle/MySQL/IBM DB2 distributed databases.

[Take minimum 3 nodes]

2. Use the above installed distributed database as backend for Assignment No. 3 & 4.

3. Demonstrate the working by connecting portals to different nodes, adding data from one node and available / display on portal connected to another node etc.

**MySQL Distributed Database Setup on Windows**

**Prerequisites**

Ensure we have the following on all three machines:

* MySQL Server installed.
* Network connectivity between the machines.
* Administrator privileges to modify MySQL configurations.

**Install MySQL on All Three Machines**

1. Download and Install MySQL:
   * Go to: [MySQL Downloads](https://dev.mysql.com/downloads/installer/).
   * Download the MySQL Installer.
   * Select MySQL Server and Workbench during installation.
2. Verify Installation:
   * Open Command Prompt.
   * Run: mysql --version

**Network Configuration**

* Assign Static IPs to the three machines:
  + Master: 192.168.1.10
  + Slave 1: 192.168.1.11
  + Slave 2: 192.168.1.12
* Allow MySQL Through Firewall:
  + Go to Windows Defender Firewall → Advanced Settings → Inbound Rules.
  + Add a new rule to allow port 3306 for MySQL.

A screenshot of a computer

AI-generated content may be incorrect.

**Configure the Master Node (Machine 1)**

**Step 1: Modify MySQL Configuration**

1. Open MySQL configuration file:

Location: C:\ProgramData\MySQL\MySQL Server x.x\my.ini

1. Add the following configuration under [mysqld]:

[mysqld]

server-id = 1

log-bin = mysql-bin

binlog-do-db = distributed\_db

1. Restart MySQL Service:

net stop mysql

net start mysql

**Step 2: Create a Replication User**

1. Open MySQL CLI: mysql -u root -p
2. Create a replication user:

CREATE USER 'replicator'@'%' IDENTIFIED BY 'password';

GRANT REPLICATION SLAVE ON \*.\* TO 'replicator'@'%';

FLUSH PRIVILEGES;

**Step 3. Get the Master Status**

Run the following command to get the master log file and position:

SHOW MASTER STATUS;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

**Configure Slave 1 (Machine 2)**

**Step 1: Modify MySQL Configuration**

1. Open my.ini file: C:\ProgramData\MySQL\MySQL Server x.x\my.ini
2. Add the following configuration:

[mysqld]

server-id = 2

relay-log = relay-log

1. Restart MySQL Service:

net stop mysql

net start mysql

**Step 2: Connect Slave 1 to Master**

1. Open MySQL CLI:

mysql -u root -p

1. Configure the slave with the master details:

CHANGE MASTER TO

MASTER\_HOST='192.168.1.10',

MASTER\_USER='replicator',

MASTER\_PASSWORD='password',

MASTER\_LOG\_FILE='mysql-bin.000001',

MASTER\_LOG\_POS=154;

1. Start the Slave:

START SLAVE;

Repeat the similar process for Slave 2 with its IP address.

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen with white text

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**Verify Replication Status**

On both Slave 1 and Slave 2, check the replication status:

SHOW SLAVE STATUS\G;

Check the following:

* Slave\_IO\_Running: Yes
* Slave\_SQL\_Running: Yes
* No replication errors.

A screenshot of a computer

AI-generated content may be incorrect.

**Testing Distributed Database**

1. **On Master:**

Create a database and table:

CREATE DATABASE distributed\_db;

USE distributed\_db;

CREATE TABLE employees (

id INT PRIMARY KEY,

name VARCHAR(50),

salary DECIMAL(10, 2)

);

INSERT INTO employees VALUES (1, 'Alice', 60000);

1. **On Slave 1 and Slave 2:**

Verify that the data is replicated:

USE distributed\_db;

SELECT \* FROM employees;

**Conclusion**

We have successfully set up a MySQL distributed database on three Windows machines using Master-Slave replication. This setup ensures data redundancy, failover protection, and better scalability.