MySQL Multi-Server Cron Job Monitoring – Complete Steps (.txt)

This document explains **ALL STEPS from zero to final result** for building a **SQL Server Agent–like job monitoring system for MySQL** using **Python + Cron + Streamlit**, supporting **multiple servers**.

---

1. FINAL RESULT (WHAT YOU WILL HAVE)

✔️ Each MySQL/Linux server runs its own cron jobs ✔️ Each server has a small Python agent (`run_agent.py`) ✔️ Every job execution (SUCCESS / FAILED) is stored in ONE central MySQL database ✔️ A Streamlit dashboard shows job status for ALL servers ✔️ Works like SQL Server Agent Job History

---

2. ARCHITECTURE OVERVIEW

```
Server-1 (172.24.11.82)
 ├─ cron
 ├─ run_agent.py
 └─ writes status → monitoring DB

Server-2 (172.24.11.83)
 ├─ cron
 ├─ run_agent.py
 └─ writes status → monitoring DB

Server-3 (172.24.11.84)
 ├─ cron
 ├─ run_agent.py
 └─ writes status → monitoring DB

Streamlit Dashboard
 └─ reads from monitoring DB
```

---

3. DATABASE SETUP (ONE TIME ONLY)

Login to MySQL on the monitoring server.

3.1 Create database

```sql
CREATE DATABASE monitoring;
```

3.2 Create job status table

```sql
CREATE TABLE job_status (
    id INT AUTO_INCREMENT PRIMARY KEY,
    server_ip VARCHAR(50),
    job_name VARCHAR(100),
    status ENUM('SUCCESS','FAILED'),
    message TEXT,
    run_time DATETIME
);
```

This table stores job history from ALL servers.

---

4. INSTALL PYTHON REQUIREMENTS (ON EACH SERVER)

```
sudo apt update
sudo apt install -y python3 python3-pip
pip3 install mysql-connector-python streamlit pandas
```

---

5. AGENT SCRIPT (ON EACH SERVER)

5.1 File location

```
/usr/local/bin/run_agent.py
```

5.2 VERY IMPORTANT

The **first line must be the shebang**. No blank line above it.

5.3 Full agent code

```python
#!/usr/bin/python3

import mysql.connector
from datetime import datetime
```

```python
import socket

SERVER_IP = socket.gethostname()

def run_job():
    try:
        # PUT YOUR REAL JOB LOGIC HERE
        # Example: backup check, script execution, etc.
        status = "SUCCESS"
        message = "Job completed successfully"

    except Exception as e:
        status = "FAILED"
        message = str(e)

    conn = mysql.connector.connect(
        host="172.24.11.82",
        user="sqlagent",
        password="sql@g3nt",
        database="monitoring"
    )

    cursor = conn.cursor()

    cursor.execute("""
        INSERT INTO job_status (server_ip, job_name, status, message, run_time)
        VALUES (%s, %s, %s, %s, %s)
    """, (
        SERVER_IP,
        "Daily Cron Job",
        status,
        message,
        datetime.now()
    ))

    conn.commit()
    cursor.close()
    conn.close()

if __name__ == "__main__":
    run_job()
```

---

6. FILE PERMISSION & TEST (ON EACH SERVER)

```
chmod +x /usr/local/bin/run_agent.py
```

Test manually:

```
/usr/bin/python3 /usr/local/bin/run_agent.py
```

Verify insert:

```sql
SELECT * FROM job_status ORDER BY run_time DESC;
```

---

7. CONFIGURE CRON (ON EACH SERVER)

Edit crontab:

```
crontab -e
```

Example: run every 5 minutes

```
*/5 * * * * /usr/bin/python3 /usr/local/bin/run_agent.py
```

Each server runs its OWN cron job.

---

8. STREAMLIT DASHBOARD (CENTRAL SERVER)

8.1 Dashboard file

```
dashboard.py
```

8.2 Full Streamlit code

```python
import streamlit as st
import mysql.connector
import pandas as pd
```

```
st.set_page_config(page_title="SQL Agent Dashboard", layout="wide")
st.title("Cron Job Monitoring Dashboard")

conn = mysql.connector.connect(
    host="172.24.11.82",
    user="sqlagent",
    password="sql@g3nt",
    database="monitoring"
)

query = """
SELECT server_ip, job_name, status, message, run_time
FROM job_status
ORDER BY run_time DESC
LIMIT 100
"""

df = pd.read_sql(query, conn)
conn.close()

if not df.empty:
    servers = df["server_ip"].unique()
    selected_server = st.selectbox("Select Server", ["ALL"] + list(servers))

    if selected_server != "ALL":
        df = df[df["server_ip"] == selected_server]

    st.dataframe(df, use_container_width=True)

    latest = df.iloc[0]
    if latest["status"] == "SUCCESS":
        st.success(f"Last Job SUCCESS on {latest['server_ip']}")
    else:
        st.error(f"Last Job FAILED on {latest['server_ip']}")
else:
    st.warning("No job data found")
```

Run dashboard:

```
streamlit run dashboard.py
```

---

9. WHY AGENT IS REQUIRED ON EACH SERVER

- Cron runs locally

- Job success/failure is local
- Logs are local
- Central server cannot detect this remotely

This is how real tools work: - SQL Server Agent - Zabbix Agent - Prometheus Node Exporter

---

10. FINAL SUMMARY

✔️ One Python agent per server ✔️One central MySQL monitoring database ✔️Cron executes jobs ✔️ Agent records job status ✔️Streamlit shows real-time job history

You have successfully built a **production-style MySQL job monitoring system**.

END OF FILE