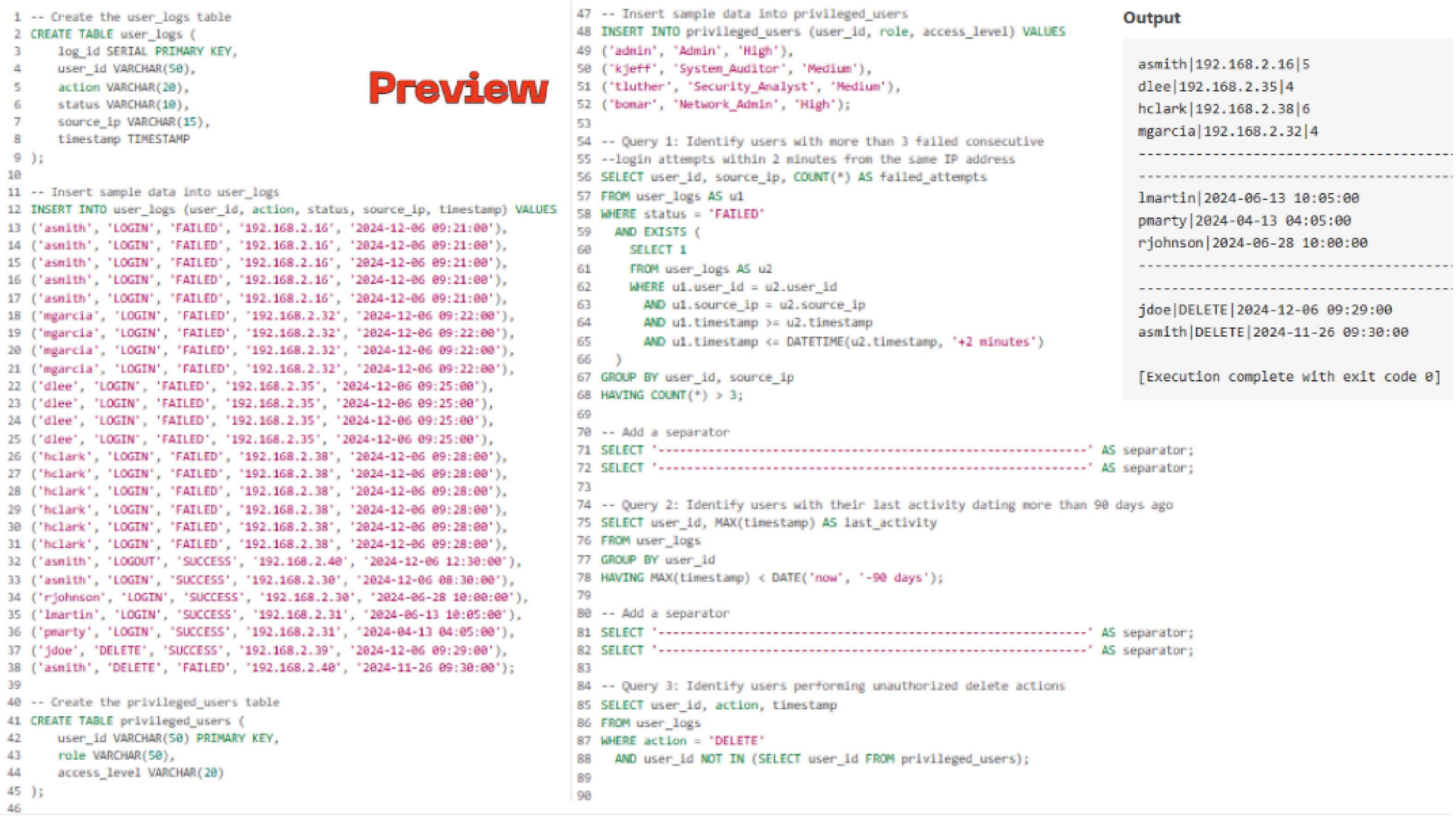
IT Audit/GRC Lab – Following NIST 800–53 Standards – Using MS SQL Database



```sql
-- Create the user_logs table
CREATE TABLE user_logs (
    log_id SERIAL PRIMARY KEY,
    user_id VARCHAR(50),
    action VARCHAR(20),
    status VARCHAR(10),
    source_ip VARCHAR(15),
    timestamp TIMESTAMP
);

-- Insert sample data into user_logs
INSERT INTO user_logs (user_id, action, status, source_ip, timestamp) VALUES
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('asmith', 'LOGOUT', 'SUCCESS', '192.168.2.40', '2024-12-06 12:30:00'),
('asmith', 'LOGIN', 'SUCCESS', '192.168.2.30', '2024-12-06 08:30:00'),
('rjohnson', 'LOGIN', 'SUCCESS', '192.168.2.30', '2024-06-28 10:00:00'),
('lmartin', 'LOGIN', 'SUCCESS', '192.168.2.31', '2024-06-13 10:05:00'),
('pmarty', 'LOGIN', 'SUCCESS', '192.168.2.31', '2024-04-13 04:05:00'),
('jdoe', 'DELETE', 'SUCCESS', '192.168.2.39', '2024-12-06 09:29:00'),
('asmith', 'DELETE', 'FAILED', '192.168.2.40', '2024-11-26 09:30:00');

-- Create the privileged_users table
CREATE TABLE privileged_users (
    user_id VARCHAR(50) PRIMARY KEY,
    role VARCHAR(50),
    access_level VARCHAR(20)
);

-- Insert sample data into privileged_users
INSERT INTO privileged_users (user_id, role, access_level) VALUES
('admin', 'Admin', 'High'),
('kjeff', 'System_Auditor', 'Medium'),
('tluther', 'Security_Analyst', 'Medium'),
('bonar', 'Network_Admin', 'High');

-- Query 1: Identify users with more than 3 failed consecutive
--login attempts within 2 minutes from the same IP address
SELECT user_id, source_ip, COUNT(*) AS failed_attempts
FROM user_logs AS u1
WHERE status = 'FAILED'
  AND EXISTS (
    SELECT 1
    FROM user_logs AS u2
    WHERE u1.user_id = u2.user_id
      AND u1.source_ip = u2.source_ip
      AND u1.timestamp >= u2.timestamp
      AND u1.timestamp <= DATETIME(u2.timestamp, '+2 minutes')
  )
GROUP BY user_id, source_ip
HAVING COUNT(*) > 3;

-- Add a separator
SELECT '------------------------------------------------------------' AS separator;
SELECT '------------------------------------------------------------' AS separator;

-- Query 2: Identify users with their last activity dating more than 90 days ago
SELECT user_id, MAX(timestamp) AS last_activity
FROM user_logs
GROUP BY user_id
HAVING MAX(timestamp) < DATE('now', '-90 days');

-- Add a separator
SELECT '------------------------------------------------------------' AS separator;
SELECT '------------------------------------------------------------' AS separator;

-- Query 3: Identify users performing unauthorized delete actions
SELECT user_id, action, timestamp
FROM user_logs
WHERE action = 'DELETE'
  AND user_id NOT IN (SELECT user_id FROM privileged_users);
```

Output

```
asmith|192.168.2.16|5
dlee|192.168.2.35|4
hclark|192.168.2.38|6
mgarcia|192.168.2.32|4
------------------------------------------------------------
------------------------------------------------------------
lmartin|2024-06-13 10:05:00
pmarty|2024-04-13 04:05:00
rjohnson|2024-06-28 10:00:00
------------------------------------------------------------
------------------------------------------------------------
jdoe|DELETE|2024-12-06 09:29:00
asmith|DELETE|2024-11-26 09:30:00

[Execution complete with exit code 0]
```

Requirements

This lab simulates IT Audit and GRC scenarios where the audit team analyzes logs directly without access to a centralized SIEM system. The setup includes:

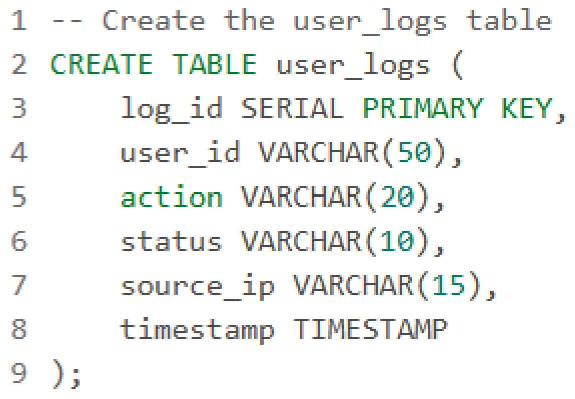
- A SQL database with two tables:
    1. **user_logs**: Records user activity (actions, status, IPs, timestamps).
    2. **privileged_users**: Lists authorized users for sensitive actions.

Objective

The lab demonstrates key IT Audit tasks aligned with **NIST 800-53** controls:

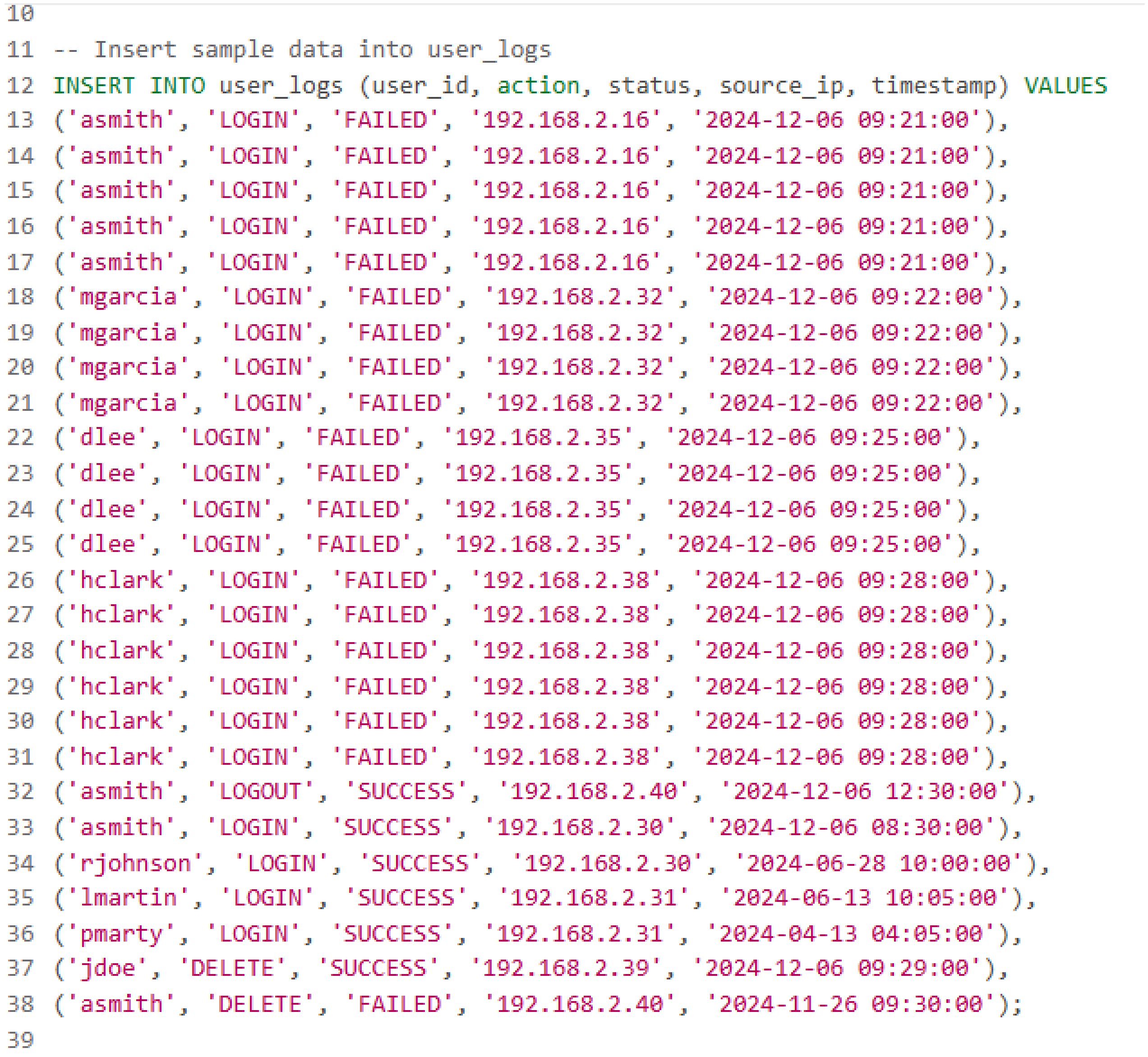
1. **AC-7 (Failed Login Attempts)**: Detects users exceeding failed login thresholds.
2. **AC-2 (Account Management)**: Identify inactive accounts over 90 days.
3. **AU-2 & AC-6 (Privileged Actions)**: Monitor unauthorized sensitive actions.

First, we will create a table containing all the necessary fields for our lab example. This table is designed to simulate a user activity log typically found in an organizational environment.



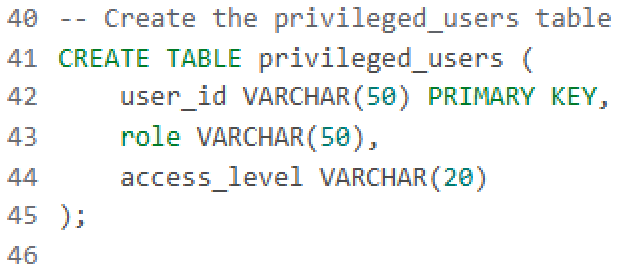
```sql
-- Create the user_logs table
CREATE TABLE user_logs (
    log_id SERIAL PRIMARY KEY,
    user_id VARCHAR(50),
    action VARCHAR(20),
    status VARCHAR(10),
    source_ip VARCHAR(15),
    timestamp TIMESTAMP
);
```

Second, we will create usernames for employees using a traditional alias format: the first letter of their first name followed by their last name. Note: all information, including usernames, IP addresses, and timestamps, is fictional and provided solely to create a sample dataset for this lab.

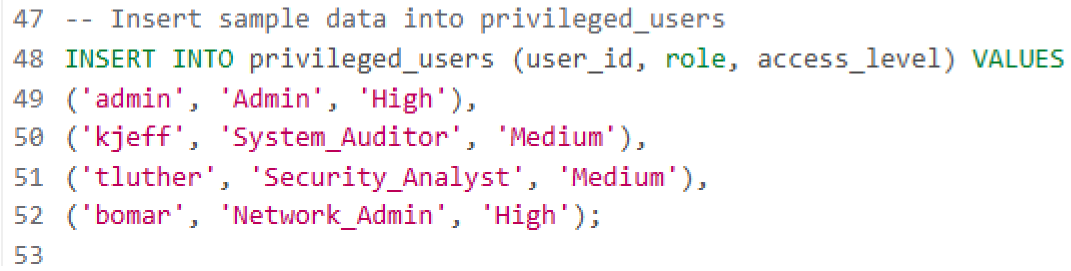


```sql

-- Insert sample data into user_logs
INSERT INTO user_logs (user_id, action, status, source_ip, timestamp) VALUES
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('asmith', 'LOGIN', 'FAILED', '192.168.2.16', '2024-12-06 09:21:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('mgarcia', 'LOGIN', 'FAILED', '192.168.2.32', '2024-12-06 09:22:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('dlee', 'LOGIN', 'FAILED', '192.168.2.35', '2024-12-06 09:25:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('hclark', 'LOGIN', 'FAILED', '192.168.2.38', '2024-12-06 09:28:00'),
('asmith', 'LOGOUT', 'SUCCESS', '192.168.2.40', '2024-12-06 12:30:00'),
('asmith', 'LOGIN', 'SUCCESS', '192.168.2.30', '2024-12-06 08:30:00'),
('rjohnson', 'LOGIN', 'SUCCESS', '192.168.2.30', '2024-06-28 10:00:00'),
('lmartin', 'LOGIN', 'SUCCESS', '192.168.2.31', '2024-06-13 10:05:00'),
('pmarty', 'LOGIN', 'SUCCESS', '192.168.2.31', '2024-04-13 04:05:00'),
('jdoe', 'DELETE', 'SUCCESS', '192.168.2.39', '2024-12-06 09:29:00'),
('asmith', 'DELETE', 'FAILED', '192.168.2.40', '2024-11-26 09:30:00');

```

3. We will create a table for privileged users to identify which users are authorized to perform privileged actions, such as deleting or modifying information.



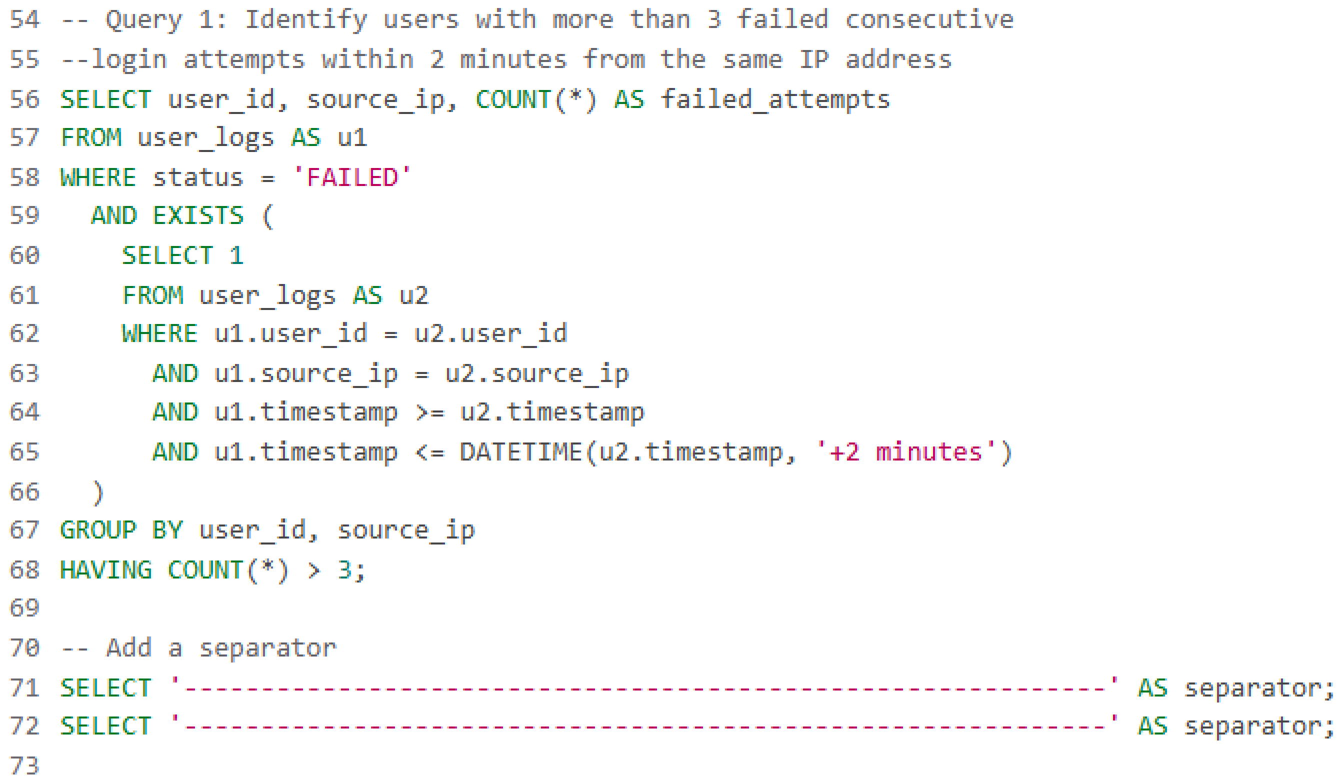
```sql
-- Create the privileged_users table
CREATE TABLE privileged_users (
    user_id VARCHAR(50) PRIMARY KEY,
    role VARCHAR(50),
    access_level VARCHAR(20)
);

```



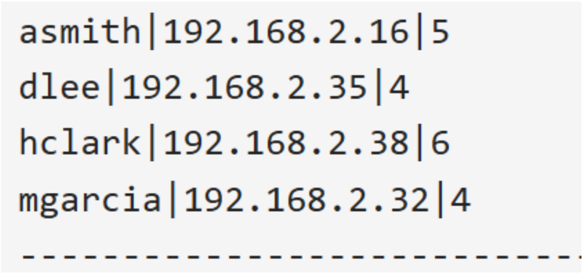
```sql
-- Insert sample data into privileged_users
INSERT INTO privileged_users (user_id, role, access_level) VALUES
('admin', 'Admin', 'High'),
('kjeff', 'System_Auditor', 'Medium'),
('tluther', 'Security_Analyst', 'Medium'),
('bomar', 'Network_Admin', 'High');

```

In our first query, we ensure that our environment aligns with **NIST Security Standard AC-7 (Unsuccessful Login Attempts)**. This control enforces thresholds for failed logins to mitigate brute-force attacks. For this lab, we set the threshold at more than three failed login attempts within a two-minute window from the same IP address. Such instances should be flagged as brute-force attempts, and the associated account is disabled as a preventive measure.

It is important to note that, in real-world scenarios, thresholds are typically higher and should be determined based on factors such as system sensitivity, compliance requirements, organizational information security policies, and other risk considerations.



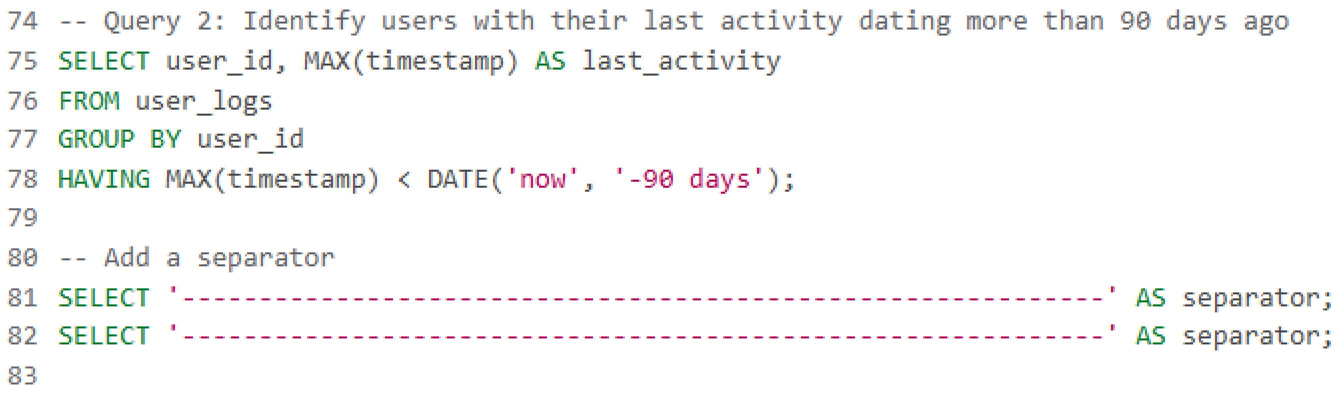
```sql
-- Query 1: Identify users with more than 3 failed consecutive
--login attempts within 2 minutes from the same IP address
SELECT user_id, source_ip, COUNT(*) AS failed_attempts
FROM user_logs AS u1
WHERE status = 'FAILED'
  AND EXISTS (
    SELECT 1
    FROM user_logs AS u2
    WHERE u1.user_id = u2.user_id
      AND u1.source_ip = u2.source_ip
      AND u1.timestamp >= u2.timestamp
      AND u1.timestamp <= DATETIME(u2.timestamp, '+2 minutes')
  )
GROUP BY user_id, source_ip
HAVING COUNT(*) > 3;

-- Add a separator
SELECT '------------------------------------------------------------' AS separator;
SELECT '------------------------------------------------------------' AS separator;

```



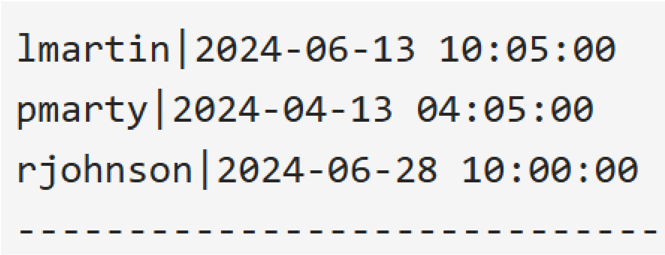
```
asmith|192.168.2.16|5
dlee|192.168.2.35|4
hclark|192.168.2.38|6
mgarcia|192.168.2.32|4
----------------------------
```

This is the result of the first query: four user accounts should have been disabled following instances of exceeding the failed login threshold. As part of our audit, we need to verify that these user accounts were indeed automatically disabled after exceeding three failed login attempts. If we confirm that the accounts were appropriately disabled, we can document that this specific system complies with **NIST Standard AC-7 (Unsuccessful Logon Attempts)**. However, if the accounts were not disabled as expected, we must document this as a failure to comply with both the NIST standard and the organization's defined policy.

For our second query, we aim to identify all users who have had no activity in the system within the last 90 days. In this example, we have set 90 days as the threshold, which is a common standard in many organizations. This query aligns with **NIST AC-2 (Account Management)**, which emphasizes tracking and managing inactive accounts to reduce security risks.



```sql
-- Query 2: Identify users with their last activity dating more than 90 days ago
SELECT user_id, MAX(timestamp) AS last_activity
FROM user_logs
GROUP BY user_id
HAVING MAX(timestamp) < DATE('now', '-90 days');

-- Add a separator
SELECT '------------------------------------------------------------' AS separator;
SELECT '------------------------------------------------------------' AS separator;

```

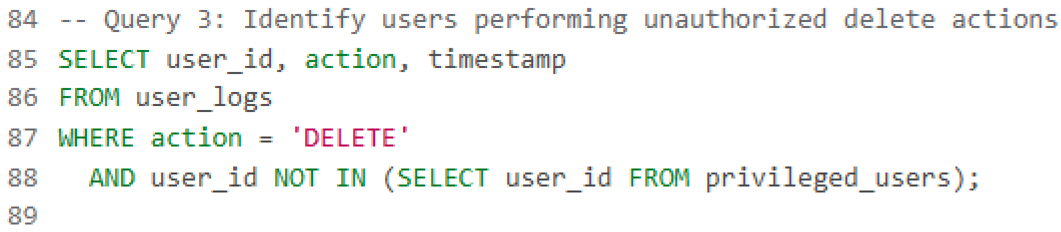


```
lmartin|2024-06-13 10:05:00
pmarty|2024-04-13 04:05:00
rjohnson|2024-06-28 10:00:00
-----------------------------
```

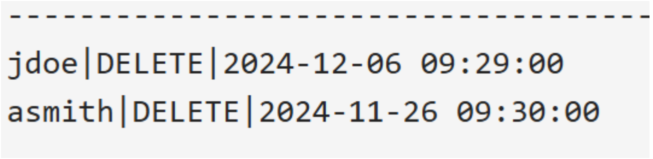
Here are the results of the second query: three users should have been automatically disabled due to inactivity. As part of our audit, we need to verify whether these accounts were indeed disabled. If we confirm this, we can document that the system complies with **NIST AC-2 (Account Management)** and the organization's policy. If not, we will document this as an area of non-compliance requiring corrective action.

\

In our final query, we aim to identify all users who performed actions they were not authorized to. This ensures compliance with **AU-2 (Audit Events)** and **AC-6 (Least Privilege)** by verifying that sensitive actions are both logged and restricted to authorized users. Specifically, users not listed in the privileged users table should not have the ability to modify or delete information. In our example, the only modification action is "delete," so we will check if any user has attempted or successfully performed a delete action without being listed in the privileged users table.



```sql
-- Query 3: Identify users performing unauthorized delete actions
SELECT user_id, action, timestamp
FROM user_logs
WHERE action = 'DELETE'
  AND user_id NOT IN (SELECT user_id FROM privileged_users);

```



```
------------------------------------
jdoe|DELETE|2024-12-06 09:29:00
asmith|DELETE|2024-11-26 09:30:00
```

Here are the results of this query: we identified two users who attempted to perform a delete action but were not listed as privileged users. In the dataset, we observed that one delete action was unsuccessful, meaning the user was unable to complete the action, while another user successfully performed the delete action.

As part of our audit, we would document this as non-compliance with **NIST AU-2 (Audit Events)** and **AC-6 (Least Privilege)**, as well as a violation of company policy. It is also crucial to determine exactly what was deleted, restore the deleted item if necessary, and monitor the failed deletion attempt for signs of ill intent or other malicious behavior. This ensures a thorough investigation and strengthens the organization's security posture.

Summary: This lab demonstrated how to conduct an IT Audit using SQL queries to ensure compliance with key **NIST 800-53** security controls. By simulating a user activity log and privileged user table, we examined common scenarios auditors may face in an organizational environment, including failed login attempts, inactive accounts, and unauthorized actions.

Key takeaways include:

1. **NIST Compliance**:
   - **AC-7**: Detected users with excessive failed login attempts to mitigate brute-force attacks.
   - **AC-2**: Identified inactive accounts for deactivation in line with account management policies.
   - **AU-2 & AC-6**: Monitored for unauthorized actions, ensuring sensitive operations were restricted to authorized users.
2. **Audit Effectiveness**:
   - Queries demonstrated practical techniques for analyzing logs and detecting security incidents.
   - Emphasis was placed on understanding the root cause of anomalies and ensuring compliance with both NIST controls and company policies.
3. **Actionable Insights**:
   - Flagging failed and successful unauthorized actions allows organizations to respond appropriately, such as restoring deleted items, monitoring suspicious behavior, and strengthening preventive measures.

This lab underscores the importance of aligning audit practices with security frameworks like **NIST 800-53** while using tools like SQL to efficiently monitor and enforce compliance.