SQL Query Portfolio: Security Investigation Analysis

Project Description
As a security professional at a large organization, I conducted an investigation into potential security issues involving login attempts and employee machines. Using SQL, I analyzed data from the organization's employees and login_attempts tables to identify suspicious activities, failed login attempts, and support security updates across different departments. This project demonstrates the application of SQL filtering techniques to enhance security monitoring and investigation capabilities.

Retrieve After Hours Failed Login Attempts
```sql
SELECT *
FROM log_in_attempts
WHERE login_time > '18:00' AND success = FALSE;
```

This query identifies failed login attempts occurring after business hours (18:00). It combines two conditions using AND:
- `login_time > '18:00'` filters for events after 6 PM
- `success = FALSE` filters for failed login attempts
This helps identify potential unauthorized access attempts during non-business hours, which could indicate suspicious activity.

Retrieve Login Attempts on Specific Dates
```sql
SELECT *
FROM log_in_attempts
WHERE login_date = '2022-05-09'
  OR login_date = '2022-05-08';
```

This query examines login attempts during a specific time window where suspicious activity was detected. It uses OR to combine two date conditions:
- Includes attempts on May 9th, 2022
- Includes attempts on May 8th, 2022
This allows for investigation of the suspicious event and potential related activities from the previous day.

Retrieve Login Attempts Outside of Mexico

```sql
SELECT *
FROM log_in_attempts
WHERE country NOT LIKE '%MEX%';
```

This query filters for login attempts originating outside Mexico. It uses:
- NOT LIKE with wildcard operator % to exclude both 'MEX' and 'MEXICO' country codes
- Pattern matching ensures comprehensive filtering regardless of country code format
This helps focus the investigation on login attempts from other countries after determining Mexico-based attempts were not suspicious.

Retrieve Employees in Marketing
```sql
SELECT *
FROM employees
WHERE department = 'Marketing'
  AND office LIKE 'East%';
```

This query identifies Marketing department employees in the East building for targeted security updates. It combines:
- Exact match on department = 'Marketing'
- LIKE with wildcard to match all East building offices (East-170, East-320, etc.)
This enables efficient targeting of security updates for specific organizational units.

Retrieve Employees in Finance or Sales
```sql
SELECT *
FROM employees
WHERE department = 'Sales'
   OR department = 'Finance';
```

This query identifies employees in either Sales or Finance departments requiring security updates. It uses:
- OR operator to combine department conditions
- Exact matching on department names
This supports efficient deployment of security updates across multiple departments simultaneously.

Retrieve All Employees Not in IT
```sql
SELECT *
FROM employees
WHERE department != 'Information Technology';
```

This query identifies employees outside the IT department who need specific security updates.
It uses:
- NOT operator (!=) to exclude IT department
- Exact match on the full department name

This ensures comprehensive coverage of security updates while avoiding redundant updates for IT staff.

Summary
Through this series of SQL queries, I demonstrated the ability to effectively filter and analyze security-relevant data across multiple database tables. The investigation covered after-hours access attempts, geographically suspicious activities, and enabled targeted security updates across different organizational units. These queries showcase the practical application of SQL operators (AND, OR, NOT, LIKE) for security investigation and system maintenance purposes.