REVATURE

Project

DENIAL OF SERVICE USING MYSQL RDS BASED ON NETWORK SECURITY

Prepared By:
Sakshi Shaha

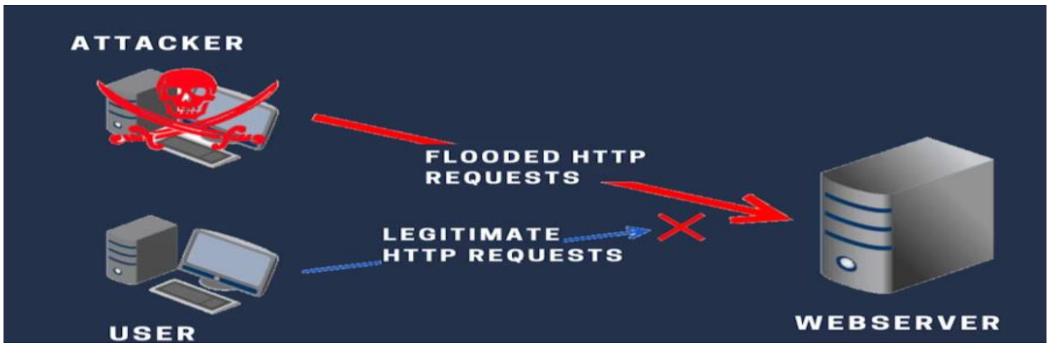
Guided By:
Zakir Hussain

Table of Contents

S No.	Content
1	Denial-Of-Service Attack
1.1	Types of DOS
1.2	Common DOS Attack Techniques
1.3	DOS Attack Can Be Launched Using Various Tools And Techniques
2	Structured Query Language
3	Databases Used In This Project
3.1	Database 1: Attack_Detection
3.2	Database 2: Network_Traffic
3.3	Database 3: System_Resources
3.4	Database 4: Incident_Response
3.5	Database 5: Security_Information
4	Tables Used In Each Databases
4.1	Database 1: attacks , attack types , sources , detection_rules , alerts
4.2	Database 2: traffic , protocols , ip_addresses , network_devices , traffic_stats
4.3	Database 3: resource_usage , resources , system_stats , process_list , user_sessions
4.4	Database 4: incidents , incident_types , response_plans ,response_teams , incident_reports
4.5	Database 5: vulnerabilities ,patches ,security_advisories , threat_intelligence , security_incidents
5	Queries Identified By The Network Infra Security Team
6	Final Goal of The Project

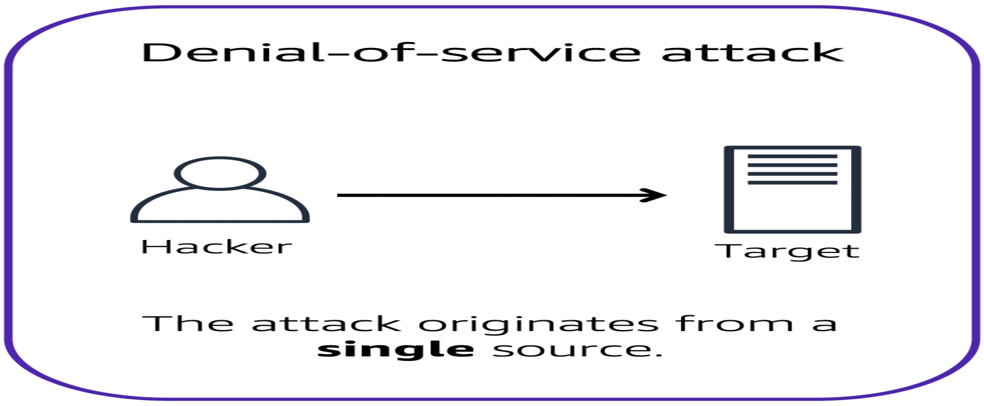
Denial-of-service attacks

A **Denial-of-Service (DoS) attack** is a deliberate attempt to make a website or application unavailable to users.

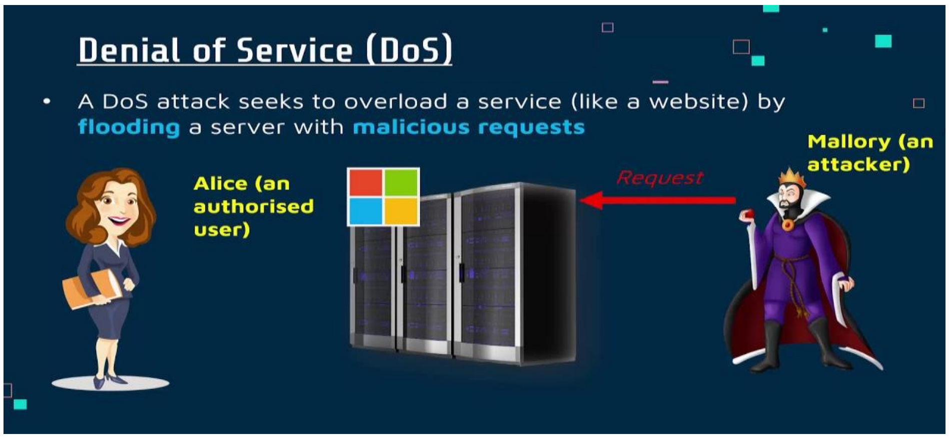


Denial of service (DOS) is a network security attack, in which, the hacker makes the system or data unavailable to someone who needs it. Hacker tries to make a network, system, or machine unavailable by flooding it with fake requests or traffic. This prevents real users from accessing it, causing anything from slowdowns to complete shutdowns.

A Denial of Service (DoS) attack is a type of cyber-attack where an attacker attempts to make a computer or network resource unavailable by overwhelming it with traffic or requests. The goal of a DoS attack is to exhaust the resource's capacity, making it unable to handle legitimate requests.



For example: An attacker might flood a website or application with excessive network traffic until the targeted website or application becomes overloaded and is no longer able to respond. If the website or application becomes unavailable, this denies service to users who are trying to make legitimate requests.

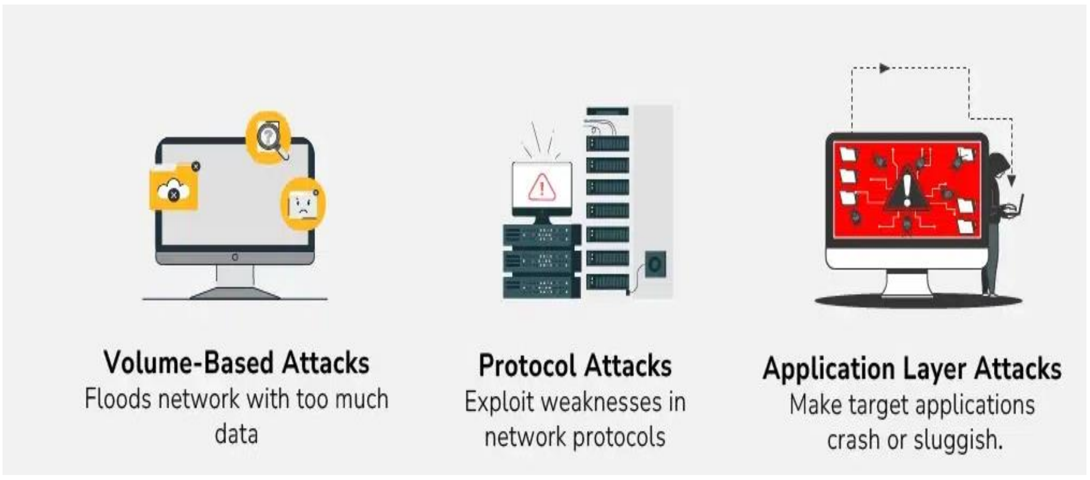

Denial of Service (DoS)
A DoS attack seeks to overload a service (like a website) by flooding a server with malicious requests
Alice (an authorised user)
Request
Mallory (an attacker)

Types of DoS attacks:

1. **Volume-Based Attacks:** Volume-based attacks flood a network with too much data, overpowering its bandwidth and making the network unusable. Examples include **UDP floods** and **ICMP floods**. In a UDP flood, attackers send many UDP packets to random ports on a server, making the server busy trying to handle all these requests, which slows down or stops legitimate traffic.

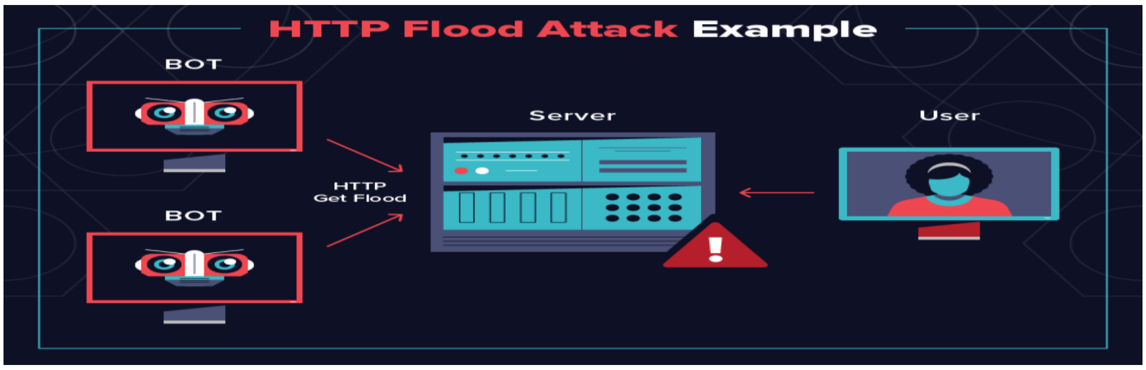
2. **Protocol Attacks:** Protocol attacks exploit weaknesses in network protocols to use up server resources. Examples are **SYN floods** and the **Ping of Death**. In a SYN flood, attackers send many SYN requests to a server but don't complete the handshake, leaving the server stuck with half-open connections. The Ping of Death involves sending oversized packets to crash or disrupt the target server.

3. **Application Layer Attacks:** Application layer attacks target specific applications or services, causing them to crash or become very slow. Examples include **HTTP floods** and **Slowloris**. In an HTTP flood, attackers send many HTTP requests to a web server, consuming its resources. Slowloris keeps many connections to the server open by sending incomplete HTTP requests, preventing the server from handling new, legitimate requests.


Volume-Based Attacks
Floods network with too much data
Protocol Attacks
Exploit weaknesses in network protocols
Application Layer Attacks
Make target applications crash or sluggish.

Common DoS attack techniques:

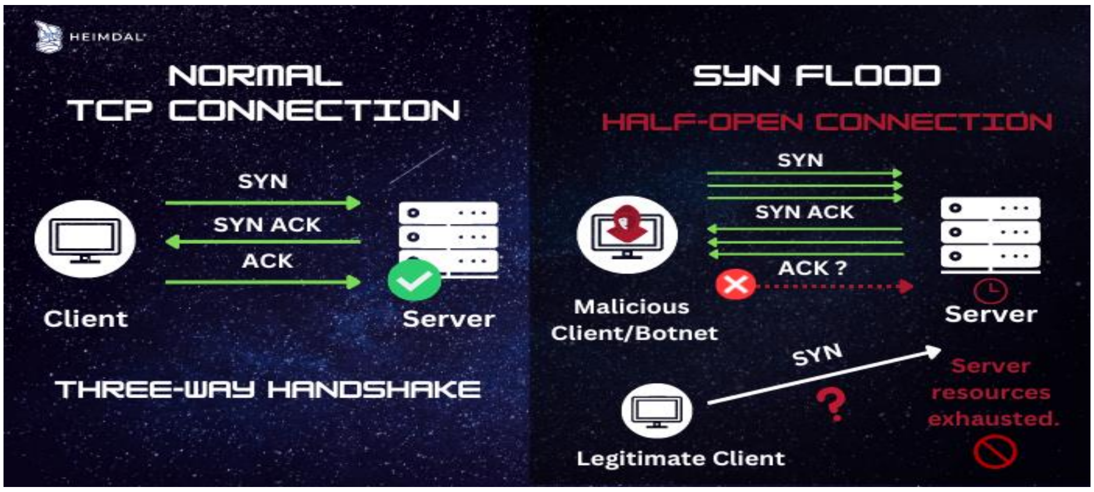
1. **Flooding:** Flooding is a common DoS attack technique where a perpetrator sends a large number of requests or traffic to overwhelm the targeted resource, **such as a website or server, making it unavailable to legitimate users.**



2. **Buffer overflow:** Buffer overflow is another common DoS attack technique where an attacker sends more data than a buffer can handle, causing it to crash or become overwhelmed. This can lead to system instability or unexpected behavior, making the resource inaccessible to legitimate users.

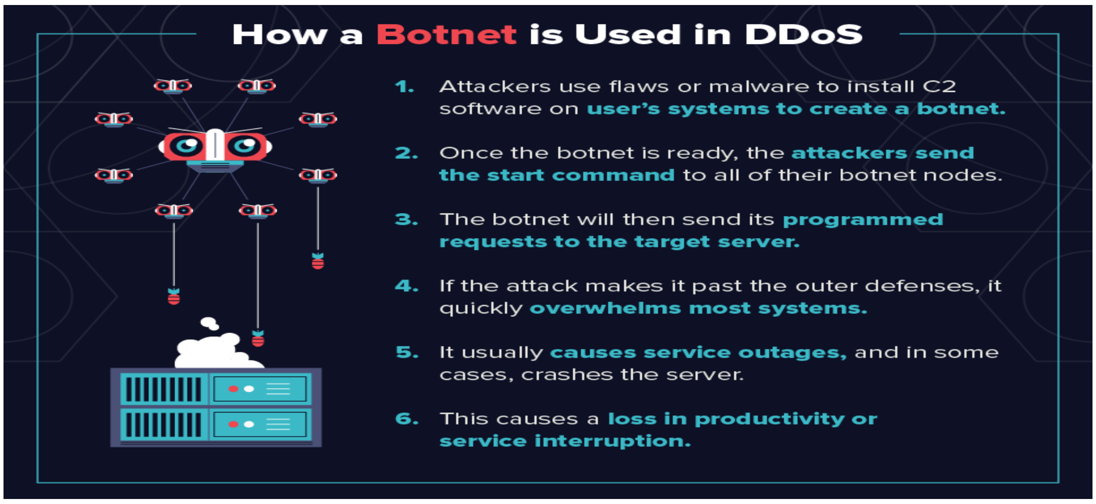
3. **Malformed packets:** Malformed packets are a type of DoS attack technique where an attacker sends packets with incorrect or malicious data to cause errors in the targeted system. This can disrupt the communication between network devices or servers, leading to downtime or service interruption for legitimate users.

4. **SYN flooding:** SYN flooding is a type of DoS attack technique where an attacker sends a large number of SYN requests to the target, such as a server, in order to fill up the cache and prevent it from accepting new connections. This type of attack can lead to service unavailability for legitimate users.



DoS attacks can be launched using various tools and techniques, including:

1. **Botnets**: Botnets are networks of compromised devices that are controlled by a single entity to carry out coordinated attacks. These devices can include computers, servers, and IoT devices that have been infected with malware. Botnets are often used in DoS attacks to overwhelm a target with a large volume of traffic, causing it to become inaccessible to legitimate users.



How a Botnet is Used in DDoS

1. Attackers use flaws or malware to install C2 software on **user's systems to create a botnet.**
2. Once the botnet is ready, the **attackers send the start command** to all of their botnet nodes.
3. The botnet will then send its **programmed requests to the target server.**
4. If the attack makes it past the outer defenses, it quickly **overwhelms most systems.**
5. It usually **causes service outages,** and in some cases, crashes the server.
6. This causes a **loss in productivity or service interruption.**

2. **Malware**: Malware is malicious software that is designed to harm or exploit systems. It can be used to infect devices within a network and turn them into part of a botnet, or to launch other forms of cyber-attacks.

3. **Scripting:** Scripting involves using scripts or automated tools to carry out attacks. This can include running scripts that flood a network with traffic or automate the process of sending malicious packets to a target.

To protect against DoS attacks, organizations can use:

1. **Firewalls:** Firewalls are a common defense mechanism used to block malicious traffic from reaching a network or resource. They can be configured to filter out unwanted traffic based on predefined rules or criteria, helping to prevent DoS attacks from overwhelming the system.

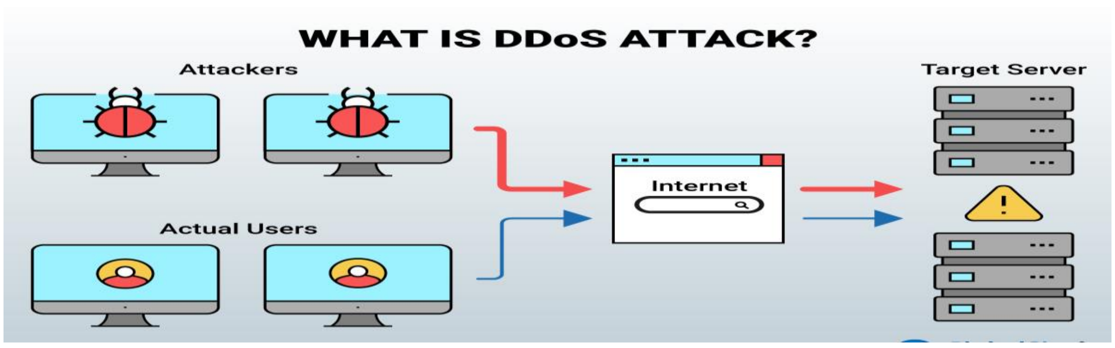
2. **Intrusion Detection/Prevention Systems (IDS/IPS):** It is a security tools that monitor network traffic for suspicious activity and take action to prevent potential attacks. They can help detect and prevent DoS attacks by analyzing traffic patterns and identifying abnormal behavior that may indicate an ongoing attack.

3. **Load balancing:** Load balancing is a technique used to distribute incoming network traffic across multiple servers or resources. By spreading the workload evenly,

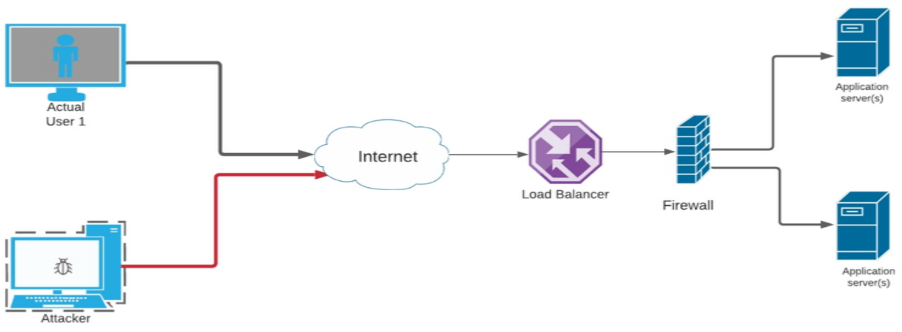
load balancing can help prevent any single resource from becoming overwhelmed during a DoS attack, ensuring that traffic is handled efficiently and effectively.

4. <u>Content Delivery Networks (CDNs):</u> Content Delivery Networks (CDNs) are distributed servers that cache content and deliver it to users based on their geographical location.

5. <u>DDoS mitigation services:</u> DDoS mitigation services are specialized services designed to detect and mitigate DoS attacks in real time.



<u>To protect against DoS attacks</u>



SQL stands for **Structured Query Language**, and it is a standard programming language used to manage and manipulate relational databases. SQL allows users to query, insert, update, and delete data from databases, as well as create and manage database schemas, tables, and indexes. It is widely used in database management systems such as MySQL, PostgreSQL, Oracle, and Microsoft SQL Server for storing, retrieving, and managing data efficiently.

MySQL is an open-source **relational database management system** that is based on SQL. It allows users to create, manage, and manipulate databases, tables, and data efficiently. MySQL is commonly used in web applications and is known for its high performance, scalability, and reliability.

<u>Database:</u> A database is a structured collection of data stored in tables and managed through a database management system like MySQL.

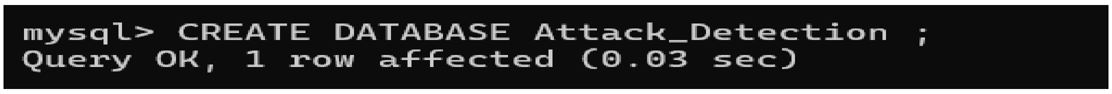
DATBASE STRCTURE:

Database 1: Attack_Detection
Database 2: Network_Traffic
Database 3: System_Resources
Database 4: Incident_Response
Database 5: Security_Information

The command to create a database in SQL:

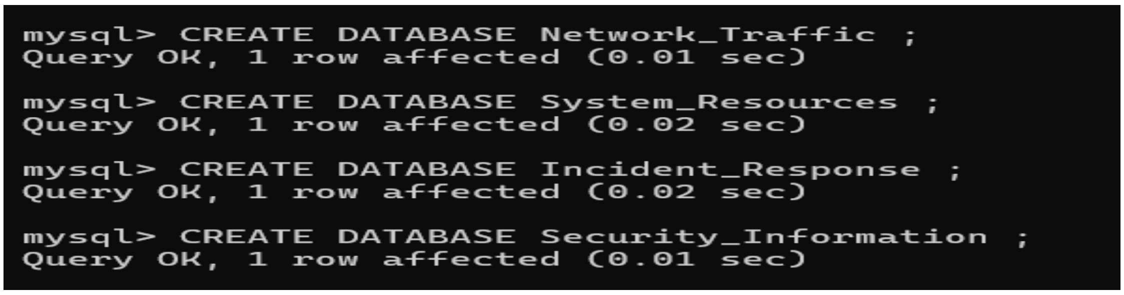
```
CREATE DATABASE <database_name>;
```

This command is used to create a new database with the specified name.
Database 1: Attack_Detection



```
mysql> CREATE DATABASE Attack_Detection ;
Query OK, 1 row affected (0.03 sec)
```

Database 2: Network_Traffic
Database 3: System_Resources
Database 4: Incident_Response
Database 5: Security_Information



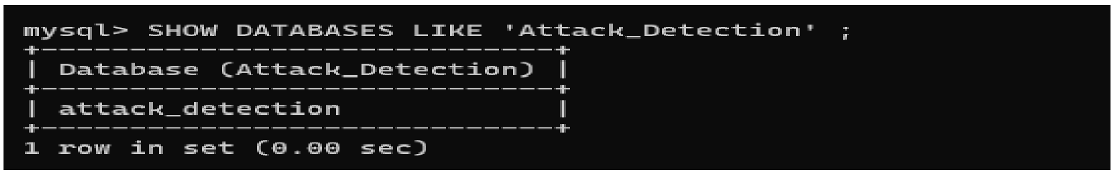
```
mysql> CREATE DATABASE Network_Traffic ;
Query OK, 1 row affected (0.01 sec)

mysql> CREATE DATABASE System_Resources ;
Query OK, 1 row affected (0.02 sec)

mysql> CREATE DATABASE Incident_Response ;
Query OK, 1 row affected (0.02 sec)

mysql> CREATE DATABASE Security_Information ;
Query OK, 1 row affected (0.01 sec)
```

The command to show a database in SQL:

```
SHOW DATABASES;
```

The command to show a Single database in SQL:

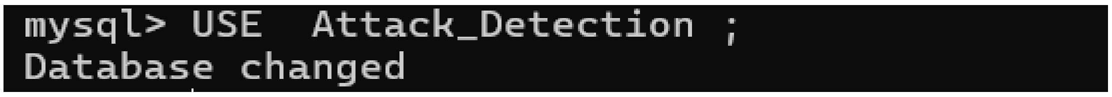
```
SHOW DATABASES LIKE ‘Attack_Detection’;
```



```
mysql> SHOW DATABASES LIKE 'Attack_Detection' ;
+-------------------------------+
| Database (Attack_Detection) |
+-------------------------------+
| attack_detection              |
+-------------------------------+
1 row in set (0.00 sec)
```

The command to USE a database in SQL:

```
USE <database_name>;
```



```
mysql> USE  Attack_Detection ;
Database changed
```

TABLE: Table are used to store data in a structured format. Each table consists of rows and columns, with each row representing a record or entry in the database, and each column representing a specific attribute or piece of information related to that record.

Database 1: Attack_Detection

Attack detection databases are used to identify and report potential security threats or attacks on a network or system.

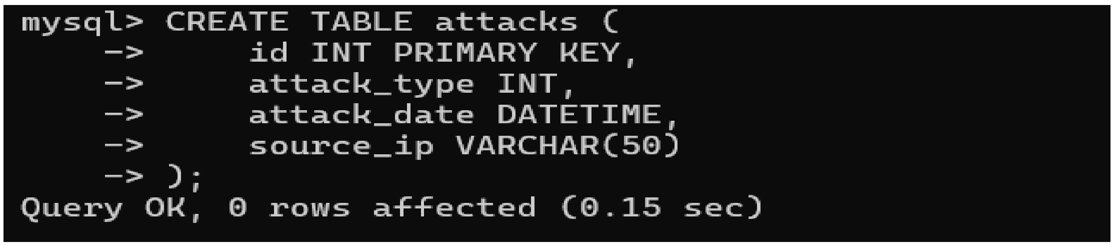
Tables:
1) attacks
2) attack_types
3) sources
4) detection_rules
5) alerts

The command to create a TABLE in SQL:

```
CREATE TABLE <table_name> (<column_name> <data_type>);
```

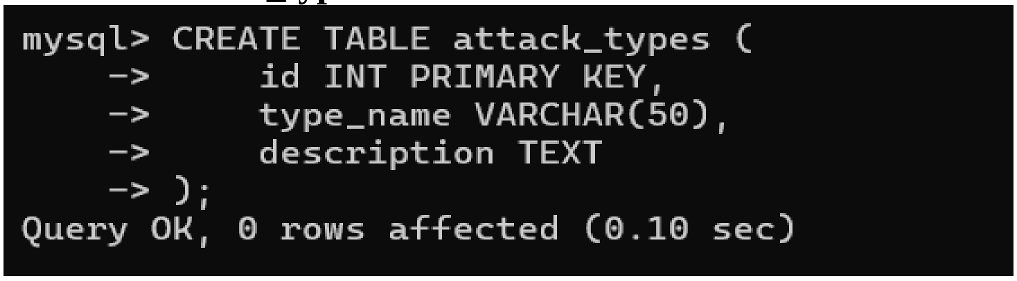
This command is used to create a new table with the specified name and define the columns with their respective data types.

1 **Create attacks table** :



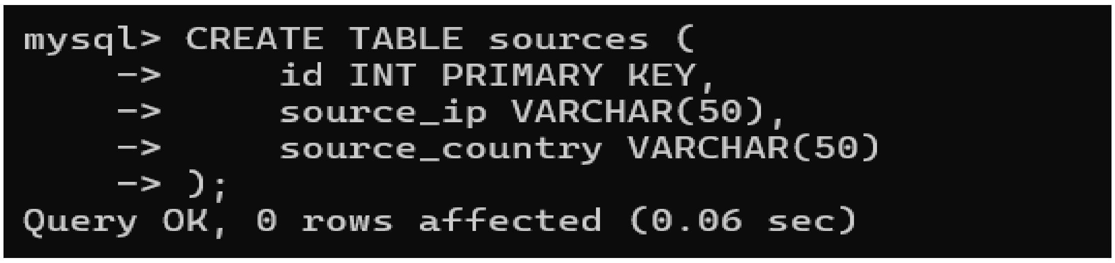
```
mysql> CREATE TABLE attacks (
    ->     id INT PRIMARY KEY,
    ->     attack_type INT,
    ->     attack_date DATETIME,
    ->     source_ip VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.15 sec)
```

2 **Create attack_types table**



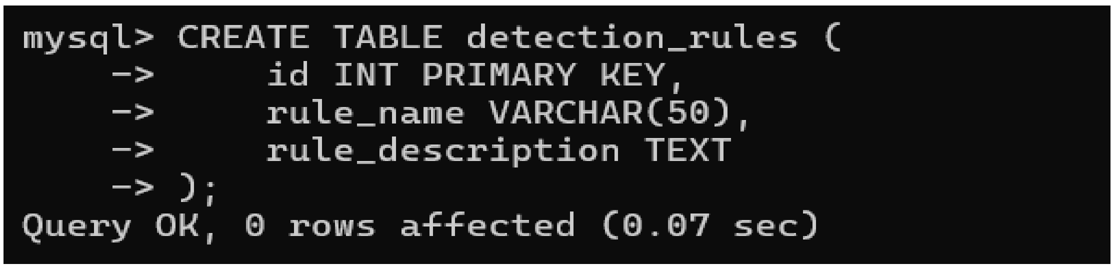
```
mysql> CREATE TABLE attack_types (
    ->     id INT PRIMARY KEY,
    ->     type_name VARCHAR(50),
    ->     description TEXT
    -> );
Query OK, 0 rows affected (0.10 sec)
```

3 Create sources table



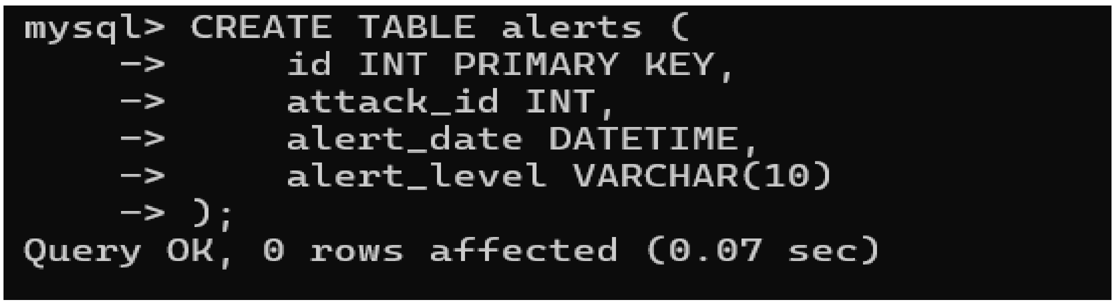
```
mysql> CREATE TABLE sources (
    ->     id INT PRIMARY KEY,
    ->     source_ip VARCHAR(50),
    ->     source_country VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.06 sec)
```

4 Create detection_rules table



```
mysql> CREATE TABLE detection_rules (
    ->     id INT PRIMARY KEY,
    ->     rule_name VARCHAR(50),
    ->     rule_description TEXT
    -> );
Query OK, 0 rows affected (0.07 sec)
```

5 Create alerts table



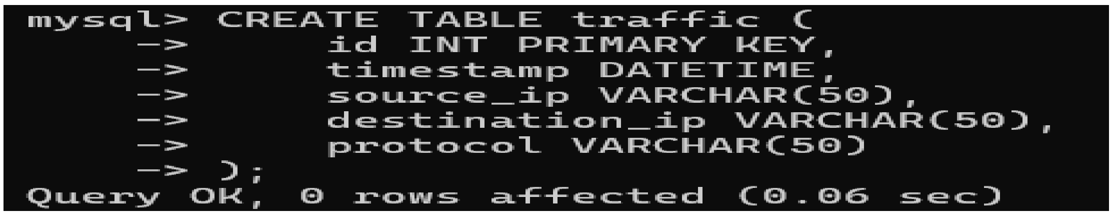
```
mysql> CREATE TABLE alerts (
    ->     id INT PRIMARY KEY,
    ->     attack_id INT,
    ->     alert_date DATETIME,
    ->     alert_level VARCHAR(10)
    -> );
Query OK, 0 rows affected (0.07 sec)
```

Database 2: Network_Traffic:

The Network_Traffic database is a collection of tables that store information related to network traffic, such as data transmitted, source and destination IP addresses, protocols used, and timestamps. This database is crucial for analyzing network usage, identifying issues, and improving network performance.

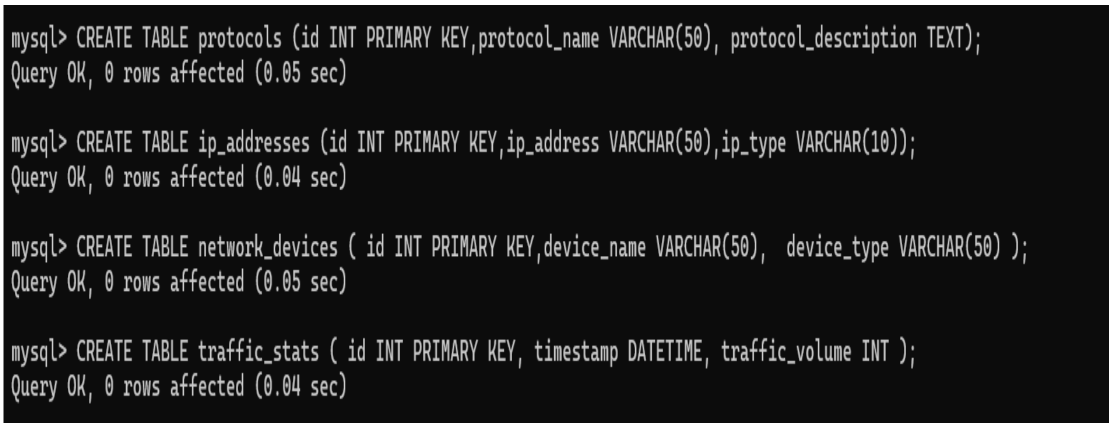
Tables:
1 traffic
2 protocols
3 ip_addresses
4 network_devices
5 traffic_stats

1 Create traffic table



```
mysql> CREATE TABLE traffic (
    ->       id INT PRIMARY KEY,
    ->       timestamp DATETIME,
    ->       source_ip VARCHAR(50),
    ->       destination_ip VARCHAR(50),
    ->       protocol VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.06 sec)
```

2 protocols
3 ip_addresses
4 network_devices
5 traffic_stats



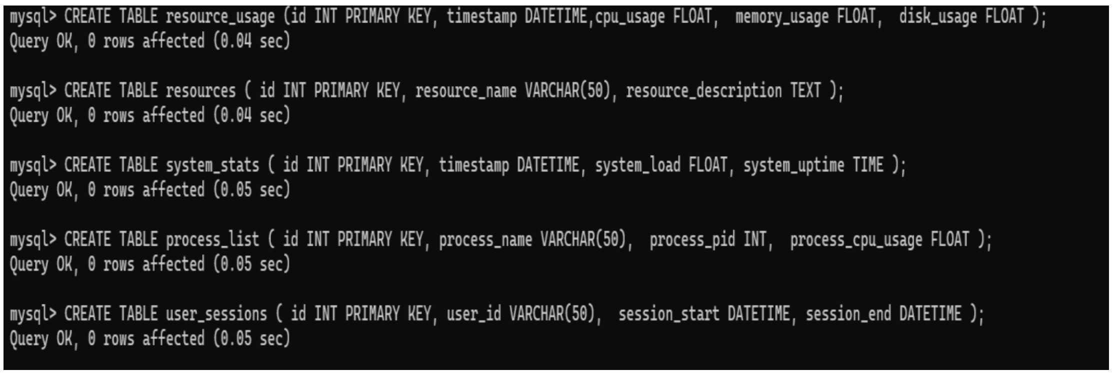
```
mysql> CREATE TABLE protocols (id INT PRIMARY KEY,protocol_name VARCHAR(50), protocol_description TEXT);
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE ip_addresses (id INT PRIMARY KEY,ip_address VARCHAR(50),ip_type VARCHAR(10));
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE network_devices ( id INT PRIMARY KEY,device_name VARCHAR(50),  device_type VARCHAR(50) );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE traffic_stats ( id INT PRIMARY KEY, timestamp DATETIME, traffic_volume INT );
Query OK, 0 rows affected (0.04 sec)
```

Database 3: System_Resources

System resources refer to the hardware and software components of a computer system that are used to perform tasks and operations. This includes but is not limited to, CPU (Central Processing Unit), memory (RAM), storage (hard drive or SSD), network resources, and peripherals such as printers or monitors. Monitoring and managing system resources is important to ensure that the system operates efficiently and effectively.

Tables:

1 resource_usage
2 resources
3 system_stats
4 process_list
5 user_sessions



```
mysql> CREATE TABLE resource_usage (id INT PRIMARY KEY, timestamp DATETIME,cpu_usage FLOAT,  memory_usage FLOAT,  disk_usage FLOAT );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE resources ( id INT PRIMARY KEY, resource_name VARCHAR(50), resource_description TEXT );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE system_stats ( id INT PRIMARY KEY, timestamp DATETIME, system_load FLOAT, system_uptime TIME );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE process_list ( id INT PRIMARY KEY, process_name VARCHAR(50),  process_pid INT,  process_cpu_usage FLOAT );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE user_sessions ( id INT PRIMARY KEY, user_id VARCHAR(50),  session_start DATETIME, session_end DATETIME );
Query OK, 0 rows affected (0.05 sec)
```

Database 4: Incident_Response:

Incident response refers to the process of reacting to and managing security incidents within an organization. It involves detecting, responding to, and resolving incidents to minimize damage and prevent future incidents.

Tables:

1 incidents
2 incident_types
3 response_plans
4 response_teams
5 incident_reports

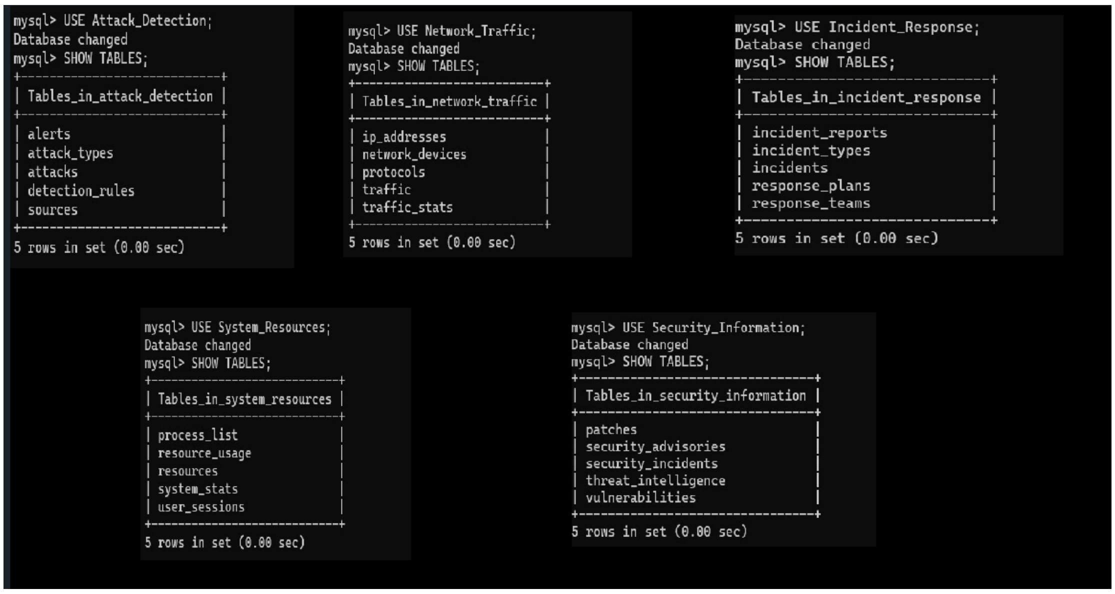
Database 5: Security_Information

The Security_Information database is a collection of tables that store information related to security incidents, threats, vulnerabilities, and security measures within an organization. This database is crucial for monitoring and managing security risks, analyzing security events, and implementing security protocols to protect sensitive information and prevent unauthorized access.

Tables:
1 vulnerabilities
2 patches
3 security_advisories
4 threat_intelligence
5 security_incidents

The command to show tables in SQL:

```
SHOW TABLES;
```



```
mysql> USE Attack_Detection;
Database changed
mysql> SHOW TABLES;
+----------------------------+
| Tables_in_attack_detection |
+----------------------------+
| alerts                     |
| attack_types               |
| attacks                    |
| detection_rules            |
| sources                    |
+----------------------------+
5 rows in set (0.00 sec)
```

```
mysql> USE Network_Traffic;
Database changed
mysql> SHOW TABLES;
+---------------------------+
| Tables_in_network_traffic |
+---------------------------+
| ip_addresses              |
| network_devices           |
| protocols                 |
| traffic                   |
| traffic_stats             |
+---------------------------+
5 rows in set (0.00 sec)
```

```
mysql> USE Incident_Response;
Database changed
mysql> SHOW TABLES;
+-----------------------------+
| Tables_in_incident_response |
+-----------------------------+
| incident_reports            |
| incident_types              |
| incidents                   |
| response_plans              |
| response_teams              |
+-----------------------------+
5 rows in set (0.00 sec)
```

```
mysql> USE System_Resources;
Database changed
mysql> SHOW TABLES;
+----------------------------+
| Tables_in_system_resources |
+----------------------------+
| process_list               |
| resource_usage             |
| resources                  |
| system_stats               |
| user_sessions              |
+----------------------------+
5 rows in set (0.00 sec)
```

```
mysql> USE Security_Information;
Database changed
mysql> SHOW TABLES;
+--------------------------------+
| Tables_in_security_information |
+--------------------------------+
| patches                        |
| security_advisories            |
| security_incidents             |
| threat_intelligence            |
| vulnerabilities                |
+--------------------------------+
5 rows in set (0.00 sec)
```

The command to insert data into a table in SQL is the INSERT INTO statement:

```
INSERT INTO table_name (column1, column2, column3 ...) VALUES (value1,
value2, value3 ...);
```

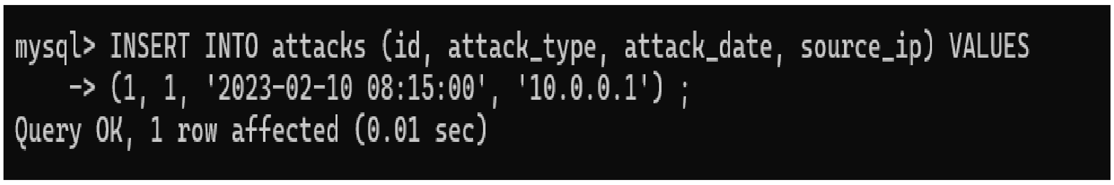
This command is used to add new records or data entries to a table in a SQL database.

Database 1: Attack_Detection

1 Attacks Table:
Insert data on detected attacks, including type, date, and source IP.

The Attacks table typically includes columns such as id (a unique identifier for each attack entry), attack_type (the type of attack that occurred), attack_date (the date and time when the attack occurred), and source_ip (the IP address of the attacker). This information helps security teams track and analyze different types of attacks on a network or system.

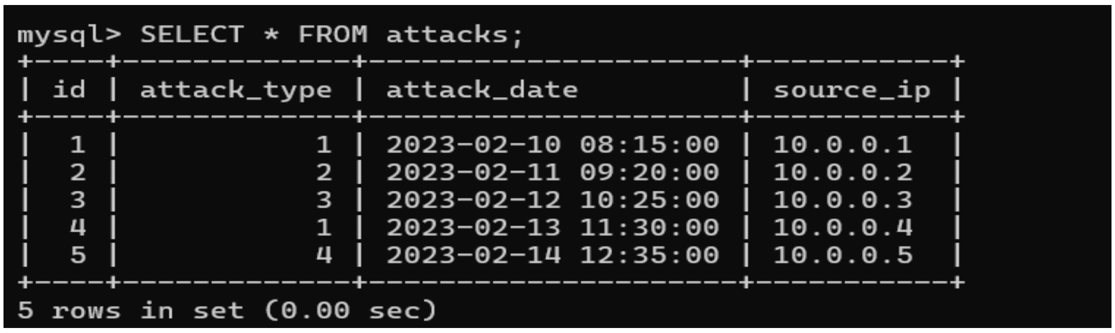
Inserting data into attacks table



```
mysql> INSERT INTO attacks (id, attack_type, attack_date, source_ip) VALUES
    -> (1, 1, '2023-02-10 08:15:00', '10.0.0.1') ;
Query OK, 1 row affected (0.01 sec)
```

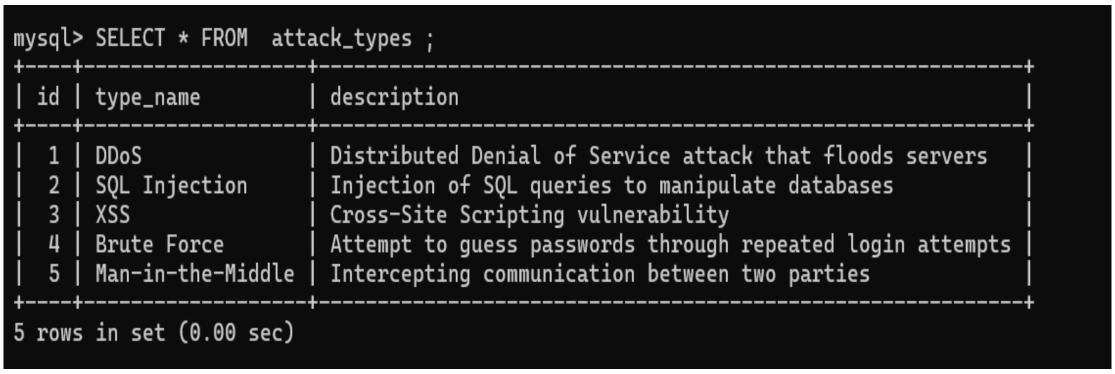
The command to DISPLAY ALL DATA of a table in SQL:

```
SELECT * FROM table_name;
```

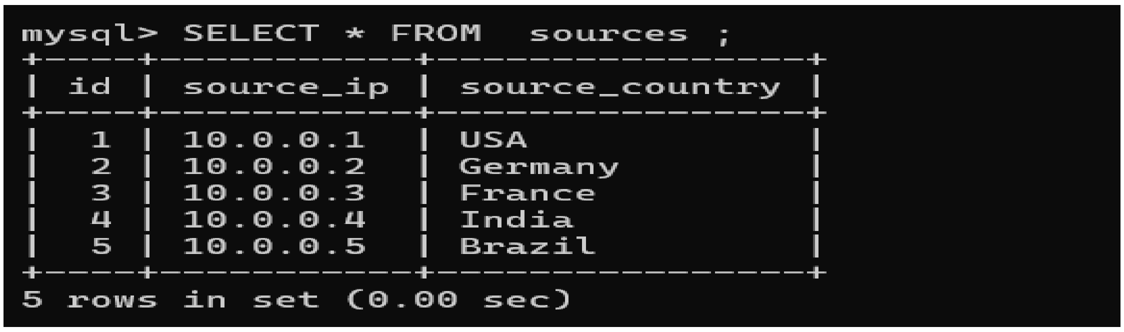


```
mysql> SELECT * FROM attacks;
+----+-------------+---------------------+-----------+
| id | attack_type | attack_date         | source_ip |
+----+-------------+---------------------+-----------+
|  1 |           1 | 2023-02-10 08:15:00 | 10.0.0.1  |
|  2 |           2 | 2023-02-11 09:20:00 | 10.0.0.2  |
|  3 |           3 | 2023-02-12 10:25:00 | 10.0.0.3  |
|  4 |           1 | 2023-02-13 11:30:00 | 10.0.0.4  |
|  5 |           4 | 2023-02-14 12:35:00 | 10.0.0.5  |
+----+-------------+---------------------+-----------+
5 rows in set (0.00 sec)
```

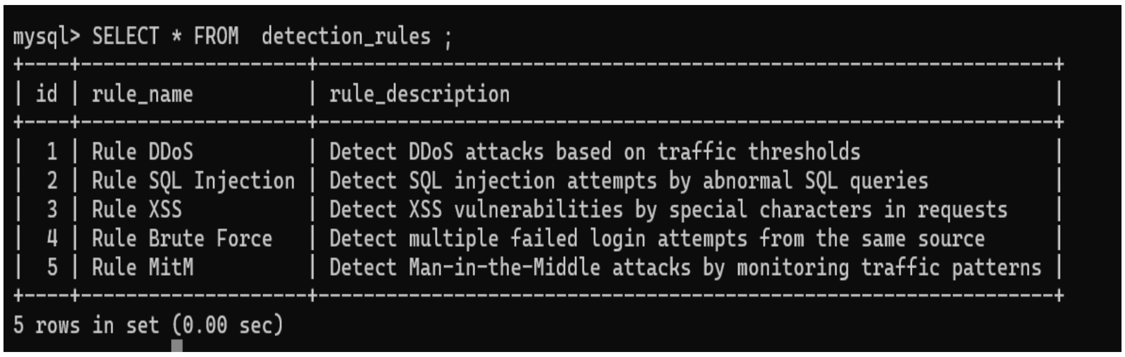
2 attack_types table: Inserting data into attack_types table


```
mysql> SELECT * FROM  attack_types ;
+----+-------------------+-------------------------------------------------------------+
| id | type_name         | description                                                 |
+----+-------------------+-------------------------------------------------------------+
|  1 | DDoS              | Distributed Denial of Service attack that floods servers    |
|  2 | SQL Injection     | Injection of SQL queries to manipulate databases            |
|  3 | XSS               | Cross-Site Scripting vulnerability                          |
|  4 | Brute Force       | Attempt to guess passwords through repeated login attempts |
|  5 | Man-in-the-Middle | Intercepting communication between two parties              |
+----+-------------------+-------------------------------------------------------------+
5 rows in set (0.00 sec)
```

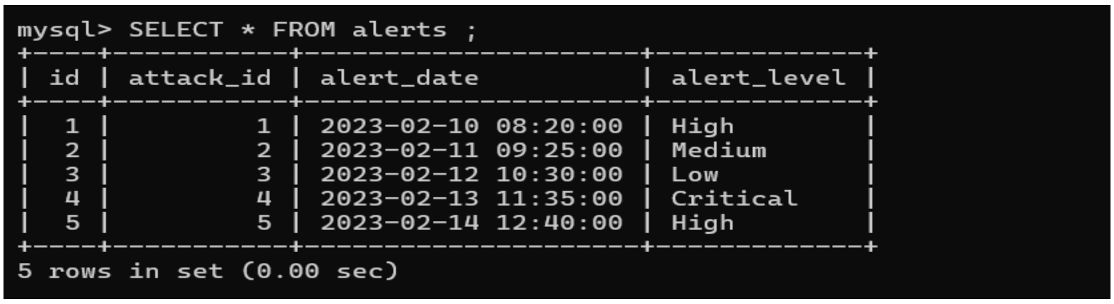
3 sources table: Inserting data into sources table


```
mysql> SELECT * FROM  sources ;
+----+------------+----------------+
| id | source_ip  | source_country |
+----+------------+----------------+
|  1 | 10.0.0.1   | USA            |
|  2 | 10.0.0.2   | Germany        |
|  3 | 10.0.0.3   | France         |
|  4 | 10.0.0.4   | India          |
|  5 | 10.0.0.5   | Brazil         |
+----+------------+----------------+
5 rows in set (0.00 sec)
```

4 detection_rules: Inserting data into detection_rules table

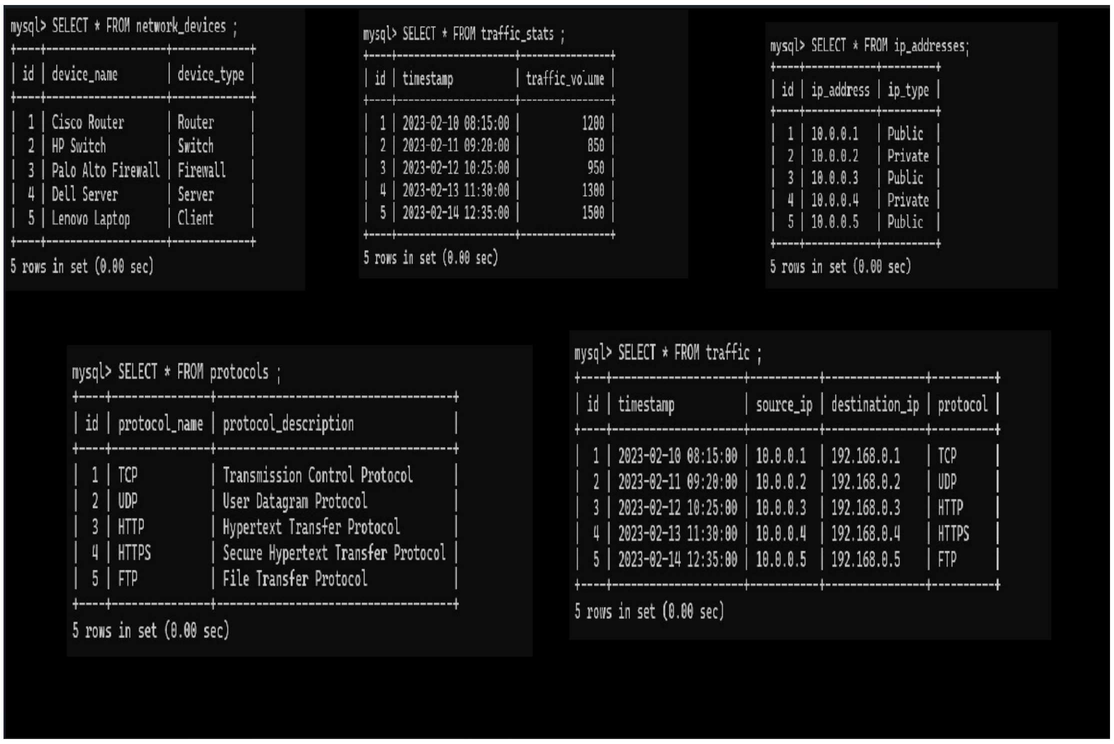

```
mysql> SELECT * FROM  detection_rules ;
+----+--------------------+----------------------------------------------------------------+
| id | rule_name          | rule_description                                               |
+----+--------------------+----------------------------------------------------------------+
|  1 | Rule DDoS          | Detect DDoS attacks based on traffic thresholds                |
|  2 | Rule SQL Injection | Detect SQL injection attempts by abnormal SQL queries          |
|  3 | Rule XSS           | Detect XSS vulnerabilities by special characters in requests   |
|  4 | Rule Brute Force   | Detect multiple failed login attempts from the same source     |
|  5 | Rule MitM          | Detect Man-in-the-Middle attacks by monitoring traffic patterns |
+----+--------------------+----------------------------------------------------------------+
5 rows in set (0.00 sec)
```

5 alerts table: Inserting data into alerts table


```
mysql> SELECT * FROM alerts ;
+----+-----------+---------------------+-------------+
| id | attack_id | alert_date          | alert_level |
+----+-----------+---------------------+-------------+
|  1 |         1 | 2023-02-10 08:20:00 | High        |
|  2 |         2 | 2023-02-11 09:25:00 | Medium      |
|  3 |         3 | 2023-02-12 10:30:00 | Low         |
|  4 |         4 | 2023-02-13 11:35:00 | Critical    |
|  5 |         5 | 2023-02-14 12:40:00 | High        |
+----+-----------+---------------------+-------------+
5 rows in set (0.00 sec)
```

Database 2: Network_Traffic

1 traffic table: Inserting data into traffic table.
2 protocols table: Inserting data into protocols table.
3 ip_addresses table: Inserting data into ip_addresses table.
4 network_devices table: Inserting data into network_devices table.
5 traffic_stats table: Inserting data into traffic_stats table.



```
mysql> SELECT * FROM network_devices ;
+----+--------------------+-------------+
| id | device_name        | device_type |
+----+--------------------+-------------+
|  1 | Cisco Router       | Router      |
|  2 | HP Switch          | Switch      |
|  3 | Palo Alto Firewall | Firewall    |
|  4 | Dell Server        | Server      |
|  5 | Lenovo Laptop      | Client      |
+----+--------------------+-------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM traffic_stats ;
+----+---------------------+----------------+
| id | timestamp           | traffic_volume |
+----+---------------------+----------------+
|  1 | 2023-02-10 08:15:00 |           1200 |
|  2 | 2023-02-11 09:20:00 |            850 |
|  3 | 2023-02-12 10:25:00 |            950 |
|  4 | 2023-02-13 11:30:00 |           1300 |
|  5 | 2023-02-14 12:35:00 |           1500 |
+----+---------------------+----------------+
5 rows in set (0.00 sec)
```

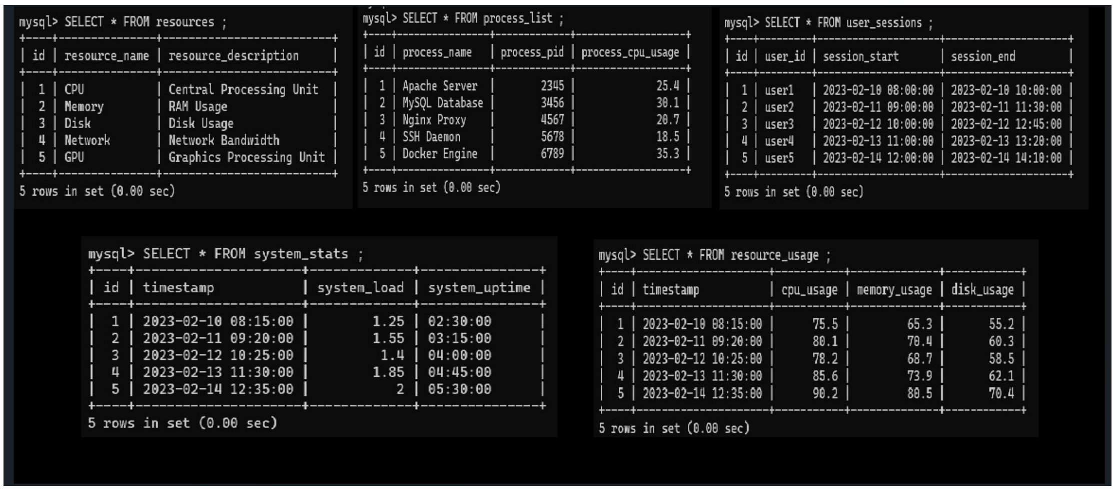
```
mysql> SELECT * FROM ip_addresses;
+----+------------+---------+
| id | ip_address | ip_type |
+----+------------+---------+
|  1 | 10.0.0.1   | Public  |
|  2 | 10.0.0.2   | Private |
|  3 | 10.0.0.3   | Public  |
|  4 | 10.0.0.4   | Private |
|  5 | 10.0.0.5   | Public  |
+----+------------+---------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM protocols ;
+----+---------------+-----------------------------------+
| id | protocol_name | protocol_description              |
+----+---------------+-----------------------------------+
|  1 | TCP           | Transmission Control Protocol     |
|  2 | UDP           | User Datagram Protocol            |
|  3 | HTTP          | Hypertext Transfer Protocol       |
|  4 | HTTPS         | Secure Hypertext Transfer Protocol|
|  5 | FTP           | File Transfer Protocol            |
+----+---------------+-----------------------------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM traffic ;
+----+---------------------+-----------+----------------+----------+
| id | timestamp           | source_ip | destination_ip | protocol |
+----+---------------------+-----------+----------------+----------+
|  1 | 2023-02-10 08:15:00 | 10.0.0.1  | 192.168.0.1    | TCP      |
|  2 | 2023-02-11 09:20:00 | 10.0.0.2  | 192.168.0.2    | UDP      |
|  3 | 2023-02-12 10:25:00 | 10.0.0.3  | 192.168.0.3    | HTTP     |
|  4 | 2023-02-13 11:30:00 | 10.0.0.4  | 192.168.0.4    | HTTPS    |
|  5 | 2023-02-14 12:35:00 | 10.0.0.5  | 192.168.0.5    | FTP      |
+----+---------------------+-----------+----------------+----------+
5 rows in set (0.00 sec)
```

Database 3: System_Resources

1 resource_usage table: Inserting data into resource_usage table
2 resources table: Inserting data into resources table
3 system_stats table: Inserting data into system_stats table
4 process_list table: Inserting data into process_list table
5 user_sessions table: Inserting data into user_sessions table



```
mysql> SELECT * FROM resources ;
+----+---------------+--------------------------+
| id | resource_name | resource_description     |
+----+---------------+--------------------------+
|  1 | CPU           | Central Processing Unit  |
|  2 | Memory        | RAM Usage                |
|  3 | Disk          | Disk Usage               |
|  4 | Network       | Network Bandwidth        |
|  5 | GPU           | Graphics Processing Unit |
+----+---------------+--------------------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM process_list ;
+----+----------------+-------------+-------------------+
| id | process_name   | process_pid | process_cpu_usage |
+----+----------------+-------------+-------------------+
|  1 | Apache Server  |        2345 |              25.4 |
|  2 | MySQL Database |        3456 |              30.1 |
|  3 | Nginx Proxy    |        4567 |              20.7 |
|  4 | SSH Daemon     |        5678 |              18.5 |
|  5 | Docker Engine  |        6789 |              35.3 |
+----+----------------+-------------+-------------------+
5 rows in set (0.00 sec)
```

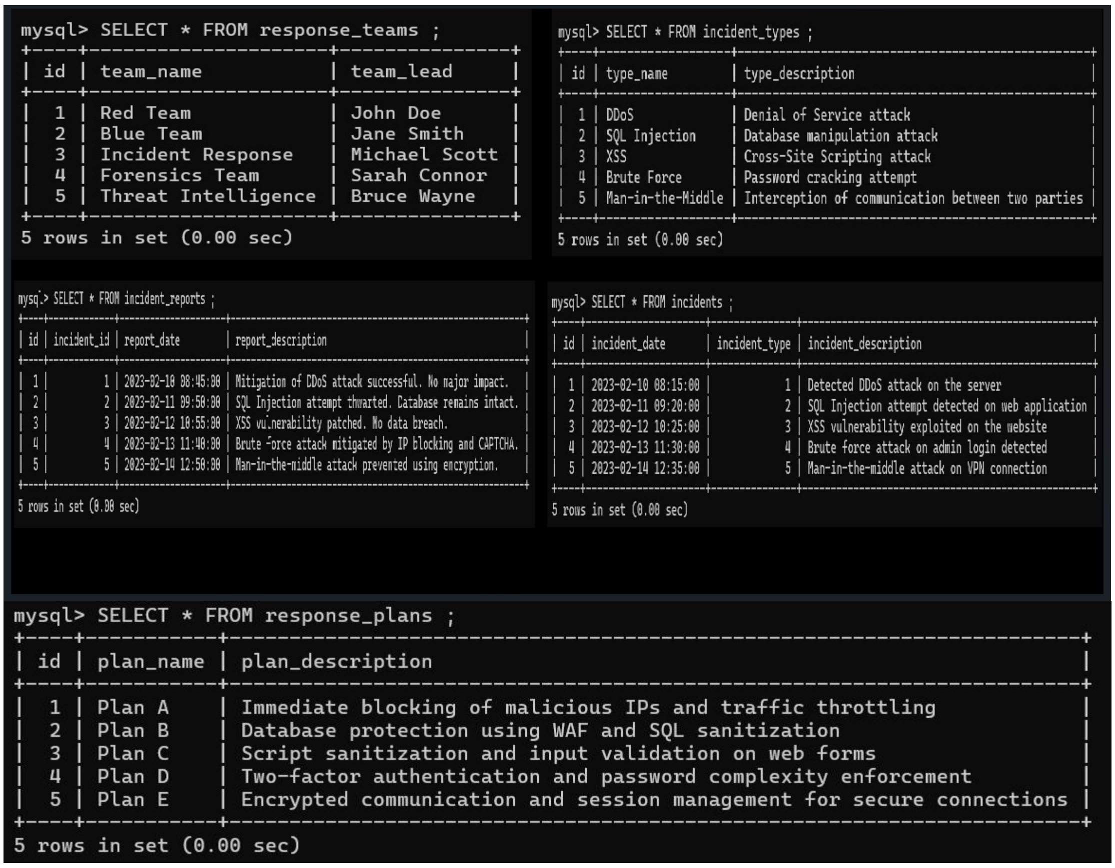
```
mysql> SELECT * FROM user_sessions ;
+----+---------+---------------------+---------------------+
| id | user_id | session_start       | session_end         |
+----+---------+---------------------+---------------------+
|  1 | user1   | 2023-02-10 08:00:00 | 2023-02-10 10:00:00 |
|  2 | user2   | 2023-02-11 09:00:00 | 2023-02-11 11:30:00 |
|  3 | user3   | 2023-02-12 10:00:00 | 2023-02-12 12:45:00 |
|  4 | user4   | 2023-02-13 11:00:00 | 2023-02-13 13:20:00 |
|  5 | user5   | 2023-02-14 12:00:00 | 2023-02-14 14:10:00 |
+----+---------+---------------------+---------------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM system_stats ;
+----+---------------------+-------------+---------------+
| id | timestamp           | system_load | system_uptime |
+----+---------------------+-------------+---------------+
|  1 | 2023-02-10 08:15:00 |        1.25 | 02:30:00      |
|  2 | 2023-02-11 09:20:00 |        1.55 | 03:15:00      |
|  3 | 2023-02-12 10:25:00 |         1.4 | 04:00:00      |
|  4 | 2023-02-13 11:30:00 |        1.85 | 04:45:00      |
|  5 | 2023-02-14 12:35:00 |           2 | 05:30:00      |
+----+---------------------+-------------+---------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM resource_usage ;
+----+---------------------+-----------+--------------+------------+
| id | timestamp           | cpu_usage | memory_usage | disk_usage |
+----+---------------------+-----------+--------------+------------+
|  1 | 2023-02-10 08:15:00 |      75.5 |         65.3 |       55.2 |
|  2 | 2023-02-11 09:20:00 |      80.1 |         70.4 |       60.3 |
|  3 | 2023-02-12 10:25:00 |      78.2 |         68.7 |       58.5 |
|  4 | 2023-02-13 11:30:00 |      85.6 |         73.9 |       62.1 |
|  5 | 2023-02-14 12:35:00 |      90.2 |         80.5 |       70.4 |
+----+---------------------+-----------+--------------+------------+
5 rows in set (0.00 sec)
```

Database 4: Incident_Response

1 incidents table: Inserting data into incidents table
2 incident_types table: Inserting data into incident_types table
3 response_plans table: Inserting data into response_plans table
4 response_teams table: Inserting data into response_teams table
5 incident_reports table: Inserting data into incident_reports table



```
mysql> SELECT * FROM response_teams ;
+----+---------------------+----------------+
| id | team_name           | team_lead      |
+----+---------------------+----------------+
|  1 | Red Team            | John Doe       |
|  2 | Blue Team           | Jane Smith     |
|  3 | Incident Response   | Michael Scott  |
|  4 | Forensics Team      | Sarah Connor   |
|  5 | Threat Intelligence | Bruce Wayne    |
+----+---------------------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM incident_types ;
+----+-------------------+------------------------------------------------+
| id | type_name         | type_description                               |
+----+-------------------+------------------------------------------------+
|  1 | DDoS              | Denial of Service attack                       |
|  2 | SQL Injection     | Database manipulation attack                   |
|  3 | XSS               | Cross-Site Scripting attack                    |
|  4 | Brute Force       | Password cracking attempt                      |
|  5 | Man-in-the-Middle | Interception of communication between two parties |
+----+-------------------+------------------------------------------------+
5 rows in set (0.00 sec)
```

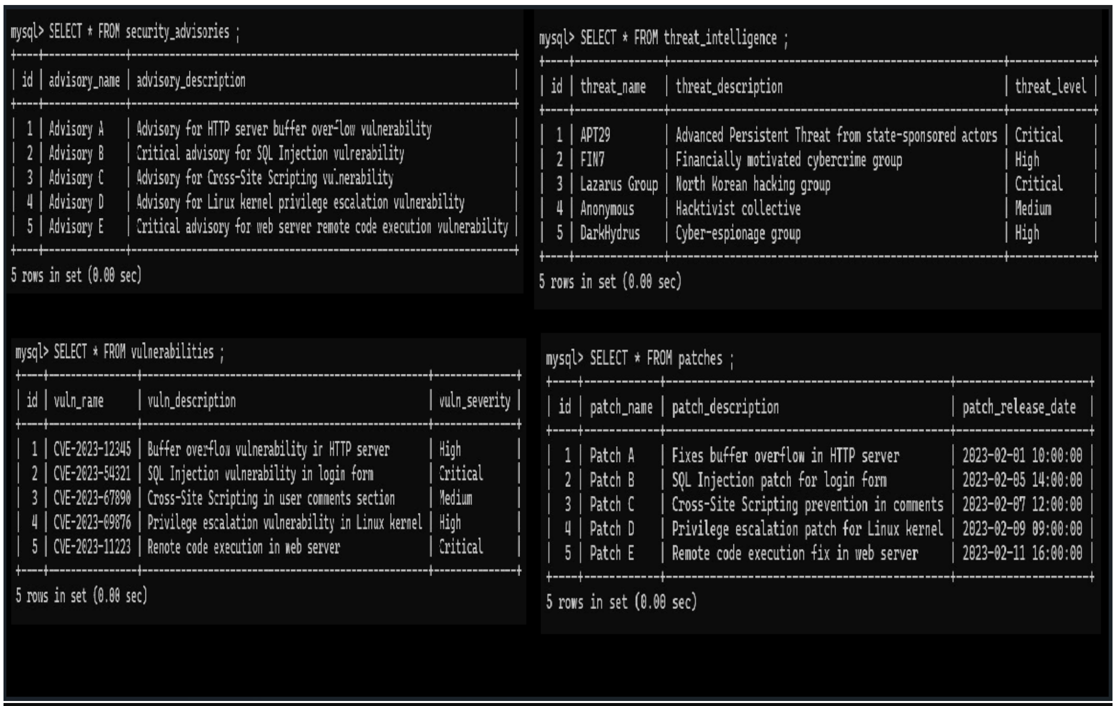
```
mysql> SELECT * FROM incident_reports ;
+----+-------------+---------------------+------------------------------------------------------+
| id | incident_id | report_date         | report_description                                   |
+----+-------------+---------------------+------------------------------------------------------+
|  1 |           1 | 2023-02-10 08:45:00 | Mitigation of DDoS attack successful. No major impact. |
|  2 |           2 | 2023-02-11 09:50:00 | SQL Injection attempt thwarted. Database remains intact. |
|  3 |           3 | 2023-02-12 10:55:00 | XSS vulnerability patched. No data breach.           |
|  4 |           4 | 2023-02-13 11:40:00 | Brute-force attack mitigated by IP blocking and CAPTCHA. |
|  5 |           5 | 2023-02-14 12:50:00 | Man-in-the-middle attack prevented using encryption. |
+----+-------------+---------------------+------------------------------------------------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM incidents ;
+----+---------------------+---------------+---------------------------------------------------+
| id | incident_date       | incident_type | incident_description                              |
+----+---------------------+---------------+---------------------------------------------------+
|  1 | 2023-02-10 08:15:00 |             1 | Detected DDoS attack on the server                |
|  2 | 2023-02-11 09:20:00 |             2 | SQL Injection attempt detected on web application |
|  3 | 2023-02-12 10:25:00 |             3 | XSS vulnerability exploited on the website        |
|  4 | 2023-02-13 11:30:00 |             4 | Brute force attack on admin login detected        |
|  5 | 2023-02-14 12:35:00 |             5 | Man-in-the-middle attack on VPN connection        |
+----+---------------------+---------------+---------------------------------------------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM response_plans ;
+----+-----------+------------------------------------------------------------------+
| id | plan_name | plan_description                                                 |
+----+-----------+------------------------------------------------------------------+
|  1 | Plan A    | Immediate blocking of malicious IPs and traffic throttling       |
|  2 | Plan B    | Database protection using WAF and SQL sanitization               |
|  3 | Plan C    | Script sanitization and input validation on web forms            |
|  4 | Plan D    | Two-factor authentication and password complexity enforcement    |
|  5 | Plan E    | Encrypted communication and session management for secure connections |
+----+-----------+------------------------------------------------------------------+
5 rows in set (0.00 sec)
```

Database 5: Security_Information

1 vulnerabilities table: Inserting data into vulnerabilities table
2 patches table: Inserting data into patches table
3 security_advisories: Inserting data into security_advisories table
4 threat_intelligence: Inserting data into threat_intelligence table
5 security_incidents: Inserting data into security_incidents table

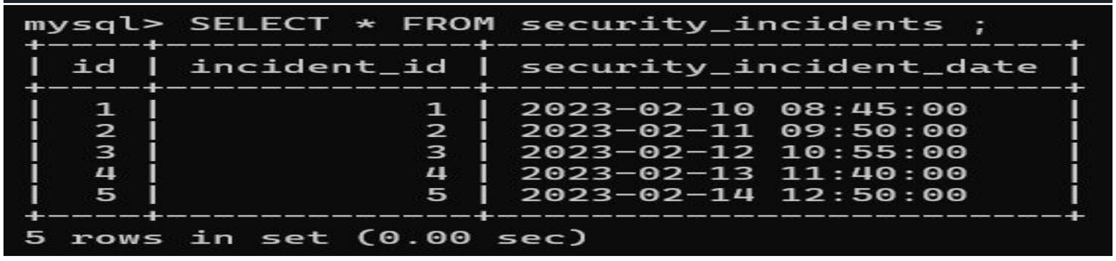


```
mysql> SELECT * FROM security_advisories ;
+----+---------------+--------------------------------------------------------------------+
| id | advisory_name | advisory_description                                               |
+----+---------------+--------------------------------------------------------------------+
|  1 | Advisory A    | Advisory for HTTP server buffer over-low vulnerability             |
|  2 | Advisory B    | Critical advisory for SQL Injection vulrerability                  |
|  3 | Advisory C    | Advisory for Cross-Site Scripting vulnerability                    |
|  4 | Advisory D    | Advisory for Lirux kernel privilege escalation vulnerability       |
|  5 | Advisory E    | Critical advisory for web server remote code execution vulnerability |
+----+---------------+--------------------------------------------------------------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM threat_intelligence ;
+----+---------------+-----------------------------------------------------+--------------+
| id | threat_name   | threat_description                                  | threat_level |
+----+---------------+-----------------------------------------------------+--------------+
|  1 | APT29         | Advanced Persistent Threat from state-sponsored actors | Critical  |
|  2 | FIN7          | Financially motivated cybercrime group              | High         |
|  3 | Lazarus Group | North Korean hacking group                          | Critical     |
|  4 | Anonymous     | Hacktivist collective                               | Medium       |
|  5 | DarkHydrus    | Cyber-espionage group                               | High         |
+----+---------------+-----------------------------------------------------+--------------+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM vulnerabilities ;
+----+----------------+------------------------------------------------+---------------+
| id | vuln_rame      | vuln_description                               | vuln_severity |
+----+----------------+------------------------------------------------+---------------+
|  1 | CVE-2023-12345 | Buffer overflow vulnerability ir HTTP server   | High          |
|  2 | CVE-2023-54321 | SQL Injection vulnerability in login form      | Critical      |
|  3 | CVE-2023-67890 | Cross-Site Scripting in user comments section  | Medium        |
|  4 | CVE-2023-09876 | Privilege escalation vulnerability in Linux kernel | High      |
|  5 | CVE-2023-11223 | Remote code execution in web server            | Critical      |
+----+----------------+------------------------------------------------+---------------+
5 rows in set (0.00 sec)
```

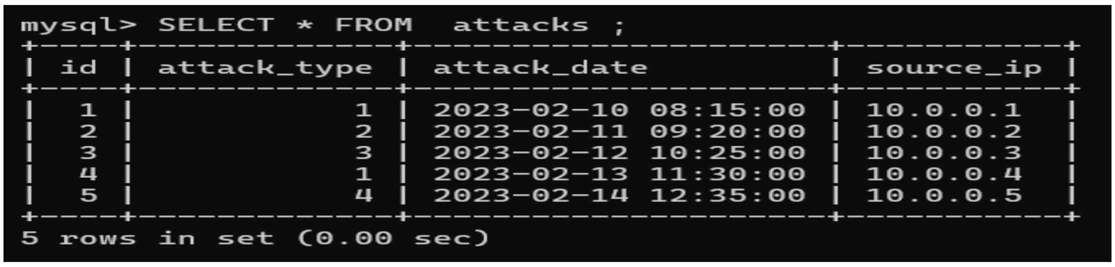
```
mysql> SELECT * FROM patches ;
+----+------------+---------------------------------------------+---------------------+
| id | patch_name | patch_description                           | patch_release_date  |
+----+------------+---------------------------------------------+---------------------+
|  1 | Patch A    | Fixes buffer overflow in HTTP server        | 2023-02-01 10:00:00 |
|  2 | Patch B    | SQL Injection patch for login form          | 2023-02-05 14:00:00 |
|  3 | Patch C    | Cross-Site Scripting prevention in comments | 2023-02-07 12:00:00 |
|  4 | Patch D    | Privilege escalation patch for Linux kernel | 2023-02-09 09:00:00 |
|  5 | Patch E    | Remote code execution fix in web server     | 2023-02-11 16:00:00 |
+----+------------+---------------------------------------------+---------------------+
5 rows in set (0.00 sec)
```



```
mysql> SELECT * FROM security_incidents ;
+----+-------------+---------------------------+
| id | incident_id | security_incident_date    |
+----+-------------+---------------------------+
|  1 |           1 | 2023-02-10 08:45:00       |
|  2 |           2 | 2023-02-11 09:50:00       |
|  3 |           3 | 2023-02-12 10:55:00       |
|  4 |           4 | 2023-02-13 11:40:00       |
|  5 |           5 | 2023-02-14 12:50:00       |
+----+-------------+---------------------------+
5 rows in set (0.00 sec)
```

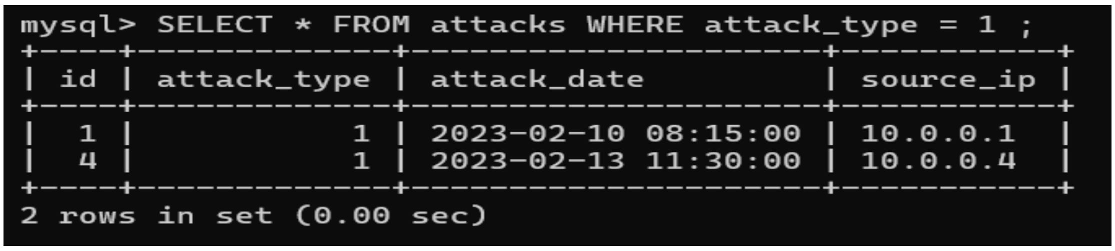
Attacks table

1. SELECT * FROM attacks;



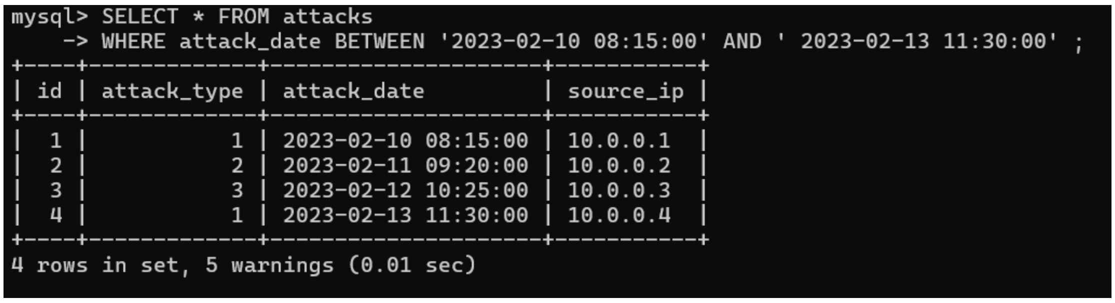
```
mysql> SELECT * FROM  attacks ;
+----+-------------+---------------------+-----------+
| id | attack_type | attack_date         | source_ip |
+----+-------------+---------------------+-----------+
|  1 |           1 | 2023-02-10 08:15:00 | 10.0.0.1  |
|  2 |           2 | 2023-02-11 09:20:00 | 10.0.0.2  |
|  3 |           3 | 2023-02-12 10:25:00 | 10.0.0.3  |
|  4 |           1 | 2023-02-13 11:30:00 | 10.0.0.4  |
|  5 |           4 | 2023-02-14 12:35:00 | 10.0.0.5  |
+----+-------------+---------------------+-----------+
5 rows in set (0.00 sec)
```

2. SELECT * FROM attacks WHERE attack_type = 1;



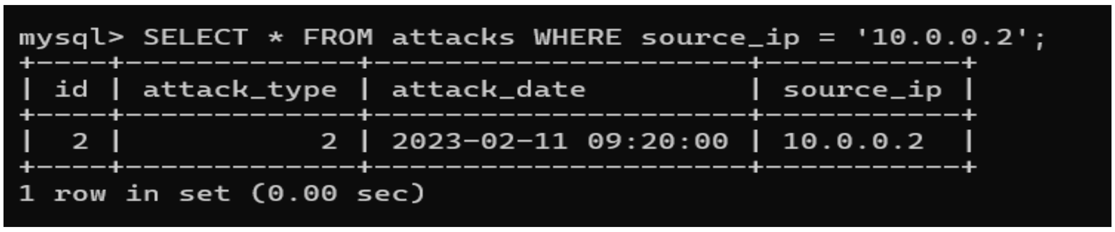
```
mysql> SELECT * FROM attacks WHERE attack_type = 1 ;
+----+-------------+---------------------+-----------+
| id | attack_type | attack_date         | source_ip |
+----+-------------+---------------------+-----------+
|  1 |           1 | 2023-02-10 08:15:00 | 10.0.0.1  |
|  4 |           1 | 2023-02-13 11:30:00 | 10.0.0.4  |
+----+-------------+---------------------+-----------+
2 rows in set (0.00 sec)
```

3. SELECT * FROM attacks WHERE attack_date BETWEEN '2023-02-10 08:15:00' AND   '2023-02-13 11:30:00' ;



```
mysql> SELECT * FROM attacks
    -> WHERE attack_date BETWEEN '2023-02-10 08:15:00' AND ' 2023-02-13 11:30:00' ;
+----+-------------+---------------------+-----------+
| id | attack_type | attack_date         | source_ip |
+----+-------------+---------------------+-----------+
|  1 |           1 | 2023-02-10 08:15:00 | 10.0.0.1  |
|  2 |           2 | 2023-02-11 09:20:00 | 10.0.0.2  |
|  3 |           3 | 2023-02-12 10:25:00 | 10.0.0.3  |
|  4 |           1 | 2023-02-13 11:30:00 | 10.0.0.4  |
+----+-------------+---------------------+-----------+
4 rows in set, 5 warnings (0.01 sec)
```

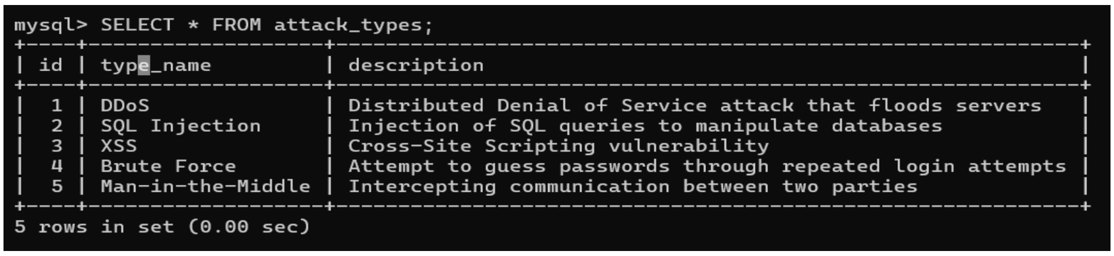
4. SELECT * FROM attacks WHERE source_ip = '10.0.0.2';



```
mysql> SELECT * FROM attacks WHERE source_ip = '10.0.0.2';
+----+-------------+---------------------+-----------+
| id | attack_type | attack_date         | source_ip |
+----+-------------+---------------------+-----------+
|  2 |           2 | 2023-02-11 09:20:00 | 10.0.0.2  |
+----+-------------+---------------------+-----------+
1 row in set (0.00 sec)
```

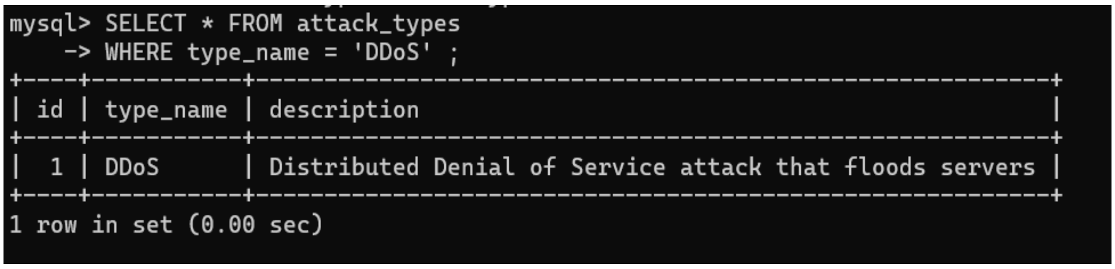
attack_types table

1. SELECT * FROM attack_types;



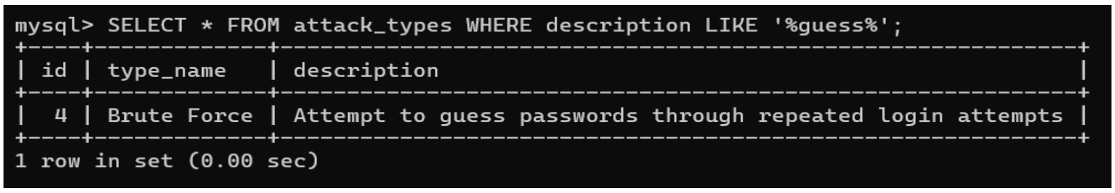
```
mysql> SELECT * FROM attack_types;
+----+-------------------+-------------------------------------------------------------+
| id | type_name         | description                                                 |
+----+-------------------+-------------------------------------------------------------+
|  1 | DDoS              | Distributed Denial of Service attack that floods servers    |
|  2 | SQL Injection     | Injection of SQL queries to manipulate databases            |
|  3 | XSS               | Cross-Site Scripting vulnerability                          |
|  4 | Brute Force       | Attempt to guess passwords through repeated login attempts |
|  5 | Man-in-the-Middle | Intercepting communication between two parties              |
+----+-------------------+-------------------------------------------------------------+
5 rows in set (0.00 sec)
```

2. SELECT * FROM attack_types WHERE type_name = 'DDoS';



```
mysql> SELECT * FROM attack_types
    -> WHERE type_name = 'DDoS' ;
+----+-----------+----------------------------------------------------------+
| id | type_name | description                                              |
+----+-----------+----------------------------------------------------------+
|  1 | DDoS      | Distributed Denial of Service attack that floods servers |
+----+-----------+----------------------------------------------------------+
1 row in set (0.00 sec)
```

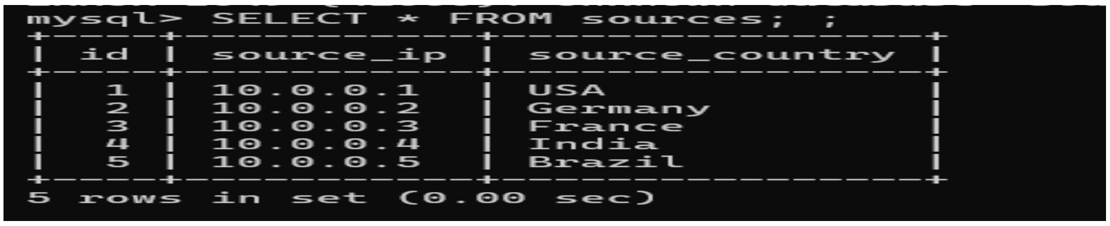
3. SELECT * FROM attack_types WHERE description LIKE '%guess%';



```
mysql> SELECT * FROM attack_types WHERE description LIKE '%guess%';
+----+-------------+------------------------------------------------------------+
| id | type_name   | description                                                |
+----+-------------+------------------------------------------------------------+
|  4 | Brute Force | Attempt to guess passwords through repeated login attempts |
+----+-------------+------------------------------------------------------------+
1 row in set (0.00 sec)
```

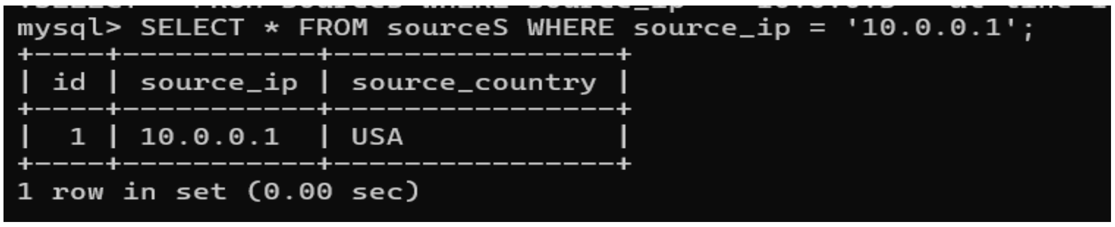
Sources table

1. SELECT * FROM sources;



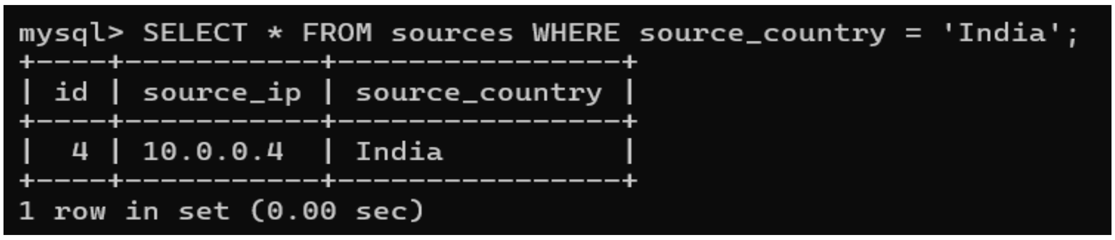
```
mysql> SELECT * FROM sources; ;
+----+------------+----------------+
| id | source_ip  | source_country |
+----+------------+----------------+
|  1 | 10.0.0.1   | USA            |
|  2 | 10.0.0.2   | Germany        |
|  3 | 10.0.0.3   | France         |
|  4 | 10.0.0.4   | India          |
|  5 | 10.0.0.5   | Brazil         |
+----+------------+----------------+
5 rows in set (0.00 sec)
```

2. SELECT * FROM sources WHERE source_ip = '10.0.0.1';



```
mysql> SELECT * FROM sourceS WHERE source_ip = '10.0.0.1';
+----+------------+----------------+
| id | source_ip  | source_country |
+----+------------+----------------+
|  1 | 10.0.0.1   | USA            |
+----+------------+----------------+
1 row in set (0.00 sec)
```

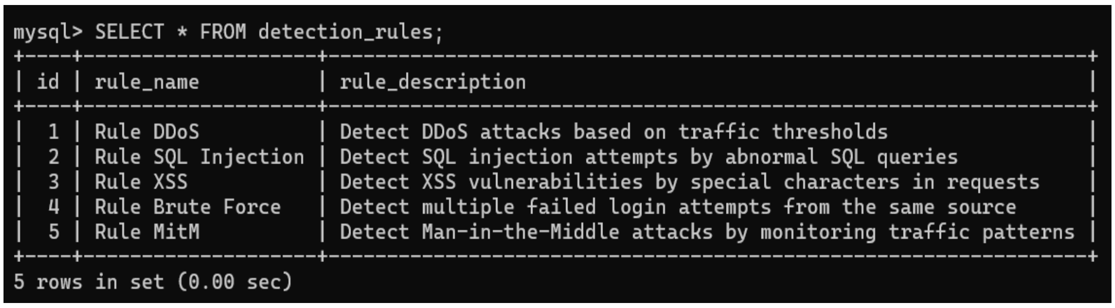
3. SELECT * FROM sources WHERE source_country = India;



```
mysql> SELECT * FROM sources WHERE source_country = 'India';
+----+------------+----------------+
| id | source_ip  | source_country |
+----+------------+----------------+
|  4 | 10.0.0.4   | India          |
+----+------------+----------------+
1 row in set (0.00 sec)
```

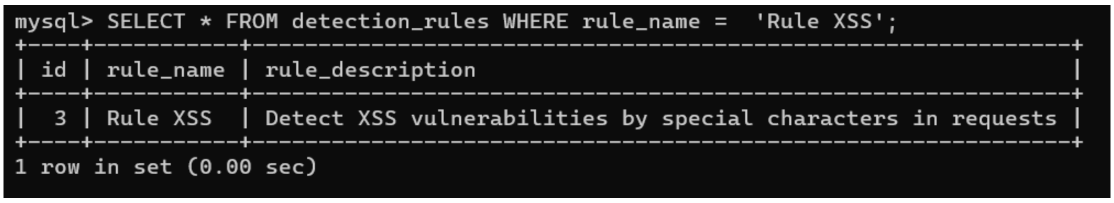
detection_rules table

1. SELECT * FROM detection_rules;



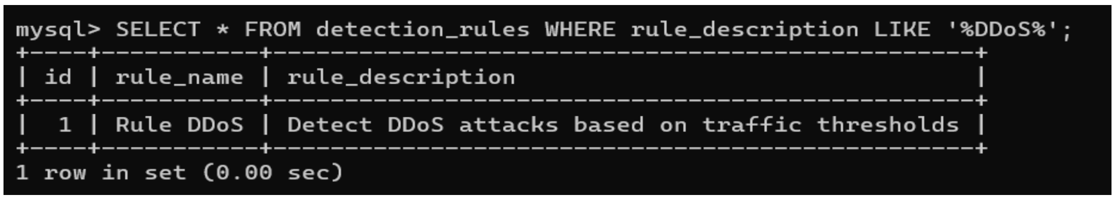
```
mysql> SELECT * FROM detection_rules;
+----+--------------------+-----------------------------------------------------------------+
| id | rule_name          | rule_description                                                |
+----+--------------------+-----------------------------------------------------------------+
|  1 | Rule DDoS          | Detect DDoS attacks based on traffic thresholds                 |
|  2 | Rule SQL Injection | Detect SQL injection attempts by abnormal SQL queries           |
|  3 | Rule XSS           | Detect XSS vulnerabilities by special characters in requests    |
|  4 | Rule Brute Force   | Detect multiple failed login attempts from the same source      |
|  5 | Rule MitM          | Detect Man-in-the-Middle attacks by monitoring traffic patterns |
+----+--------------------+-----------------------------------------------------------------+
5 rows in set (0.00 sec)
```

2. SELECT * FROM detection_rules WHERE rule_name = 'Rule XSS';



```
mysql> SELECT * FROM detection_rules WHERE rule_name =  'Rule XSS';
+----+-----------+--------------------------------------------------------------+
| id | rule_name | rule_description                                             |
+----+-----------+--------------------------------------------------------------+
|  3 | Rule XSS  | Detect XSS vulnerabilities by special characters in requests |
+----+-----------+--------------------------------------------------------------+
1 row in set (0.00 sec)
```

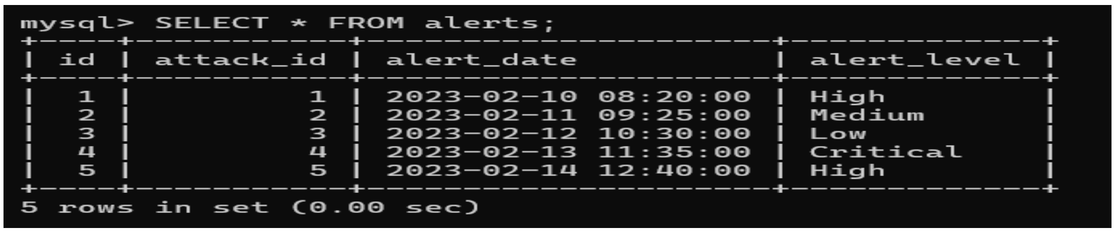
3. SELECT * FROM detection_rules WHERE rule_description LIKE '%DDoS%';



```
mysql> SELECT * FROM detection_rules WHERE rule_description LIKE '%DDoS%';
+----+-----------+-------------------------------------------------+
| id | rule_name | rule_description                                |
+----+-----------+-------------------------------------------------+
|  1 | Rule DDoS | Detect DDoS attacks based on traffic thresholds |
+----+-----------+-------------------------------------------------+
1 row in set (0.00 sec)
```

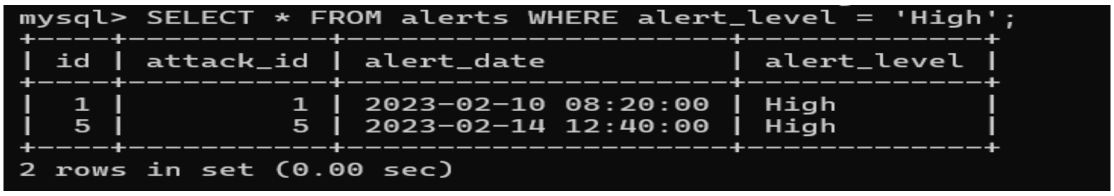
Alerts table

1. SELECT * FROM alerts;



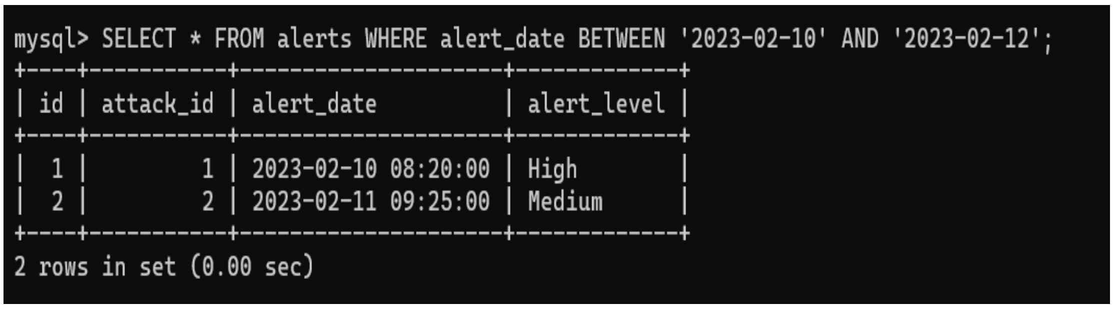
```
mysql> SELECT * FROM alerts;
+----+-----------+---------------------+-------------+
| id | attack_id | alert_date          | alert_level |
+----+-----------+---------------------+-------------+
|  1 |         1 | 2023-02-10 08:20:00 | High        |
|  2 |         2 | 2023-02-11 09:25:00 | Medium      |
|  3 |         3 | 2023-02-12 10:30:00 | Low         |
|  4 |         4 | 2023-02-13 11:35:00 | Critical    |
|  5 |         5 | 2023-02-14 12:40:00 | High        |
+----+-----------+---------------------+-------------+
5 rows in set (0.00 sec)
```

2. SELECT * FROM alerts WHERE alert_level = 'High';



```
mysql> SELECT * FROM alerts WHERE alert_level = 'High';
+----+-----------+---------------------+-------------+
| id | attack_id | alert_date          | alert_level |
+----+-----------+---------------------+-------------+
|  1 |         1 | 2023-02-10 08:20:00 | High        |
|  5 |         5 | 2023-02-14 12:40:00 | High        |
+----+-----------+---------------------+-------------+
2 rows in set (0.00 sec)
```

3. SELECT * FROM alerts WHERE alert_date BETWEEN '2023-02-10' AND '2023-02-12';



```
mysql> SELECT * FROM alerts WHERE alert_date BETWEEN '2023-02-10' AND '2023-02-12';
+----+-----------+---------------------+-------------+
| id | attack_id | alert_date          | alert_level |
+----+-----------+---------------------+-------------+
|  1 |         1 | 2023-02-10 08:20:00 | High        |
|  2 |         2 | 2023-02-11 09:25:00 | Medium      |
+----+-----------+---------------------+-------------+
2 rows in set (0.00 sec)
```

Goal of This Project:

Based on the schema and tables created, is to build a comprehensive **Cyber security Attack Detection and Response System** that integrates various components to effectively detect, analyze, and respond to security threats. Here's a breakdown of the core objectives of the project:

1. Attack Detection

- **Goal:** Detect various types of cyber-attacks in real-time, such as Distributed Denial of Service (DDoS), SQL Injection, Cross-Site Scripting (XSS), Brute Force attacks, and more.

2. Incident Management

- **Goal:** Respond to detected threats with a defined incident response plan that mitigates the impact of security breaches.

3. Resource Monitoring and Optimization

- **Goal:** Continuously monitor system resource usage (CPU, memory, disk) to ensure optimal performance and detect abnormal usage patterns that may indicate an attack.

4. Threat Intelligence

- **Goal:** Leverage external threat intelligence data to stay ahead of known and emerging threats.

5. Security Automation

- **Goal:** Automate the detection, response, and recovery from cyber-attacks to reduce human error and reaction time

6. Reporting and Auditing

- **Goal:** Generate detailed security reports and maintain logs for auditing and compliance purposes.