Importing JSON into MySQL with PHP

- Creating a Security Events Table
- Importing JSON into MySQL with PHP
- Querying Imported Data with SQL
- Building a Security Dashboard Using PHP and MySQL

Creating a Security Events Table

In this tutorial, you will create a MySQL table to store security event data imported from a JSON file.

The JSON file contains homes, devices and events. Rather than storing the entire JSON structure, each event will be stored as a separate row in a database table.

This approach makes it easier to search, filter and analyse the data using SQL.

Open phpMyAdmin

Open phpMyAdmin in your browser.

Example:

http://localhost/phpmyadmin/

Select your existing database:

```
project_db
```

Create the Security Events Table

Select the **SQL** tab and run:

```
CREATE TABLE security_events (

    event_id INT AUTO_INCREMENT PRIMARY KEY,

    device_id VARCHAR(50) NOT NULL,

    device_type VARCHAR(100) NOT NULL,
```

```
    room VARCHAR(100) NOT NULL,
    event_timestamp DATETIME NOT NULL,
    event_type VARCHAR(100) NOT NULL,
    data_transmitted VARCHAR(100) NOT NULL,
    severity VARCHAR(20) NOT NULL,
    access_result VARCHAR(20) NOT NULL
);
```

This creates a table capable of storing all event information from the JSON file.

Review the Table Structure

The completed table should contain the following fields:

Field	Purpose
event_id	Unique identifier for each event
device_id	Device that generated the event
device_type	Type of device
room	Location of the device
event_timestamp	Date and time of the event
event_type	Type of activity
data_transmitted	Data involved in the event
severity	Event severity level
access_result	Result of the event

View the Table Structure

Run:

```
DESCRIBE security_events;
```

You should see a structure similar to:

Field	Type
event_id	int
device_id	varchar(50)
device_type	varchar(100)
room	varchar(100)
event_timestamp	datetime
event_type	varchar(100)
data_transmitted	varchar(100)
severity	varchar(20)
access_result	varchar(20)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**event_id**	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
2	**device_id**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
3	**device_type**	varchar(100)	utf8mb4_general_ci		No	*None*			Change Drop More
4	**room**	varchar(100)	utf8mb4_general_ci		No	*None*			Change Drop More
5	**event_timestamp**	datetime			No	*None*			Change Drop More
6	**event_type**	varchar(100)	utf8mb4_general_ci		No	*None*			Change Drop More
7	**data_transmitted**	varchar(100)	utf8mb4_general_ci		No	*None*			Change Drop More
8	**severity**	varchar(20)	utf8mb4_general_ci		No	*None*			Change Drop More
9	**access_result**	varchar(20)	utf8mb4_general_ci		No	*None*			Change Drop More

Insert a Test Record

Before importing JSON data, insert a test record to verify the table works correctly.

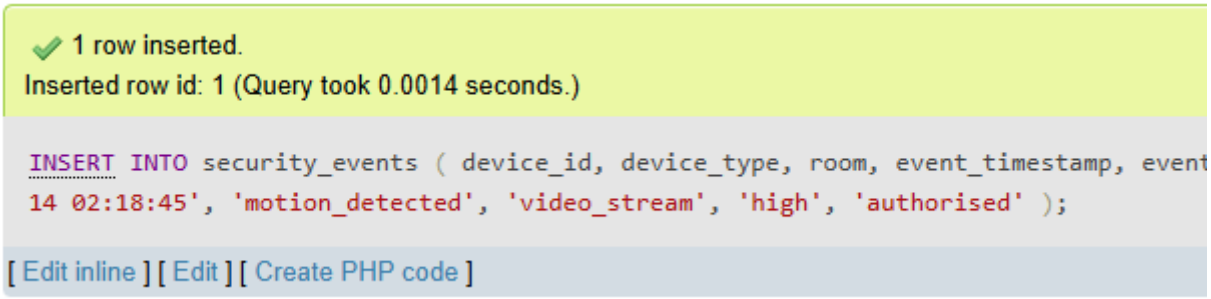
Run:

```
INSERT INTO security_events (

    device_id,
    device_type,
    room,
```

```
    event_timestamp,
    event_type,
    data_transmitted,
    severity,
    access_result

)
VALUES (

    'CAM-01',
    'Security Camera',
    'Front Entrance',
    '2026-03-14 02:18:45',
    'motion_detected',
    'video_stream',
    'high',
    'authorised'

);
```

If successful, MySQL will report:

```
1 row inserted
```



View the Data

Run:

```
SELECT * FROM security_events;
```

You should see:

event_id	device_id	device_type	room	event_timestamp	event_type	severity
1	CAM-01	Security Camera	Front Entrance	2026-03-14 02:18:45	motion_detected	high

Delete the Test Record

The test record is no longer required.

Run:

```
DELETE FROM security_events;
```

Verify the table is empty:

```
SELECT * FROM security_events;
```

You should see:

```
Empty set
```

Complete SQL Script

```
CREATE TABLE security_events (

    event_id INT AUTO_INCREMENT PRIMARY KEY,

    device_id VARCHAR(50) NOT NULL,

    device_type VARCHAR(100) NOT NULL,

    room VARCHAR(100) NOT NULL,

    event_timestamp DATETIME NOT NULL,

    event_type VARCHAR(100) NOT NULL,
```

```
    data_transmitted VARCHAR(100) NOT NULL,

    severity VARCHAR(20) NOT NULL,

    access_result VARCHAR(20) NOT NULL

);

INSERT INTO security_events (

    device_id,
    device_type,
    room,
    event_timestamp,
    event_type,
    data_transmitted,
    severity,
    access_result

)
VALUES (

    'CAM-01',
    'Security Camera',
    'Front Entrance',
    '2026-03-14 02:18:45',
    'motion_detected',
    'video_stream',
    'high',
    'authorised'

);

SELECT * FROM security_events;

DELETE FROM security_events;
```

You now have a database table ready to receive event data from a JSON file.

Next tutorial: **Importing JSON into MySQL with PHP**.

Importing JSON into MySQL with PHP

In the previous tutorial, you created a `security_events` table. In this tutorial, you will read data from a JSON file and import it into MySQL using PHP.

The JSON file contains devices and events. Each event will be inserted into the `security_events` table as a separate database record.

Project Structure

Your project should contain:

```
json-import
├── import_json.php
└── smart_security_data.json
```

Copy the JSON file into your project folder.

Create the Import Script

Create a new file called:

```
import_json.php
```

Add the following code:

```
<?php

$conn = new mysqli(
    "localhost",
    "root",
```

```
    "",
    "project_db"
);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

echo "Database connection successful.";

?>
```

Test the Database Connection

Open:

```
http://localhost/json-import/import_json.php
```

You should see:

```
Database connection successful.
```

Read the JSON File

Add the following code underneath the database connection:

```
$json = file_get_contents(
    "smart_security_data.json"
);

echo $json;
```

Refresh the page.

The contents of the JSON file should be displayed in the browser.

{ "homeId": "GC-HOME-014", "location": "Gold Coast", "devices": [{ "deviceId": "CAM-01", "deviceType": "Security Car
"video_stream", "severity": "high", "accessResult": "authorised" }, { "timestamp": "2026-03-14T02:19:10", "eventType": "re
"LOCK-02", "deviceType": "Smart Door Lock", "room": "Back Door", "events": [{ "timestamp": "2026-03-13T23:47:02", "

Convert the JSON into a PHP Array

Replace:

```
echo $json;
```

with:

```
$data = json_decode(
    $json,
    true
);

print_r($data);
```

Refresh the page.

You should now see a PHP array structure.

Array ([homeId] => GC-HOME-014 [location] => Gold Coast [devices] => Array ([0] => Array ([deviceId] =
[eventType] => motion_detected [dataTransmitted] => video_stream [severity] => high [accessResult] => auth
=> critical [accessResult] => blocked))) [1] => Array ([deviceId] => LOCK-02 [deviceType] => Smart Doo
=> access_log [severity] => medium [accessResult] => denied)))))

Loop Through the Devices

Replace:

```
print_r($data);
```

with:

```
foreach ($data["devices"] as $device) {

    echo "<h3>";
```

```
    echo $device["deviceId"];
    echo "</h3>";

}
```

Refresh the page.

You should see:

```
CAM-01


LOCK-02
```

Loop Through the Events

Replace the previous code with:

```
foreach ($data["devices"] as $device) {

    foreach ($device["events"] as $event) {

        echo $event["eventType"];
        echo "<br>";

    }

}
```

Refresh the page.

You should see:

```
motion_detected
remote_access_attempt
unlock_attempt
```

Prepare the INSERT Statement

Add the following code before the loops:

```
$stmt = $conn->prepare(
    "INSERT INTO security_events (

        device_id,
        device_type,
        room,
        event_timestamp,
        event_type,
        data_transmitted,
        severity,
        access_result

    )
    VALUES (?, ?, ?, ?, ?, ?, ?, ?)"
);
```

This statement will be reused for every event.

Insert the Events

Replace the existing loops with:

```
foreach ($data["devices"] as $device) {

    foreach ($device["events"] as $event) {

        $stmt->bind_param(
            "ssssssss",

            $device["deviceId"],
            $device["deviceType"],
            $device["room"],

            $event["timestamp"],
            $event["eventType"],
            $event["dataTransmitted"],
```

```
            $event["severity"],
            $event["accessResult"]

        );

        $stmt->execute();

    }

}
```

Display a Success Message

Add:

```
echo "Import completed successfully.";
```

after the loops.

The completed import should now run automatically when the page loads.

Run the Import

Open:

```
http://localhost/json-import/import_json.php
```

You should see:

```
Import completed successfully.
```

Verify the Imported Data

Open phpMyAdmin.

Run:

```
SELECT * FROM security_events;
```

You should now see three imported events from the JSON file.

Example:

event_id	device_id	event_type	severity
1	CAM-01	motion_detected	high
2	CAM-01	remote_access_attempt	critical
3	LOCK-02	unlock_attempt	medium

Complete import_json.php File

```
<?php

$conn = new mysqli(
    "localhost",
    "root",
    "",
    "project_db"
);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$json = file_get_contents(
    "smart_security_data.json"
);

$data = json_decode(
    $json,
    true
);

$stmt = $conn->prepare(
```

```
    "INSERT INTO security_events (

        device_id,
        device_type,
        room,
        event_timestamp,
        event_type,
        data_transmitted,
        severity,
        access_result

    )
    VALUES (?, ?, ?, ?, ?, ?, ?, ?)"
);

foreach ($data["devices"] as $device) {

    foreach ($device["events"] as $event) {

        $stmt->bind_param(
            "ssssssss",

            $device["deviceId"],
            $device["deviceType"],
            $device["room"],

            $event["timestamp"],
            $event["eventType"],
            $event["dataTransmitted"],
            $event["severity"],
            $event["accessResult"]

        );

        $stmt->execute();

    }

}
```

```
echo "Import completed successfully.";


?>
```

You have successfully imported JSON data into MySQL using PHP.

Next tutorial: **Querying Imported Data with SQL**.

Querying Imported Data with SQL

In the previous tutorial, you imported JSON data into the `security_events` table. In this tutorial, you will use SQL queries to analyse the imported data.

View All Events

Open phpMyAdmin and select the:

```
security_events
```

table.

Run:

```
SELECT *
FROM security_events;
```

This displays all imported records.

Count All Events

To find the total number of events:

```
SELECT COUNT(*) AS total_events
FROM security_events;
```

Example result:

total_events
3

This query is useful for dashboards and reporting.

Find Critical Events

To display only critical events:

```
SELECT *
FROM security_events
WHERE severity = 'critical';
```

Example result:

device_id	event_type	severity
CAM-01	remote_access_attempt	critical

Count Events by Severity

Run:

```
SELECT
    severity,
    COUNT(*) AS total
FROM security_events
GROUP BY severity;
```

Example result:

severity	total
critical	1
high	1
medium	1

This is often used to build dashboard summary cards.

Find Events from a Specific Device

To display events generated by the security camera:

```
SELECT *
FROM security_events
WHERE device_id = 'CAM-01';
```

Example result:

device_id	event_type
CAM-01	motion_detected
CAM-01	remote_access_attempt

Count Events Per Device

Run:

```
SELECT
    device_id,
    COUNT(*) AS total_events
FROM security_events
GROUP BY device_id;
```

Example result:

device_id	total_events
CAM-01	2
LOCK-02	1

This identifies which devices are generating the most activity.

Find High-Risk Events

To display high and critical events:

```
SELECT *
FROM security_events
WHERE severity IN ('high', 'critical');
```

Example result:

device_id	event_type	severity
CAM-01	motion_detected	high
CAM-01	remote_access_attempt	critical

This query could be used to generate security alerts.

Sort Events by Time

To view the newest events first:

```
SELECT *
FROM security_events
ORDER BY event_timestamp DESC;
```

This is commonly used in event logs and monitoring systems.

Useful Dashboard Queries

Total Events

```
SELECT COUNT(*) AS total_events
FROM security_events;
```

Critical Events

```
SELECT COUNT(*) AS critical_events
FROM security_events
WHERE severity = 'critical';
```

Device Count

```
SELECT COUNT(DISTINCT device_id) AS total_devices
FROM security_events;
```

Events by Severity

```
SELECT
    severity,
    COUNT(*) AS total
FROM security_events
GROUP BY severity;
```

These queries are commonly used when building dashboards.

Complete Practice Queries

```
SELECT *
FROM security_events;

SELECT COUNT(*) AS total_events
FROM security_events;

SELECT *
FROM security_events
WHERE severity = 'critical';

SELECT
    severity,
    COUNT(*) AS total
FROM security_events
GROUP BY severity;

SELECT
    device_id,
    COUNT(*) AS total_events
FROM security_events
GROUP BY device_id;
```

```
SELECT *
FROM security_events
ORDER BY event_timestamp DESC;
```

You have successfully queried imported JSON data using SQL.

Next tutorial: **Building a Security Dashboard Using PHP and MySQL**.

Building a Security Dashboard Using PHP and MySQL

In the previous tutorial, you used SQL queries to analyse imported JSON data. In this tutorial, you will build a dashboard that displays security statistics directly from the database.

The dashboard will display:

- Total Devices
- Total Events
- Critical Events
- High Events
- Recent Security Events

Create the Dashboard File

Create a new file called:

```
dashboard.php
```

Add the database connection:

```
<?php

$conn = new mysqli(
    "localhost",
    "root",
    "",
    "project_db"
);
```

```
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

?>
```

Get the Total Number of Events

Add:

```
$totalEventsQuery = "
    SELECT COUNT(*) AS total_events
    FROM security_events
";

$totalEventsResult =
    $conn->query($totalEventsQuery);

$totalEvents =
    $totalEventsResult->fetch_assoc()["total_events"];
```

Get the Number of Devices

Add:

```
$totalDevicesQuery = "
    SELECT COUNT(DISTINCT device_id)
    AS total_devices
    FROM security_events
";

$totalDevicesResult =
    $conn->query($totalDevicesQuery);

$totalDevices =
```

```
    $totalDevicesResult->fetch_assoc()["total_devices"];
```

Get Critical Events

Add:

```
$criticalEventsQuery = "
    SELECT COUNT(*) AS critical_events
    FROM security_events
    WHERE severity = 'critical'
";

$criticalEventsResult =
    $conn->query($criticalEventsQuery);

$criticalEvents =
    $criticalEventsResult->fetch_assoc()["critical_events"];
```

Get High Events

Add:

```
$highEventsQuery = "
    SELECT COUNT(*) AS high_events
    FROM security_events
    WHERE severity = 'high'
";

$highEventsResult =
    $conn->query($highEventsQuery);

$highEvents =
    $highEventsResult->fetch_assoc()["high_events"];
```

Create the Dashboard Page

Add the following HTML underneath the PHP code:

```
<!DOCTYPE html>
<html>
<head>
    <title>Security Dashboard</title>
</head>
<body>

<h1>Security Dashboard</h1>

<div>
    <h2>Total Devices</h2>
    <p><?php echo $totalDevices; ?></p>
</div>

<div>
    <h2>Total Events</h2>
    <p><?php echo $totalEvents; ?></p>
</div>

<div>
    <h2>Critical Events</h2>
    <p><?php echo $criticalEvents; ?></p>
</div>

<div>
    <h2>High Events</h2>
    <p><?php echo $highEvents; ?></p>
</div>

</body>
</html>
```

Test the Dashboard

Open:

```
http://localhost/json-import/dashboard.php
```

Using the sample data, you should see something similar to:

```
Security Dashboard

Total Devices: 2
Total Events: 3
Critical Events: 1
High Events: 1
```

Display Recent Events

Add the following query before the HTML:

```
$recentEventsQuery = "
    SELECT *
    FROM security_events
    ORDER BY event_timestamp DESC
";

$recentEvents =
    $conn->query($recentEventsQuery);
```

Create the Events Table

Add the following underneath the dashboard cards:

```
<h2>Recent Events</h2>

<table border="1">
```

```
<tr>
    <th>Device</th>
    <th>Event Type</th>
    <th>Severity</th>
    <th>Timestamp</th>
</tr>

<?php

while (
    $row =
    $recentEvents->fetch_assoc()
) {

?>

<tr>
    <td><?php echo $row["device_id"]; ?></td>
    <td><?php echo $row["event_type"]; ?></td>
    <td><?php echo $row["severity"]; ?></td>
    <td><?php echo $row["event_timestamp"]; ?></td>
</tr>

<?php

}

?>

</table>
```

Refresh the page.

You should now see all imported events underneath the dashboard.

Add Basic Styling

Inside the `<head>` section add:

```
<style>

body {
    font-family: Arial, sans-serif;
}

.card {
    border: 1px solid #cccccc;
    border-radius: 8px;
    padding: 15px;
    margin-bottom: 10px;
}

table {
    border-collapse: collapse;
    width: 100%;
}

th,
td {
    border: 1px solid #cccccc;
    padding: 10px;
}

th {
    background-color: #f2f2f2;
}

</style>
```

Then update each dashboard `<div>`:

```
<div class="card">
```

The dashboard should now look much more professional.

Complete Dashboard Features

Your dashboard now:

- Connects to MySQL
- Runs SQL queries
- Displays summary statistics
- Displays recent events
- Updates automatically whenever new JSON data is imported

This is one of the major advantages of importing JSON into a database rather than displaying the JSON directly.

Next tutorial: **Creating Filters and Search Options for the Dashboard**.