SOFTWARE TESTING ASSIGNMENT

MODULE – 4

DARSHAN GOHIL

WHAT IS RDBMS?

- A Relational Database Management System (RDBMS) is a software application used to efficiently store, manage, and retrieve data that is organized in a structured format using tables.
- These tables, also known as relations, consist of rows and columns—where each row represents a single record, and each column represents a specific attribute of the data.
- One of the defining features of an RDBMS is its use of Structured Query Language (SQL), which allows users to perform operations such as inserting, updating, deleting, and querying data with precision and flexibility.
- RDBMS ensures data integrity and consistency through the use of constraints like primary keys (which uniquely identify each record), foreign keys (which define relationships between tables), and other rules.
- Additionally, it follows ACID properties (Atomicity, Consistency, Isolation, Durability) to guarantee reliable and secure transaction processing.
- Popular examples of RDBMS include MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server, and SQLite. These systems are widely used in everything from small applications to large enterprise systems due to their robustness, scalability, and ability to handle complex relationships between data.

WHAT IS SQL?

- SQL (Structured Query Language) is a standard programming language specifically designed for managing and manipulating data in a Relational Database Management System (RDBMS).
- It allows users to interact with the database by writing queries to perform tasks such as retrieving, inserting, updating, and deleting data stored in tables.
- SQL is essential for working with relational databases because it provides a clear and consistent way to access and modify structured data.
- SQL is widely used in many database systems like MySQL, PostgreSQL, Oracle, and Microsoft SQL Server.
- It supports a wide range of commands grouped into different categories:
  - **DQL – DATA QUERY LANGUAGE**
    - SELECT – Retrieves data from one or more table.
  - **DDL – DATA DEFINITION LANGUAGE**
    - CREATE, ALTER, DROP – Used to create or modify tables & other database projects.
  - **DML – DATA MANIPULATION LANGUAGE**
    - INSERT, UPDATE, DELETE – Used to add, change or remove data.
  - **DCL – DATA CONTROL LANGUAGE**
    - GRANT, REVOKE – Used to give or take away user permission.

WHAT IS JOIN

- In a relational database, data is often split into multiple tables to reduce redundancy.
- A JOIN helps you bring that data back together when you need to see related information from different tables.
- Types Of JOIN:
  - **INNER JOIN:**
    - Returns records that have matching values in both tables.
  - **LEFT JOIN:**
    - Returns all record from the left table, & the matched records from the right table.
    - If no match, the result is **NULL** on the right side.
  - **RIGHT JOIN:**
    - Returns all records from the right table, & the matched records from the left table.
    - If no match, the result is **NULL** on the left side.
  - **FULL JOIN:**
    - Returns all records when there is a match in either left or right table.
    - If there is no match, the result is **NULL** from the missing side.

DIFFERENCE – DBMS & RDBMS

FEATURE	DBMS	RDBMS
Data Storage	Stores data as files or in a hierarchical form.	Stores data in tables (rows & columns).
Data Relationships	No relationships between data.	Relationships between tables using foreign keys.
Normalization	Not supported.	Supports data normalization to avoid redundancy.
Data Integrity	Less focus on data accuracy & integrity.	Ensures data integrity using constraints.
Examples	File System, XML, MS Access.	MySQL, Oracle, PostgreSQL, MS SQL Server.
Security	Basic security features.	Advanced user-level security features.
Concurrency Control	Limited.	Supports multi-user access & transactions
ACID Compliance	May not supported.	Fully supports ACID (Atomicity, Consistency, Isolation, Durability) properties.
Scalability & Performance	Suitable for small systems.	Designed for large, complex, multi-user applications.

WHAT IS ALIAS IN SQL?

- In SQL, an alias is a temporary name given to a table or a column, mainly used to make queries more readable and the results more understandable.
- Aliases are especially helpful when working with complex queries, long table or column names, or when performing joins between multiple tables.
- There are two main types of aliases: column aliases and table aliases.
- A column alias is used to rename the output of a column in the result set.
- A table alias is commonly used in queries involving multiple tables, especially joins.
- It allows you to assign a short name to a table, making the SQL statement easier to write and read.

WHAT IS API TESTING?

- API Testing is a type of software testing that focuses on verifying whether an Application Programming Interface (API) behaves as expected in terms of functionality, reliability, performance, and security.
- An API allows different software systems to communicate with each other. It defines a set of rules and protocols for accessing a web-based software application or tool.
- In API testing, testers send various types of requests (like GET, POST, PUT, DELETE) to the API endpoints and examine the responses.
  - The response data is correct (e.g., correct value, type, status code).
  - The response time is acceptable.
  - The API handles edge cases and errors properly.

TYPES OF API TESTING – OPEN API TESTING

- Open API testing involves validating APIs that are documented and described using the OpenAPI Specification.
- This approach provides a clear and standardized way to understand what an API does, what inputs it expects, and what outputs it returns.
- Using this specification, developers and testers can generate test cases automatically, simulate different request scenarios, and verify the accuracy of responses.
- With OpenAPI, testing becomes more efficient because changes in the API specification can be quickly reflected in the testing process.
- This ensures consistency between the API documentation and its actual behavior, reduces human error, and speeds up the quality assurance phase.
- Overall, Open API testing plays a critical role in modern API development by supporting automation, improving accuracy, and enhancing collaboration between teams.

TYPES OF API TESTING – PARTNER API TESTING

- Partner API testing is the process of verifying and validating APIs that are exposed to external business partners.
- Unlike internal APIs, which are used within a company, partner APIs must be carefully tested to ensure they are secure, stable, and perform well under various external conditions.
- This testing includes checking how the API handles authentication, authorization, and data exchange with outside systems.
- Since partner APIs often involve sensitive business logic or customer data, security testing is critical to prevent unauthorized access and data breaches.
- Overall, partner API testing helps build trust with external users, ensures smooth integration with other platforms, and supports the long-term stability of cross-company collaborations.

Types of API Testing – Internal API Testing

- Internal API testing focuses on validating the APIs that are used within a company's own systems, such as those connecting backend services, databases, or microservices.
- Since these APIs are not accessible by external users, the main goal is to ensure that all internal components communicate correctly, efficiently, and reliably.
- Testing includes checking data flow between services, verifying business logic, and ensuring that performance and response times meet expected standards.
- Internal API testing helps development teams catch bugs early, maintain high system reliability, and ensure a smooth end-user experience, even if the user never directly interacts with those APIs.

WHAT IS RESPONSIVE TESTING?

- Responsive testing is a crucial part of front-end testing that ensures a website or web application looks and functions properly across a variety of devices and screen sizes.
- With the wide range of smartphones, tablets, and desktop monitors available today, it's important that the user interface adjusts fluidly to provide a consistent and user-friendly experience.
- During responsive testing, testers check layout alignment, font sizes, image scaling, button placements, navigation behavior, and overall usability on different devices and browsers.
- This can be done manually by resizing browser windows or using device simulators and emulators, as well as through automated tools like BrowserStack, LambdaTest, or Selenium with responsive frameworks.
- The main goal is to ensure that the content remains accessible, readable, and interactive, regardless of the screen being used.
- Responsive testing helps maintain a professional appearance, improves user satisfaction, and supports accessibility standards in modern web development.

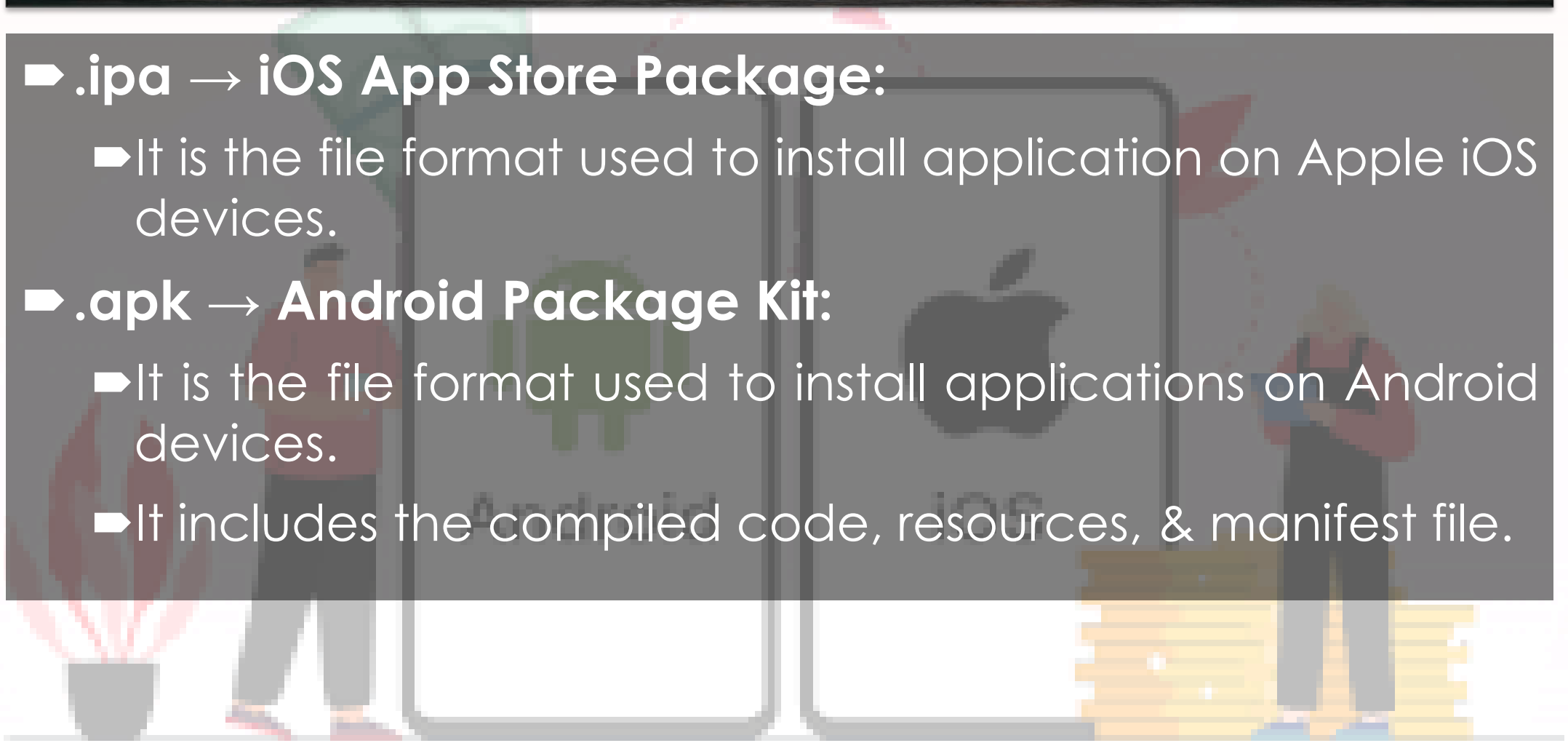
WHICH TYPES OF TOOLS ARE AVAILABLE FOR RESPONSIVE TESTING

- **RESPONSIVE TESTING TOOLS:**
- LT Browser
- Lembda Testing
- Google Resizer
- I am responsive
- Pixel tuner

- **EXAMPLE:**
  - I AM RESPONSIVE WEBSITE.

WHAT IS FULL FORM OF .ipa, .apk

- **.ipa → iOS App Store Package:**
  - It is the file format used to install application on Apple iOS devices.
- **.apk → Android Package Kit:**
  - It is the file format used to install applications on Android devices.
  - It includes the compiled code, resources, & manifest file.



How to create step for to open developer option mode on? - Android

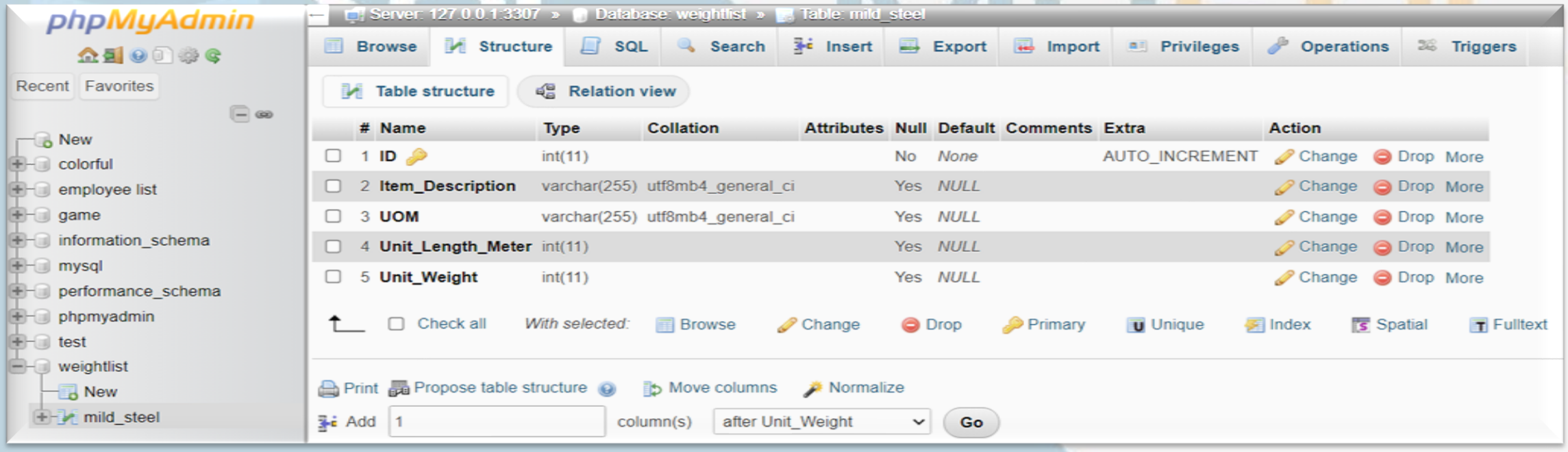
- Open the Settings app on your Android device.
- Scroll down and tap on "About phone" (or "About device", depending on your phone model).
- Find the option called "Build number".
- Tap on "Build number" 7 times continuously.
  - You may be prompted to enter your device PIN or password.
- After that, you'll see a message:
  - "You are now a developer!"
- Go back to the Settings main menu.
- You will now see a new option called "Developer options" (usually under System or near the bottom).

How to create step for to open Developer option mode on? - iPhone

- Connect your iPhone or iPad to a Mac using a USB cable.
- Open Xcode on your Mac.
  - If you don’t have it, you can download Xcode from the Mac App Store.
- In Xcode, go to Window > Devices and Simulators.
- Your connected iOS device should appear in the list. Select it.
- If prompted, allow the device to be used for development and trust the computer on the iPhone.
- After a moment, your iPhone will show a popup asking:
  - "Developer Mode Required
  - "Tap “Turn On”.
- Your iPhone will then restart to enable Developer Mode.
- After restarting, you'll get another popup asking to confirm.
  - Tap "Enable".

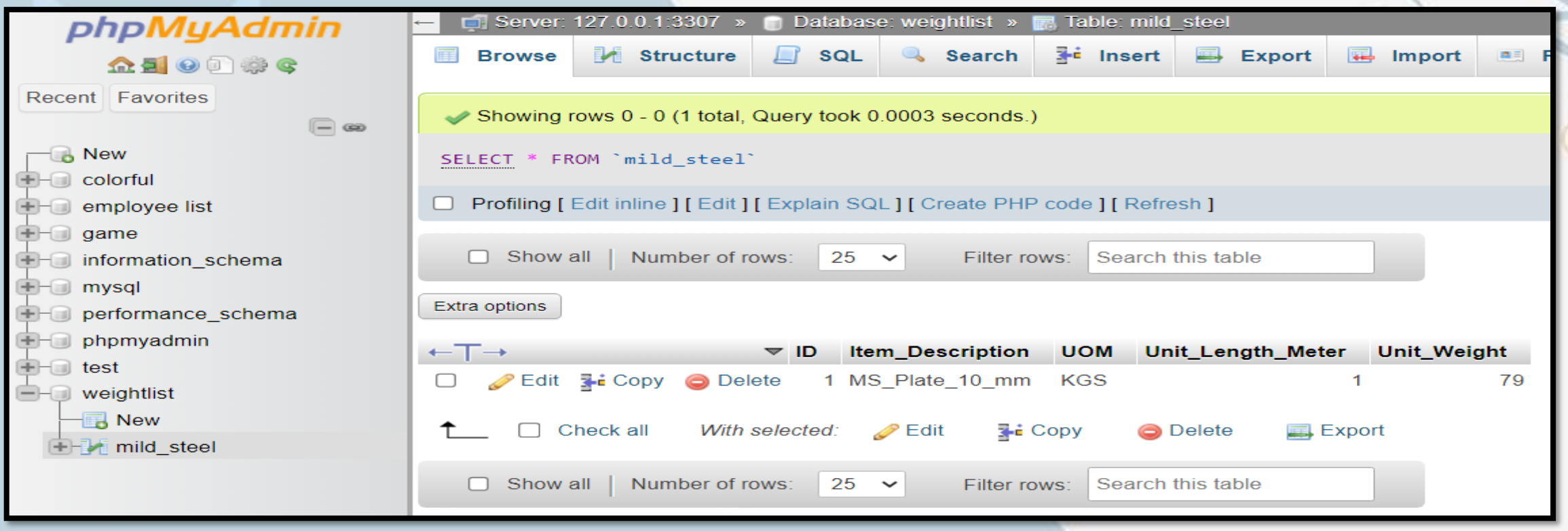
WRITE A QUERY TO CREATE THE TABLE IN SQL

- CREATE DATABASE ***"DATABASE NAME"***
- CREATE TABLE ***"TABLE NAME"*** (***"FIELD NAME WITH VARIABLE"***, ***FIELD NAME WITH VARIABLE"***)
- EXAMPLE:
  - CREATE DATABASE WeightList
  - CREATE TABLE Mild_Steel (ID int PRIMARY KEY AUTO_INCREMENT, Item_Description varchar(255), UOM varchar(255), Unit_Length_Meter int, Unit_Weight int)



WRITE A QUERY TO INSERT DATA INTO TABLE

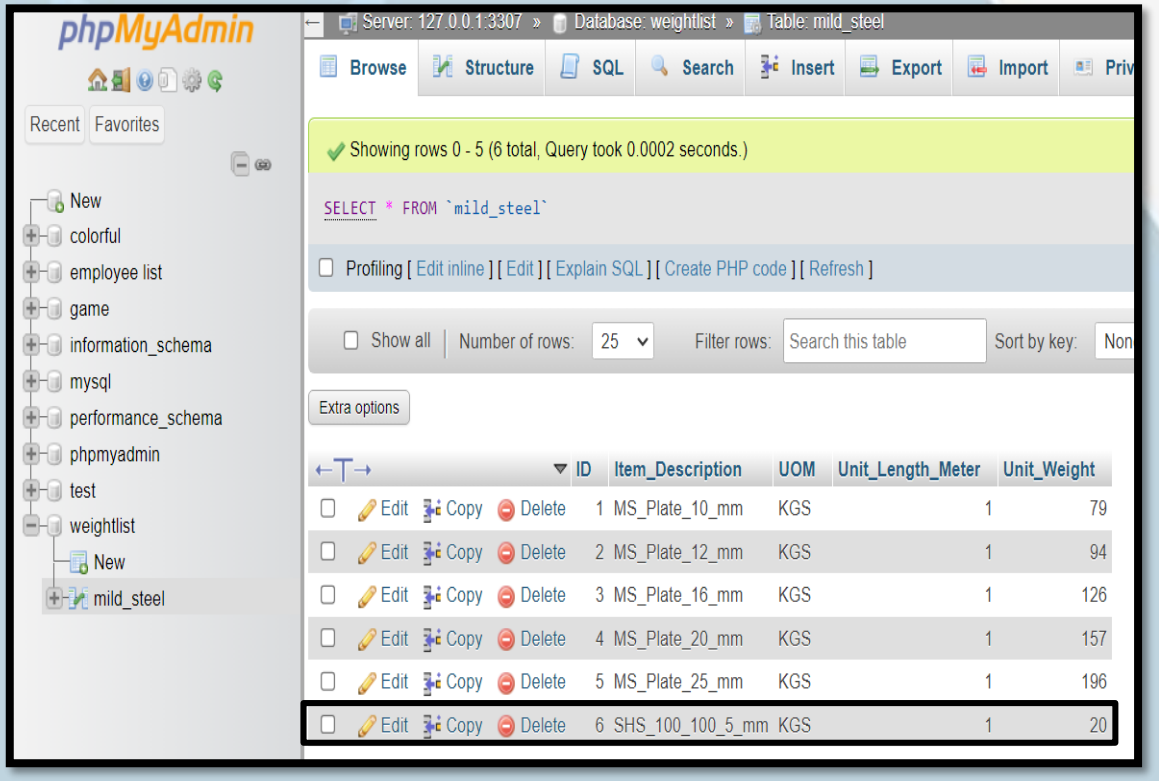
- INSERT INTO ***"TABLE NAME"*** (***"FIELD NAME WITH VARIABLE"***, ***FIELD NAME WITH VARIABLE"***) VALUES ('***INSERT DATA***', '***INSERT DATA***')
- EXAMPLE:
  - INSERT INTO mild_steel (Item_Description,UOM,Unit_Length_Meter,Unit_Weight) VALUES ('MS_Plate_10_mm','KGS','1.000','78.5')



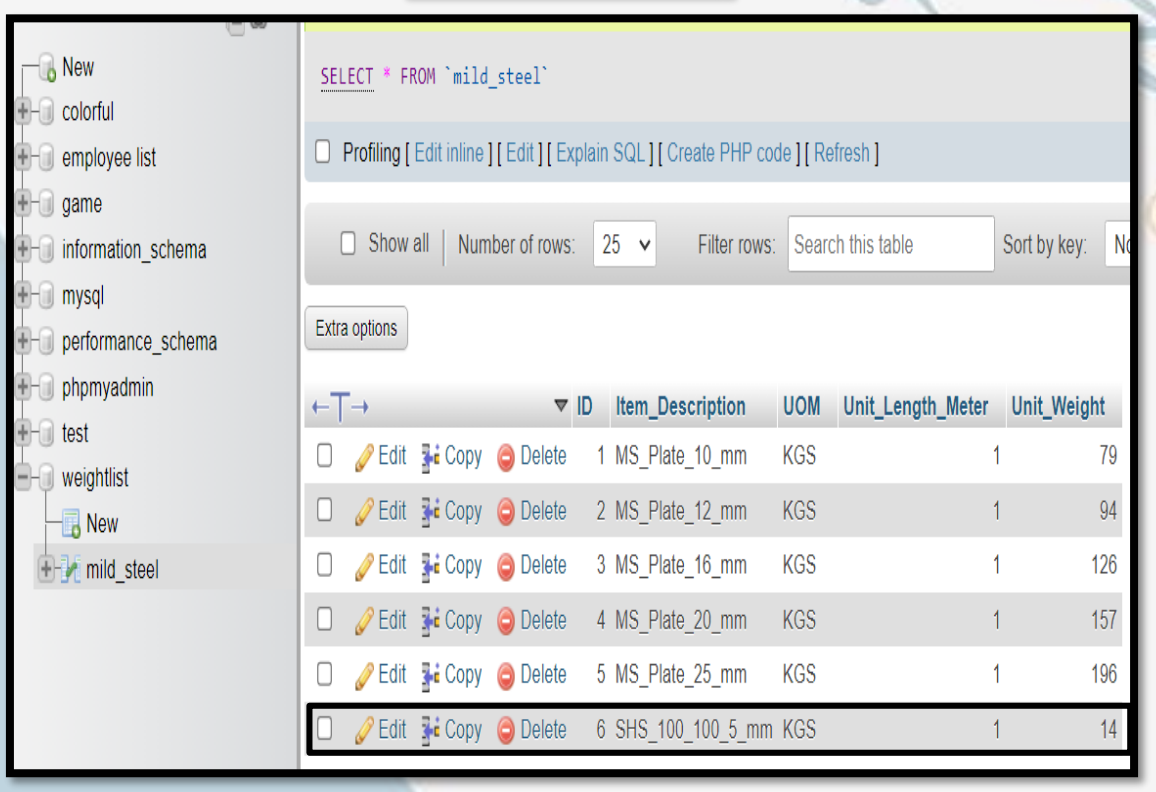
WRITE A QUERY TO UPDATE DATA INTO TABLE WITH VALIDATIONS

- UPDATE '***TABLE NAME***' SET '***FIELD NAME***'='***DATA***' WHERE id='***ID NUMBER***'
- EXAMPLE:
  - UPDATE mild_steel SET Unit_Weight='14.41' WHERE id='6'

BEFORE



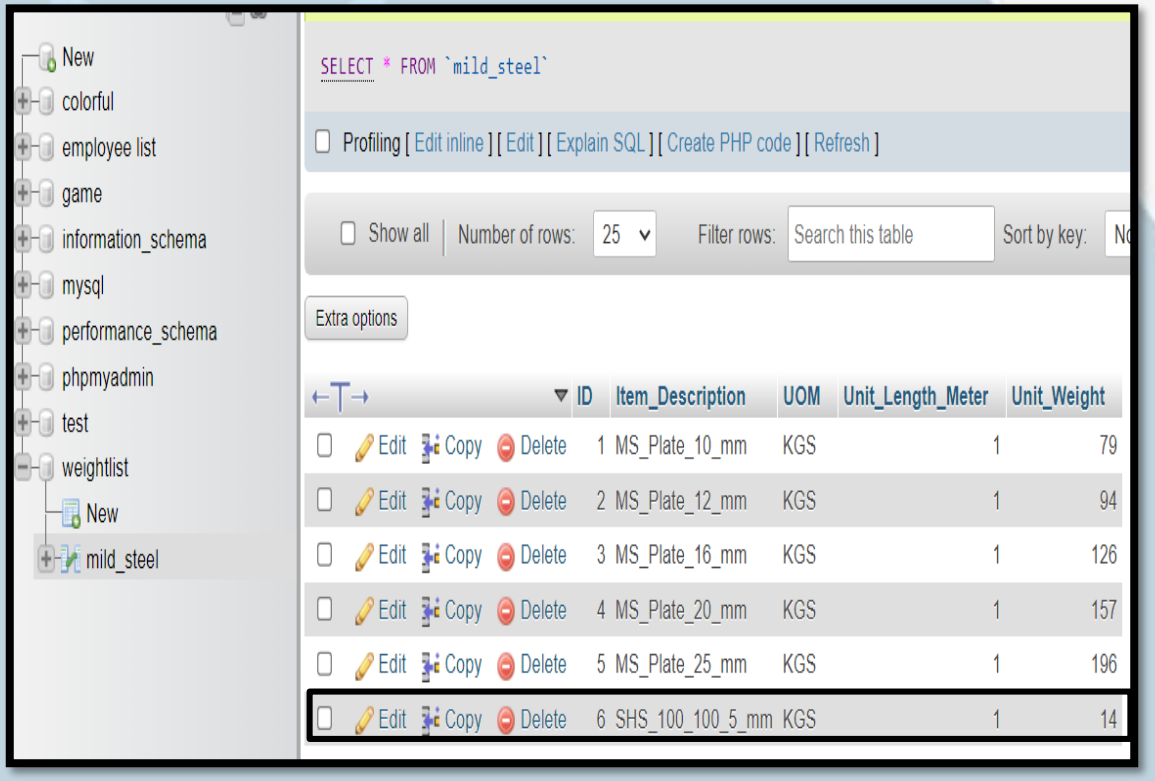
AFTER



WRITE A QUERY TO DELETE DATA INTO TABLE WITH VALIDATIONS

- DELETE FROM '***TABLE NAME***' WHERE id='***ID NUMBER***'
- EXAMPLE:
  - DELETE FROM mild_steel WHERE id='6'

BEFORE

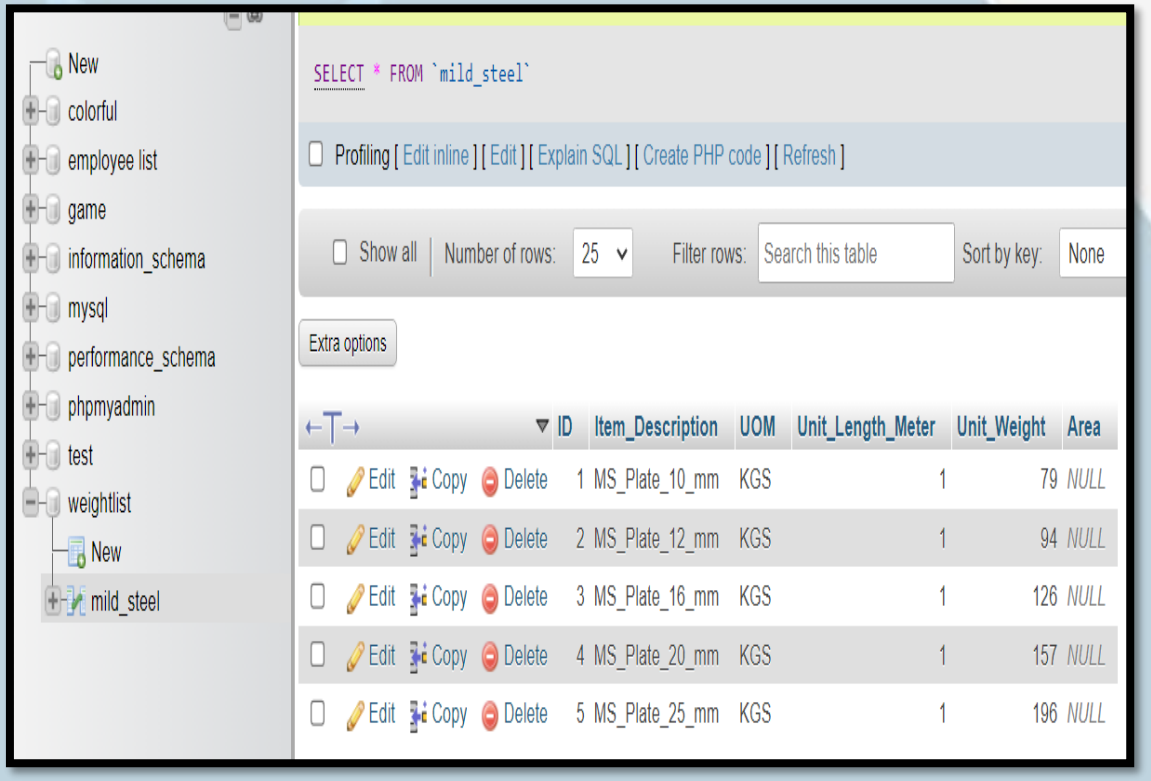


AFTER

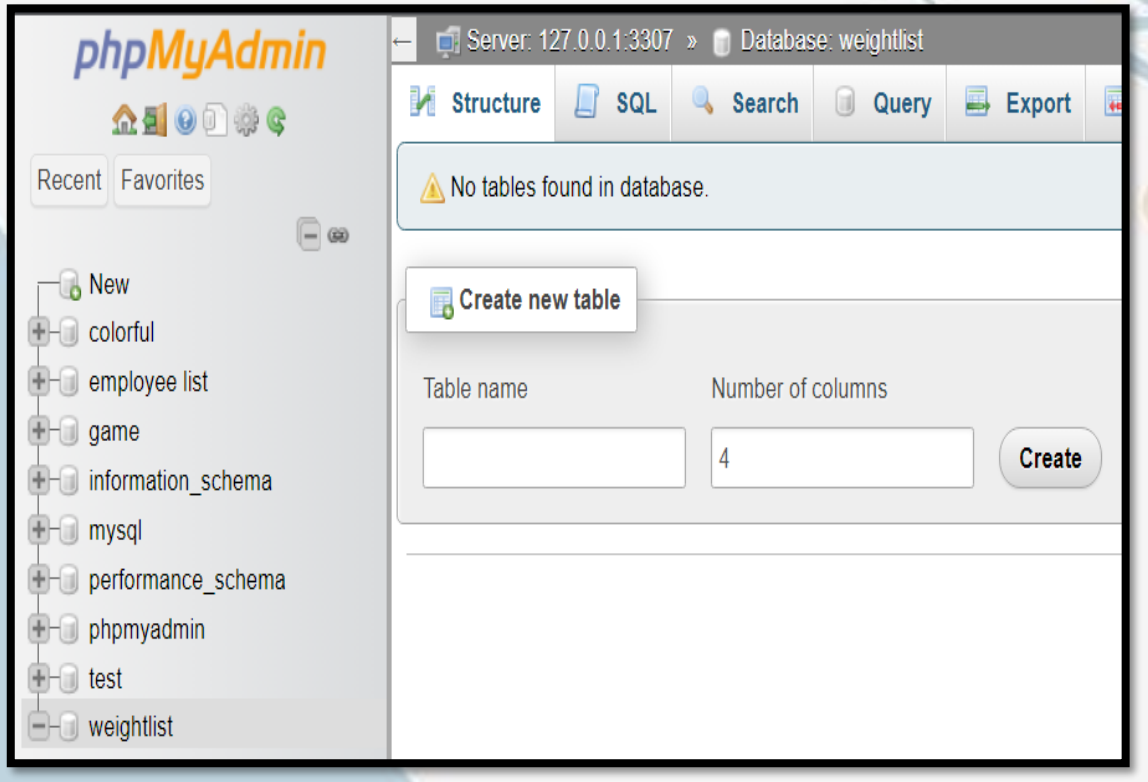
WRITE A QUERY TO DROP TABLE

- DROP TABLE '***TABLE NAME***'
- EXAMPLE:
  - DROP TABLE mild_steel

BEFORE



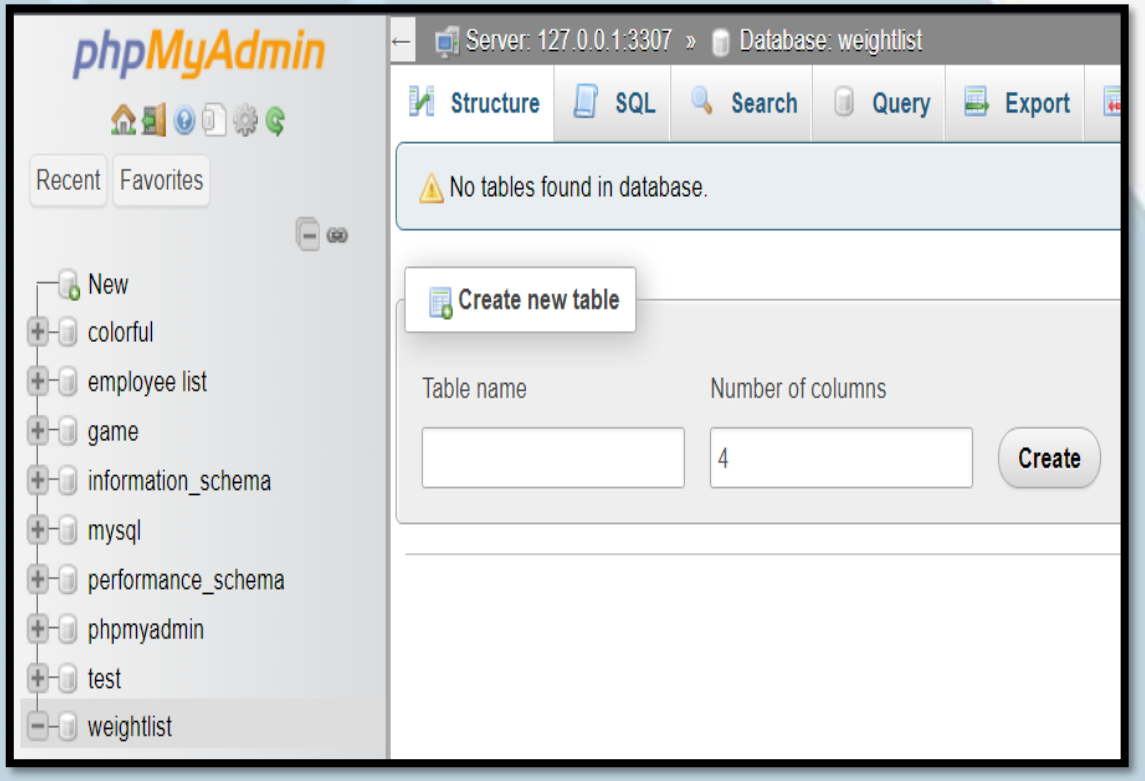
AFTER



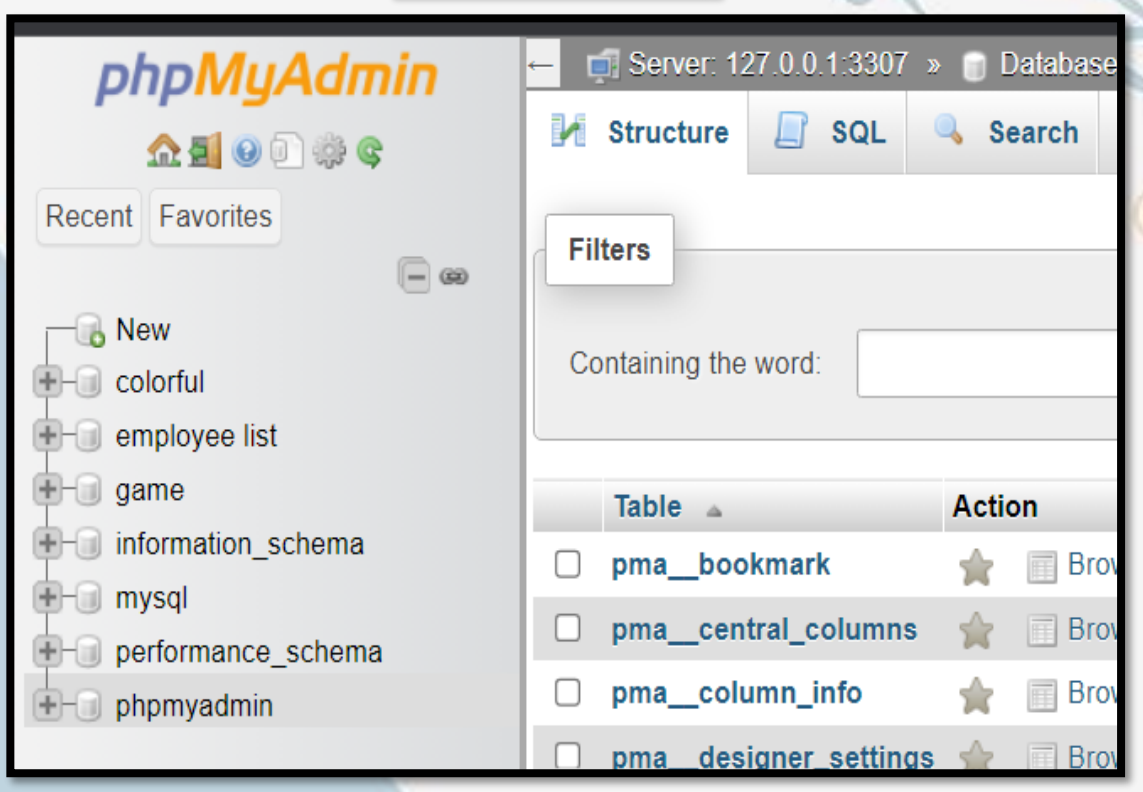
WRITE A QUERY TO DROP TABLE

- DROP DATABASE '***DATABASE NAME***'
- EXAMPLE:
  - DROP DATABASE weightlist

BEFORE

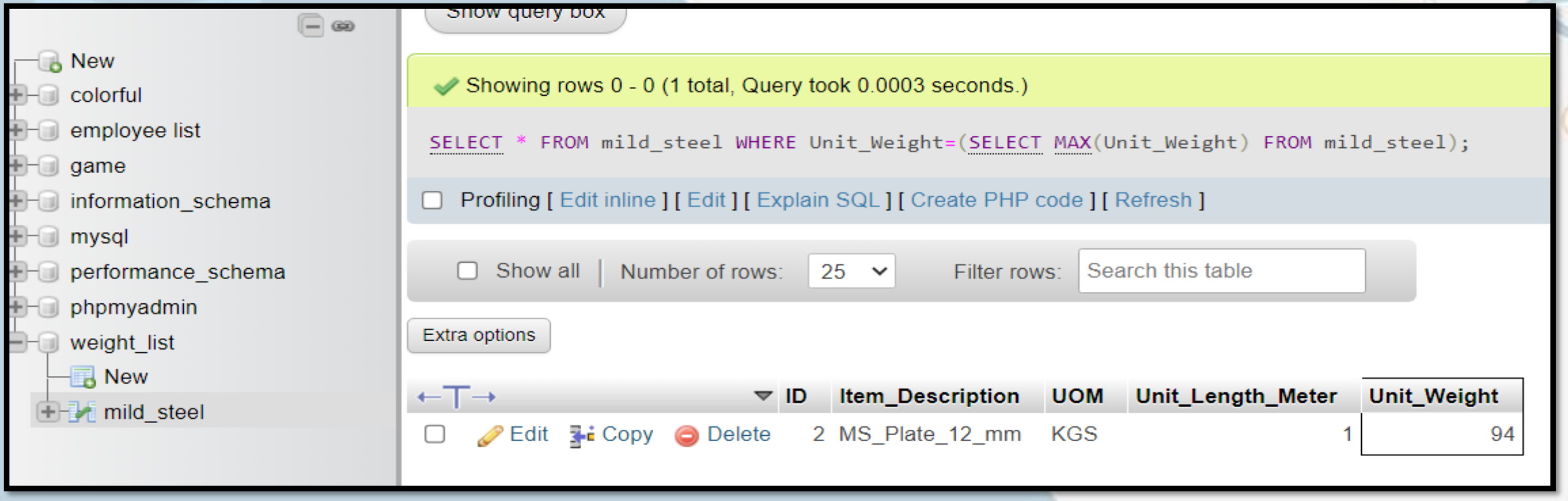


AFTER



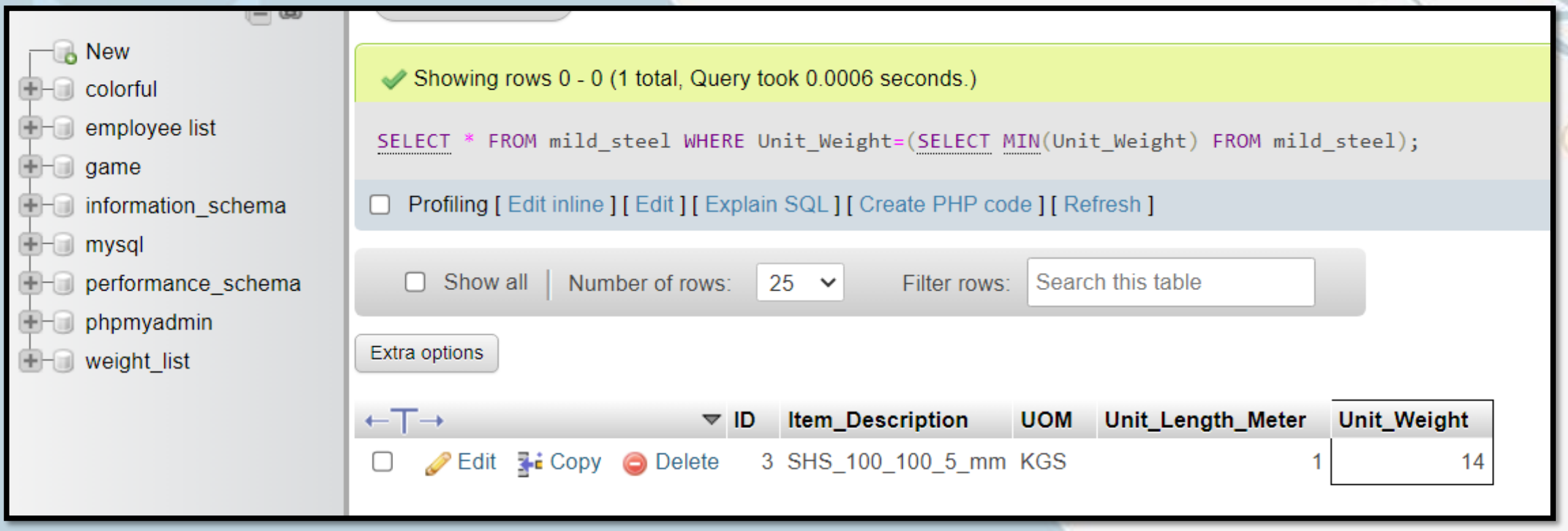
WRITE A QUERY TO FIND MAX VALUE FROM TABLE

- SELECT * FROM '***TABLE NAME***' WHERE '***COLUMN NAME***' = (SELECT MAX('***COLUMN NAME***') FROM '***TABLE NAME***'
- EXAMPLE:
  - SELECT * FROM mild_steel WHERE Unit_Weight=(SELECT MAX(Unit_Weight) FROM mild_steel)



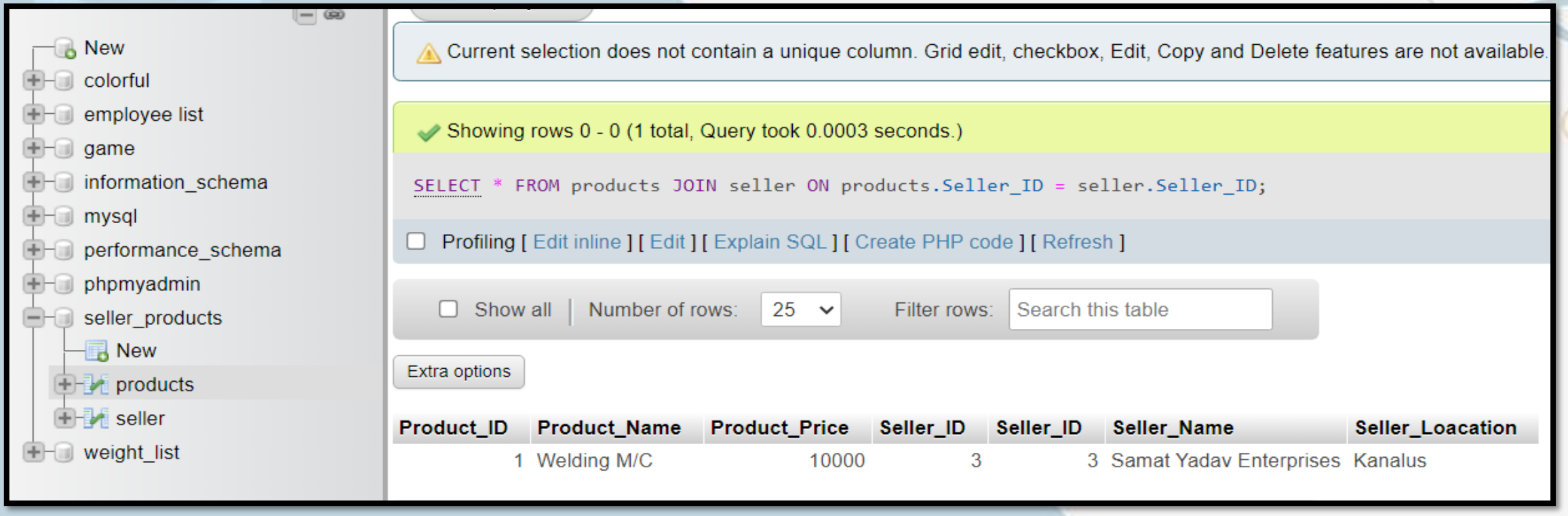
WRITE A QUERY TO FIND MINIMUM VALUE FROM TABLE

- SELECT * FROM '***TABLE NAME***' WHERE '***COLUMN NAME***' = (SELECT MIN('***COLUMN NAME***') FROM '***TABLE NAME***'
- EXAMPLE:
  - SELECT * FROM mild_steel WHERE Unit_Weight=(SELECT MIN(Unit_Weight) FROM mild_steel)



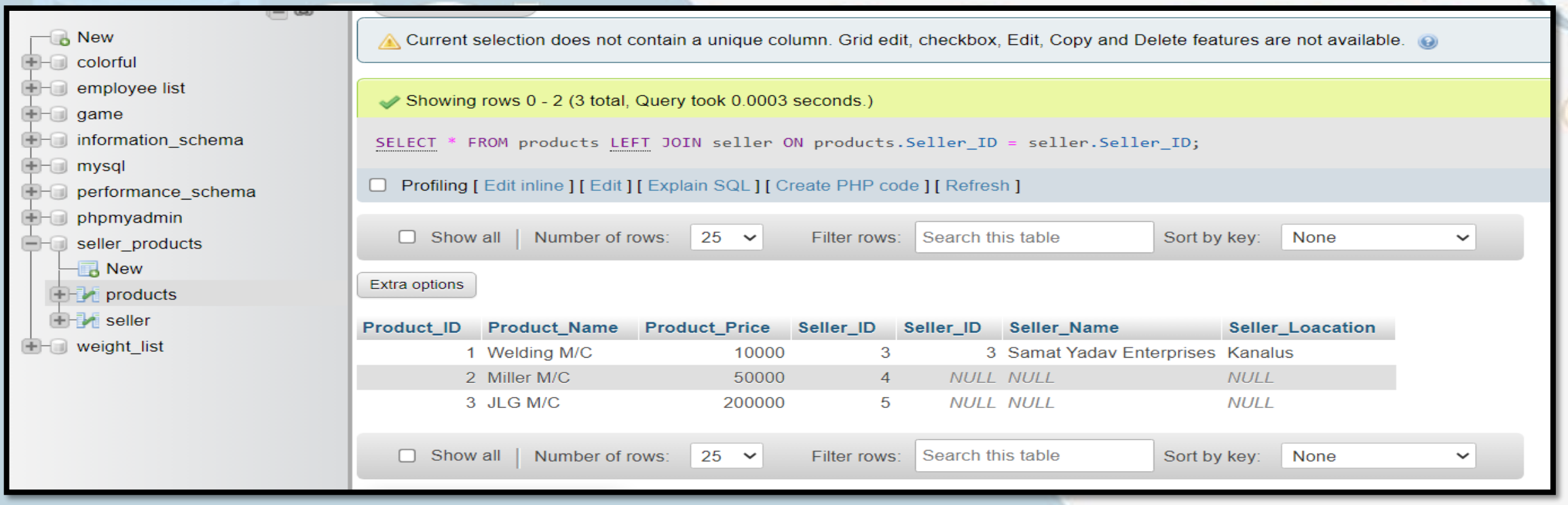
CREATE TWO TABLES NAMED SELLER & PRODUCT APPLY FOREIGN KEY IN PRODUCT TABLE. FETCH DATA FROM BOTH TABLE USING DIFFERENT JOINS.

- **INNER JOIN**
- SELECT * FROM '***TABLE NAME-2***' JOIN '***TABLE NAME-1***' ON '***TABLE NAME-2***'. '***TABLE NAME-2 COLUMN***' = '***TABLE NAME-1 COLUMN***'
- EXAMPLE:
  - SELECT * FROM products JOIN seller ON products.Seller_ID = seller.Seller_ID



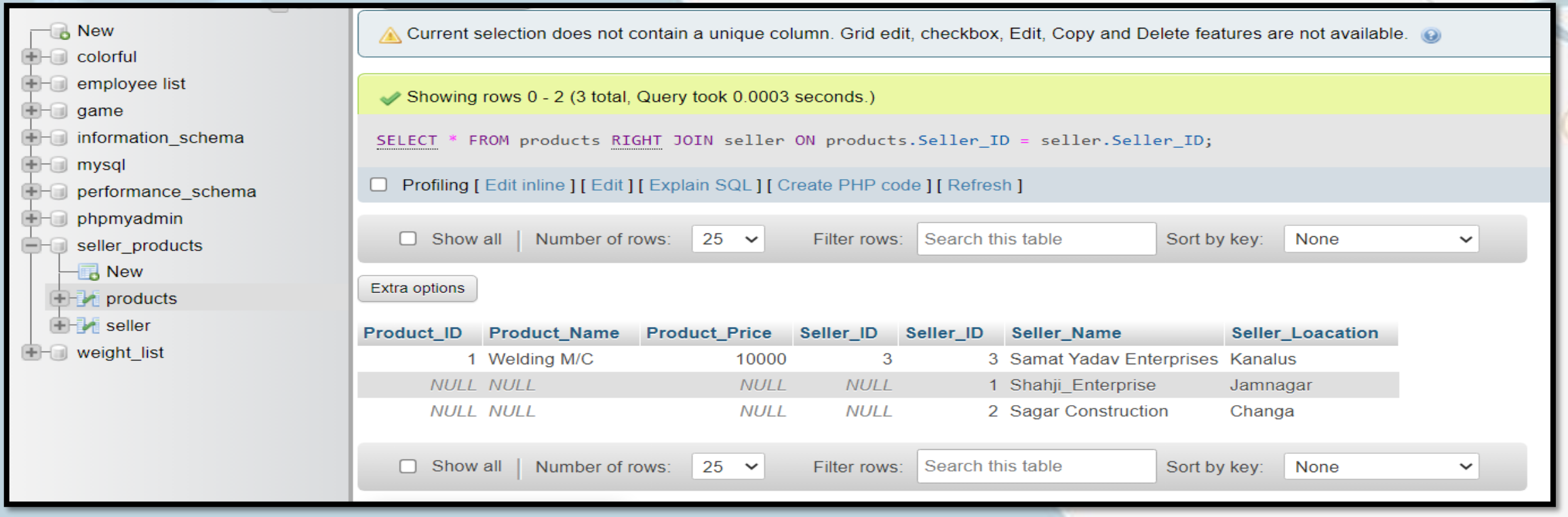
CREATE TWO TABLES NAMED SELLER & PRODUCT APPLY FOREIGN KEY IN PRODUCT TABLE. FETCH DATA FROM BOTH TABLE USING DIFFERENT JOINS.

- **LEFT JOIN**
- SELECT * FROM '***TABLE NAME-2***' LEFT JOIN '***TABLE NAME-1***' ON '***TABLE NAME-2***'. '***TABLE NAME-2 COLUMN***' = '***TABLE NAME-1 COLUMN***'
- EXAMPLE:
  - SELECT * FROM products LEFT JOIN seller ON products.Seller_ID = seller.Seller_ID



CREATE TWO TABLES NAMED SELLER & PRODUCT APPLY FOREIGN KEY IN PRODUCT TABLE. FETCH DATA FROM BOTH TABLE USING DIFFERENT JOINS.

- **RIGHT JOIN**
- SELECT * FROM '***TABLE NAME-2***' RIGHT JOIN '***TABLE NAME-1***' ON '***TABLE NAME-2***'. '***TABLE NAME-2 COLUMN***' = '***TABLE NAME-1 COLUMN***'
- EXAMPLE:
  - SELECT * FROM products RIGHT JOIN seller ON products.Seller_ID = seller.Seller_ID



CREATE TWO TABLES NAMED SELLER & PRODUCT APPLY FOREIGN KEY IN PRODUCT TABLE. FETCH DATA FROM BOTH TABLE USING DIFFERENT JOINS.

- **FULL JOIN**
- SELECT * FROM '***TABLE NAME-2***' RIGHT JOIN '***TABLE NAME-1***' ON '***TABLE NAME-2***'. '***TABLE NAME-2 COLUMN***' = '***TABLE NAME-1 COLUMN***'
- **UNION**
- SELECT * FROM '***TABLE NAME-2***' LEFT JOIN '***TABLE NAME-1***' ON '***TABLE NAME-2***'. '***TABLE NAME-2 COLUMN***' = '***TABLE NAME-1 COLUMN***'
- EXAMPLE:
  - SELECT * FROM products RIGHT JOIN seller ON products.Seller_ID = seller.Seller_ID
  - UNION
  - SELECT * FROM products LEFT JOIN seller ON products.Seller_ID = seller.Seller_ID