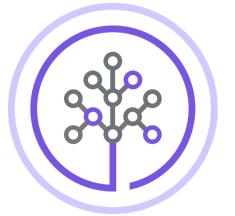


Lab: MySQL User Management, Access Control, and Encryption

Estimated time needed: 30 minutes



**Skills
Network**

Objectives

After completing this lab, you will be able to:

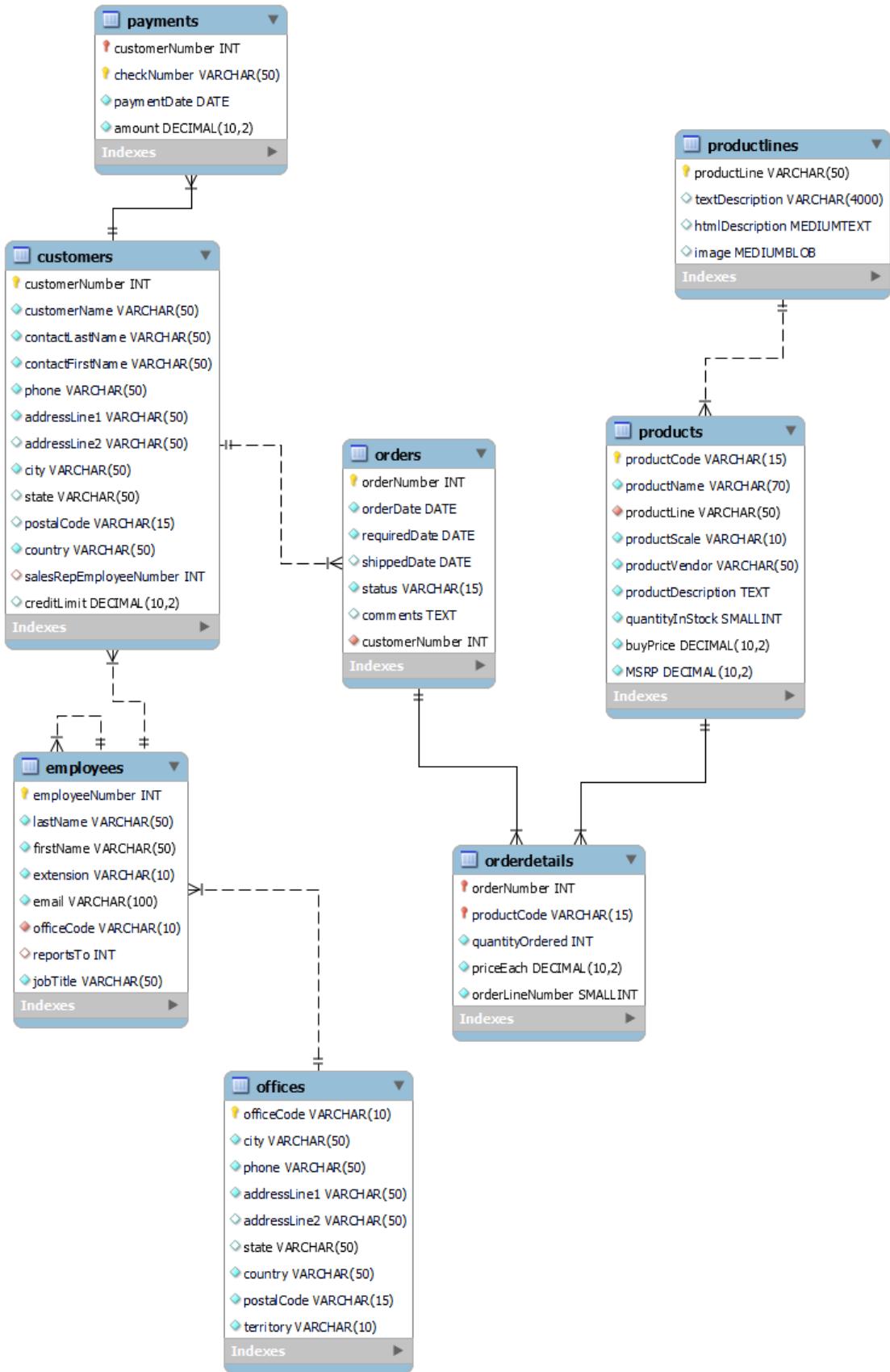
- Manage MySQL user accounts and roles using the phpMyAdmin graphical user interface (GUI) tool
- Control access to MySQL databases and their objects
- Secure your data by adding an extra layer of security using data encryption

Prerequisites

Database

In this lab, you will use a customer orders database, which is a modified version of the source database. It is recommended that you use the given database rather than the database from the original source to follow the lab instructions successfully.

The following ERD diagram shows the schema of the customer orders database.



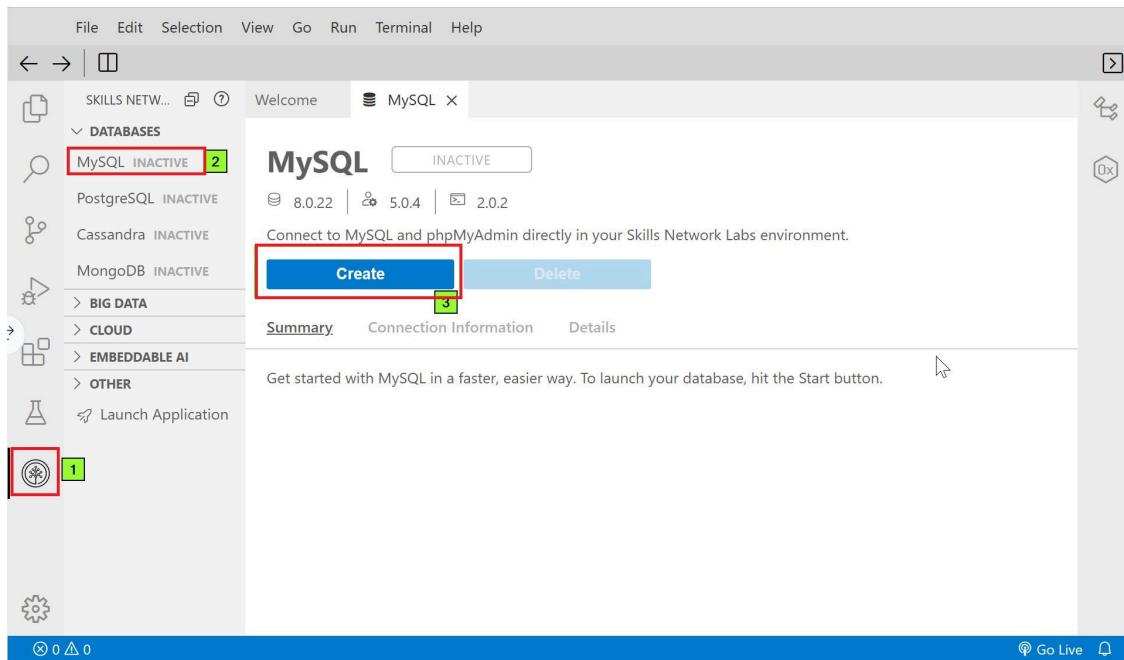
Exercise 1: Manage MySQL user accounts and roles

In this exercise, you will learn how to manage MySQL user accounts and roles using phpMyAdmin. User management is the process of controlling which users are allowed to connect to the MySQL server and what permissions they have on each database. phpMyAdmin does not handle user management; rather, it

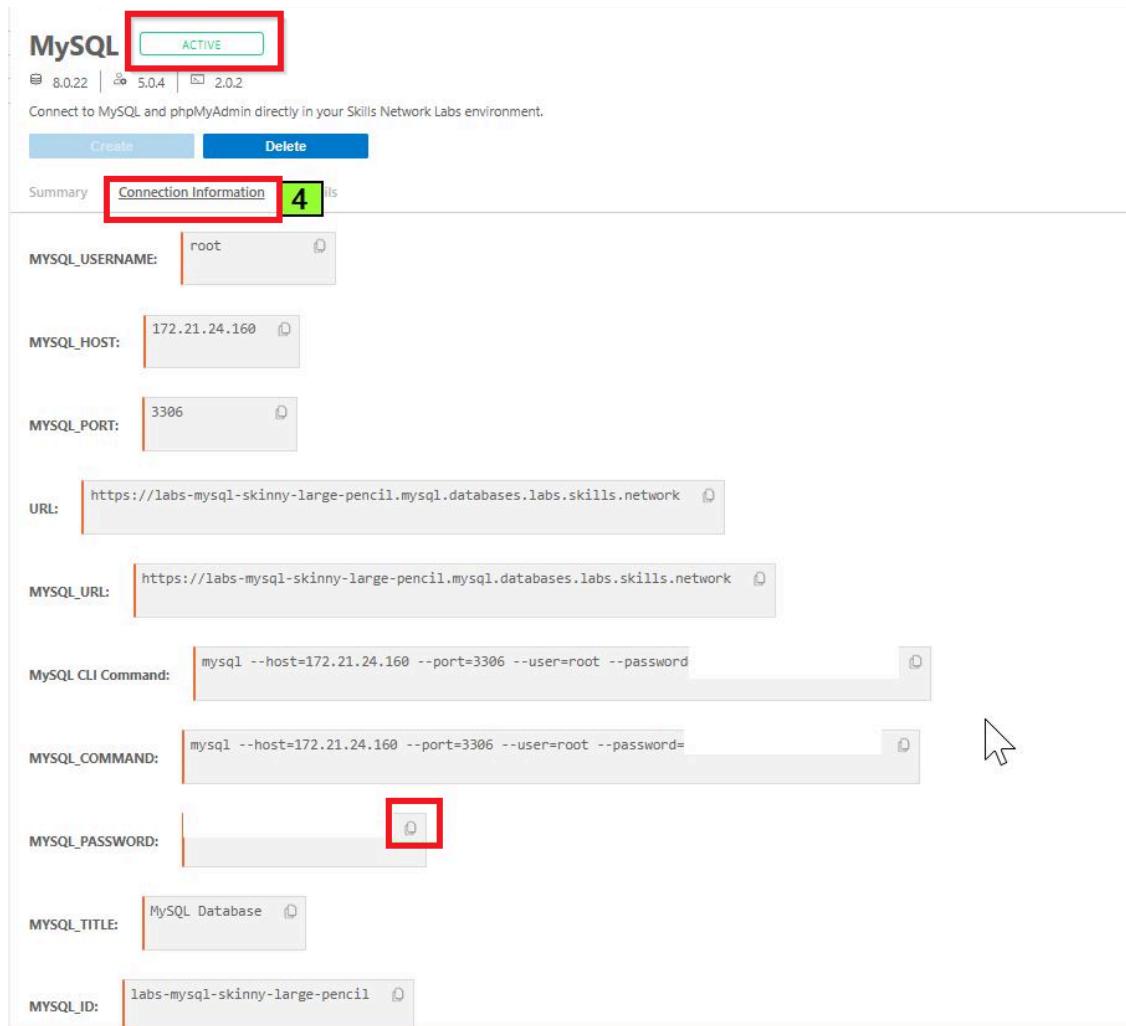
passes the username and password on to MySQL, which then determines whether a user is permitted to perform a particular action. Within phpMyAdmin, administrators have full control over creating users, viewing and editing privileges for existing users, and removing users.

Task 1

1. Go to **Skills Network Toolbox** by clicking the following icon from the side-by-side launched Cloud IDE.
2. From the **Databases** drop-down menu, click **MySQL** to open the MySQL service session tab.
3. Click the **Create** button and wait until the MySQL service session gets launched.



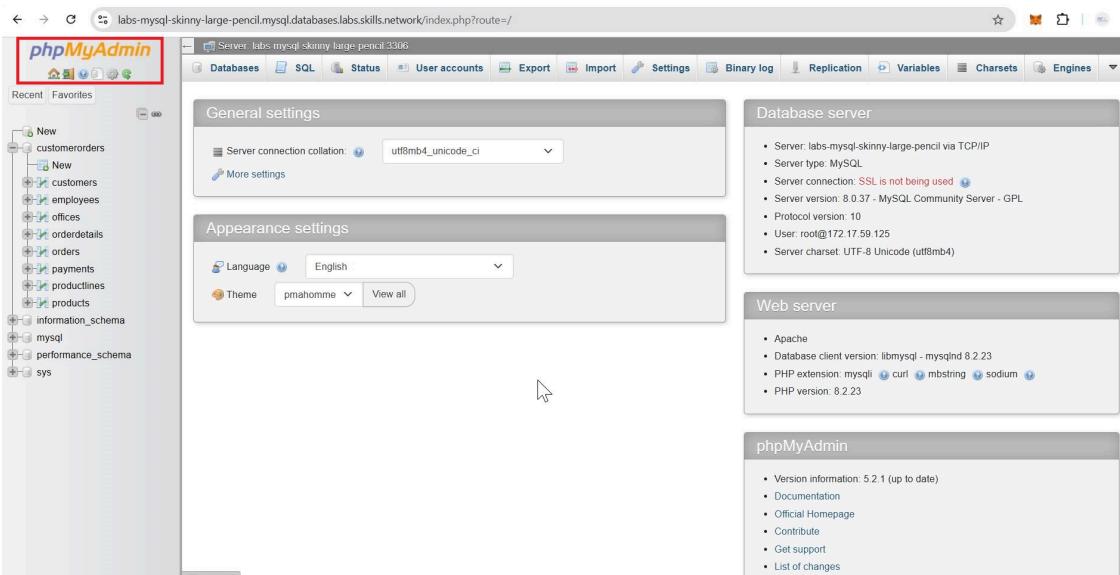
The MySQL server will take a few moments to start. Once it is ready, you will see the green "Active" label at the top of the window.



4. Whenever you are required to enter your MySQL service session password from the MySQL service session tab at any step of the lab, copy the password by clicking on the small copy button on the right of the password block. Paste the password into the terminal using **Ctrl + V** (Mac: **⌘ + V**), and press **Enter** on the keyboard. For security reasons, you will not see the password as it is entered on the terminal.

Task 2

1. Click **phpMyAdmin** button from the mysql service session tab. You will see the phpMyAdmin GUI tool.



2. In the tree view, click **New** to create a new empty database. Then, enter **customerorders** as the name of the database and click **Create**.

3. Go to the **Import** tab. Upload the following SQL script file using the **Choose File** button (first, right-click this [SQL script file](#) to download it to your local computer storage). Then click the **Import** button at the bottom. You will be notified when the import is successfully finished. Click the **Home** icon.

The screenshot shows the phpMyAdmin interface for the 'customerorders' database. The left sidebar lists tables: customers, employees, offices, orderdetails, orders, payments, productlines, products, and information_schema. The main area displays the table structure for each table, including columns, data types, and constraints. A green message bar at the top right indicates that an import operation was successful.

This screenshot shows the same phpMyAdmin interface as above, but with the 'customerorders' table selected and highlighted by a red box in the tree view on the left. The main table structure view remains the same.

4. Now, you will create a user account with the custom role "sales_rep." sales_rep role will have access to limited tables. Go to the **User accounts** tab and click **Add user account**.

This screenshot shows the phpMyAdmin configuration page. The 'User accounts' tab is selected and highlighted with a red box. The page includes sections for General settings, Appearance settings, Database server, Web server, and phpMyAdmin footer information.

User accounts overview

User name	Host name	Password	Global privileges	Grant	Action
mysql.infoschema	localhost	Yes	SELECT	No	<input type="button" value="Edit privileges"/> <input type="button" value="Export"/> <input type="button" value="Unlock"/>
mysql.session	localhost	Yes	SHUTDOWN, SUPER	No	<input type="button" value="Edit privileges"/> <input type="button" value="Export"/> <input type="button" value="Unlock"/>
mysql.sys	localhost	Yes	USAGE	No	<input type="button" value="Edit privileges"/> <input type="button" value="Export"/> <input type="button" value="Unlock"/>
root	%	Yes	ALL PRIVILEGES	Yes	<input type="button" value="Edit privileges"/> <input type="button" value="Export"/> <input type="button" value="Lock"/>
	localhost	Yes	ALL PRIVILEGES	Yes	<input type="button" value="Edit privileges"/> <input type="button" value="Export"/> <input type="button" value="Lock"/>

New With selected:

Add user account

Remove selected user accounts

Revoke all active privileges from the users and delete them afterwards.

Drop the databases that have the same names as the users.

Note: phpMyAdmin gets the users' privileges directly from MySQL's privilege tables. The content of these tables may differ from the privileges the server uses, if they have been changed manually. In this case, you should [reload the privileges](#) before you continue.

Console

5. Fill in the Login Information as shown in the following image (enter your own password). Under Global privileges, click select option SELECT, INSERT, UPDATE under Data. Scroll down and click **Go**.

Add user account

Login Information

User name: sales_rep

Host name: %

Password: Strength: Strong

Re-type:

Authentication plugin: Caching sha2 authentication

Generate password:

Global privileges Check all

Note: MySQL privilege names are expressed in English.

<input checked="" type="checkbox"/> Data	<input type="checkbox"/> Structure	<input type="checkbox"/> Administration	<input type="checkbox"/> Resource limits
<input checked="" type="checkbox"/> SELECT <input checked="" type="checkbox"/> INSERT <input checked="" type="checkbox"/> UPDATE <input type="checkbox"/> DELETE <input type="checkbox"/> FILE	<input type="checkbox"/> CREATE <input type="checkbox"/> ALTER <input type="checkbox"/> INDEX <input type="checkbox"/> DROP <input type="checkbox"/> CREATE TEMPORARY TABLES <input type="checkbox"/> SHOW VIEW <input type="checkbox"/> CREATE ROUTINE <input type="checkbox"/> ALTER ROUTINE <input type="checkbox"/> EXECUTE <input type="checkbox"/> CREATE VIEW <input type="checkbox"/> EVENT <input type="checkbox"/> TRIGGER	<input type="checkbox"/> GRANT <input type="checkbox"/> SUPER <input type="checkbox"/> PROCESS <input type="checkbox"/> RELOAD <input type="checkbox"/> SHUTDOWN <input type="checkbox"/> SHOW DATABASES <input type="checkbox"/> LOCK TABLES <input type="checkbox"/> REFERENCES <input type="checkbox"/> REPLICATION CLIENT <input type="checkbox"/> REPLICATION SLAVE <input type="checkbox"/> CREATE USER	Note: Setting these options to 0 (zero) removes the limit. MAX QUERIES PER HOUR: 0 MAX UPDATES PER HOUR: 0 MAX CONNECTIONS PER HOUR: 0 MAX USER_CONNECTIONS: 0

SSL

6. You have successfully created a user account with appropriate privileges.

The screenshot shows the MySQL Workbench interface with the 'User accounts' tab selected. The title bar reads 'Server: labs-mysql-slimy-tinkling-application:3306'. The main area displays a table titled 'User accounts overview' with columns: User name, Host name, Password, Global privileges, Grant, and Action. The 'sales_rep' user is highlighted with a red border in the table. The 'Action' column for this user contains three buttons: 'Edit privileges', 'Export', and 'Lock'.

User name	Host name	Password	Global privileges	Grant	Action
mysql.infoschema	localhost	Yes	SELECT	No	
mysql.session	localhost	Yes	SHUTDOWN, SUPER	No	
mysql.sys	localhost	Yes	USAGE	No	
root	%	Yes	ALL PRIVILEGES	Yes	
root	localhost	Yes	ALL PRIVILEGES	Yes	
sales_rep	%	Yes	SELECT, INSERT, UPDATE	No	

Below the table, there are buttons for 'Check all', 'With selected:', 'Export', and a 'New' button.

Exercise 2: Control access to MySQL databases and their objects

In this exercise, you will learn how to control access to MySQL databases and their objects.

1. Making an exception to the user definition of the sales_rep role you created earlier, you will modify the privileges of this user. You will remove access to payments tables to sales_rep user and restrict sales_rep from updating all the other columns except the column creditLimit of the table customers of the database customerorders.

Go to **Home > User accounts** tab. Click the **Edit privileges** option of the **sales_rep** user name.

This screenshot is identical to the one above, showing the MySQL Workbench 'User accounts' table with the 'sales_rep' user selected and highlighted with a red border. The table structure and columns are the same, showing various privilege grants and action buttons.

2. Under **Database** sub-tab, select **customerorders** database and click **Go**.

The screenshot shows the MySQL Workbench User Accounts interface. The top navigation bar includes tabs for Databases, SQL, Status, User accounts, Export, Import, and Settings. Below the navigation bar, there are four sub-tabs: Global, Database (which is highlighted with a red box), Change password, and Login Information. The main content area displays the title "Edit privileges: User account 'sales_rep'@'%'". Under this title, a section titled "Database-specific privileges" is shown. It includes a sub-tab menu with Database, Privileges, Grant, Table-specific privileges, and Action. The Action sub-tab is currently selected and shows the value "None". A dropdown menu is open, listing "customerorders", "mysql", and "sys". Below the dropdown, a note says "Add privileges on the following database(s):" followed by an empty input field and a "Go" button.

3. Under **Database-specific privileges**, select SELECT, INSERT, and UPDATE options and click **Go** at the bottom.

The screenshot shows the MySQL Workbench User Accounts interface, similar to the previous one but with a different sub-tab selection. The top navigation bar and sub-tabs are identical. The main content area displays the title "Edit privileges: User account 'sales_rep'@'%' - Database customerorders". Under this title, a section titled "Database-specific privileges" is shown with a "Check all" checkbox. A note below says "Note: MySQL privilege names are expressed in English." The "Data" sub-tab is selected and highlighted with a red box. Under the "Data" sub-tab, the "SELECT", "INSERT", and "UPDATE" checkboxes are checked. Other sub-tabs like "Structure" and "Administration" are also present but not selected. At the bottom, a "Go" button is highlighted with a red box.

4. Switch to **Table** sub-tab. Select the **table** payments from the drop-down menu and click **Go**.

Table-specific privileges

Table	Privileges	Grant	Column-specific privileges	Action
customers	SELECT,INSERT,REFERENCES	No	Yes	Edit privileges Revoke

Add privileges on the following table: Use text field:

Use text field: employees
offices
orderdetails
orders
payments
productlines
products

5. Click None option under all sections of SELECT, INSERT, UPDATE, REFERENCES and click **Go**.

Table-specific privileges

Note: MySQL privilege names are expressed in English.

SELECT	INSERT	UPDATE	REFERENCES	DELETE
customerNumber checkNumber paymentDate amount cardNumber	customerNumber checkNumber paymentDate amount cardNumber	customerNumber checkNumber paymentDate amount cardNumber	customerNumber checkNumber paymentDate amount cardNumber	<input type="checkbox"/> DELETE <input type="checkbox"/> CREATE <input type="checkbox"/> DROP <input type="checkbox"/> GRANT <input type="checkbox"/> INDEX <input type="checkbox"/> ALTER <input type="checkbox"/> CREATE VIEW <input type="checkbox"/> SHOW VIEW <input type="checkbox"/> TRIGGER

Select all Or None Select all Or None Select all Or None Select all Or None

[Go](#)

[Database customerorders: Structure] [Table payments: Browse]

6. As a practice exercise perform steps to remove access to employees, offices tables for **sales_rep** user.

7. Switch to **Table** sub-tab. Select the table **customers** from the drop-down menu and click **Go**.

The screenshot shows the MySQL Workbench interface under the 'Table' tab. The title bar indicates the connection is to 'Server: labs-mysql-slimy-linking-application:3306'. The top navigation bar includes 'Databases', 'SQL', 'Status', 'User accounts', 'Export', 'Import', 'Settings', 'Binary log', and 'Replic'. Below the navigation bar, tabs for 'Database', 'Table' (which is selected and highlighted with a red box), 'Routine', and 'Login Information' are visible. The main area is titled 'Edit privileges: User account 'sales_rep'@'%' - Database customerorders'. A sub-section titled 'Table-specific privileges' is shown, with tabs for 'Table', 'Privileges', 'Grant', 'Column-specific privileges', and 'Action'. The 'Action' tab is selected and shows 'None'. Below this, there is a dropdown menu labeled 'Add privileges on the following table:' with options 'Use text field:' and 'Use text field:'. A list of tables is displayed: customers, employees, offices, orderdetails, orders, payments, productlines, and products. The 'customers' table is highlighted with a red box.

8. Under **Table-specific privileges**, configure all the SQL commands and their custom access to the columns of the table **customers**. Then click **Go**. Such table-specific privilege configuration will restrict **sales_rep** from updating all the other columns except the column **creditLimit** of the table **customers** of the database **customerorders**.

9. As a practice exercise restrict access to update product table "buyPrice" column by sales_rep user.

Exercise 3: Secure data using encryption

In this exercise, you will learn how to secure your data by adding an extra layer of security using data encryption. Certain parts of your database may contain sensitive information that should not be stored in plain text. This is where encryption comes in.

You will implement encryption and decryption of a column in the customerorders database using the official AES (Advanced Encryption Standard) algorithm. AES is a symmetric encryption where the same key is used

to encrypt and decrypt the data. The AES standard permits various key lengths. By default, a key length of 128 bits is used. Key lengths of 196 or 256 bits can be used. The key length is a trade-off between performance and security.

1. Click the **MySQL CLI** button from the mysql service session tab.

2. First, you will need to hash your passphrase (consider your passphrase is **My secret passphrase**) with a specific hash length (consider your hash length is **512**) using a hash function (here you will use the hash function from **SHA-2** family). It is good practice to hash the passphrase you use since storing the passphrase in plaintext is a significant security vulnerability. Use the following command in the terminal to use the SHA2 algorithm to hash your passphrase and assign it to the variable key_str:

```
SET @key_str = SHA2('My secret passphrase', 512);
```

3. Now, let's take a look at the customerorders database. First, you will connect to the database by entering the following command in the CLI:

```
USE customerorders;
```

4. Next, let's take a quick look at the customers table in our database with the following command.

```
SELECT * FROM customers LIMIT 5;
```

For demonstration purposes, suppose that the last column in the table, labelled addressLine1, contains sensitive data; storing such sensitive data in plain text is an enormous security concern, so let's go ahead and encrypt that column.

5. To encrypt the addressLine1 column, you will first convert the data in the column into binary byte strings of length 255 by entering the following command into the CLI.

```
ALTER TABLE customers MODIFY COLUMN addressLine1 varbinary(255);
```

6. Now, to encrypt the addressLine1 column, execute the following command using the AES encryption standard and our hashed passphrase.

```
UPDATE customers SET addressLine1 = AES_ENCRYPT(addressLine1 , @key_str);
```

7. Let's go ahead and see if the column was successfully encrypted by taking another look at the customers table. Run the same command as in step 4.

```
SELECT * FROM customers LIMIT 5;
```

8. The supposedly sensitive data is now encrypted and secured from prying eyes. However, we should still have a way to access the encrypted data when needed. To do this, we use the AES_DECRYPT command, and since AES is symmetric, we use the same key for both encryption and decryption. In our case, recall that the key was a passphrase, which was hashed and stored in the variable key_str. Suppose we need to access the sensitive data in that column. We can do so by entering the following command in the CLI:

```
SELECT cast(AES_DECRYPT(addressLine1 , @key_str) as char(255)) FROM customers;
```

9. As a practice exercise you need to encrypt/decrypt cardNumber column in the payments table.

Summary

Congratulations! In this lab, you learned how to manage MySQL user accounts and roles using the phpMyAdmin graphical user interface (GUI) tool. You also learned how to control access to MySQL

databases and their objects. Finally, you learned how to secure your data, adding an extra layer of security using data encryption.

Author(s)

Parikshit Jain

Other Contributor(s)

Rav Ahuja

[Abhishek Gagneja](#)

© IBM Corporation. All rights reserved.