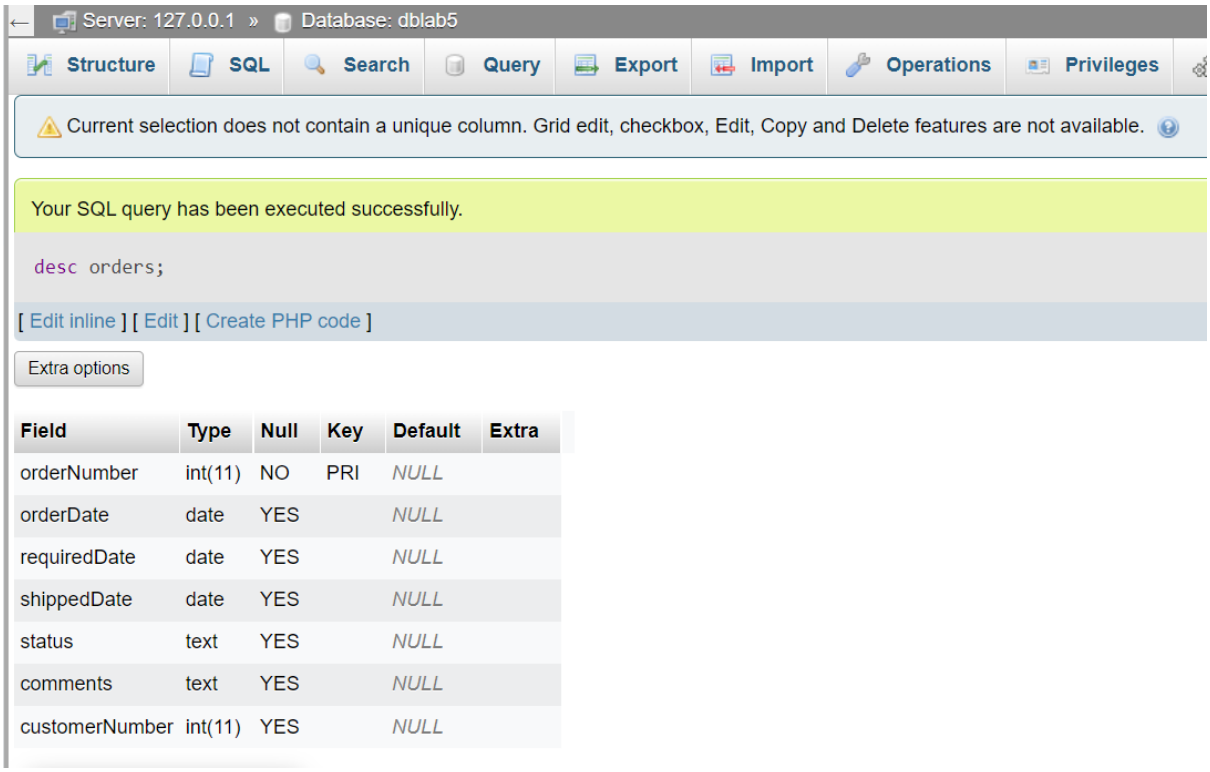
Objective: The objective of this lab is to utilize a stored procedure, transactions, and conditionals to rollback a transaction if a certain value is found, if a certain value is not found, insert into the database table.

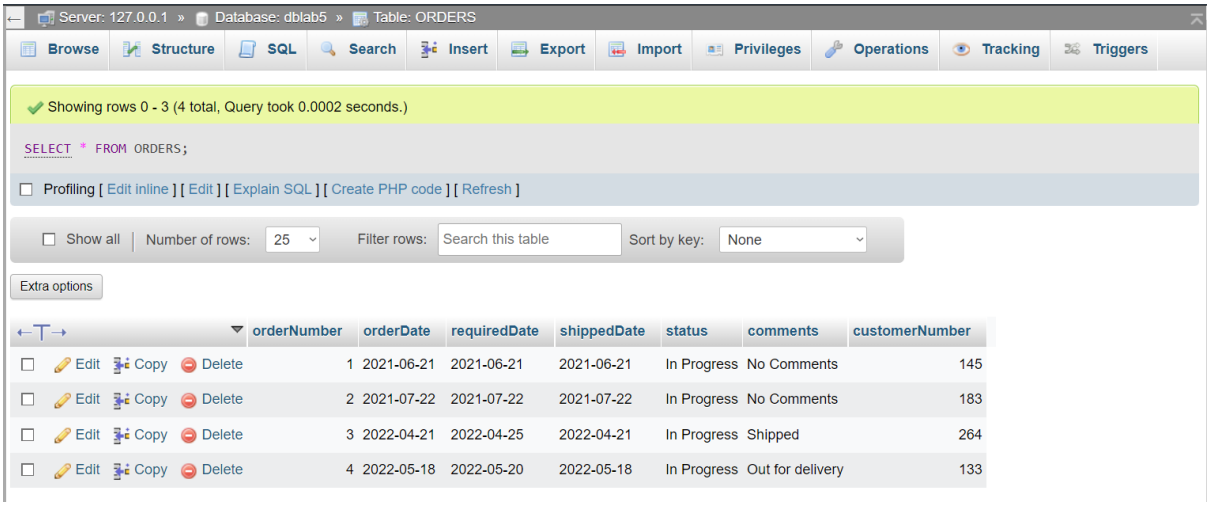
1.

a) Create a database and use the CREATE TABLE query for the orders table from the week 6 lecture.



Field	Type	Null	Key	Default	Extra
orderNumber	int(11)	NO	PRI	*NULL*	
orderDate	date	YES		*NULL*	
requiredDate	date	YES		*NULL*	
shippedDate	date	YES		*NULL*	
status	text	YES		*NULL*	
comments	text	YES		*NULL*	
customerNumber	int(11)	YES		*NULL*	

b) Insert four rows of dummy data (the data can be whatever you like) into the orders table. Screenshot the result of a SELECT * query to prove that this step was completed.



	orderNumber	orderDate	requiredDate	shippedDate	status	comments	customerNumber
Edit Copy Delete	1	2021-06-21	2021-06-21	2021-06-21	In Progress	No Comments	145
Edit Copy Delete	2	2021-07-22	2021-07-22	2021-07-22	In Progress	No Comments	183
Edit Copy Delete	3	2022-04-21	2022-04-25	2022-04-21	In Progress	Shipped	264
Edit Copy Delete	4	2022-05-18	2022-05-20	2022-05-18	In Progress	Out for delivery	133

2. Using a stored procedure and a condition statement (if statement), start a transaction that inserts a new row into the orders table, the transaction should rollback if the orderNumber being inserted already exists. If the orderNumber does not exist, then it should insert into the table and commit. Include the stored procedure in the PDF that you submit for this lab. Prove that your stored procedure works by running it and screenshotting the result.

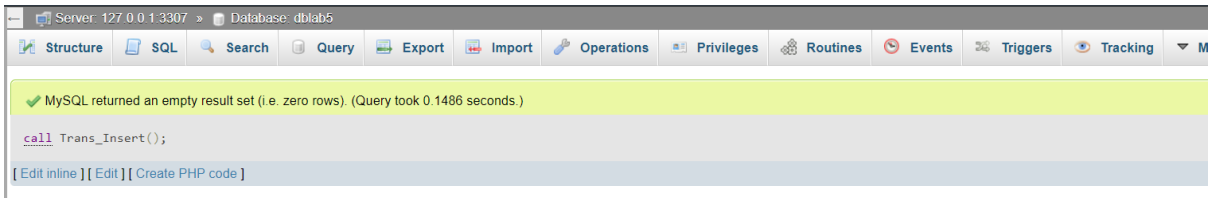
Stored procedure:

```
-- Creating the Procedure and setting the Delimiter to $$
DELIMITER $$
CREATE OR REPLACE PROCEDURE Trans_Insert()
LANGUAGE SQL
DETERMINISTIC
SQL SECURITY DEFINER
COMMENT 'Using Transactions with Stored procedures'

BEGIN
        -- Declaring Variables 'torder' and 'torderNumber' to store the count of order numbers and the latest value of order number
        DECLARE torder DECIMAL(10,2) DEFAULT 0;
        DECLARE torderNumber DECIMAL(10,2) DEFAULT 0;

    -- Starting the Transaction
    START TRANSACTION;

    -- Getting the latest order number to our variable 'torderNumber'
    SELECT
    MAX(orderNumber) + 1
    INTO torderNumber
        FROM
    orders;

    -- Finding the Count of Ordernumber and storing the value in our variable 'torder'
    SELECT COUNT(orderNumber)
    INTO torder
    FROM orders
    WHERE orderNumber = torderNumber;

    -- Checking for condition to evaluate if ordernumber already exists
    IF (torder>0) THEN
        -- Transaction rolls back if condition is true
        ROLLBACK;
    ELSE
        -- Inserting new values

        INSERT INTO orders(orderNumber,orderDate,requiredDate,shippedDate,status,comments,customerNumber)
        VALUES(torderNumber,'2022-07-01','2022-07-03','2022-07-02','No Comment','At a hub near you',369);
```

```
        -- Committing changes to the database
        COMMIT;
    End IF;
END;
$$
-- End of Procedure
```

Calling the Stored procedure to show the result:

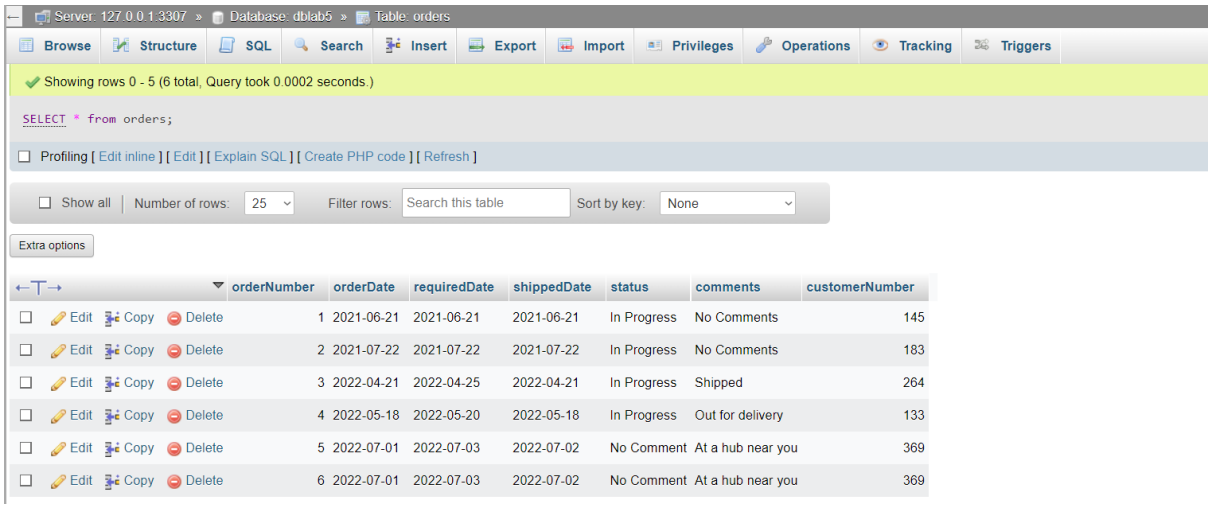
Call Trans_Insert();



As we can see the output below, changes are committed only when an order is assigned a unique order number every time the stored procedure is run, otherwise the transaction will be rolled back.

(I have run the stored procedure twice before taking the screenshot, hence the row is inserted twice.)

*SELECT * from orders;*



Attribution:

*These assignments were completed by **Ravi Chandan Pandi,** and they represent his original work completed for academic purposes during his studies and self-learning purposes.*

Please note that the documents shared here are intended for educational and informational purposes only. Any unauthorized use or reproduction is strictly prohibited. If you have any questions or would like to reach out to Ravi, you can contact him on LinkedIn. [https://www.linkedin.com/in/ravichandan/].
