

# SQL Stored Procedures

---

**CS 4750**  
**Database Systems**

# Stored Procedures

- Allow business logic to be stored in the database and executed from SQL statements
- Pre-defined operations (collection of pre-compiled SQL statements)
- Can pass input parameters to the procedure
- Can return values

## Note:

- If your local *phpMyAdmin* is the same version as *phpMyAdmin* on the CS server, your local DBMS also supports stored procedures.
- Syntax may vary, depending on DBMS you are using

# How Do Stored Procedure Work?

```
CREATE PROCEDURE <procedure_name>  
    [ ( IN | OUT <parameter> <datatype> ) ]  
  
BEGIN  
    <local-declarations>  
    <procedure-body>  
END;
```

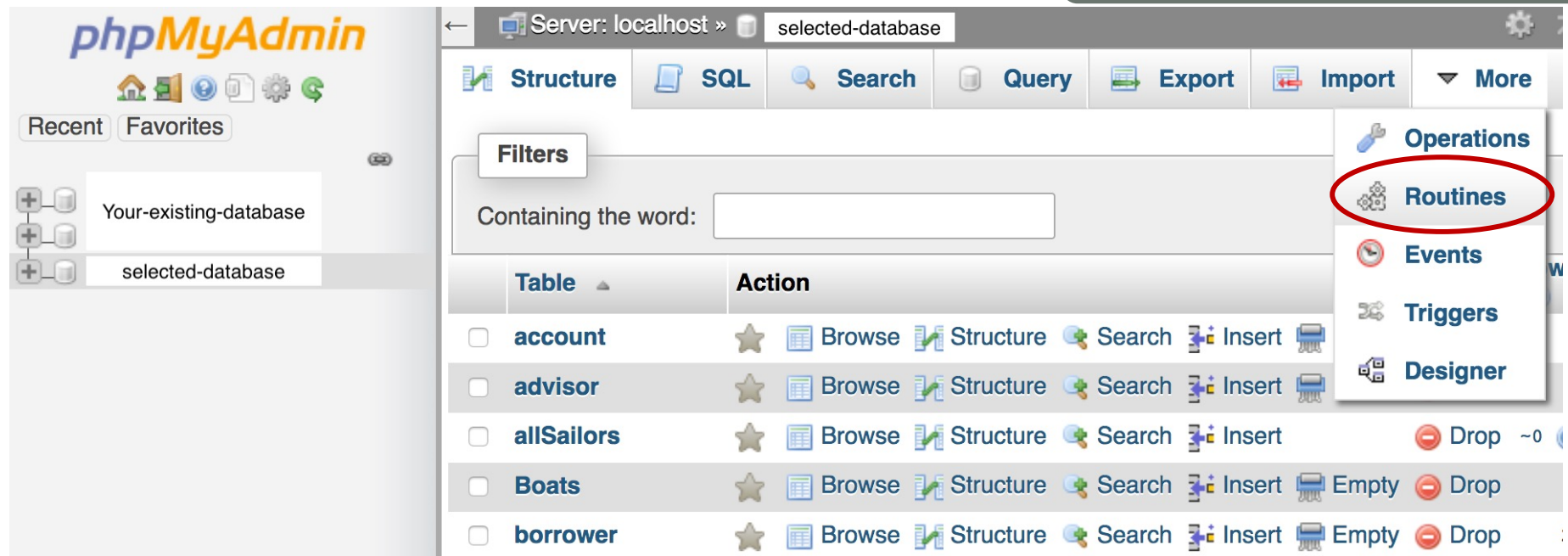
```
CALL <procedure-name> ( <argument-list> )
```

```
DROP PROCEDURE [ IF EXISTS ] <procedure-name>
```

# Create PROCEDURE (CS Server)

- Select the database you would like to work on
- You may create a stored procedure using the Routines feature or write SQL manually

Use Routines feature

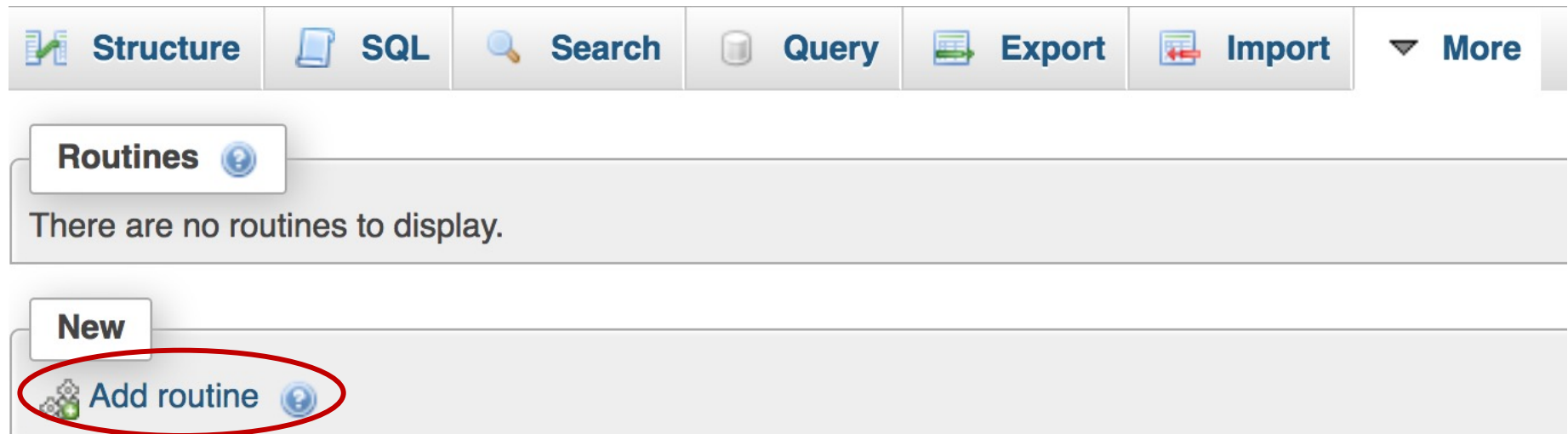


The screenshot shows the phpMyAdmin interface. On the left, there's a sidebar with 'Recent' and 'Favorites' sections. The 'selected-database' is chosen. The main area shows a table list with columns 'Table' and 'Action'. A dropdown menu is open, showing 'Operations', 'Routines' (highlighted with a red circle), 'Events', 'Triggers', and 'Designer'.

Table	Action
<input type="checkbox"/> account	★ Browse Structure Search Insert
<input type="checkbox"/> advisor	★ Browse Structure Search Insert
<input type="checkbox"/> allSailors	★ Browse Structure Search Insert Drop ~0
<input type="checkbox"/> Boats	★ Browse Structure Search Insert Empty Drop
<input type="checkbox"/> borrower	★ Browse Structure Search Insert Empty Drop

# Create PROCEDURE (CS Server)

- Select Add routine



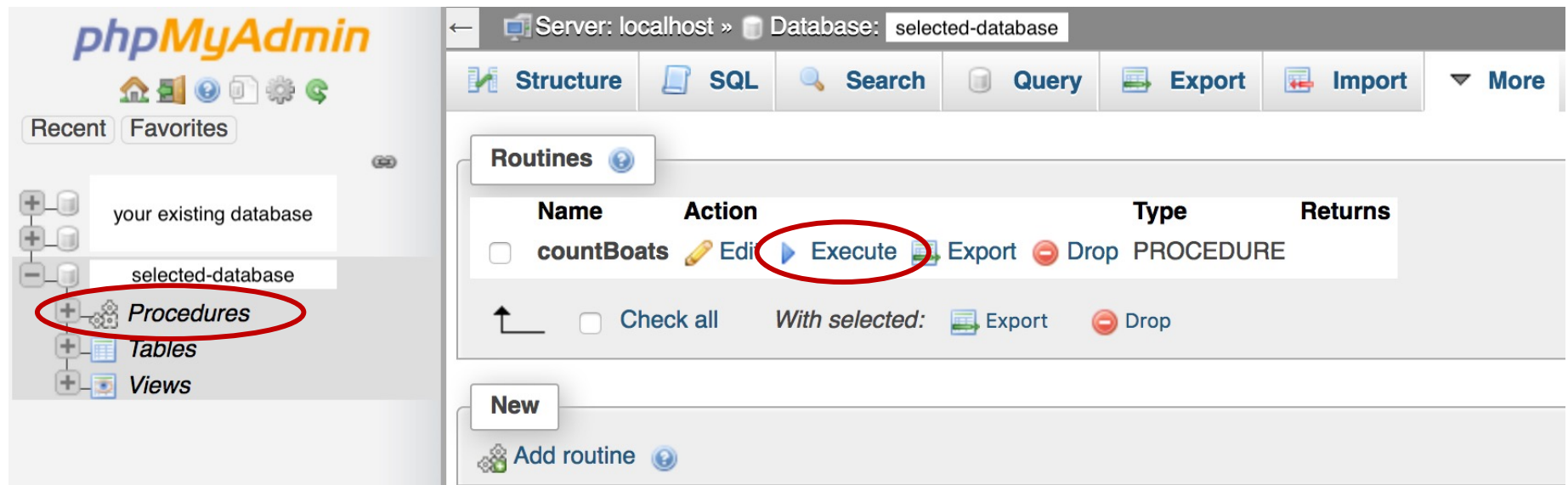
# Create PROCEDURE (CS Server)

- Enter the procedure
- Then, click the Go button

<b>Routine name</b>	<input type="text" value="countBoats"/>				
<b>Type</b>	<input type="text" value="PROCEDURE"/>				
<b>Parameters</b>	<b>Direction</b>	<b>Name</b>	<b>Type</b>	<b>Length/Values</b>	<b>Options</b>
	↓ OUT ↓	param1	INT ↓	<input type="text"/>	<input type="text"/> Drop
<b>Add parameter</b>					
<b>Definition</b>	<pre>1 SELECT COUNT(*) INTO param1 FROM Boats</pre>				
<b>Is deterministic</b>	<input type="checkbox"/>				
<b>Definer</b>	<input type="text"/>				
<b>Security type</b>	<input type="text" value="DEFINER"/>				
<b>SQL data access</b>	<input type="text" value="NO SQL"/>				
<b>Comment</b>	<input type="text"/>				

# Run PROCEDURE (CS Server)

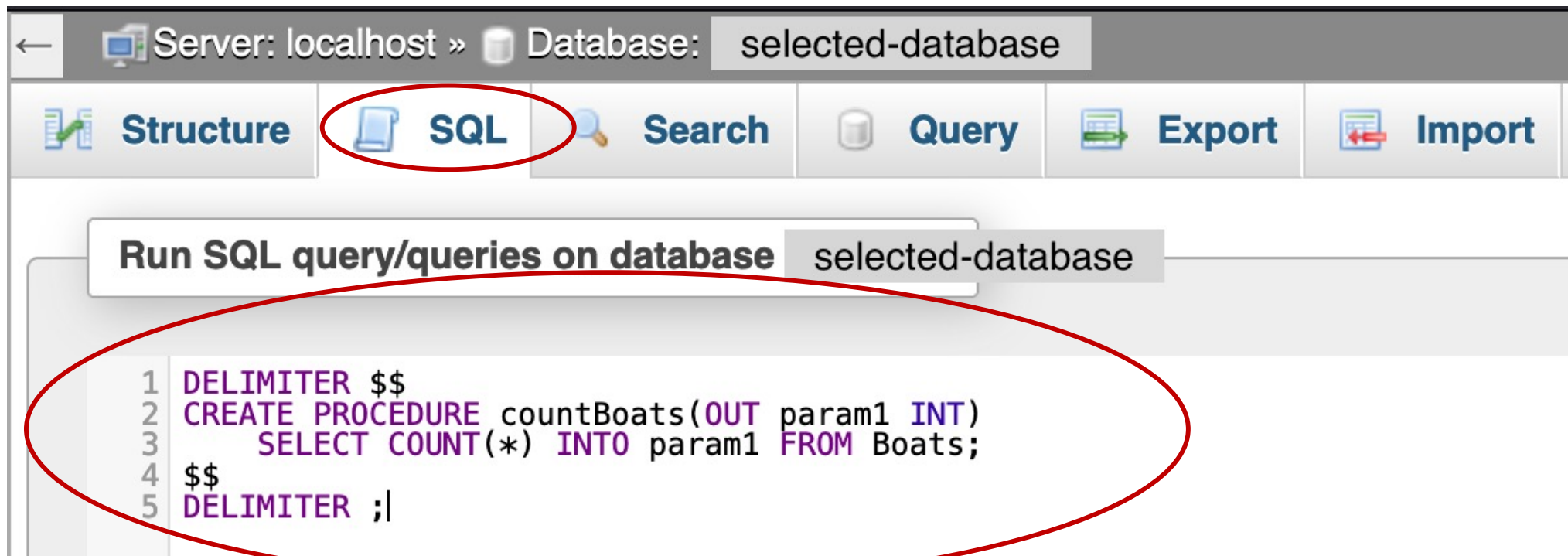
- To run the procedure, under your database, select Procedures
- Locate the procedure you would like to run, select Execute



# Create PROCEDURE (CS Server)

- Select SQL tab
- Enter SQL statement
- Click the Go button

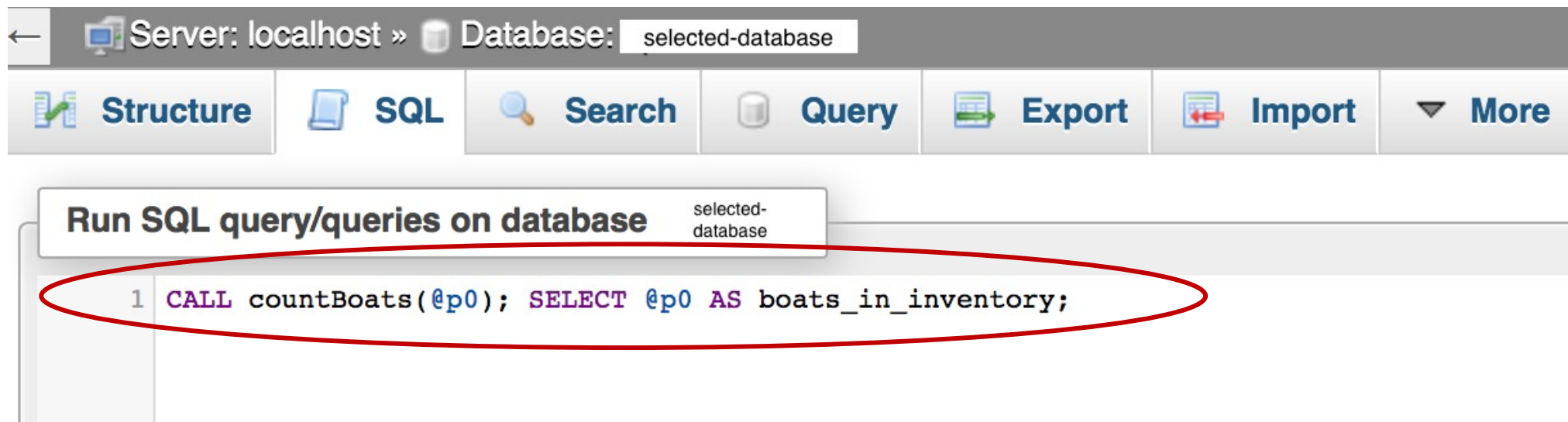
Use SQL statement





# Run PROCEDURE (CS Server)

- Select SQL tab
- Enter SQL statement
- Click the Go button



# Another Example (CS Server)

Create a procedure named raiseSalary

```
DELIMITER $$  
CREATE PROCEDURE raiseSalary (IN oldSalary DECIMAL(8,2),  
                                IN newSalary DECIMAL(8,2))  
UPDATE instructor  
SET salary = newSalary  
WHERE salary = oldSalary;  
$$  
DELIMITER ;
```

Pass two input values and run the procedure

```
SET @p0='62000'; SET @p1='64000';  
CALL raiseSalary(@p0, @p1);
```

# Create PROCEDURE (GCP)

- Start your GCP Cloud SQL instance
- Connect to SQL instance (assume, we are using GCP Cloud Shell)
- Login to MySQL
- Enter SQL statement

```
USE cs4750;      -- (assume) use a database named cs4750
```

```
CREATE PROCEDURE countBoats
```

```
(OUT param1 INT)
```

```
    SELECT COUNT(*) INTO param1 FROM Boats
```

```
$$
```

# Run PROCEDURE (GCP)

```
CALL countBoats(@p0);  
SELECT @p0 AS boats_in_inventory;  
$$
```

# Another Example (GCP)

Create a procedure named raiseSalary

```
CREATE PROCEDURE raiseSalary (IN oldSalary DECIMAL(8,2),  
                                IN newSalary DECIMAL(8,2))  
  
UPDATE instructor  
SET salary = newSalary  
WHERE salary = oldSalary;  
$$
```

Pass two input values and run the procedure

```
SET @p0='62000'; SET @p1='64000';  
CALL raiseSalary(@p0, @p1);  
$$
```