/*
-1. In the Sakila database you create a procedure called insertMyself that retrieves the input parameters:
name VARCHAR(45), cognom VARCHAR(45), date TIMESTAMP and x INT.

This procedure will insert x times the actor with name and cognom into the Sakila.actor table.
As actor_id is INT AUTOINCREMENTAL, we do not have to specify its value. That is to say, we will use an indirect insert.

-0. In the Sakila database create a procedure called myFilms that will retrieve the input parameters: name VARCHAR(45), cognom VARCHAR(45), title VARCHAR (128).

This procedure will insert in the Sakila.film_actor table the information as the actor passed by parameter has participated in the movie in question at today's date.

You have to search first the ids of the actor and the movie. Also check that the actor and the movie exist and if not, show a WARNING.

Also note that if the identical data already exists, it will cause an FK violation error that you have to prevent.

*/

/*

1. In the Sakila DB we will create a showMeSomeActors() procedure that will have two input INT parameters:

valor1 serà id numèric d'actor del primer actor a mostrar-li el nom.
value2 will be the numeric id of the last actor to be shown to display the name
Use a for-loop or a while and a variable @aux that allows you to iterate between Value1 and Value2.
Perform a SELECT on the actor table using WHERE on the actor_id column.

The procedure has to show the name and cognom of the actors the actor_id of which is comprised between value1 and value2.
*/

2.

Go to Northwind.
Create a procedure that shows the shippers data on screen.
*/

CREATE PROCEDURE showShippers() ...

/*

3.

Go to the Northwind DB.

Look at the customers table.

Create a new procedure that checks if a CustomerID passed by parameter exists. If it does exist, return its ContactName.
If it does not exist, to show an error message.

This procedure must have two parameters (1 input and 1 output).
*/

CREATE PROCEDURE checkCustomerID(IN vCustomerID VARCHAR(???), OUT vContactName VARCHAR(??))

4. Within the world database, create a procedure that allows inserting new data into the City table.

5. Create a procedure to export the data from the CountryLanguage table. The name of the fitxer must be passed as a parameter of the user's choice.

8./*

Go to the World DB.
Look at the Cities tab.
Create a procedure that is able to export the cities of a country in a fitxer using a dynamic statement.

The procedure has to receive two input parameters: vCC so that the user specifies from which country he wants to export the cities at the moment of calling the procedure and vFile so that the user specifies the name of the file to be exported where we will also concatenate at the end the date of the current day and the .csv extension.

Before exporting directly, the procedure has to check if there are cities in the last country for each parameter. If there is no data for the requested country, it will display an error message and not export anything.

*/

CREATE PROCEDURE wCityExport (IN vCC VARCHAR(3), IN vFile VARCHAR(15))