**Database Testing**

1. Database Testing overview

* Introduction
* What is Database Testing
* Difference between User Interface Testing and Database Testing
* Types of Database Testing

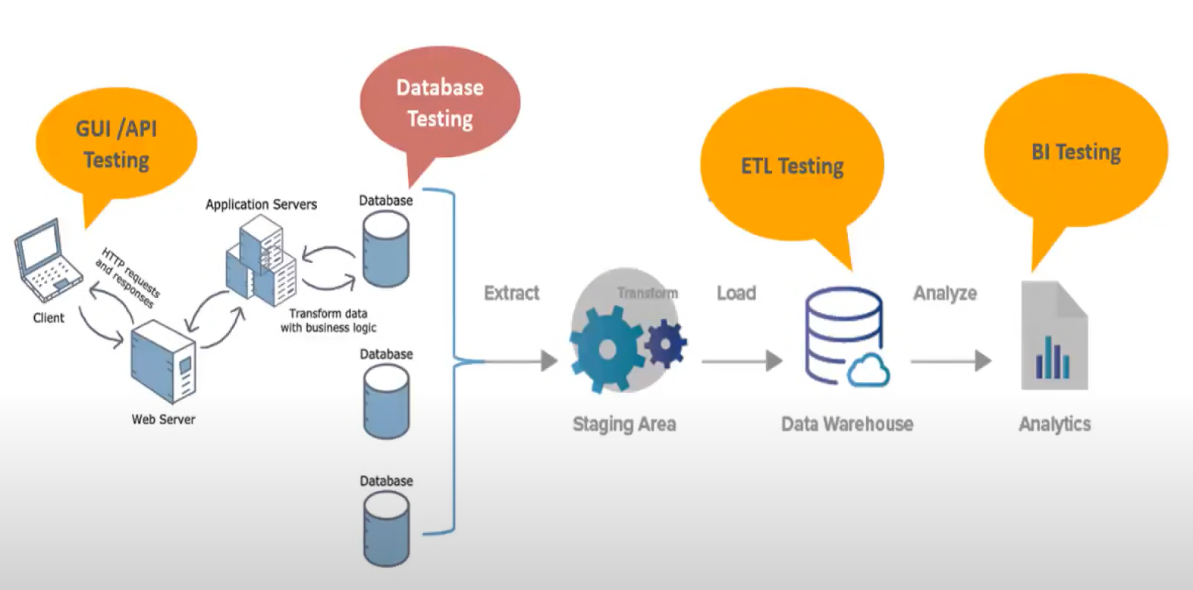
Database: Storage Area where actual application data is stored.

SQL: Structured Query Language (Used to communicate with Database)

So, now we are going to talk about completely on database backend testing.

Database Testing: Mainly we will test Database operations or database object (triggers, joints, indexes, functions, tables, rows, views).

So, database testing sometimes comes under black box testing as well as white box testing.



Explanation: (when a particular testing takes place)

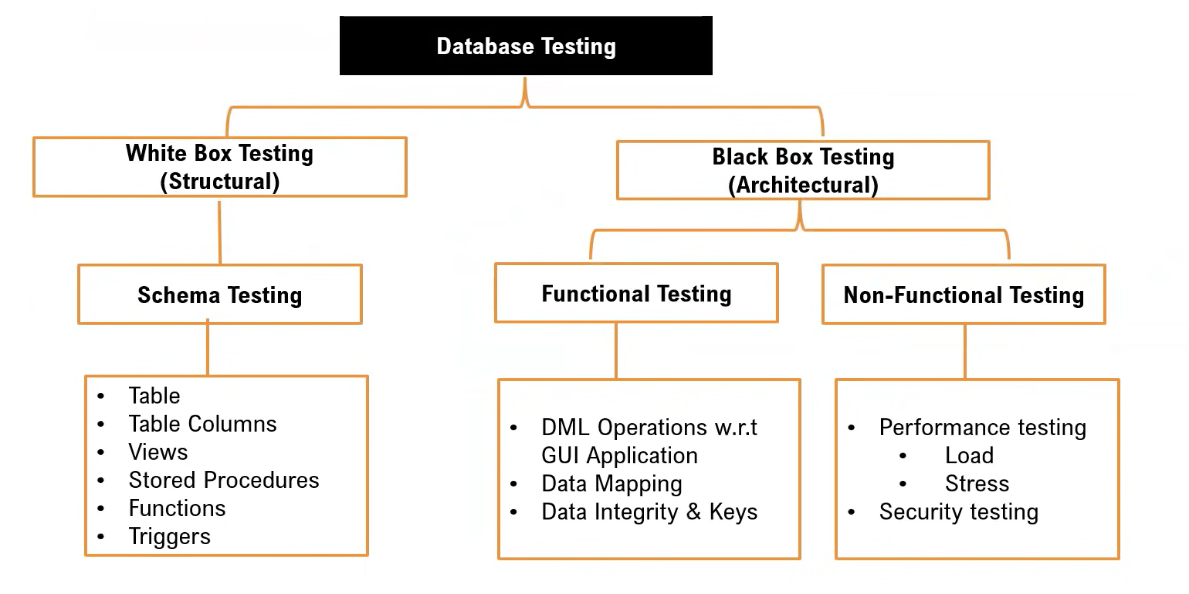
In the above picture, the GUI testing done on the UI of the application using Selenium tool (Web Testing). In this GUI testing, the user sends the request and receives the response from the API through HTTP protocol.

API testing takes place as the client and server involves.

Database contains day to day transactional data. We are going to validate all the database objects are properly working or not. So, to validate those database objects database testing takes place.

After that extract, the data and apply some rules and push the data into a single database (Data Warehouse). To check all the data are perfectly stored in the database we going to do ETL (Extracting Transforming Load) testing.

To analyze those data and to take some business ideas we need BI (Business Intelligence) testing using salesforce or tableau.



Explanation:

Database Testing are mainly divided into two ways – White Box Testing and Black Box Testing.

White Box Testing also known as Structural Testing and Black Box Testing also known as Architectural Testing.

White Box Testing mainly focus on logic. In White Box Testing, we must know how to write queries and programming logic.

Black Box Testing divides into two parts – Functional and Non-Functional Testing.

Functional – Behavior of the Database. It mainly focuses on DML operations (insert, update, select), Data Mapping (whatever data sending from the client application same data is storing in a proper related columns or not), and Data Integrity & Keys (relation between tables, primary key, and foreign key).

1. Environment Setup

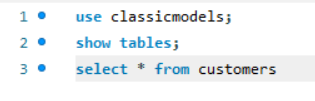
Before going to test database, we need to set environment.

Step – 1: Download and install MySQL Workbench

Step – 2: Download MySQL sample model database (classic models)

Step – 3: After successful setup of workbench, create a database classic model and import the sample database using Data Import.

Step – 4: Perform the operations like select, show tables.



1. How to test Schema of Database Table

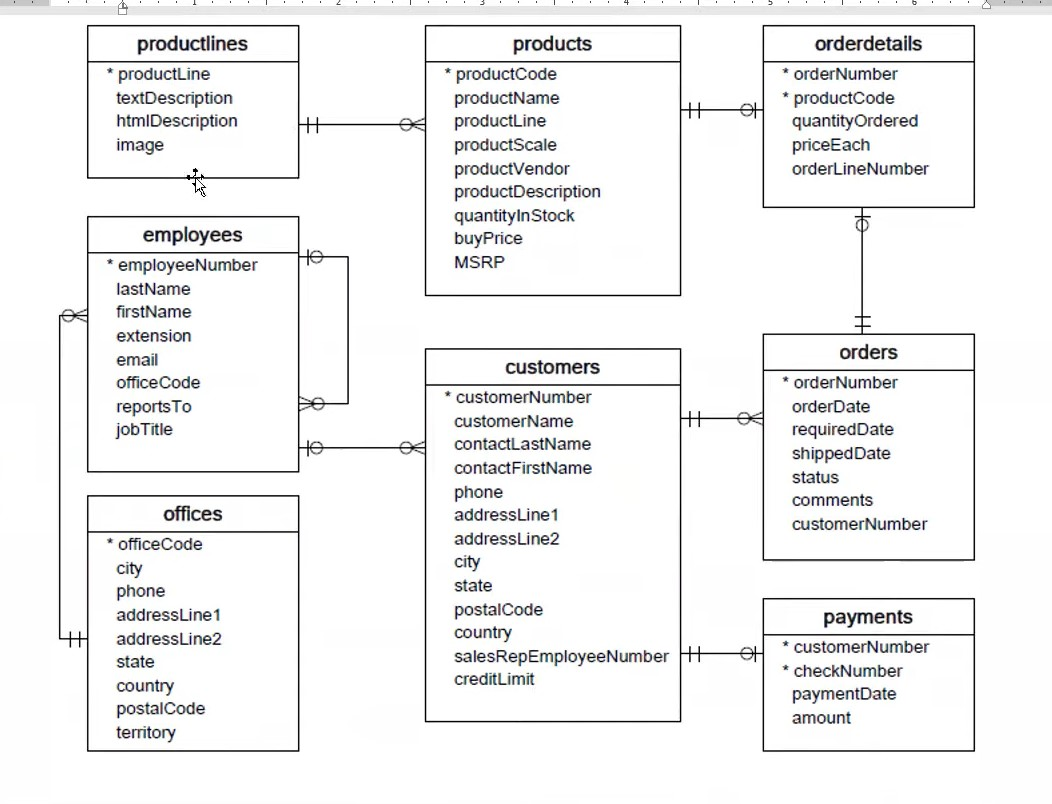
* Table:

Table is a basic database element in which actual data is stored in multiple columns and rows format.

Before going to test we need to know the documentation of database (size and column names of database).

MySQL sample database schema consists of the following tables:

* **Customers** stores customer’s data.
* **Products** stores a list of scale model cars.
* **ProductLines** stores a list of product line categories.
* **Orders** stores sales orders placed by customers.
* **OrderDetails** stores sales order line times for each sales order.
* **Payments** stores payments made by customers based on their accounts.
* **Employees** stores all employee information as well as the organization structure such as who reports to whom.
* **Offices** stores sales office data.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | Columns | Data Type & Size | Null | Keys |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Customers | customerNumber | int(11) | NO | PRI |
|  | customerName | varchar(50) | NO |  |
|  | contactLastName | varchar(50) | NO |  |
|  | contactFirstName | varchar(50) | NO |  |
|  | phone | varchar(50) | NO |  |
|  | addressLine1 | varchar(50) | NO |  |
|  | addressLine2 | varchar(50) | YES |  |
|  | city | varchar(50) | NO |  |
|  | state | varchar(50) | YES |  |
|  | postalCode | varchar(50) | YES |  |
|  | country | varchar(50) | NO |  |
|  | salesRepEmployeeNumber | int(11) | YES | MUL |
|  | creditLimit | decimal(10,2) | YES |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Products | productCode | varchar(15) | NO | PRI |
|  | productName | varchar(70) | NO |  |
|  | productLine | varchar(50) | NO | MUL |
|  | productScale | varchar(10) | NO |  |
|  | productVendor | varchar(50) | NO |  |
|  | productDescription | text | NO |  |
|  | quantityInStock | smallint(6) | NO |  |
|  | buyPrice | decimal(10,2) | NO |  |
|  | MSRP | decimal(10,2) | NO |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ProductLines | productLine | varchar(50) | NO | PRI |
|  | textDescription | varchar(4000) | YES |  |
|  | htmlDescription | mediumtext | YES |  |
|  | image | mediumblob | YES |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Orders | orderNumber | int(11) | NO | PRI |
|  | orderDate | date | NO |  |
|  | requiredDate | date | NO |  |
|  | shippedDate | date | YES |  |
|  | status | varchar(15) | NO |  |
|  | comments | text | YES |  |
|  | customerNumber | int(11) | NO | MUL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OrderDetails | orderNumber | int(11) | NO | PRI |
|  | productCode | varchar(15) | NO | PRI |
|  | quantityOrdered | int(11) | NO |  |
|  | priceEach | decimal(10,2) | NO |  |
|  | OrderLineNumber | smallint(6) | NO |  |

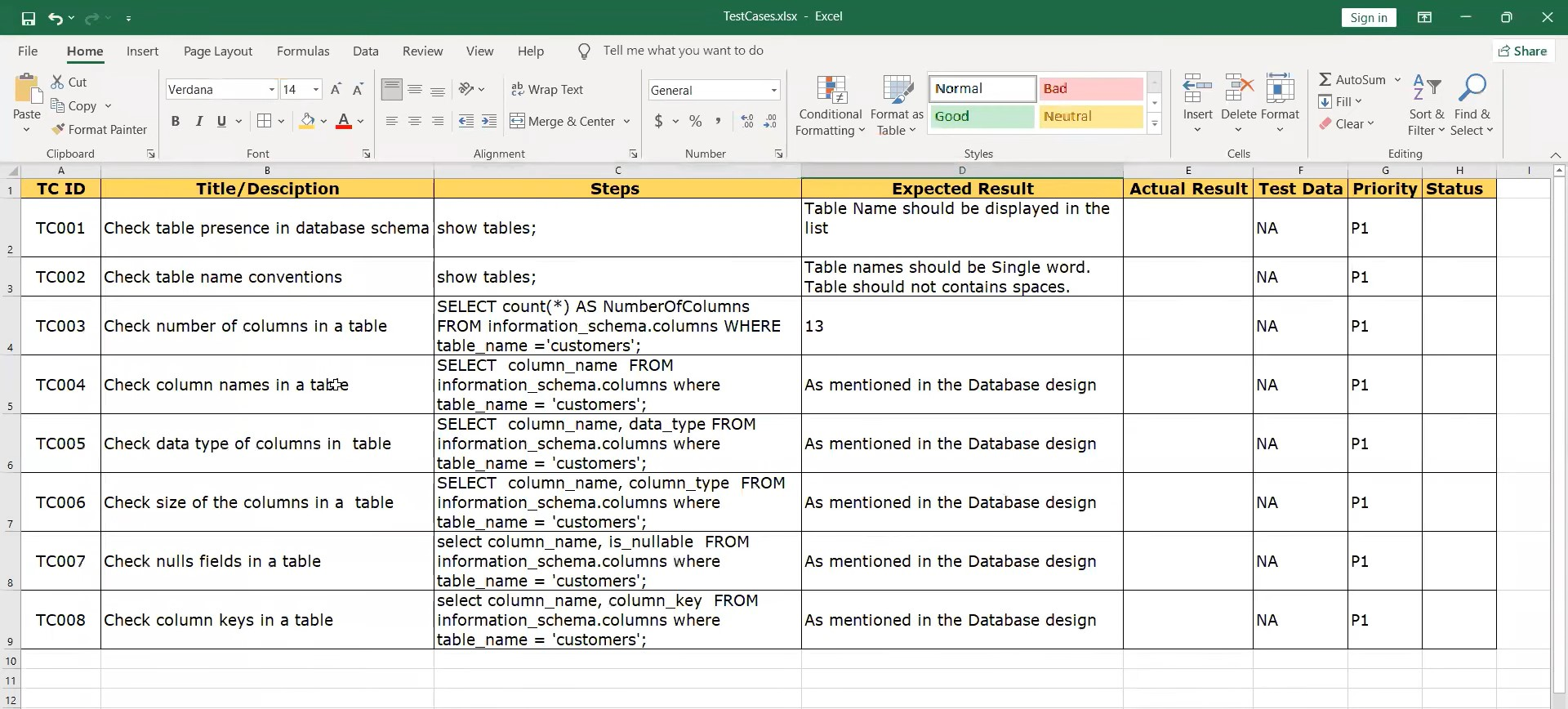
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Payments | customerNumber | int(11) | NO | PRI |
|  | checkNumber | varchar(50) | NO | PRI |
|  | paymentDate | date | NO |  |
|  | amount | decimal(10,2) | NO |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Employees | employeeNumber | int(11) | NO | PRI |
|  | lastName | varchar(50) | NO |  |
|  | firstName | varchar(50) | NO |  |
|  | extension | varchar(10) | NO |  |
|  | email | varchar(100) | NO |  |
|  | officeCode | varchar(10) | NO | MUL |
|  | reportsTo | int(11) | YES | MUL |
|  | jobTitle | varchar(50) | NO |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Offices | officeCode | varchar(10) | NO | PRI |
|  | city | varchar(50) | NO |  |
|  | phone | varchar(50) | NO |  |
|  | addressLine1 | varchar(50) | NO |  |
|  | addressLine2 | varchar(50) | YES |  |
|  | state | varchar(50) | YES |  |
|  | country | varchar(50) | NO |  |
|  | postalCode | varchar(15) | NO |  |
|  | territory | varchar(10) | NO |  |

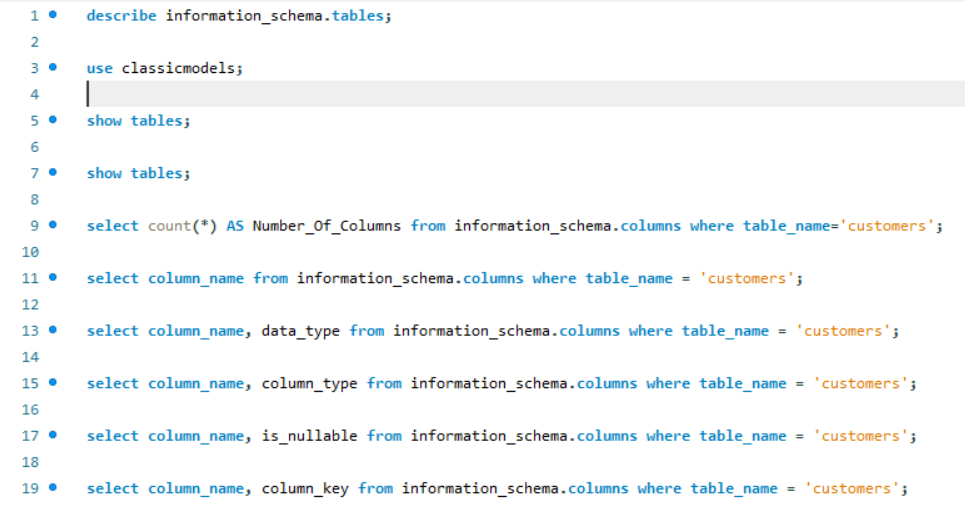
By referring this document, we must write testcases to test database table.

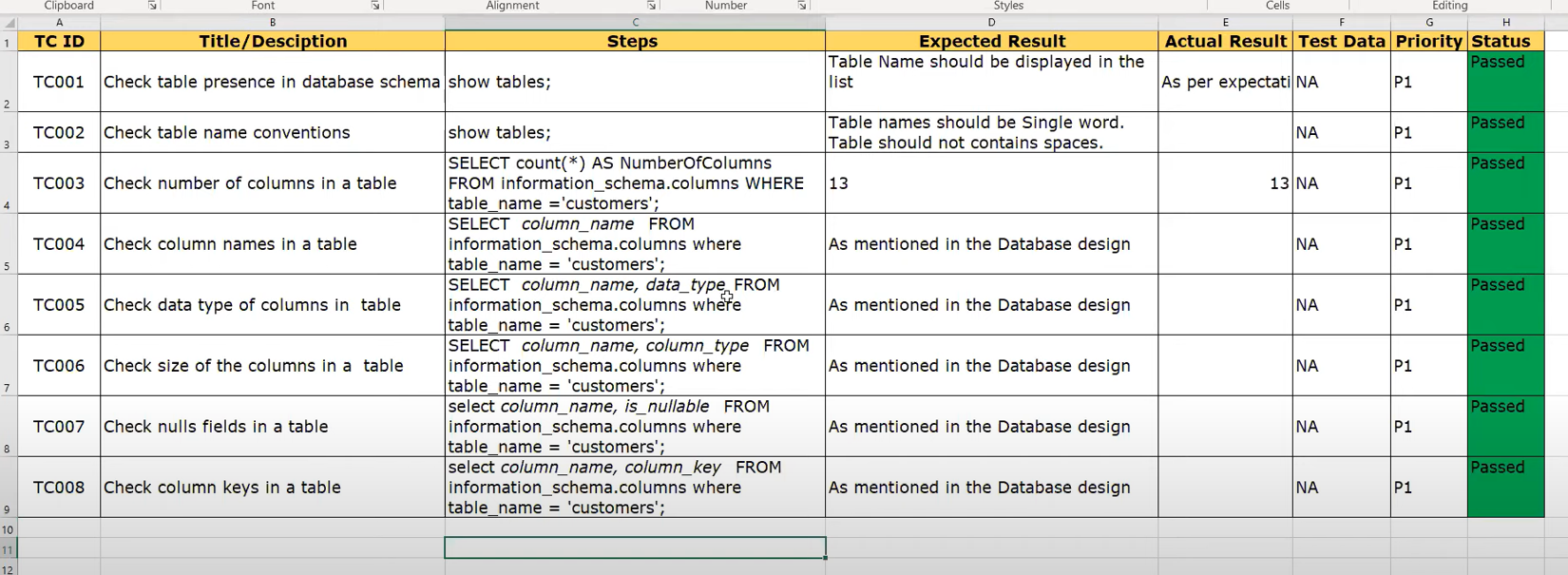
* Test Cases
* Check table presence in database schema.
* Check table name conventions.
* Check number of columns in a table.
* Check column names in a table.
* Check data type of columns in a table.
* Check size of the columns in a table.
* Check nulls fields in a table.
* Check columns keys in a table.



Now, use describe command in the MySQL – describe command will display all the metadata information of the table. And run the test cases prepared in the excel.

Compile these commands to run the testcases in the MySQL work bench.





1. Stored Procedure Testing

* What is Stored Procedure & Advantages of Stored Procedure?

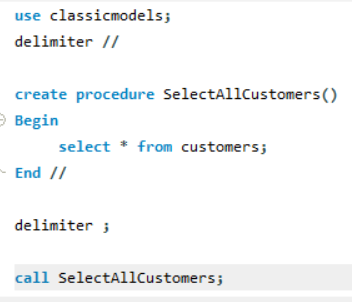
**Stored Procedure:**

* A Stored Procedure is block of SQL statements.
* We can save stored procedure and can be reuse multiple times.
* We can also pass parameters to the stored procedure.

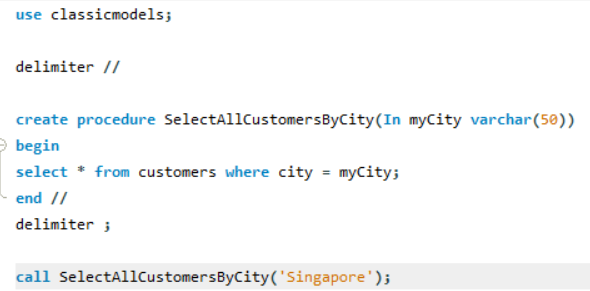
**Advantages:**

* Reduce network traffic.
* Centralize business logic in the database.
* Make database more secure.
* How to create Stored Procedure?
* How to call Stored Procedure?

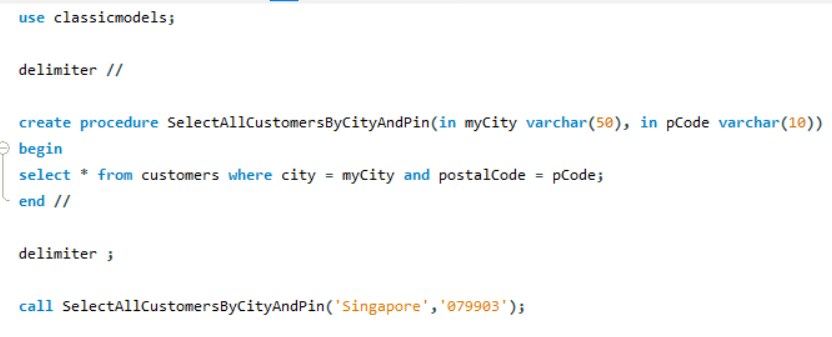
**Basic Stored Procedure:**



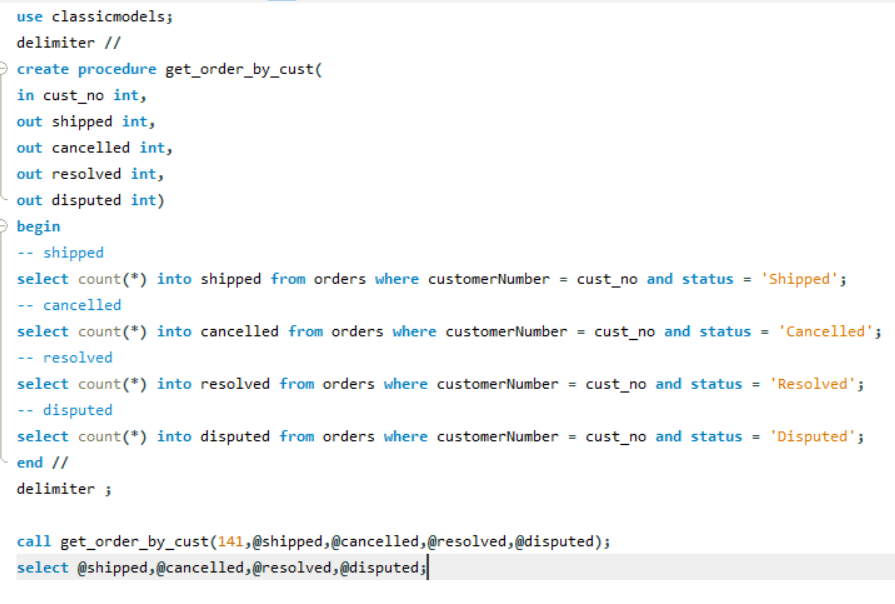
**Single Parameter Stored Procedure:**

****

**Two Parameter Stored Procedure:**

****

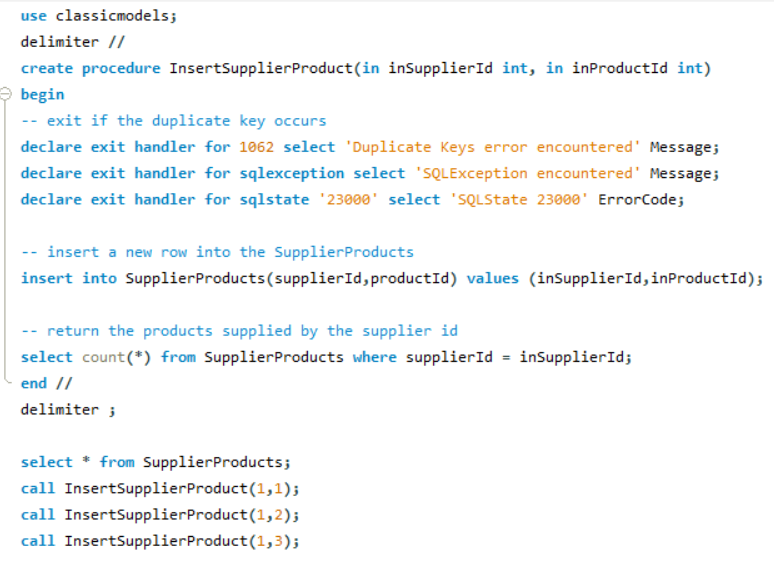
**New Stored Procedure:**

****

**Stored Procedure with Multiple Conditions:**

****

**Stored Procedure with Multiple Errors:**

****

* How to write test cases for Stored Procedures?
* How to test Stored Procedures?