**ASTANA IT UNIVERSITY**

**SOFTWARE QUALITY ASSURANCE**

**TEST CASES FOR DATABASE TESTING**

**Project Name:** Customer Relationship Management Database Testing

**Test Case ID:** DBT-001

**Version:** 1.0

**Date Prepared:** 31.05.2023

**Last Updated By:**  Abu Mariyam

**Last Updated Date:** 01.06.2023

**Test Environment:** PostgreSQL 13.2, Windows 10

**Testing Approach:** Manual testing

**Test Coverage:** Data integrity, Data retrieval, Security

**Prepared By:** Amangeldi Aruzhan, Abu Mariyam, Darmenkyzy Nurbakyt

**Reviewed By:** Tleubayeva A.O.

**Link from Roadmap:** [Link](https://darmenqyzy.atlassian.net/jira/software/c/projects/DBT/boards/9/roadmap?shared=&atlOrigin=eyJpIjoiOWM2NzE3Y2E1YTk3NDNkM2JmNDNmZTk1Njk3NmQyMjgiLCJwIjoiaiJ9)

**Link to the Github repository:** [Link](https://github.com/Maribuoo/CRM_database_testing/tree/main)

**Astana, 2023**

**Test Case ID: TC001**

**Test Case Description:** Verify that a new customer can be successfully created and added to the database.

**Test Objective:** To verify that a new data can be added to the Staff table

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Enter valid user details (First name: James, Last name: Bond, Email: bond@gmail.com, Password: 777bb).
4. Submit the registration form.
5. Execute an SQL INSERT query to add a new staff member with the provided details to the Staff table.
6. Verify that the query execution is successful.
7. Query the Staff table to ensure that the newly created staff member exists.

**Expected Results:** Insert query is succeeded and new staff is added to the table

**Test Data:**

*INSERT INTO customer (customer\_id, address\_id, store\_id, first\_name, last\_name, email)*

*VALUES (1002, 2, 2, 'James', 'Bond', '*[*bond@gmail.com*](mailto:bond@gmail.com)*');*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC002**

**Test Case Description:** Verify that searching for a specific customer by ID and name returns the correct result.

**Test Objective:** To verify that a search by ID and name works correctly.

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of the database.
3. Enter the customer ID and name to search for.
4. Execute an SQL SELECT query to search for a staff member with the provided ID and name.
5. Verify that the query execution is successful.
6. Compare the retrieved result with the expected staff member details.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

*SELECT \**

*FROM customer*

*WHERE customer\_id = 1001*

*AND first\_name = 'James'*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC003**

**Test Case Description:** Validate that the database supports retrieving a list of all customers.

**Test Objective:** To check the correctness of retrieving all customers

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Execute an SQL SELECT query to retrieve all staff members from the Staff table.
4. Verify that the query execution is successful.
5. Compare the retrieved result with the expected list of staff members.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

*SELECT \**

*FROM customer;*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC004**

**Test Case Description:** Verify that validation rules are applied when updating customer data, such as field constraints or data type checks.

**Test Objective:** To check validation and data types

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Select the customer record to update.
4. Modify the customer data with invalid or mismatched values, violating the validation rules or data type checks.
5. Submit the update form.
6. Verify that an error message is displayed indicating the validation or data type error.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

*UPDATE customer*

*SET activebool = true*

*WHERE customer\_id = 1001;*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC005**

**Test Case Description:** Validate the deletion of a customer record and ensure that it is no longer present in the database.

**Test Objective:** To check the deletion process.

**Preconditions:**Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Select the customer record to delete.
4. Confirm the deletion action.
5. Execute an SQL DELETE query to remove the selected customer record from the Staff table.
6. Verify that the query execution is successful.
7. Query the Staff table to ensure that the deleted customer record is no longer present.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

*DELETE FROM customer*

*WHERE customer\_id = 1001;*

*SELECT \**

*FROM customer*

*WHERE customer\_id = 1001;*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC006**

**Test Case Description:** Verify that the database enforces a primary key constraint on the customer ID field, preventing duplicate or null values.

**Test Objective:** To check the duplication or null problem

**Preconditions**Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Enter customer details, including a duplicate or null value for the customer ID field.
4. Submit the registration form.
5. Verify that an error message is displayed indicating the violation of the primary key constraint.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

*INSERT INTO customer (customer\_id, first\_name, last\_name, email)*

*VALUES (NULL, 'Jane', 'Bond', 'janebond@gmail.com');*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC007**

**Test Case Description:** Validate the behavior of the foreign key constraints when a referenced customer is deleted or updated.

**Test Objective:** To verify the correct behavior of foreign key

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Select a customer record to delete or update.
4. Confirm the deletion or update action.
5. Verify that the foreign key constraints defined in the database schema prevent the deletion or update if there are dependent records in the Orders table.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

*UPDATE customer SET customer\_id = 1003 WHERE customer\_id = 1001;*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC008**

**Test Case Description:** Validate that the database enforces appropriate data types for customer fields, preventing invalid or mismatched data.

**Test Objective:** To check prevention of invalid data

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Enter customer details with invalid or mismatched data types for certain fields (e.g., entering a string value in a numeric field).
4. Submit the registration form.
5. Verify that an error message is displayed indicating the data type mismatch or invalid data.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

*INSERT INTO customer (customer\_id, first\_name, last\_name, email)*

*VALUES ('ABC', 'James', 'Bond', 'bond@gmail.com');*

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC009**

**Test Case Description:** Verify that sensitive data, such as passwords or credit card numbers, is stored securely using encryption techniques

**Test Objective:** To verify the secured fields

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Navigate to the customer details page.
4. Retrieve the sensitive data fields (e.g., passwords or credit card numbers) of a customer.
5. Verify that the data is stored using encryption techniques, ensuring that it is not in plaintext.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None

**Test Case ID: TC010**

**Test Case Description:** Validate the encryption and decryption processes to ensure the data is stored and retrieved correctly.

**Test Objective:** To verify the decryption process

**Preconditions:** Open postgresql and database

**Test Steps:**

1. Connect to the database.
2. Navigate to the Query tool of database
3. Retrieve the encrypted data fields of a customer.
4. Decrypt the encrypted data using the appropriate decryption algorithm and key.
5. Verify that the decrypted data matches the original plaintext data before encryption.

**Expected Results:** Select query is succeeded and got a result as a listed rows

**Test Data:**

**Test Environment:** PostgerSQL 14

**Test Priority:** High

**Test Status:** Pass

**Test Notes:** None