**ASTANA IT UNIVERSITY**

**SOFTWARE QUALITY ASSURANCE**

**Test Reporting and Documentation**

**DATABASE TESTING IN THE CONTEXT OF SOFTWARE QUALITY ASSURANCE (SQA)**

**Project:** CRM Database Testing

**Test Phase:** Quality Assurance

**Version:** 1.0

**Date Prepared:** 31.05.2023

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

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

**Last Updated:** 01.06.2023

**Last Updated By:** Aruzhan Amangeldi

**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 Execution Summary:**

* Date: May 31, 2023
* Testing Duration: 2 days
* Test Environment:
  + Database System: PostgreSQL 14
  + Operating System: Windows 10, Macbook Pro 13
  + Hardware: AMD RYZEN5, 16GB RAM

1. **Test Requirements Executed:**

| **Requirement ID** | **Requirement Description** | **Test Cases Executed** | **Test Case Status** |
| --- | --- | --- | --- |
| DBT-REQ-001 | Database should support CRUD operations for customer data | TC001, TC002, TC003, TC004, TC005 | Passed |
| DBT-REQ-002 | Data integrity should be maintained in the database | TC006, TC007, TC008 | Passed |
| DBT-REQ-003 | Database should verify the secure storage of data and ensure protection. | TC007, TC008, TC009 | Passed |

1. **Test Cases Executed:**

| **Test Case ID** | **Description** | **Status** |
| --- | --- | --- |
| TC001 | Verify that a new customer can be successfully created and added to the database. | Pass |
| TC002 | Verify that searching for a specific customer by ID and name returns the correct result. | Pass |
| TC003 | Validate that the database supports retrieving a list of all customers. | Pass |
| TC004 | Verify that validation rules are applied when updating customer data, such as field constraints or data type checks. | Pass |
| TC005 | Validate the deletion of a customer record and ensure that it is no longer present in the database. | Pass |
| TC006 | Verify that the database enforces a primary key constraint on the customer ID field, preventing duplicate or null values. | Pass |
| TC007 | Validate the behavior of the foreign key constraints when a referenced customer is deleted or updated. | Pass |
| TC008 | Validate that the database enforces appropriate data types for customer fields, preventing invalid or mismatched data. | Pass |
| TC009 | Verify that sensitive data, such as passwords or credit card numbers, is stored securely using encryption techniques. | Pass |
| TC010 | Validate the encryption and decryption processes to ensure the data is stored and retrieved correctly. | Pass |

1. **Test Execution Results:**

* Total Test Cases Executed:
* Passed:
* Failed:
* Test Coverage: 100%

1. **Test Findings:**

| **Defect ID** | **Description** | **Severity** | **Steps to Reproduce** | **Actual Results** | **Expected Results** | **Status** | **Assigned To** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| DEF001 | Inconsistent Customer Data | High | Access the Customer table in the CRM database through pgAdmin.  Verify the customer records for accuracy and completeness. | Some customer records have missing or inconsistent data, such as missing email addresses or incorrect contact information. | All customer records should contain complete and accurate data, including valid email addresses and correct contact information. | Open | Mariyam |
| DEF002 | Slow Query Performance | Medium | Execute a query to retrieve customer data, such as selecting all customers.  Measure the time taken for the query to execute. | The query takes an unusually long time (e.g., more than 10 seconds) to retrieve customer data. | The query should execute within a reasonable time frame (e.g., within 2 seconds) to ensure efficient performance. | Open | Nurbakyt |
| DEF003 | Allowance of duplicate email addresses | High | Access the CRM database in the pgAdmin tool.  Attempt to insert a new customer record with a duplicate email address.  Observe the system's response. | The system allows the insertion of a customer record with a duplicate email address, violating the uniqueness constraint on the email field. | The system should reject the insertion of a customer record with a duplicate email address, enforcing the uniqueness constraint on the email field. | Open | Aruzhan |

Recommendations:

1. Optimize the database queries related to the Inventory table to improve performance and reduce execution time. Consider indexing appropriate columns, revising query logic, or implementing caching mechanisms to enhance query performance.

2. Investigate and resolve the issue of missing data in the Customer table. Conduct a thorough analysis to identify the root cause of the missing records and implement necessary measures to ensure data consistency and integrity.

Test Summary:

The test execution for the Customer Relationship Management (CRM) database was mostly successful, achieving a high test coverage of 95%.

Two issues were identified during testing: one critical performance issue and one medium-severity data inconsistency. These issues require further investigation and resolution to ensure the reliability and efficiency of the CRM database.

Recommendations have been provided to address the identified issues and improve the overall quality and performance of the database system.

Test Documentation Organization:

All test documentation is stored in the designated "CRM Test Documentation" folder within the project repository. The documentation is organized as follows:

- Test Plans: Contains the detailed plans and strategies for testing the CRM database.

- Test Requirements: Specifies the functional and non-functional requirements that the CRM database must fulfill.

- Test Cases: Includes the individual test cases with their steps, expected results, and actual results.

- Test Reports: Documents the outcomes of the test execution, including any issues, defects, and test coverage.

- Defect Logs: Logs and tracks all identified defects, their severity, steps to reproduce, and status.

Note: It is recommended to follow your organization's preferred documentation and version control practices to maintain consistency and traceability of the test artifacts.

* Test Completion and Evaluation:
* After executing the test cases for the Customer Relationship Management (CRM) database, it is important to review the test execution results and evaluate the overall testing process. Here are the key steps to be followed:
* 1. Review Test Execution Results: Compare the actual test results against the expected outcomes outlined in the test cases. Verify that all test cases have been executed and determine if any failures or discrepancies were encountered during testing.
* 2. Analyze Issues and Defects: Evaluate the issues and defects identified during testing. Assess their severity and impact on the CRM database. Prioritize them based on their potential risks and the criticality of their impact on the system.
* 3. Assess Testing Effectiveness: Reflect on the overall effectiveness of the testing process. Consider factors such as test coverage, test environment stability, adherence to test plans, and the efficiency of defect tracking and resolution. Identify any areas for improvement or potential bottlenecks in the testing workflow.
* 4. Generate Final Test Report: Summarize the testing activities, findings, and recommendations in a comprehensive test report. Include details about the test execution results, identified defects, test coverage, and the overall assessment of the testing process. Provide actionable recommendations for addressing the identified issues and improving the quality and performance of the CRM database.
* Test Completion Summary:
* - Total Test Cases Executed: 10
* - Passed: 10
* - Failed: 0
* - Defects Identified: 3
* - Test Coverage: 100%
* - Testing Duration: 2 days
* Note: It is essential to adapt the test completion summary to reflect the specific results, duration, and coverage achieved during the testing of the CRM database in the pgAdmin NoSQL environmen**t.**

**Test Reporting and Documentation Sign-off:**

Tester: Darmenkyzy Nurbakyt Date: 01.06.2023

Project Manager: Amangeldi Aruzhan Date: 01.06.2023

Documentation: Abu Mariyam Date: 01.06.2023

1. **Test Documentation Organization:**

Ensure all test documentation is properly organized and easily accessible for future reference. The following structure can be used:

1. Test Plans:

- [Insert test plan files]: This folder should contain all the test plan documents created for the database testing. Each test plan should be appropriately named and versioned for easy identification.

2. Test Requirements:

[Upload test requirements files in Jira and GitHub]: This folder should contain all the test requirements documents specifying the functional and non-functional requirements for the database testing.

3.Test Cases:

- [Upload test case files in Jira and GitHub]: This folder should contain all the test case documents developed for the database testing. Test cases should be organized based on the specific components or modules being tested. Each test case should have a clear and descriptive name for easy reference.

4. Test Reports:

- [Upload test report files in Jira and GitHub]: This folder should contain all the test execution reports generated during the testing process. The reports should be organized based on the testing phases or iterations. Each test report should include relevant details such as the test case status, issues encountered, and any recommendations.

5. Defect Logs:

- [Upload defect log files in Jira and GitHub]: This folder should contain all the defect logs created during the database testing. Each defect log should include information such as the defect description, severity level, steps to reproduce, and the person responsible for resolving the defect.

It is important to maintain a consistent naming convention for the files and folders to ensure clarity and ease of navigation. Additionally, consider utilizing version control or document management systems to track changes, revisions, and updates to the test documentation.