Database Table Editor

Test plan

Version 2

April 22, 2020

Document Control

Approval

The Guidance Team and the customer shall approve this document.

Document Change Control

|  |  |
| --- | --- |
| Initial Release: | April 8, 2020 |
| Current Release: | April 10, 2020 |
| Indicator of Last Page in Document: | \*\* |
| Date of Last Review: |  |
| Date of Next Review: |  |
| Target Date for Next Update: |  |

Distribution List

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members: Dr. Roach

Team Members: Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio

Change Summary

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
| 1 | April 8, 2020 | Andrea Torres | Initial Release of Document |
| 2 | April 19 2020 | Andrea Torres | Table is updated based on reviewers’ comments |
| 3 | April 22 | Andrea Torres | Tests for comparison are added. |

Table of Contents

[Document Control ii](#_Toc38500740)

[Approval ii](#_Toc38500741)

[Document Change Control ii](#_Toc38500742)

[Distribution List ii](#_Toc38500743)

[Change Summary ii](#_Toc38500744)

[1. Introduction 1](#_Toc38500745)

[1.1. Purpose 1](#_Toc38500746)

[1.2. Scope 1](#_Toc38500747)

[1.3. System Overview 1](#_Toc38500748)

[1.4. Suspension and Exit Criteria 1](#_Toc38500749)

[1.5. Document Overview 1](#_Toc38500750)

[1.6. References 1](#_Toc38500751)

[2. Test Items and Features 2](#_Toc38500752)

[3. Testing Approach 3](#_Toc38500753)

[4. Compare Tests 4](#_Toc38500754)

[4.1. Test DBComp1 4](#_Toc38500755)

[4.2. Test DBComp2 3](#_Toc38500762)

[5. Duplicate Tests 3](#_Toc38500763)

[5.1. Test DBDup1 3](#_Toc38500764)

[5.2. Test DBDup2 2](#_Toc38500765)

[5.3. Test DBDup3 2](#_Toc38500766)

[6. Test 04 2](#_Toc38500767)

[6.1. Test DBDup2 2](#_Toc38500768)

[7. User Interface Testing 3](#_Toc38500769)

[8. Test Schedule 4](#_Toc38500770)

[9. Other Sections 5](#_Toc38500771)

[10. Appendix 6](#_Toc38500772)

# Introduction

<< This section gives introductory information regarding the project, the system to be tested, and the testing approach.>>

## Purpose

<< Identify the project and stipulate the test plan purpose by indicating what the document contains (e.g., organizational responsibilities, test approach, test schedule. There are generally four different types of test plans: project test plan that describes the overall strategy for testing; the system test plan that describes the system from the customer’s point of view; integration test plan that describes integration of units and subsystems; unit test plan that describes modules or classes. This section needs to identify which of these this document is.>>

## Scope

<<Specify the project software releases/versions encompassed by the plan. >>

## System Overview

<<Describe the system to be exercised by the testing approach specified in the plan. This overview serves to identify aspects of the system operation that will be the focus of the plan’s testing approach. This should align with the systems overview of other documents in the project.>>

## Suspension and Exit Criteria

<< “suspension criteria” describes when we suspend testing, to be resumed at a later time. For example, if 40% of the test cases fail, or if any of the critical test cases fail. If there are no suspension criteria, indicate that all tests cases will be executed. “Exit criteria” indicates when testing stops. This could be based on run rate (number of test cases run divided by number of test cases specified) or pass rate (number of test cases passed divided by number of test cases run, or test cases passed divided by number of test cases specified). Nominally, we expect to run all of the specified tests. We want the pass rate to be high. We might specify that all critical tests must pass, and 90% of the non-critical must pass. In general, we want this to be high. >>

## Document Overview

<<Describe the remainder of the document.>>

## References

<<List all the references applicable to the test plan. Generally, this includes project standards, SRS, SDD, and a product assurance plan.>>

# Test Items and Features

<< This section describes the test items (e.g., components, classes, functions or methods) and the features to be tested. It may also list features not to be tested. A class diagram is useful. A table of features is useful. >>

# Testing Approach

These tests will determine the accuracy of the compare and duplicate row features. These testes were created to determine if the features work as expected. These are not stress tests, they are barley acceptance tests in that if these tests pass the system displays expected performance, and if the test fails then these features are not working and we are able to conclude that the features are not implemented in a manner that is acceptable for general release.

Table 1: Test Plan

|  |  |  |
| --- | --- | --- |
| **TEST SUITE DBCompDup** | | |
| **Description of Test Suite** | This test suite will befocused on the ability to compare files to other versions and check for duplicate rows | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| DBComp1 | **To determine if the compare option works as expected when a row is added.** | **Medium** |
| DBComp2 | **To determine if the compare option works as expected when a row is deleted.** | **Medium** |
| DBComp3 | **To determine if the compare option works as expected when a row is edited.** | **Medium-High** |
| DBDup1 | **To determine if the duplicate option works as expected when there are no duplicates.** | **Medium** |
| DBDup2 | **To determine if the duplicate option works as expected when a new unique row is added.** | **Medium** |
| DBDup3 | **To determine if the duplicate option works as expected when a new duplicate row is added.** | **Medium** |

# Compare Tests

The purpose of this section is to:

* Determine if the Compare option compares two versions of a table.

for each test case, complete the following:

* open dbEdit software
* Select file
* Select Open
* Select Test\_DB

## Test DBComp1

**Objective: Compare two tables that share only one row change.**

**Notes:** This test is to help determine if the software is able to detect change in one row.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DBComp1 | | | | | Current Status: Fail | |
| Test title: Data Base Comparison one. | | | | | | |
| Testing approach: For this test we will be using the TEST\_DB\_\_CONSTRAINT\_TABLE, this will be done manually. | | | | | | |
| STEP  1  2  3  4  5  6  7  8  9  10  11  12  13 | OPERATOR ACTION  Select CONSTRAINT\_TABLE  Select “Edit” from menu bar.  Select “Insert Row” from “Edit” Dropdown menu  Type in 55 into the first column of the table in the last row.  Type in 1 into the second column of the table in the last row.  Select File drop down menu.  Select “Save as” from “File” dropdown menu.  Type “TEST\_DB\_CONSTRAINTCHANGE\_TABLE” and select save.  Input “4/20/2020” into the date box. Input “Tester” into the author box. Input “Added a row” into the description box.  Select “OK” in the “File write” popup box.  Go to “CONSTRAINT\_TABLE” view.  Open “File” dropdown menue. Select “Compare table to another version”  Select “TEST\_DB\_\_CONSTAINT\_CHANGE-TABLE.XML” Press “Open” | PURPOSE  To have a table that is the foundation for the test.  Get to edit menu so that we may change the table.  To have a new row to make changes to the table.  To have a row that is different from the original table.  To have a row that is different from the original table.  To open the file menu so that we may be able to create a second table that is similar.  To start the process of saving a new table that is like the original table.  To save a new table.  To fill in requested information.  To proceed with the workflow.  To proceed with the workflow.  To begin the comparison.  To select the table to compare. | EXEPCTED RESULTS  A new window is open with CONSTRAINT\_Table  Dropdown with edit results appear.  A new empty row is added to the bottom of the table.  “55” is in the first column last row.  “1” is in the second column last row.  Dropdown of the file menu appears.  Save File window opens.  XML History Input window opens.  File write popup appears showing that the table was saved. The file is created and saved.  “File write” popup box is closed. New table is created.  Table is displayed in window.  Open a file window is displayed with available tables.  Compare result popup window opens and shows 1 mismatch. Added row is shown in a new window. | | | COMMENTS |
| Concluding Remarks: This test does not pass. Although in an attempt of this test I selected the original file after the save as and the change was registered. I believe that the steps of the test need rewriting, because when we do the save as the window of the table is changed to the new table and not the original table. | | | | | | |
| Testing Team:  Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio | | | | Date Completed: | | |

## Test DBComp2

**Objective: <**< Define the objective of Test XX.Y. >>

**Notes:** <<This area provides general notes concerning the test procedure. Such notes might include comments on how to execute the test procedure, an estimate of the test duration, the requirements of the procedure tests, or a statement of resources needed for this test.>>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DBComp2 | | | | Current Status: << Passed / Failed / Pending >> | | |
| Test title: <<This line contains the long title of the test procedure.>> | | | | | | |
| Testing approach: <<Included in this section is a description of test harnesses, testing frameworks, environmental requirements, test tools and test automation that will be employed to achieve testing. Include naming conventions for tests and test scripts if appropriate. Provide requirements traceability and test priority.  >> | | | | | | |
| STEP  <<N>> | OPERATOR ACTION  Describe the actions taken by the person executing the test procedure. Include the test suite, or the name of the test file (in this case, the contents of the file should be given in the appendix). | PURPOSE  Describe the reason for the step. | | | EXEPCTED RESULTS  Describe the expected response of the system being tested to the action specified under OPERATOR ACTION. This should be derived from the SRS and SDD. Clearly indicate how we determine whether the step passes. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team:  Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio | | | Date Completed: | | | |

## Test DBComp2

**Objective: <**< Define the objective of Test XX.Y. >>

**Notes:** <<This area provides general notes concerning the test procedure. Such notes might include comments on how to execute the test procedure, an estimate of the test duration, the requirements of the procedure tests, or a statement of resources needed for this test.>>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DBComp2 | | | | Current Status: << Passed / Failed / Pending >> | | |
| Test title: <<This line contains the long title of the test procedure.>> | | | | | | |
| Testing approach: <<Included in this section is a description of test harnesses, testing frameworks, environmental requirements, test tools and test automation that will be employed to achieve testing. Include naming conventions for tests and test scripts if appropriate. Provide requirements traceability and test priority.  >> | | | | | | |
| STEP  <<N>> | OPERATOR ACTION  Describe the actions taken by the person executing the test procedure. Include the test suite, or the name of the test file (in this case, the contents of the file should be given in the appendix). | PURPOSE  Describe the reason for the step. | | | EXEPCTED RESULTS  Describe the expected response of the system being tested to the action specified under OPERATOR ACTION. This should be derived from the SRS and SDD. Clearly indicate how we determine whether the step passes. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team:  Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio | | | Date Completed: | | | |

# Duplicate Tests

<<The purpose of this section is to:

* document test input, specific test procedures, and outcomes.
* establish test methods,
* explain the nature and extent of each test >>

<< for each test case, complete the following: >>

## Test DBDup1

**Objective:** The goal of Test 03.1 is to make sure that when there are no duplicates then none of the rows are highlighted yellow.

**Notes:** <<This area provides general notes concerning the test procedure. Such notes might include comments on how to execute the test procedure, an estimate of the test duration, the requirements of the procedur

e tests, or a statement of resources needed for this test.>>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DBDup1 | | | | Current Status: Pending | | |
| Test title: Database Duplicate No Duplicates | | | | | | |
| Testing approach: For this test we will be using the Dup\_DB\_No\_Dups\_Table, this will be done manually. To conduct this test first open the dbEdit.jar and then run Dup\_DB file. No\_Dups\_Table. This will open the No\_Dups\_Table, it is observeable that there are no duplicates between any row. | | | | | | |
| STEP  1  2 | OPERATOR ACTION  Run dbEdit.jar open Dup\_DB, select No\_Dups\_Table  View the table makesure that no row is colored yellow. | PURPOSE  This step is to start the test properly.  To visually verify that the software did not mistakenly highlight a row. | | | EXEPCTED RESULTS  No\_Dups\_Table is opened. The table does not have any duplicate rows.  No rows are highlighted yellow because the table does not have any duplicates. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team:  Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio | | | Date Completed: | | | |

## Test DBDup2

**Objective:** The goal of Test 03.2 is to make sure that when there are no duplicates then none of the rows are highlighted yellow when a nonduplicate row is added.

**Notes:** <<This area provides general notes concerning the test procedure. Such notes might include comments on how to execute the test procedure, an estimate of the test duration, the requirements of the procedure tests, or a statement of resources needed for this test.>>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DBDup2 | | | | Current Status: Pending | | |
| Test title: Database Duplicate No Duplicates add a row that is not a duplicate. | | | | | | |
| Testing approach: For this test we will be using the Dup\_DB\_No\_Dups\_Table, this will be done manually. To conduct this test first open the dbEdit.jar and then run Dup\_DB file. No\_Dups\_Table. This will open the No\_Dups\_Table, it is observable that there are no duplicates between any row. The tester will manually add a row that does not have a duplicate. The table has 3 columns if something was to go wrong when inputting new information it should not fail in the first middle or last columns. After each entry the row should still be unique and there should be no yellow highlighting because all the rows are unique and not duplicates of each other. | | | | | | |
| STEP  1  2  3  4  5  6 | OPERATOR ACTION  Run dbEdit.jar open Dup\_DB, select No\_Dups\_Table  View the table make sure that no row is colored yellow.  Select “Edit” from the top tool bar. Select “Insert Row” from the dropdown menu.  Populate the first column “First Name” with “Jane”.  Populate the second column “Last Name” with “Doe”.  Populate the third column “Age” with 37 | PURPOSE  This step is to start the test properly.  To visually verify that the software did not mistakenly highlight a row.  To create a new empty row to later in the test populate.  To populate the first cell in the new row.  To populate the middle cell in the new row.  To populate the last cell in the new row. | | | EXEPCTED RESULTS  No\_Dups\_Table is opened. The table does not have any duplicate rows.  No rows are highlighted yellow because the table does not have any duplicates.  A new row is added, the row is not highlighted yellow because the empty row is unique.  The row is still unique no row is highlighted yellow.  The row is still unique no row is highlighted yellow.  The row is still unique no row is highlighted yellow. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team:  Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio | | | Date Completed: | | | |

## Test DBDup3

**Objective:** The goal of Test 03.3 is to make sure that when there are no duplicates then none of the rows are highlighted yellow. In This test we will add a row that will become a duplicate.

**Notes:** <<This area provides general notes concerning the test procedure. Such notes might include comments on how to execute the test procedure, an estimate of the test duration, the requirements of the procedure tests, or a statement of resources needed for this test.>>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DBDup3 | | | | Current Status: Pending | | |
| Test title: Database Duplicate No Duplicates | | | | | | |
| Testing approach: For this test we will be using the Dup\_DB\_No\_Dups\_Table, this will be done manually. To conduct this test first open the dbEdit.jar and then run Dup\_DB file. No\_Dups\_Table. This will open the No\_Dups\_Table, it is observeable that there are no duplicates between any row. | | | | | | |
| STEP  1  2 | OPERATOR ACTION  Run dbEdit.jar open Dup\_DB, select No\_Dups\_Table  View the table makesure that no row is colored yellow. | PURPOSE  This step is to start the test properly.  To visually verify that the software did not mistakenly highlight a row. | | | EXEPCTED RESULTS  No\_Dups\_Table is opened. The table does not have any duplicate rows.  No rows are highlighted yellow because the table does not have any duplicates. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team:  Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio | | | Date Completed: | | | |

# Test 04

<<The purpose of this section is to:

* document test input, specific test procedures, and outcomes.
* establish test methods,
* explain the nature and extent of each test >>

<< for each test case, complete the following: >>

## Test DBDup2

**Objective: <**< Define the objective of Test XX.Y. >>

**Notes:** <<This area provides general notes concerning the test procedure. Such notes might include comments on how to execute the test procedure, an estimate of the test duration, the requirements of the procedure tests, or a statement of resources needed for this test.>>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DBDup2 | | | | Current Status: << Passed / Failed / Pending >> | | |
| Test title: <<This line contains the long title of the test procedure.>> | | | | | | |
| Testing approach: <<Included in this section is a description of test harnesses, testing frameworks, environmental requirements, test tools and test automation that will be employed to achieve testing. Include naming conventions for tests and test scripts if appropriate. Provide requirements traceability and test priority.  >> | | | | | | |
| STEP  <<N>> | OPERATOR ACTION  Describe the actions taken by the person executing the test procedure. Include the test suite, or the name of the test file (in this case, the contents of the file should be given in the appendix). | PURPOSE  Describe the reason for the step. | | | EXEPCTED RESULTS  Describe the expected response of the system being tested to the action specified under OPERATOR ACTION. This should be derived from the SRS and SDD. Clearly indicate how we determine whether the step passes. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team:  Torres, Andrea | Pincus, Nicholas R.| De La Cruz, Julio | | | Date Completed: | | | |

# User Interface Testing

<<This section focuses on the interaction between the user and the system. For testing the user interface, consider the following traits:

* Consistent terminology, shortcut keys, menu selections, and presentation
* Correct language, spelling, and grammar.
* Flexibility in navigation between windows and interface elements.
* Error handling that will inform user of critical operations.
* Follows standards and guidelines such as placement of scroll bars, windows, and menu items.

This section could be integrated into Section 4.

>>

# Test Schedule

<< Specify the schedule for testing activities. A table with the order and completion dates of the tests is useful. The table below might be useful.>>

|  |  |  |
| --- | --- | --- |
| **Task and date** | **People** | **Description** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Other Sections

<< Other sections that may appear in a test plan (but not required for this course) are:

* Test Management Requirements: how testing is to be managed; a delineation of responsibilities of each project organization involved with testing
* Staffing and training needs: delineate the responsibilities of those individuals who are to perform the testing, level of skill required, and training to be provided
* Environmental Requirements: describe the hardware (including communication and network equipment) needed to support testing; describe configuration of hardware components on which software and database to be tested are to operate.
* Software Requirements: describe the software needed to support testing; include the software code and databases that are object of the testing. Also include software tools such as compilers, CASE instruments and simulators that are needed to model the user’s operational environment.
* Risk and contingencies
* Cost: include an estimate of costs.
* Approvals
* Test Deliverables

>>

# Appendix

<< possibly more readable to put the expected output here and refer to it in the previous sections. Might also provide explicit directions for analysis of output, if it’s easier to read as an appendix or if analysis is post execution. >>