Database-Edit

**Test plan**

**Version 1.0**

04/14/2020

**Document Control**

**Approval**

The Guidance Team and the customer shall approve this document.

**Document Change Control**

|  |  |
| --- | --- |
| Initial Release: | 04/05/2020 |
| Current Release: | 04/14/2020 |
| 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:

Customer:

Software Team Members:

**Change Summary**

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
| 1.0 | 04/14/2020 | Ricardo Sanchez | Created Test Plan Peer-Unrevised Version |
|  |  |  |  |
|  |  |  |  |

Note: The template presented in this document was taken from:

Donaldson, S., and S. Siegel, *Successful Software Development*. Upper Saddle River, NJ: Prentice Hall, 2001, pp. 321-323.

Note: The template presented in this document was taken from: Donaldson, S., and S. Siegel, *Successful Software Development*. Upper Saddle River, NJ: Prentice Hall, 2001, pp. 321-323 and modified by Humberto Mendoza and Steve Roach.

Supplementary information is from:

Pfleeger, S. *Software Engineering, Theory and Practice*. Upper Saddle River, NJ: Prentice Hall, 1998, p. 365.

**Table of Contents**

[**DOCUMENT CONTROL II**](#_heading=h.30j0zll)

**Approval ii**

**Document Change Control ii**

**Distribution List ii**

**Change Summary ii**

[**1.**](#_heading=h.3dy6vkm) **INTRODUCTION 1**

**1.1.** **Purpose 1**

**1.2.** **Scope 1**

**1.3.** **System Overview 1**

**1.4.** **Suspension and Exit Criteri a 1**

**1.5.** **Document Overview 1**

**1.6.** **References 1**

[**2.**](#_heading=h.qsh70q) **TEST ITEMS AND FEATURES 2**

[**3.**](#_heading=h.35nkun2) **TESTING APPROACH 3**

[**4.**](#_heading=h.44sinio) **TEST G4 4**

**4.1.** **Test G4.C1 4**

**4.1.** **Test G4.C2 4**

**4.1.** **Test G4.D1 4**

**4.1.** **Test G4.D2 4**

**4.1.** **Test G4.D3 4**

[**5.**](#_heading=h.3as4poj) **USER INTERFACE TESTING 5**

[**6.**](#_heading=h.3j2qqm3) **TEST SCHEDULE 6**

[**7.**](#_heading=h.1y810tw) **OTHER SECTIONS 7**

[**8.**](#_heading=h.1pxezwc) **APPENDIX 8**

# 1. **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**

The suspension criteria of the test are the following: each evaluator must be able to successfully run each test and get the expected results shown below. However, only the critical test cases must pass to suspend testing.

## **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**

The items to be tested are the following functions of the dbEdit program:

* The File/Compare option allows a user to compare two versions of a table. This compare can be restricted to selected columns by using the Options/Column Selection feature. The output is rather crude and is not intended for large change lists.
* File/Check for Duplicate Rows should list rows that have identical contents. This check is restricted to the columns selected in Options/Select Columns.

# **Testing Approach**

These tests are designed to test program functions in sequences that approximate operational use of the system. These tests are the most significant types of operations, therefore 80% of them are critical, if any of the critical tests can’t be completed, i.e. they fail, testing must continue until they all pass (the critical ones).

**Table 1: Test Plan**

|  |  |  |
| --- | --- | --- |
| **TEST SUITE <Identifier>** | | |
| **Description of Test Suite** | The test suite will focus on testing the ‘compare file to other version’ and ‘check for duplicate rows’ functions of the database table editor. | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| G4.C1 | Compare 2 different versions of a table | Critical |
| G4.C2 | Compare 2 different tables | Critical |
| G4.D1 | Check for duplicate rows in an empty table | Critical |
| G4.D2 | Check for duplicate rows in a table | Critical |
| G4.D3 | Check for duplicate rows in a table (special case) | Normal |

# **Test** G4

<<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** G4.C1

**Objective:** The objective of this test case is to compare two different versions of the same table, and confirm whether or not the program compares and displays the differences of the right fields.

**Notes:** This test case should take less than 5 minutes to successfully execute. The resources needed to successfully run this test are the dbEdit.jar executable, and the input files: G4\_\_TEST\_TABLE.XML, G4\_\_TEST\_TABLE~.XML provided in the G4\_TestFiles folder provided over on the Group 6 Team 4 GitHub repository. This test case tests 2 versions are of the same table, with all columns fields selected (that is, without manually selecting any specific columns on the options menu tab, also named the default option).

Note that the two files: G4\_\_TEST\_TABLE.XML and G4\_\_TEST\_TABLE~.XML are two different files. Also, the screenshots provided below were resized to fit in the table, actual results might vary on size but not on content. E.g. the name of the window, and the actual content inside of the window should be the same as below.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 1 | | | | Current Status: Pending | | |
| Test title: Compare 2 different table versions | | | | | | |
| Testing approach: This test will be conducted using the dbEdit.jar file on either windows or macOS. Behavior is observed on the simulation screen once the program is running. The sample screenshots on the expected results column should match after performing the corresponding operator action. | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | | | EXPECTED RESULTS | COMMENTS |
| 1 | Run the dbEdit executable | Initial condition | | |  |  |
| 2 | Click “File” on the menu bar, then click the “Open” option.  After the file search path window opens, select and open the file: G4\_\_TEST\_TABLE.XML | Open the newest version of the table. | | |  | Output: A window with the newest version named: G4\_TEST\_TABLE.XML  \*Ignore the red highlighted field |
| 3 | Click “File” on the menu bar, then click the “Compare table to another version” option.  After the file search path window opens, select the file: G4\_\_TEST\_TABLE~.XML | Compare the old version (G4\_\_TEST\_TABLE~.XML) to the newest one (G4\_\_TEST\_TABLE.XML) | | |  | First output: the Compare Result window.  Second: a window with the old version named: G4\_TEST\_TABLE~.XML  Third: a window listing the differences of the two files. |
| Concluding Remarks: | | | | | | |
| Testing Team:  Producer: Ricardo Sanchez  Evaluators: Sebastian Nunez, Brian Cardiel | | | Date Completed: | | | |

## Test G4.C2

**Objective:** The objective of this test case is to compare two different table files and observe how the system handles this operation.

**Notes:** This test case should take less than 5 minutes to successfully execute. This test case compares 2 different tables, all columns fields are used to compare. The key element to note is how both tables share only 1 common column field.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 1 | | | | Current Status: Pending | | |
| Test title: Compare 2 different tables (that is with completely different column fields), with only 1 column in common (the selected column field appears in both tables). | | | | | | |
| Testing approach: This test will be conducted using the dbEdit.jar file on either windows or macOS. Behavior is observed on the simulation screen once the program is running. The sample screenshots on the expected results column should match after performing the corresponding operator action. | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | | | EXPECTED RESULTS | COMMENTS |
| 1 | Run the dbEdit executable | Initial condition | | |  |  |
| 2 | Click “File” on the menu bar, then click the “Open” option.  After the file search path window opens, select and open the file: G4\_\_TEST\_TABLE.XML | Open the first of 2 tables to compare. | | |  | Output: A window with the newest version named: G4\_TEST\_TABLE.XML  \*Ignore the red highlighted field. |
| 3 | Click “File” on the menu bar, then click the “Compare table to another version” option.  After the file search path window opens, select the file: G4\_\_CONSTRAINT\_TABLE.XML | Compare: G4\_\_TEST\_TABLE.XML to a completely different format of table: G4\_\_CONSTRAINT\_TABLE.XML | | |  | First output: a window with the columns that are not contained in both tables.  Second: window with the number of total mismatches.  Third: window listing the different rows of both table 1 and table 2.  Fourth: the table G4\_\_CONSTRAINT\_TABLE.XML. |
| Concluding Remarks: | | | | | | |
| Testing Team:  Producer: Ricardo Sanchez  Evaluators: Sebastian Nunez, Brian Cardiel | | | Date Completed: | | | |

## Test G4.D1

**Objective:** The objective of this test case is to check for duplicate rows in an empty table (0 rows, 0 columns).

**Notes:** This test case should take less than 5 minutes to successfully execute. This case uses the default configuration of selecting columns, in other words, all columns are taken into consideration.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 1 | | | | Current Status: Pending | | |
| Test title: Check for duplicate rows in an empty table. | | | | | | |
| Testing approach: This test will be conducted using the dbEdit.jar file using either windows or macOS. Behavior is observed on the simulation screen. The sample screenshots on the expected results column should match the operator’s after performing the corresponding operator action. | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | | | EXPECTED RESULTS | COMMENTS |
| 1 | Run the dbEdit executable | Initial condition | | |  |  |
| 2 | Click “File” on the menu bar, then click the “Open” option.  After the file search path window open, select and open the file: G4\_\_EMPTY\_TABLE.XML | Open and display the empty table: G4\_\_EMPTY\_TABLE.XML | | |  | Output: window with the table: G4\_EMPTY\_TABLE.XML |
| 3 | Click “File” on the menu bar, then click the “Check for duplicate rows”. | Check for duplicates in a table with no column fields or row fields. | | |  | Output: window with the number of duplicates. In this case, since the table is empty, it found 0 duplicates. |
| Concluding Remarks: | | | | | | |
| Testing Team:  Producer: Ricardo Sanchez  Evaluators: Sebastian Nunez, Brian Cardiel | | | Date Completed: | | | |

## Test G4.D2

**Objective:** The objective of this test case is to compare two different versions of the same table file.

**Notes:** This test case should take less than 5 minutes to successfully execute.

The only difference in both files is the row with index 2,

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 1 | | | | Current Status: Pending | | |
| Test title: Check for duplicate rows in a table with 1 duplicate. | | | | | | |
| Testing approach: This test will be conducted using the dbEdit.jar file using either windows or macOS. Behavior is observed on the simulation screen. The sample screenshots on the expected results column should match after performing the corresponding operator action. | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | | | EXPECTED RESULTS | COMMENTS |
| 1 | Run the dbEdit executable | Initial condition | | |  |  |
| 2 | Click “File” on the menu bar, then click the “Open” option.  After the file search path window open, select and open the file: G4\_\_TEST\_TABLE~.XML | Open and display a table with repeated column fields. | | |  | Output: window with the table: G4\_\_TEST\_TABLE~.XML |
| 3 | Click “Options” on the menu bar, then click the “Column selection” option. | Open the Enable/Disablecolumns window. | | |  | Output: window with the column fields of the table. |
| 4 | Once the Enable/Disable window is open, unselect the fields: INDEX and DESCRIPTION. | Leave only one field to compare. | | |  | This window can be left open or closed after unselecting the fields. |
| 5 | On the table window (the one on step 2), click “File” on the menu bar, then click the “Check for duplicate rows”. | Check for duplicates with only one column selected | | |  | First output: window with the number of duplicates found  Second: window with value of the duplicate key.. |
| Concluding Remarks: | | | | | | |
| Testing Team:  Producer: Ricardo Sanchez  Evaluators: Sebastian Nunez, Brian Cardiel | | | Date Completed: | | | |

## Test G4.D3

**Objective:** The objective of this test case is to observe how the program handles a boundary when checking for duplicate rows.

**Notes:** This test case should take less than 5 minutes to successfully execute. The test consists of checking for duplicates on a table with no duplicates while at the same time deselecting all columns with the column selection feature.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 1 | | | | Current Status: Pending | | |
| Test title: Check for duplicate rows in a table with no duplicates | | | | | | |
| Testing approach: This test will be conducted using the dbEdit.jar file using either windows or macOS. Behavior is observed on the simulation screen. | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | | | EXPECTED RESULTS | COMMENTS |
| 1 | Run the dbEdit executable | Initial condition | | |  |  |
| 2 | Click “File” on the menu bar, then click the “Open” option.  After the file search path window open, select and open the file: G4\_\_TEST\_TABLE.XML | Open a table with 0 row repetitions, both individually and as a whole.. | | |  | Output: window with the table: G4\_\_TEST\_TABLE.XML |
| 3 | Click “Options” on the menu bar, then click the “Column selection” option. | Open the Enable/Disablecolumns window. | | |  | Output: window with the column fields of the table. |
| 4 | Once the Enable/Disable window is open, unselect the fields: INDEX and DESCRIPTION. | Disable all column fields when comparing. | | |  | This window can be left open or closed after unselecting the fields. |
| 5 | On the table window (the one on step 2), click “File” on the menu bar, then click the “Check for duplicate rows”. | See if the program handles this operation correctly. | | |  | First output: window with the number of duplicates found.  Second: 2 windows with the values of the duplicate keys found.  Third: Changes on the window with the table (from step 2), the highlights in yellow indicate duplicates. |
| Concluding Remarks: | | | | | | |
| Testing Team:  Producer: Ricardo Sanchez  Evaluators: Sebastian Nunez, Brian Cardiel | | | 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. >>

$