X-Audit  
Tutorial

Software version 13.3.02, 20230822

Contents

[Contents 2](#_Toc143165584)

[Publication Information 4](#_Toc143165585)

[Preface 5](#_Toc143165586)

[About this guide 5](#_Toc143165587)

[Version 5](#_Toc143165588)

[How to use this guide 5](#_Toc143165589)

[Prerequisites 6](#_Toc143165590)

[What is new in the latest release? 7](#_Toc143165591)

[Overview of X-Audit 8](#_Toc143165592)

[Metrics Analysis 9](#_Toc143165593)

[Metrics Toolbar 16](#_Toc143165594)

[Screen Metrics 22](#_Toc143165595)

[Screen Metrics Toolbar 25](#_Toc143165596)

[File Metrics 27](#_Toc143165597)

[Business Process Logic Metrics 29](#_Toc143165598)

[Specialized Analysis 32](#_Toc143165599)

[Options On Report 35](#_Toc143165600)

[Specialized Analysis Toolbar 40](#_Toc143165601)

[Problem Analysis 46](#_Toc143165602)

[Object Allocation 50](#_Toc143165603)

[Database Summary 52](#_Toc143165604)

[Summary Report 54](#_Toc143165605)

[Audit Report 55](#_Toc143165606)

[Initialize Source Archiving 59](#_Toc143165607)

[Generate Metrics and Problem Analysis 63](#_Toc143165608)

[Edit Problem Audit Limit 66](#_Toc143165609)

[Edit Problem Categories 68](#_Toc143165610)

[Track Database Changes 73](#_Toc143165611)

[View Database Size Statistics 77](#_Toc143165612)

[Inter-Repository Options 84](#_Toc143165613)

[Generate Difference Analysis 86](#_Toc143165614)

[Display Difference Analysis 87](#_Toc143165615)

[Customized Libraries 88](#_Toc143165616)

[Generate PTF Analysis 89](#_Toc143165617)

[PTF Analysis 89](#_Toc143165618)

[Manage Linked Repositories 93](#_Toc143165619)

[Linking IBM i Repositories 93](#_Toc143165620)

[Appendix A – Code Review feature (TD/OMS Support) 97](#_Toc143165621)

[Appendix B – Initializing Source Archiving 102](#_Toc143165622)

[Appendix C – Adding new Problem Analysis Category 104](#_Toc143165623)

[Prerequisites 105](#_Toc143165624)

[Programming Guidelines 108](#_Toc143165625)

[User Guidelines 113](#_Toc143165626)

[Index 115](#_Toc143165627)

Publication Information

© 2023 Fresche Solutions Inc.

**Published by:**

Fresche Solutions Inc.  
995 Wellington Suite 200  
Montreal, QC  
Canada, H3C 1V3

Telephone numbers:

(514) 747.7007  
(toll-free in US and Canada): 1.800.361.6782  
(toll-free in Belgium, France, Germany, UK): 00 800 361 67 82 0  
(toll-free in Australia): 0011 800 361 6782 0

E-mail for support: [support@freschesolutions.com](mailto:support@freschesolutions.com?subject=Support)

E-mail for inquiries: [info@freschesolutions.com](mailto:inquiry@freschesolutions.com?subject=inquiry)

Web: [www.freschesolutions.com](http://www.freschesolutions.com/)

Title: X-Audit Tutorial, Software version 13.3.02

Publication Date: August 2023

**Trademarks**

X-Analysis and X-Analysis Professional are trademarks or registered trademarks of Fresche Solutions Inc. <iSeries, Power Systems, Power8> are registered trademarks of IBM Corporation. Microsoft and Microsoft Windows are registered trademarks of Microsoft Corporation. All other brand and product names are trademarks or registered trademarks of their respective companies.

No part of this document may be reproduced or transmitted in any form or by any means, without prior permission in writing from Fresche Solutions.

The information in this manual is believed to be correct at the time of publication. However, Fresche Solutions Inc. makes no warranty, express or implied, about the accuracy of this information and reserves the right to revise this document or make changes to the products described herein at any time without notice and without obligation. Fresche Solutions Inc. is not liable for any loss of data, damage to databases or other software, or any other losses arising from the use of this manual.

Preface

About this guide

This X-Audit guide outlines the X-Audit module. It explains how to use X-Audit to measure, compare, and manage the quality, metrics, and complexity of the IBM i applications. In particular, it discusses the following topics:

* Metrics Analysis
* Screen Metrics
* File Metrics
* Business Process Logic Metrics
* Specialized Analysis
* Problem Analysis
* Object Allocation
* Database Summary
* Summary Report
* Initialize Source Archiving
* Generate Metrics Analysis
* Generate Problem Analysis
* Track Database Changes
* View Database Size Statistics
* Difference Analysis
* PTF Analysis

Version

This guide covers X-Audit, Software version 13.3.02.

How to use this guide

Each chapter in this guide focuses on a significant aspect of the X-Audit module. The X-Audit features and options are explained in the same sequence as they appear in the application. The explanations are accompanied by relevant screens that make understanding the module easier for the users.

The X-Analysis suite of products contains a total of eight modules. This guide only covers the X-Audit module. For information about the other modules, please contact your Fresche Solutions representative, or visit us at:  
<https://freschesolutions.com/products/x-analysis-suite/>

Prerequisites

Before starting this tutorial, the following are required:

* The X-Analysis server and the client components should be installed on an IBM i server and a Windows PC respectively.
* The demo library – XAN4CDEM and its cross-reference library – XAN4CDXA should be successfully restored on the IBM i.

For details on how to implement these prerequisites, refer to the document “X-Analysis\_User\_Manual” for this release.

What is new in the latest release?

|  |  |
| --- | --- |
|  | For this release, we updated this document. Here is the change that were made to this document. |

* [Initialize Source Archiving](#_Ref-596753523): Describe the commands XA4MAINT and XA4SRCMNT. Also, added note regarding these two commands.
* [Display Difference Analysis](#_Ref-2056668205): Described the option Display Difference Analysis more elaborately.
* [Metrics Analysis](#_Ref801080524), [Screen Metrics](#_Ref-883077166), [File Metrics](#_Ref1984196772), [Business Process Logic Metrics](#_Ref105704089), [Specialized Analysis](#_Ref-1592660068), [Problem Analysis](#_Ref3020133), [Object Allocation](#_Ref-434811608), [Database Summary](#_Ref2009537555), [Summary Report](#_Ref1657982129), [Initialize Source Archiving](#_Ref-596753523), [Edit Problem Audit Limit](#_Ref2114457060), [Edit Problem Categories](#_Ref288995958), [Track Database Changes](#_Ref-867225387), [Appendix C – Adding new Problem Analysis Category](#_Ref1919873333), and [Appendix A – Code Review feature (TD/OMS Support)](#_Ref-1114179681): Replaced the image for Audit Options.
* [Generate Metrics and Generate Problem Analysis](#_Ref-1474115974): Added new option under the section X-Audit.
* [Adding new Problem Analysis Category](#_Ref1919873333): Following changes were performed in this section.
  + Revised the topic title from Adding new Problem Analysis Category using Service Program to Adding new Problem Analysis Category.
  + [Prerequisites](#_Ref726615572): Editorial changes regarding adding new problem category under the section Adding new Problem Analysis Category.
  + [Programming Guidelines](#_Ref1201848044): Revised the topic title from Programmer Guide to Programming Guidelines.
  + [User Guidelines](#_Ref-73173543): Revised the topic title from User Guide to User Guidelines.

Overview of X-Audit

The main premise behind any audit exercise is to evaluate changes. For software applications, technological innovations can be abundant; this makes auditing an indispensable task for systems analysts. Keeping track of these modifications is easier with X-Audit#. It is one of the most prominent modules of X-Analysis. The tool provides quality and complexity metrics, cross-application impact, and difference analyses for development and analytical purposes by evaluating data with details of numbers and graphical trends.

There are numerous practical benefits of utilizing the advanced features and options of X-Audit. The options help users build an in-depth understanding of the complex codes in the system and plan their operations accordingly. They support refitting customization projects and assessing differences – both at the design level and the code level. X-Audit also prewarns the users of harmful system conditions so as to prevent the production problems.

|  |  |
| --- | --- |
|  | The Audit Options functionality is implicitly dependent on X-Rules for certain problem analysis categories and screen metrics. The user must own the X-Rules license to generate Business Rules (for the X-Ref library) before using them.  The X-Audit features dependent on X-Rules license are "Program Code Alerts" under Problem Analysis and Screen Metrics information.  The X-Analysis repertoire also features the TD/OMS support. The details are covered in the [Appendix A – Code Review feature (TD/OMS Support)](#_Ref-1114179681). |

**# The product is under continuous improvement, hence, there may be differences in the screens.**

Metrics Analysis

“As a system evolves, its complexity increases, unless steps are taken to reduce it.”   
 Professor Meir Lehman

The Metrics Analysis solution set contains advanced computing solutions because X-Analysis recognizes the importance of monitoring, measuring, and managing the complexity and changes to the databases of legacy applications.

This module provides quality and complexity metrics for development and analytical purposes.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Options | **Brief Description** | | Metrics Analysis | Advanced computing solutions based on different complexity levels of programs. | | Metrics History | Record of all Metrics data generated previously. | | Screen Metrics | Displays information of all the screen functions of a program. | | File Metrics | Presents Metrics-related information of all the files. | | Business Process Logic Metrics | Migrated logic data in metrics form. | | Specialised Analysis | Helps the user create customized reports using the metrics database. | | Problem Analysis | Displays data discrepancies present in specific files. | | Object Allocation | Displays information about objects and the application areas they belong to. | | Database Summary | Summarized database report for the cross-reference library. | | Summary Report | Structured report for Metrics Analysis and Problem Analysis. | | Initialize Source Archiving | Refers to feature enabling data records of all the source types. | | Generate Metrics and Problem Analysis | Option to generate Metrics and Problem Analysis data. | |

Metrics Analysis classifies each program based on low, average, and high complexity. This is calculated based on allocated weights in the Metrics Preference Setting dialog. The user may decide to specify one or more criteria. In following example, 1000 or higher Source lines would put a program in High complexity, 750-1000 in Average complexity, and lower than 750 in a Low complexity level. There is an exception with the Maintainability Index, where a lower value means High complexity and higher value means Low complexity.

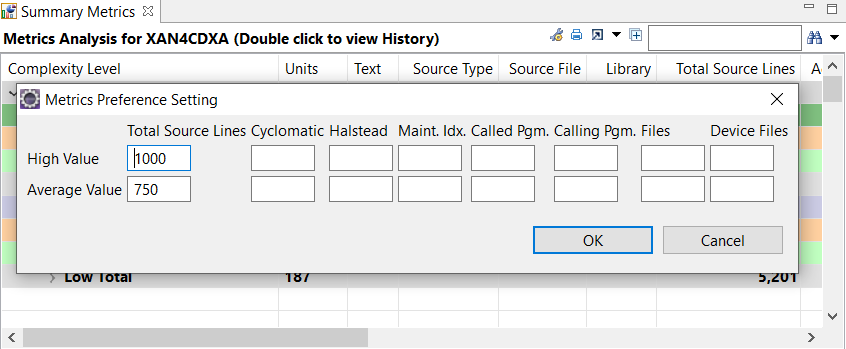


Fig. 1.1.1 – Summary Metrics

Under the Metrics Analysis, there are parameters like Cyclomatic Complexity, Halstead, and Maintainability Index, etc. The conventions used for processing these are given below:

Cyclomatic Complexity – This is a count of the number of decision points and exit points in a module. In RPG, this would be a count for each subroutine or procedure (plus mainline) of how many statements have these operation codes:

– IFxx  
– ELSEIF  
– DOW  
– DOU  
– CABxx  
– WHxx  
– CASxx  
– FOR

Halstead – This is calculated using four different values (N1, N2, η1, and η2) obtained from the source code of each subroutine. These values show numbers of different op codes or number of different variables used in the subroutines, whether once or multiple times.

η1 – number of distinct operators  
η2 – number of distinct operands  
N1 – total number of operators  
N2 – total number of operands

From these four numbers, different measurements are calculated:

**Program Length N = N1 + N2**  
**Program Vocabulary η = η1 + η2**  
**Halstead Volume (HVOL) V=N×logη**  
**Difficulty = (η1/2) \* (N2/η2)**  
**Effort = Volume \* Difficulty**

Maintainability Index – Hewlett-Packard engineers put forward this formula. According to this, the number derived is directly proportional to the maintainability factor. The maintainability index is calculated using lines-of-code measures, Cyclomatic Complexity and Halstead complexity measures. The value can range from negative values to value higher than 180. This Maintainability Index value must be interpreted the following way:

1. Value above 85 percent: the code is considered highly maintainable component.
2. Value between 85 and 65 percent: the code maintainability is considered moderately maintainable component.
3. Value lower than 65 percent: the code is considered as difficult to maintain.

The calculation in X-Analysis limits the low value to 1. A value of zero indicates that the Maintainability index could not be properly calculated due to one of the other metrics containing an invalid value.

This also applies at the subroutine level.

The formula used to calculate MI (Maintainability Index) is shown below:

**MI = 171 - 5.2 \* ln(HVOL) - 0.23 \* CC - 16.2 \* ln(LOC) + 50 \* sin(sqrt(2.46 \* CD))**

HVOL - Halstead Volume  
CC - McCabe’s Cyclomatic Complexity, that is, Complexity of the subroutine/procedure/program expressed as the number of independent control ﬂow paths in it.  
LOC - Lines Of Code, that is, Number of non-empty and non-comment code lines of the subroutine/procedure/program.  
CLOC - Comment Lines Of Code, that is, Number of comment and documentation code lines of the subroutine/procedure/program.  
CD - Comment Density, that is, Ratio of the comment lines of the subroutine/procedure/program (CLOC) to the sum of its comment (CLOC) and logical lines of code (LOC)

The Metrics Analysis option provides low, medium, and high complexity classification of each program, based on the following attributes of a program:

* Source Type
* Source Lines
* Cyclomatic Complexity
* Halstead
* Maintainability Index
* Files
* Device files comprising Display Files and Printer Files
* Called Programs
* Calling Programs

The Metrics information is broken down at the highest level into the following categories:

* Batch programs
* Interactive programs

These two highest levels are then further sub-categorized as High, Average, and Low.

The user-specified values determine where a program fits into these sub-categories. By default, these preferences are blank, so every program is ‘Low’ complexity until the values have been set. Check the Metrics Preferences section (below) to set these values.

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | Please execute the ‘Generate Metrics and Problem Analysis’ option before selecting the ‘Metrics Analysis’ option. | |

The Metrics Analysis option is shown in the screen below:

|  |
| --- |
| Fig. 1.1.2 – Audit Options – Metrics Analysis option |

The following screen displays the Metrics information for the cross-reference library, XAN4CDXA.

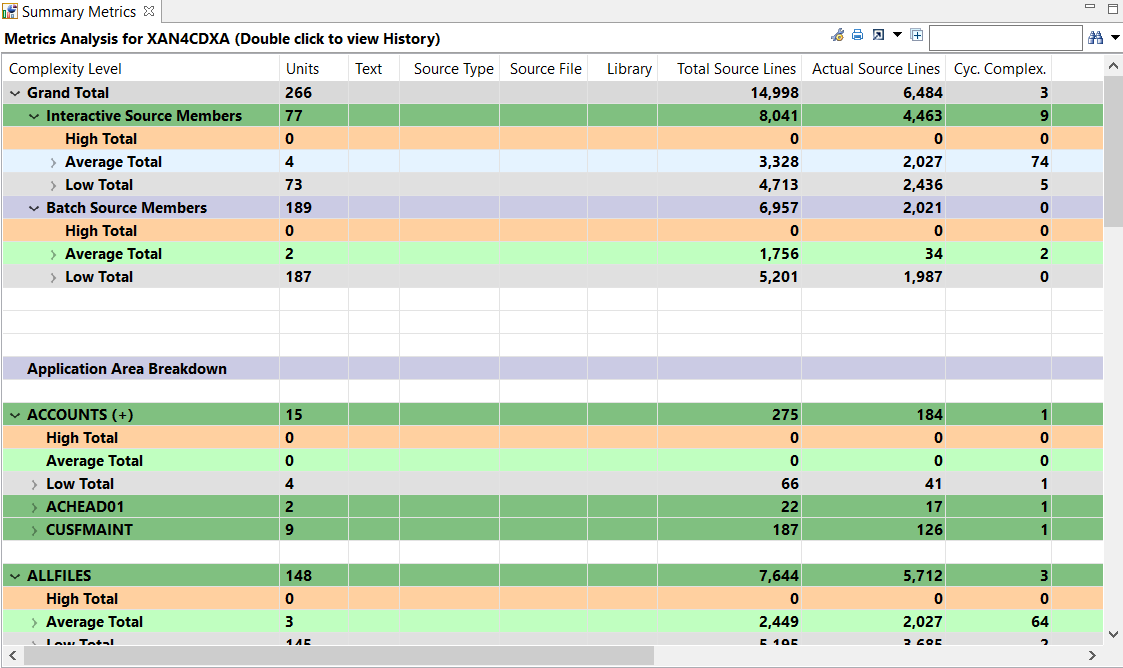


Fig. 1.1.3 – Metrics Analysis information for XAN4CDXA

The Metrics information displayed above has the following columns:

1. Complexity Level: For each attribute, for example, each file usage equals 1, and so on. The sum total of these per program will be calculated, and the Complexity level for the program worked out. For example, Points <10 - LOW, 10-20 - AVERAGE, >20 - HIGH. If required, a user can allocate values for each complexity classification using the Metrics Preferences option.
2. Units: When collapsed, the total number of programs displays. When expanded, the name of the program displays.
3. Text: Displays the description of the object.
4. Source Type: Informs about the type of source.
5. Source File: Displays the name of the Source File.
6. Library: Displays the name of the Source library.
7. Source Lines: The total number of Source Lines in a source member.
8. Cyclomatic Complexity: Measures the amount of decision logic in a program. Shows the total number of conditional statements used.
9. Halstead: Displays program's complexity directly from the source code based upon the operators and operands used.
10. Maintainability Index: The maintainability index calculated with certain formulae from lines-of-code measures, Cyclomatic Complexity and Halstead complexity measures.
11. Files: The total number of Files used by a source member.
12. Device Files: The total number of Display and Printer Files used by source member.
13. Called Programs: Displays the total number of called programs from a source member.
14. Calling Programs: Displays the total number of calling programs from a source member.

|  |  |
| --- | --- |
|  | The Cyclomatic Complexity, Halstead, and Maintainability Index columns in the Metrics window display the average. User can see this in both the Group Total and the Grand Total.  In the Metrics display, the members with no associated objects are displayed in blue. If the user checks the ‘Show only Objects in Metrics’ box in the X-Analysis Preferences window, then such members will not be shown. |

Metrics Toolbar

The Metrics toolbar comprises various options which are described ahead.

**Metrics Preferences**

Set the Metrics Preferences as per requirements.

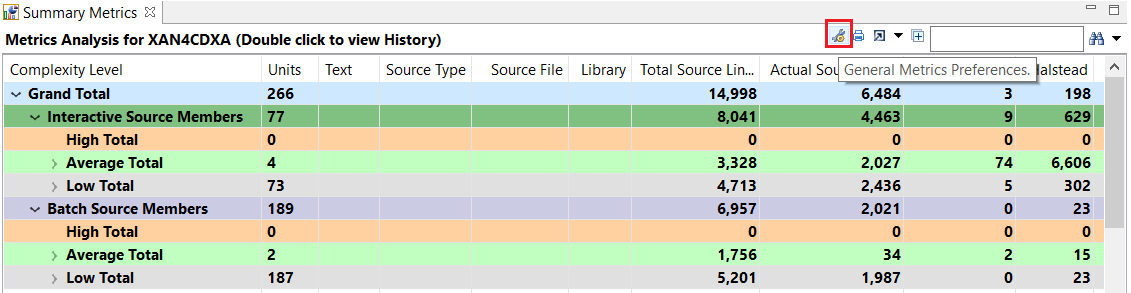


Fig.1.2.1 – General Metrics Preferences icon

Click the General Metrics Preferences icon to invoke the relevant dialog:

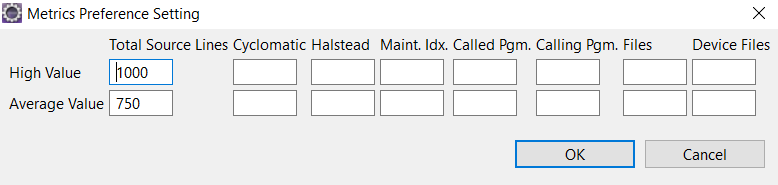


Fig.1.2.2 – Metrics Preference Setting dialog

Specify a high value, so that any value above this will determine that the program is of higher complexity. Any program that falls below an average setting will be indicative of lower complexity.

**Print Metrics**

To print the Metrics information, click the Print button.

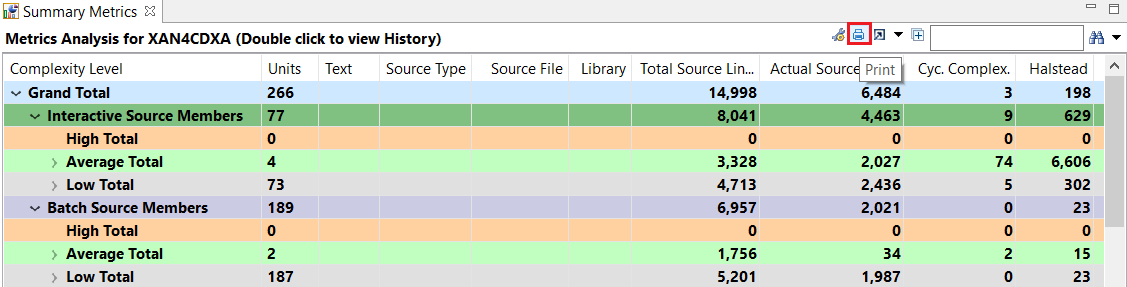


Fig.1.2.3 – Print icon on Metrics Analysis toolbar

**Export Metrics**

Click the Export Options icon. Select from the Export to PDF, Export to MS Word or Export to MS Excel options.

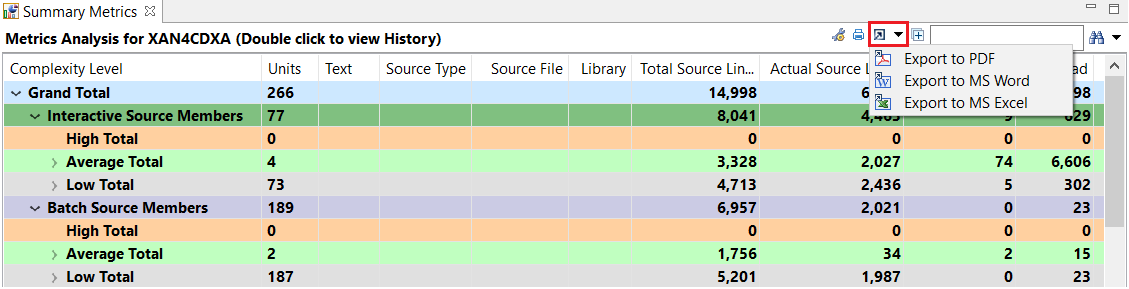


Fig.1.2.4 – Export Options on Metrics Analysis window toolbar

**Expand All**

Click the Expand All icon to expand all the data:

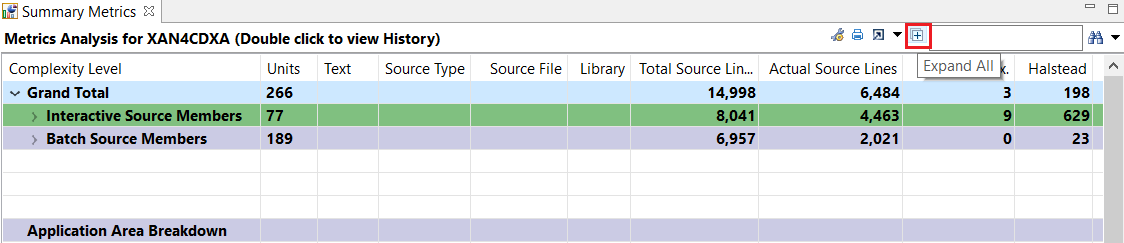


Fig.1.2.5 – Expand All option on Metrics Analysis toolbar

The screen below shows the expanded window after the Expand All icon is clicked:

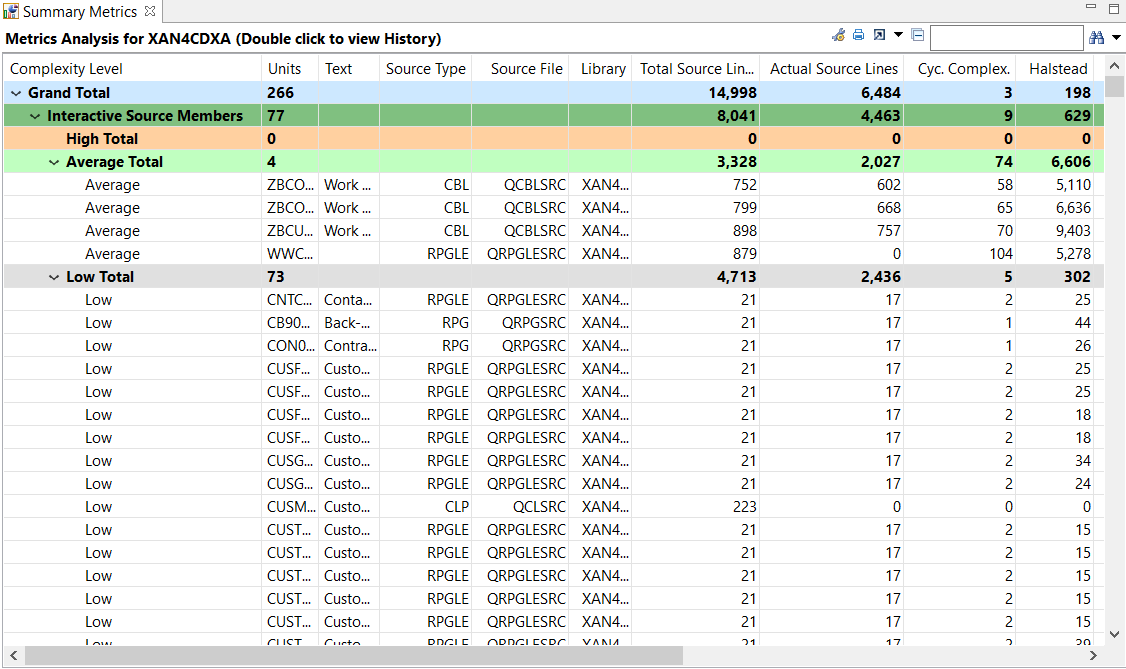


Fig.1.2.6 – Expanded Metrics Analysis window

**Metrics History**

The Metrics History displays all previous instances of the generated metrics data. The Metrics History data will only be available when the user generate the metrics data more than once.

The user can generate the Metrics History for all the individual levels – High, Average, and Low Totals for Batch / Interactive Programs and for the Grand Total.

Double-click on the Interactive Source Members row to generate the history.

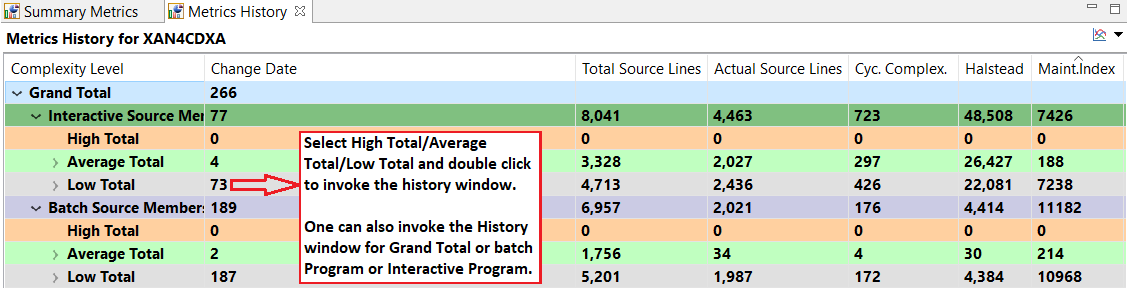


Fig.1.2.7 – Invoking Metrics History for Interactive Programs

Double-click Low Total row to invoke the Metrics History window.

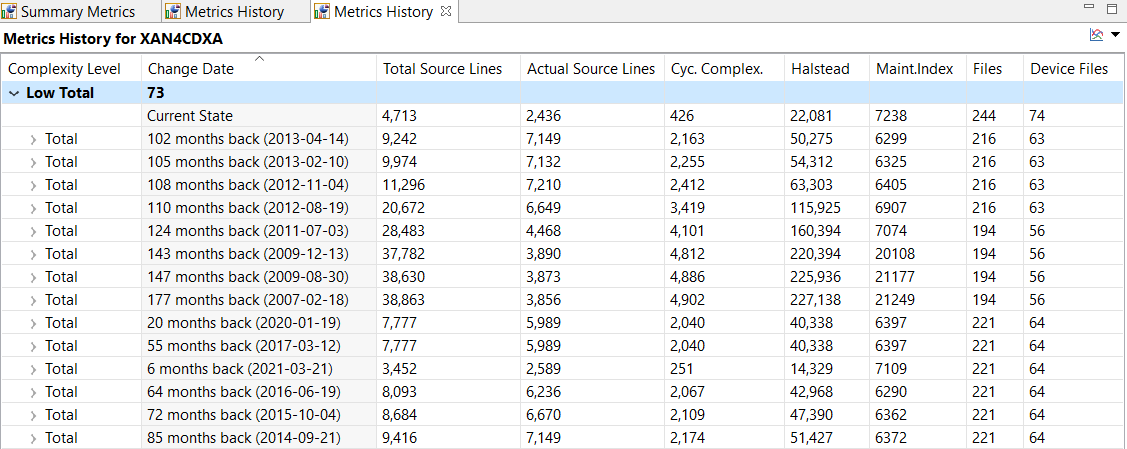


Fig.1.2.8 – Metrics History window for Interactive Programs

Metrics History Chart

The Metrics History can also be displayed in chart form for all the individual parameters. The Metrics History toolbar contains a drop-down button called Metrics Analysis.

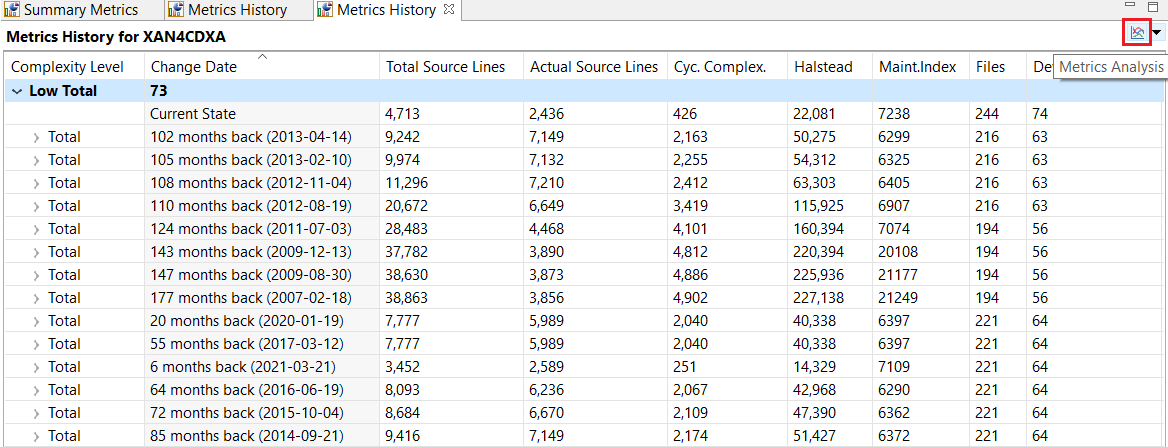


Fig.1.2.9 – Metrics Analysis drop-down button

When the user clicks the Metrics Analysis button, it will display a list of parameters against which a chart can be generated.

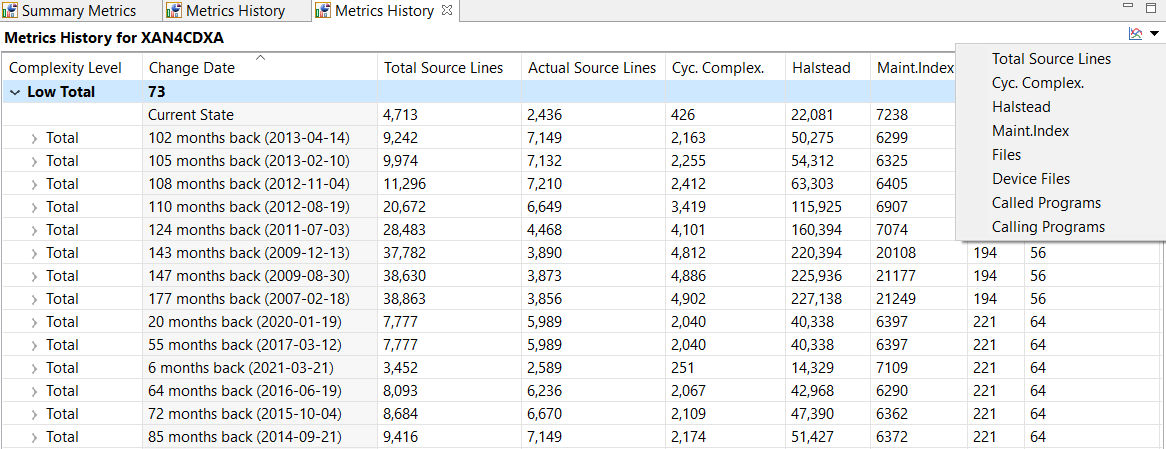


Fig.1.2.10 – Metrics Analysis drop-down options

Select Cyc. Complex. option from the drop-down menu.

This invokes the Metrics History chart displaying Cyclomatic Complexity.

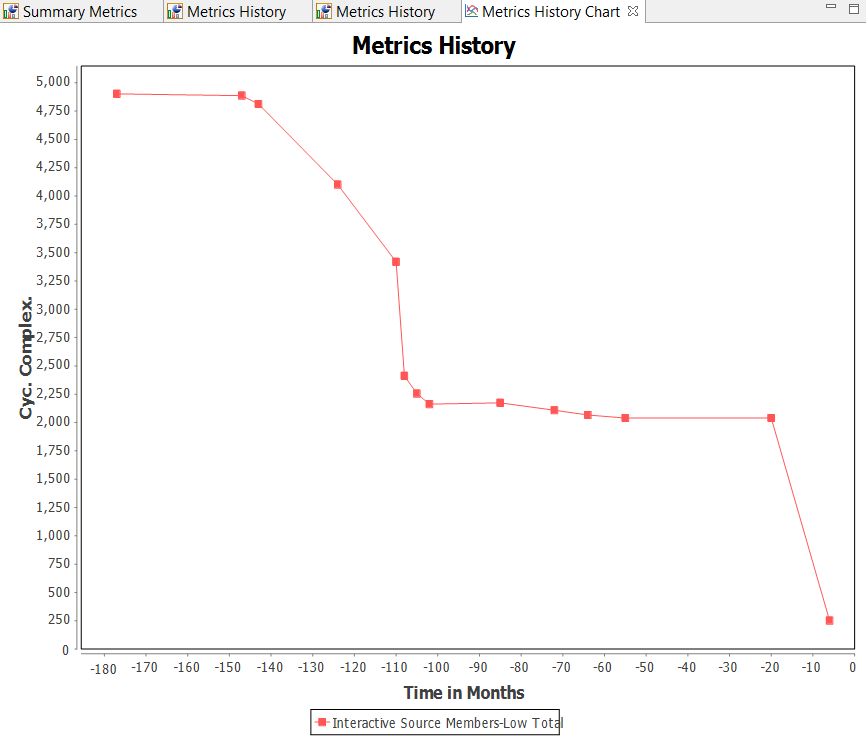


Fig.1.2.11 – Metrics History Chart displaying Cyclomatic Complexity

Similarly, the user can generate charts for all the other parameters.

The Metrics History for all the Interactive/Batch Programs and Grand Total can be checked from the Metrics window.

**Source code tracking**

The Metrics History also provides a facility to see changes in the source code. To invoke a Source Compare editor, expand one of the listed programs from the Metrics History list.

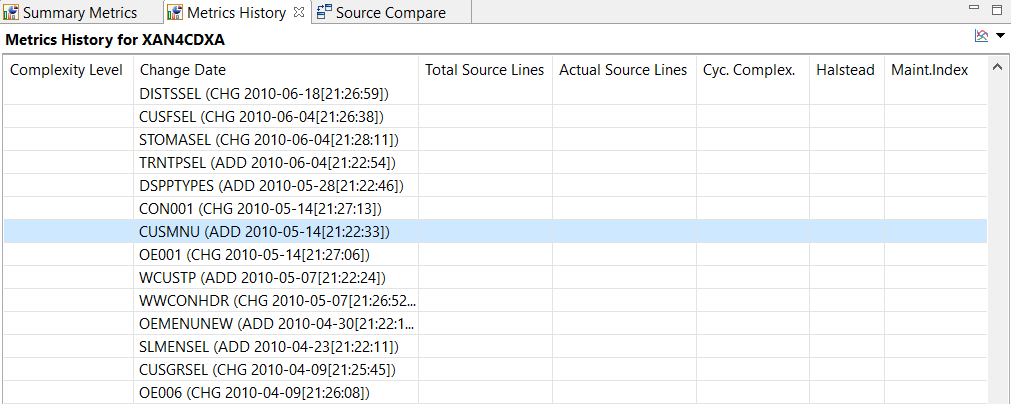


Fig.1.2.12 – Program selected for Source Code Comparison

Double-click on the selected program to invoke the Source Compare editor.

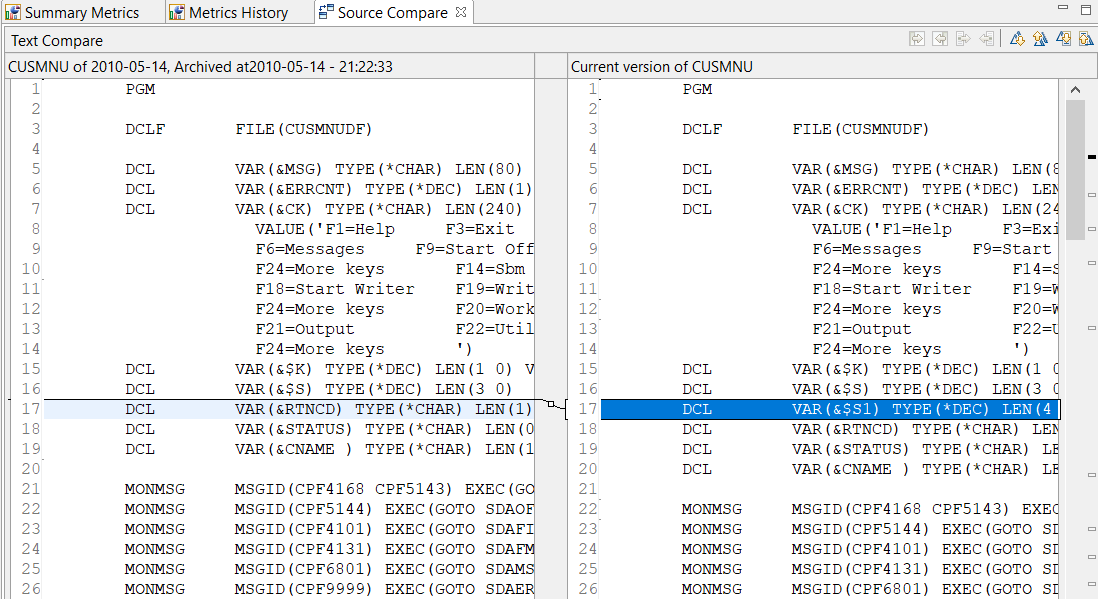


Fig. 1.2.13 – Source Compare Editor displaying the Code Differences

Screen Metrics

|  |  |
| --- | --- |
|  | Business Rules must first be generated in order to view the Screen Metrics. |

Each specific program has various screen functions. Screen Metrics displays these screen functions for a complete application and for all the application areas. The following screen shows the Screen Metrics option.

|  |
| --- |
| Fig. 2.1.1 – Audit Options – Screen Metrics option |

Screen Metrics are designed to help with re-facing projects and UI rewrites. They describe designs, uses and relationships of file complexities, which help to measure the database complexities.

The Complexity Level on Screen Metrics displays two groups that are the Grand Total and Application Area totals. Hereafter, all functions in the application areas or cross-reference application are grouped on the basis of complexity.

The Screen Metrics option provides low, medium, and high complexity classification of the available screen functions. The Screen Metrics information is displayed for the complete application and its application areas.

The following screen displays the Screen Metrics information for the cross-reference library, XAN4CDXA:

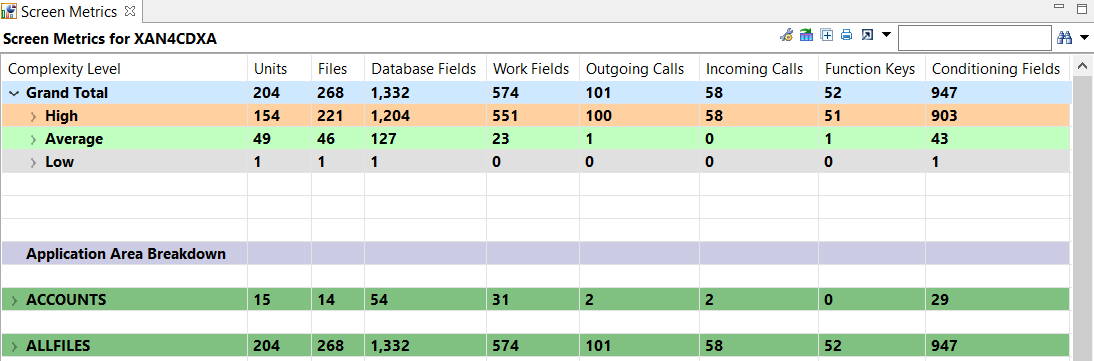


Fig. 2.1.2 – Screen Metrics for cross-reference library, XAN4CDXA

The Metrics information displayed above has the following columns:

1. Complexity Level: This column shows various groups. The topmost group is either the Grand Total or App area totals. All functions in the application area or cross-reference application are then grouped based on complexity, and then on type of function.
2. Units: The number of functions in the group.
3. Files: The total number of Files used by screen function.
4. Database Fields: The total number of screen fields which are read from database fields.
5. Work Fields: The total number of work fields in the group.
6. Outgoing Calls: The number of functions called by this function.
7. Incoming Calls: The number of functions calling this function.
8. Function Keys: The total number of Function key-based actions called by a function which displays a screen (EXFMT). Function Keys value in Screen Metrics do not include options or the function keys F1 = Help, F3 = Exit and F12 = Cancel; these are used for general green screen actions and do not proceed to further action. In other words, no function is generated for them.
9. Conditioning Fields: The number of fields on which there is a conditioning logic. It is the total of conditional indicators used in Display File for the below points.
   * Field constant and the Fields with conditional Indicators.
   * Each DSPATR / COLOR attribute with indicator on the Field constants / Fields.

|  |  |
| --- | --- |
|  | While calculating Metrics in a Synon/2E application, X-Analysis uses the Generated Source and not the Action Diagram. Refer to the screenshot. |

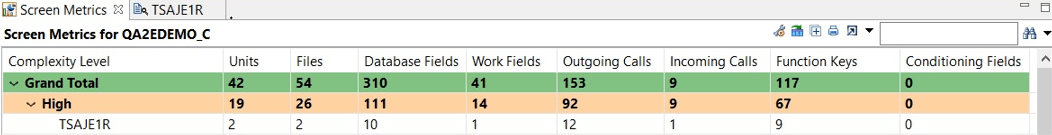


Fig. 2.1.3 – Screen Metrics

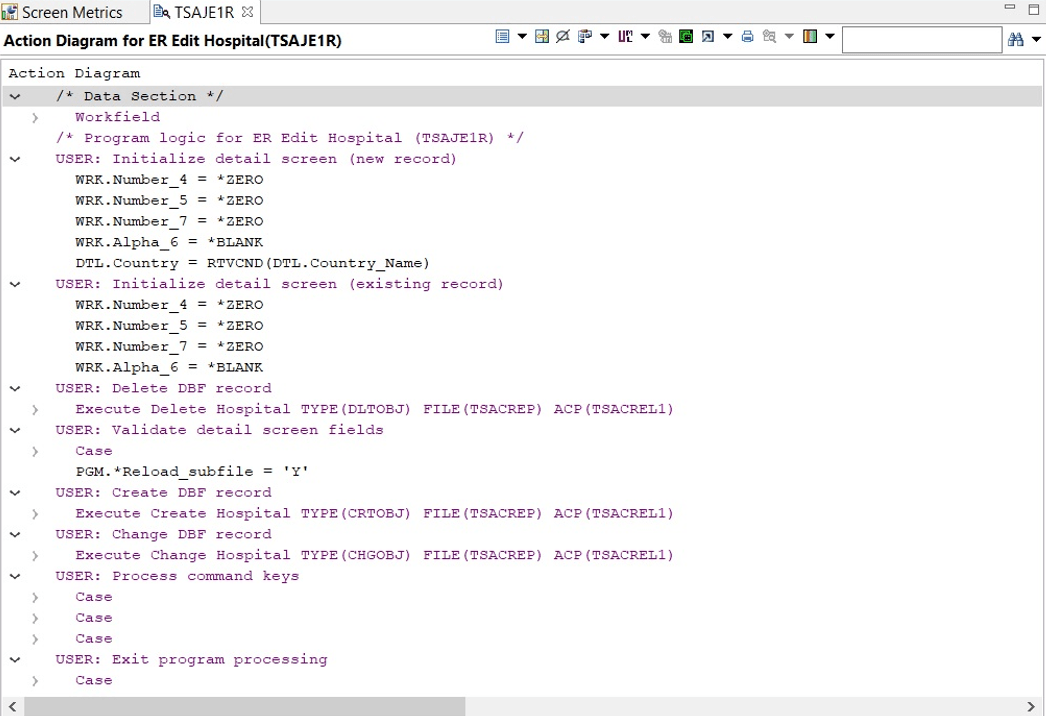


Fig. 2.1.4 – Action Diagram

Screen Metrics Toolbar

The Screen Metrics toolbar consists of various options discussed ahead.

**Screen Metrics Settings**

Set the Screen Metrics Settings as per the requirements.

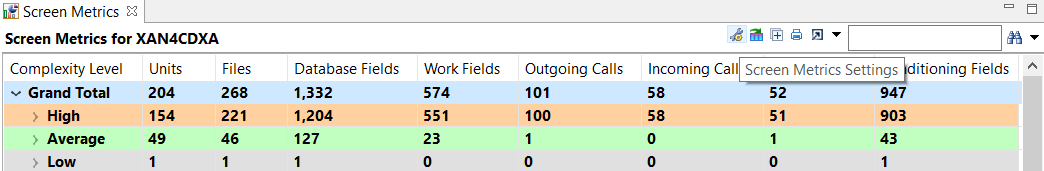


Fig. 2.2.1 – Screen Metrics Settings button on the Screen Metrics toolbar

Click the Screen Metrics Settings icon to invoke the relevant dialog:

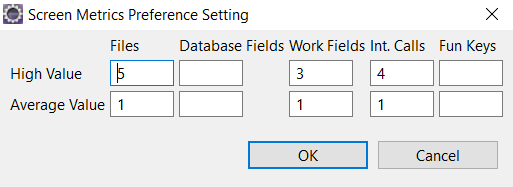


Fig. 2.2.2 – Screen Metrics Preference Setting dialog

The user can set the criteria for High/Average using the above dialog.

**Expanded Screen Metrics**

Click Expanded Screen Metrics to switch the mode from one entry per screen program to one entry per screen format.

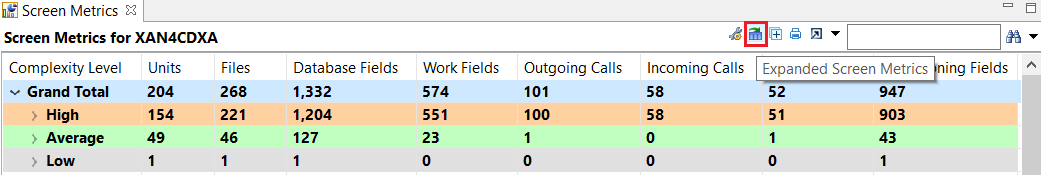


Fig. 2.2.3 – Expanded Screen Metrics button on the Screen Metrics toolbar

**Expand All Screen Metrics Data**

Click Expand All icon which will expand all the data.

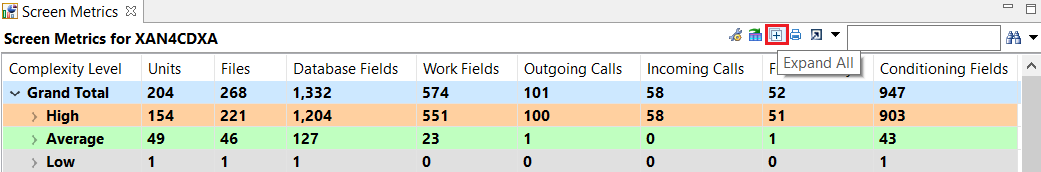


Fig. 2.2.4 – Expand All button on the Screen Metrics Toolbar

**Print Screen Metrics**

To print the Screen Metrics information, click the Print icon on the toolbar.

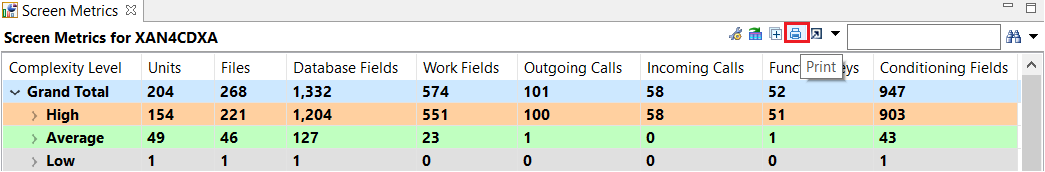


Fig. 2.2.5 – Print icon on Metrics toolbar

**Export Screen Metrics**

Click Export Options and select the Export to PDF, Export to MS Word or Export to MS Excel option, as required:

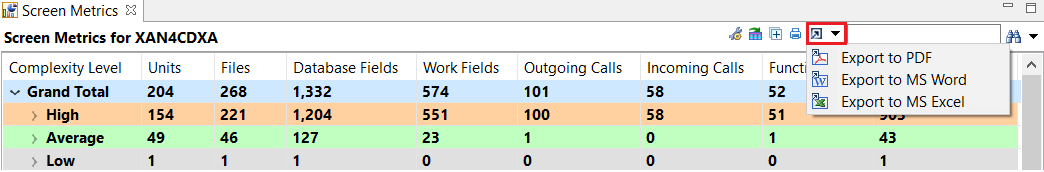


Fig. 2.2.6 – Export Options on Screen Metrics toolbar

File Metrics

The File Metrics displays metrics-related information on all the files. The user will get the overview of metrics data for the complete application and/or for all the application areas. The following screen displays the File Metrics option:

|  |
| --- |
| Fig. 3.1.1 – Audit Options – File Metrics option |

The File Metrics displays information under the following headings:

1. Units: Displays total number of files.
2. Fields: Displays the total number of fields available in the file.
3. Access Paths: The total number of access paths for the file.
4. Creating Programs: The number of programs creating records in this file.
5. Reading Programs: The number of programs reading this file.
6. Updating Programs: The number of programs updating records in this file.
7. Deleting Programs: The number of programs deleting records from this file.
8. Total References: The total number of programs referencing this file. In effect, this is a total of the Creating, Reading, Updating, and Deleting Programs columns.

The following screen displays the File Metrics information for the cross-reference library, XAN4CDXA.

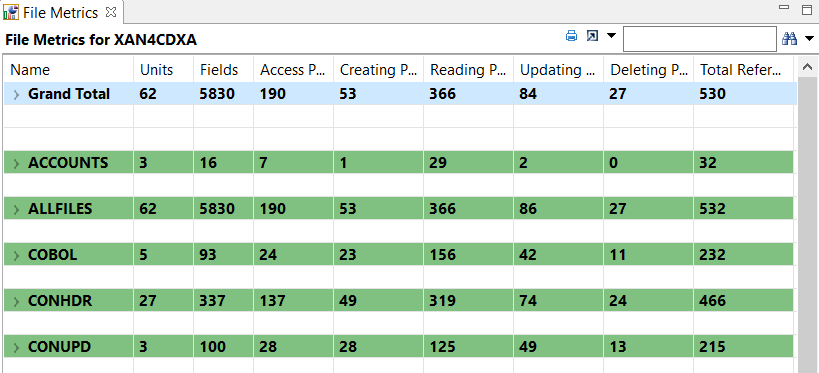


Fig. 3.1.2 – Window displaying File Metrics for XAN4CDXA

|  |  |
| --- | --- |
|  | The File Metrics will now show the long name of Files when the long names exist and the Windows>Preferences>X-Analysis XX.X.X>Show Long Names for Files option is selected. |

Business Process Logic Metrics

|  |  |
| --- | --- |
|  | The Business Process Logic Metrics option is only available for the users who have the X-RPG migrate license. This option is hidden for the users who do not have the X-RPG migrate license. |

The Business Process Logic Metrics report displays the migrated logic data in metrics form – Total Lines, Excluded Lines, Controller Lines, and Process Logic. The Business Process Logic Metrics option is displayed below:

|  |
| --- |
| Fig. 4.1.1 – Audit Options – Business Process Logic Metrics option |

The following screen displays the Business Process Logic Audit Report under the cross-reference node XAN4CDXA:

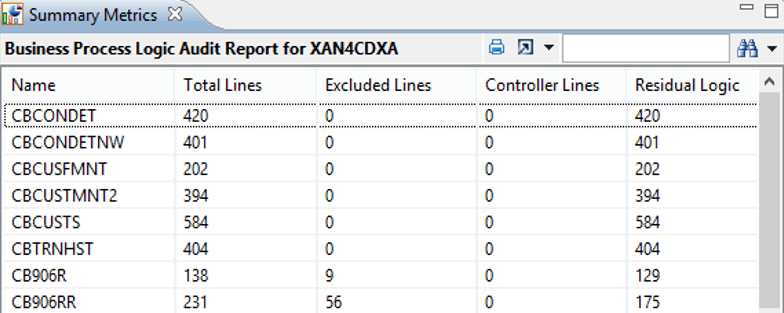


Fig. 4.1.2 – Business Process Logic Audit Report for XAN4CDXA

Specialized Analysis

Measuring and managing the quality and complexity of a codebase is systems analyst’s biggest priority. Specialized Analysis is an invaluable feature that provide users a complete freedom to create customized reports using the full metrics database in the defined system repository. One can choose to view the report as a Static Report or a Difference Report. The latter plots changes in values over a period, thus giving a better idea of measurement.

Select the Specialized Analysis option from the Audit Options submenu.

|  |
| --- |
| Fig. 5.1.1 – Audit Options – Specialized Analysis option |

From version 13.2.10 Metrics Dashboard is now updated to display Special Reports as Specialized Analysis.

Refer to the screen below displaying the Metrics Dashboard.

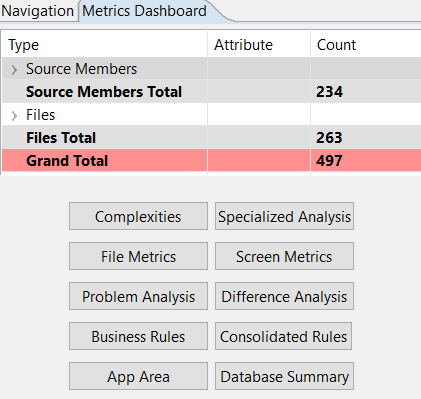


Fig. 5.1.2 – Metrics Dashboard

The following screen displays the Specialized Analysis window:

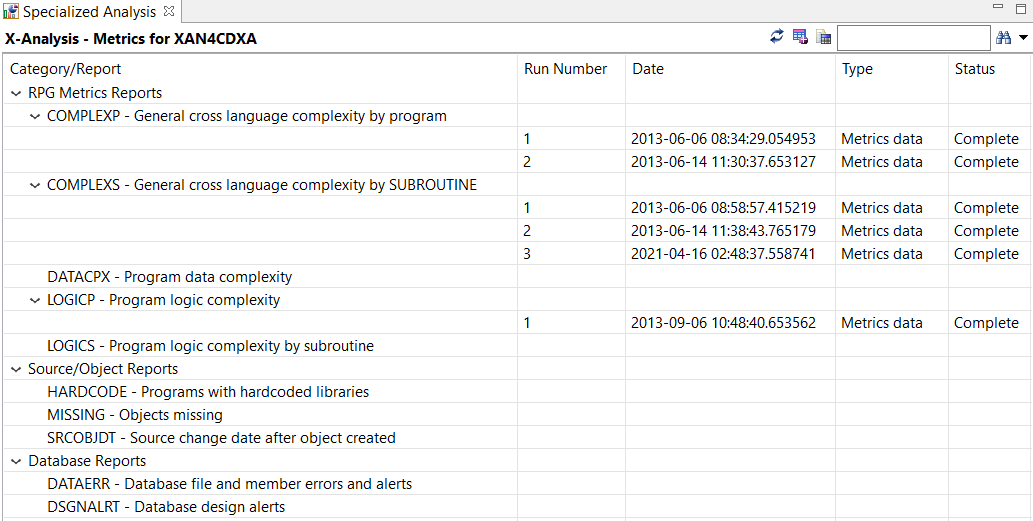


Fig. 5.1.3 – Specialized Analysis window for XAN4CDXA

The user has the option of generating the pre-configured reports. Select any report from the listed category and right-click to view the context menu, as shown below:

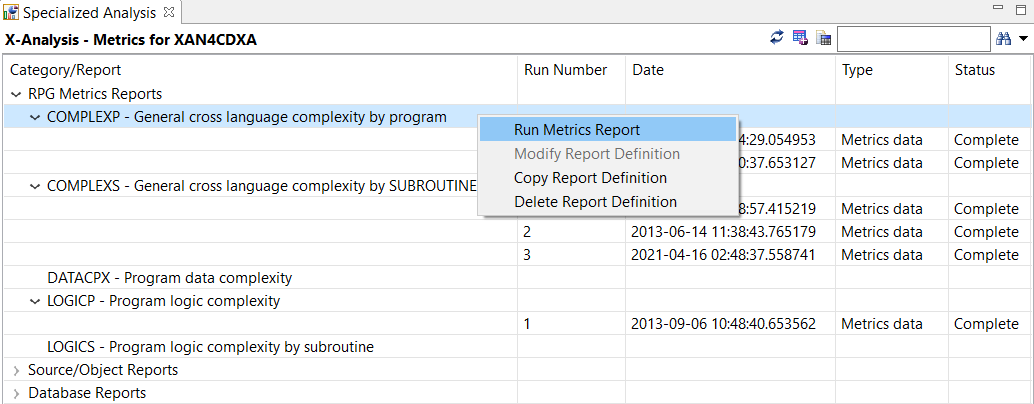


Fig. 5.1.4 – Context Menu on Report Definition

Options On Report

Right-click on a selected report to invoke a context menu listing four Report options:

* Run Metrics Report
* Modify Report Definition
* Copy Report Definition
* Delete Report Definition

Choose any of these options by clicking it.

**Run Metrics Report**

Selecting this option invokes the pre-configured Report Definition dialog box:

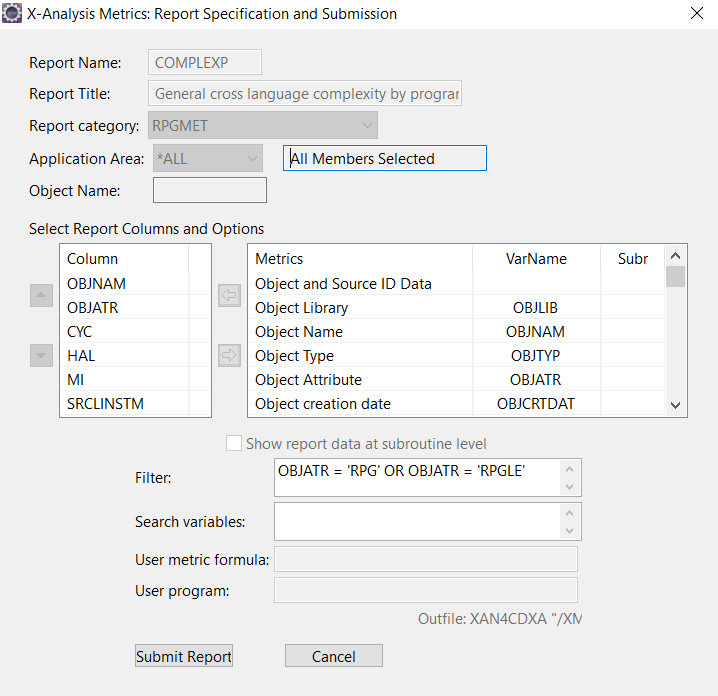


Fig. 5.2.1 – Report Definition dialog box

Click Submit Report to generate the report. A batch job is submitted and, once completed, the Specialized Analysis window updates itself:

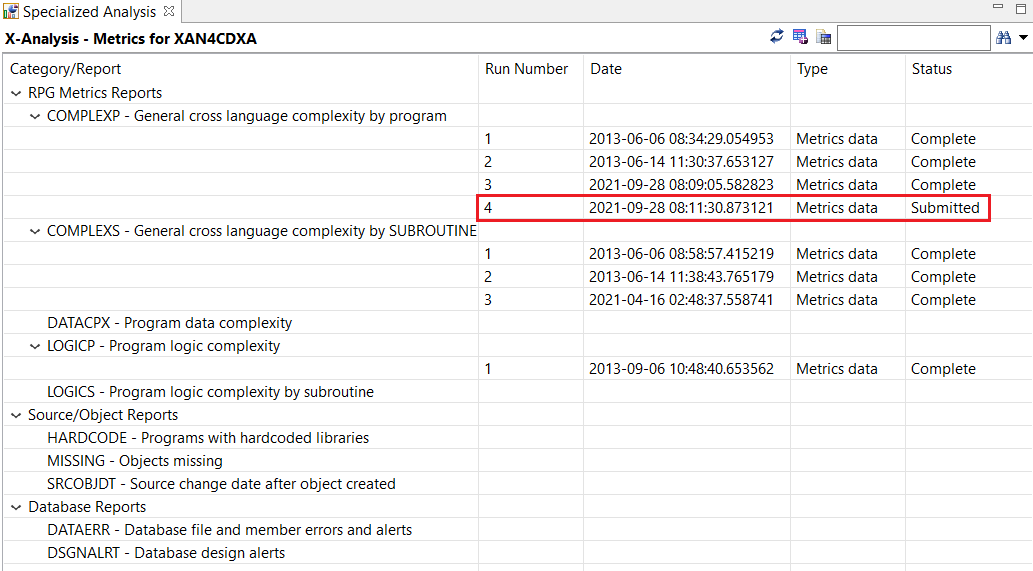


Fig. 5.2.2 – Specialized Analysis window

To view the generated report, select the report’s row, right-click on it and choose View Report from the context menu. The user can also delete the report by selecting the Delete Report option from the context menu.

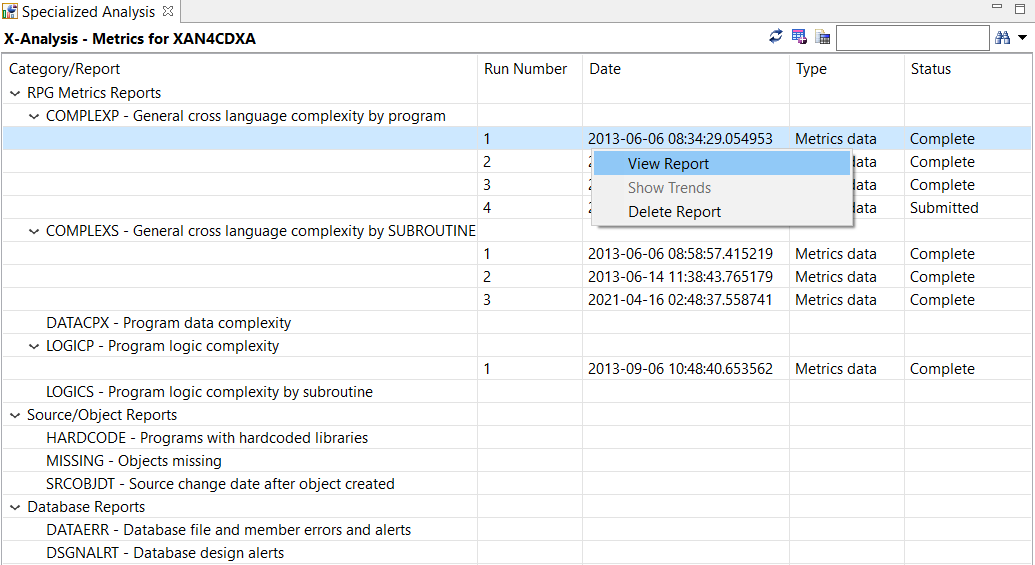


Fig. 5.2.3 – Updated Specialized Analysis window

Refer to the below report displayed below:

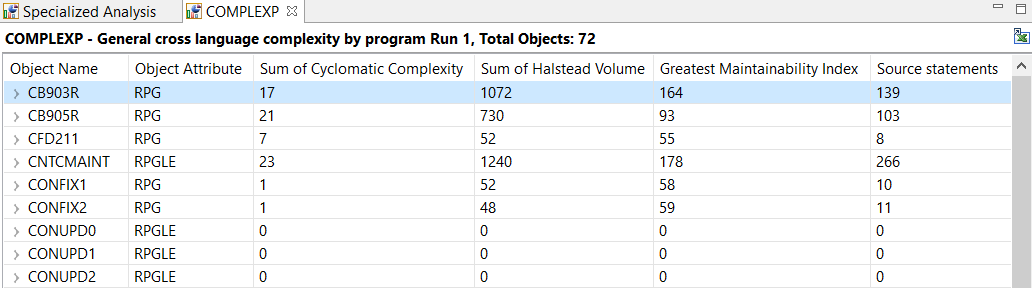


Fig. 5.2.4 – COMPLEXP Report

If required, export this report to MS Excel.

**Modify Report Definition**

|  |  |
| --- | --- |
|  | If any instance user has generated the report, then Modify Report Definition option is not allowed. For example, if a report has been run once, the user cannot modify that report definition. |

The user has the option to modify an existing Report Definition.

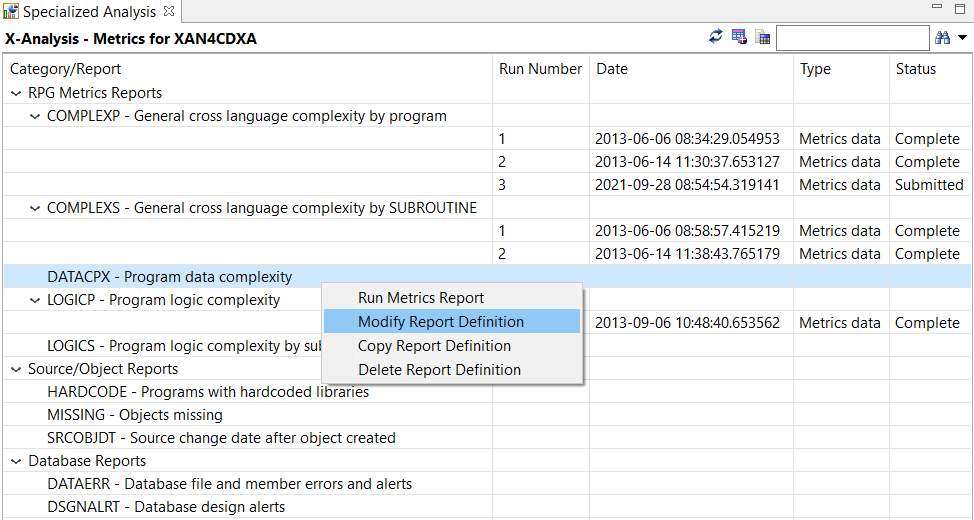


Fig. 5.2.5 – Modify Report Definition option

Select the Modify Report Definition option to invoke the report configuration dialog:

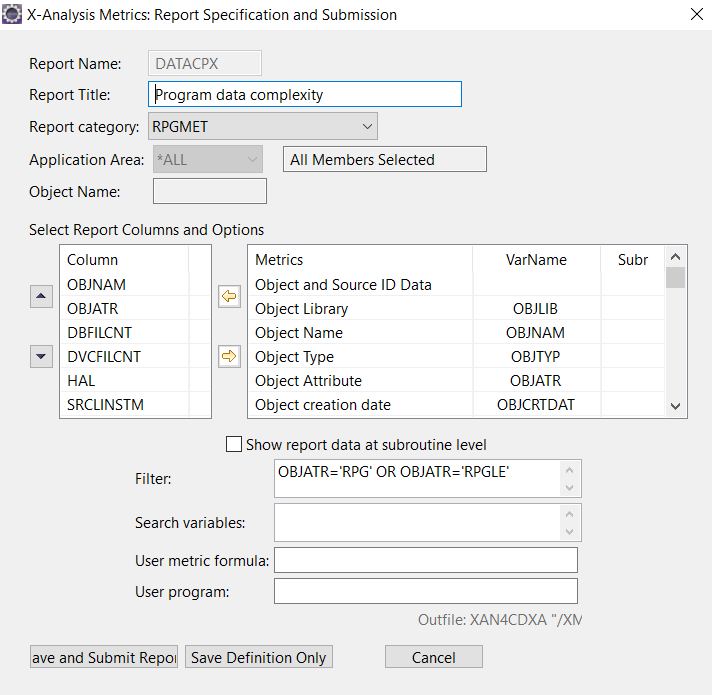


Fig. 5.2.6 – Report Definition dialog

After changing the report definition, click Save Definition Only.

To generate a report, click Save and Submit Report.

|  |  |
| --- | --- |
|  | Only those fields can be used in the filter which have been selected in Select Report Columns and Options category. |

**Copy Report Definition**

Copy an existing Report Definition by using this option. On selecting this option, the user can retain an existing report besides the customized report.

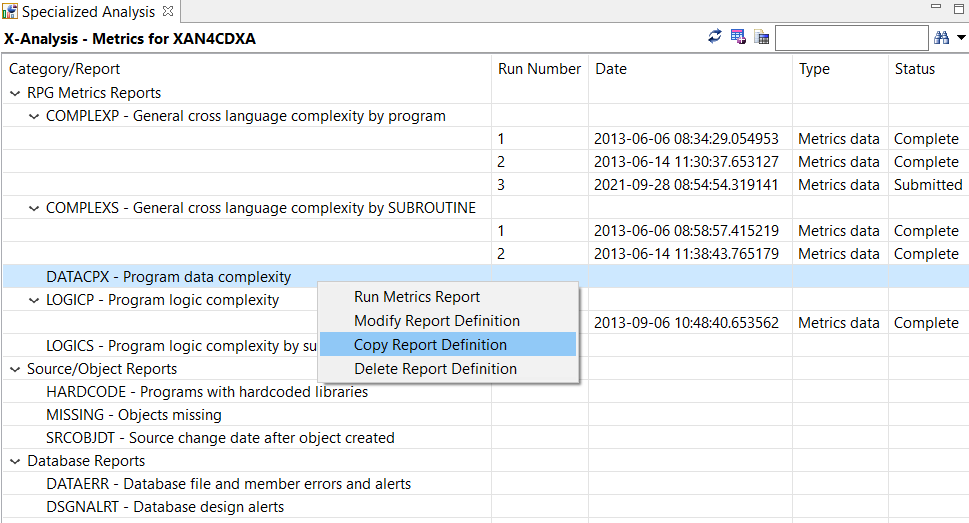


Fig. 5.2.7 – Copy Report Definition option

**Delete Report Definition**

If required, delete an existing Report Definition by selecting this option.

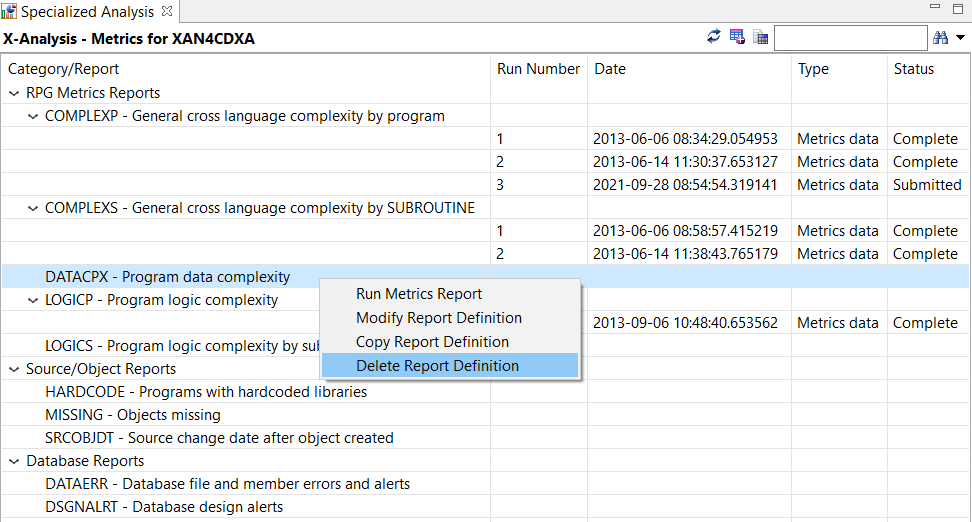


Fig. 5.2.8 – Delete Report Definition

Specialized Analysis Toolbar

**Refresh Metrics**

The Refresh Metrics icon performs the refresh function.

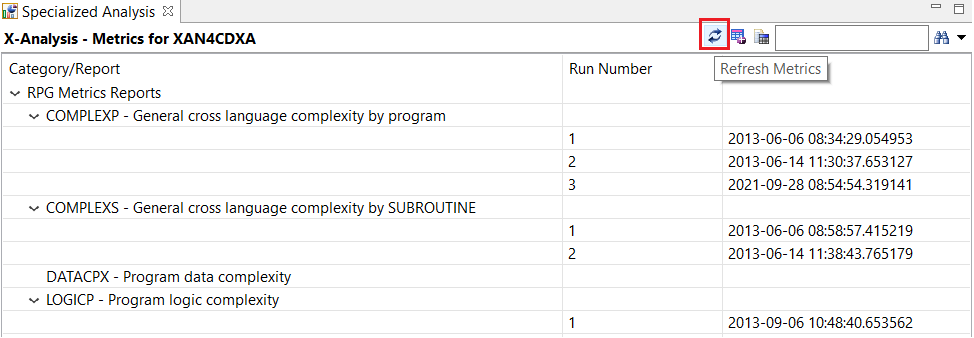


Fig. 5.3.1 – Refresh Metrics icon

**Create New Report**

If required, use the Create New Report icon for creating a customized report.

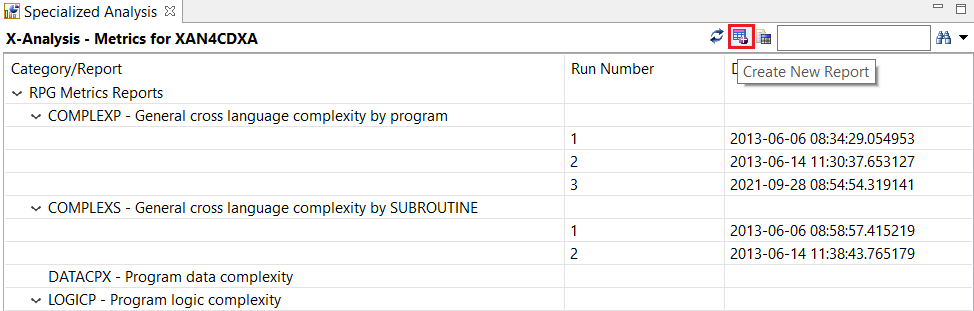


Fig. 5.3.2 – Create New Report icon

Clicking the Create New Report icon invokes the report definition dialog box.

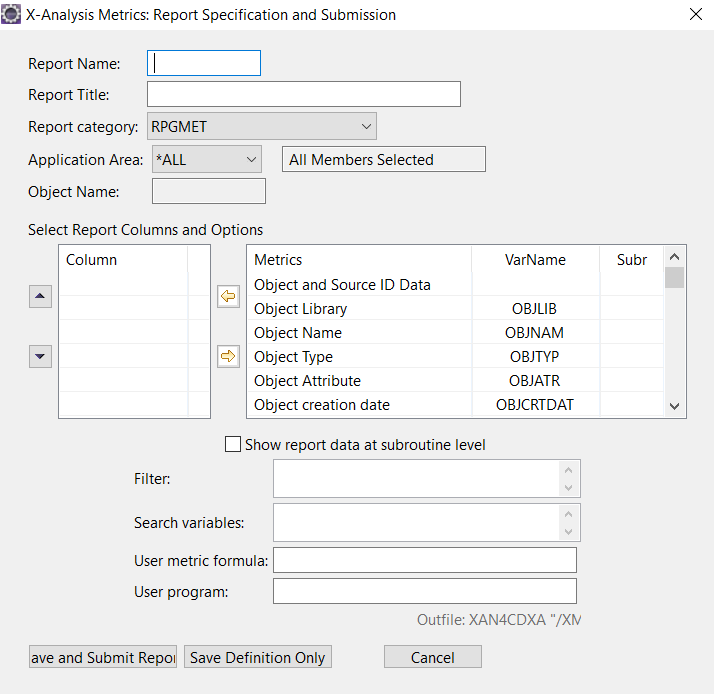


Fig. 5.3.3 – Create New Report dialog box

***Select Report Columns and Options***

Use this list to select which column the user want in their report.

|  |  |
| --- | --- |
| **Options** | **Description** |
| Column | This indicate that the metrics should appear on the user's report and what its sequence should be, left to right, in the column across the top of the report. |
| Metrics | Description of the metrics. For detailed descriptions of each metrics, click here. |
| VarName | A variable name to use when referring to this metrics in the Filter or User Metrics Formula. |
| Subr | A "Y" indicates that the metrics is reportable at the subroutine level. |

***Show report data at subroutine level***

If a user checks this box, then the report will show metrics for each subroutine in all programs. In other words, there will be one row for each subroutine in all programs. If the user leave it unchecked then there will be one row per program, and subroutines will not be shown.

|  |
| --- |
| Filter  Use this field to select which metrics records will appear in the report. The syntax is same as that of a DB2 SQL WHERE clause, without entering WHERE itself. Use the variable names from the metrics list in the VarName column.  For example, to select only RPGLE programs with GOTOs, enter this: OBJATR = 'RPGLE' AND GOTCNT > 0  The LIKE operator may be employed in the filter by using standard SQL wildcard characters: OBJNAM LIKE ‘OE%’ OBJNAM LIKE 'CUSN%'  The IN operator may be employed in the filter by using standard SQL wildcard characters: OBJTYP IN ('PGM', 'FILE')  The <> operator may be employed in the filter by using standard SQL wildcard characters: OBJATR <> 'RPG'  The OR operator may be employed in the filter by using standard SQL wildcard characters: DBFILCNT > 5 OR DBFILCNT < 1 |

**Search variables**

Here, the user can specify up to three field names from their application software. Each field name should be on a separate line.

On the new report, the column(s) get created in the field name(s) specified. The data results will show the field(s) usage count in that object. The user can also use these fields in the ‘Filter’ box above to restrict the results based on the number of occurrences of the specified field(s).

***User metrics formula***

Use this field to create custom combinations of metrics. To use this, a user needs to do the following steps:

1. Select the metrics USRFORM (User defined formula) in the column list, by giving it a column number for the report.
2. Enter a formula.

For example, to combine the number of database files with the number of device files, the user could enter the following:  
DVFILCNT + DBFILCNT

For this to appear in the report, the user also has to designate the USRFORM metrics as a column somewhere in the report.

The user's program should be updated in some way the file of the name and library shown below, "Outfile".

***Outfile***

This is the name of the library and file that will contain user’s report data. It is the same as the name of the user's report with a prefix of "XM". For example - XAN4CDXA/XMCOMPLEXP.

|  |  |
| --- | --- |
|  | The Use Metric Formula option allows the user to describe an arithmetic expression on the numeric columns selected for the report. This is shown as an additional column (USRFORMULA) on the generated report. The columns which are not selected before the report creation, shall not be displayed.  The User Program field is not used for now and can be blank. |

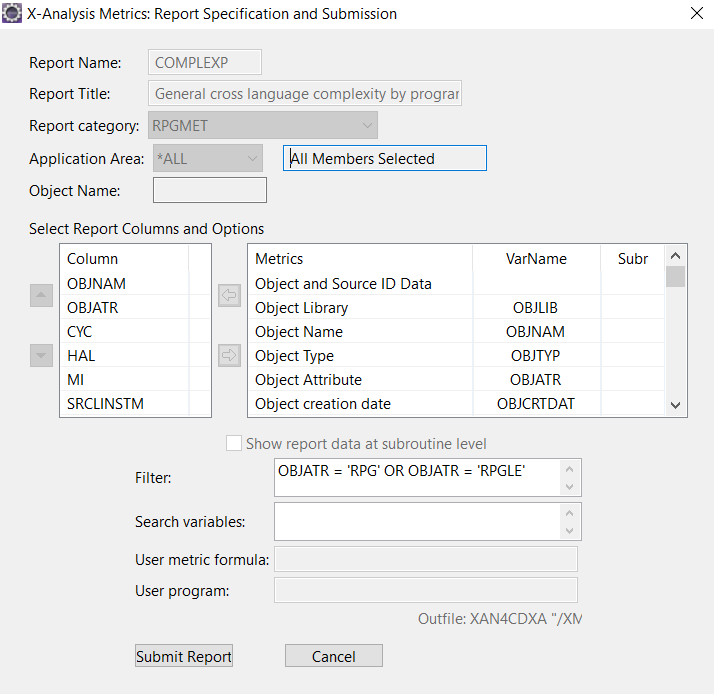


Fig. 5.3.4.A – Specialized Analysis

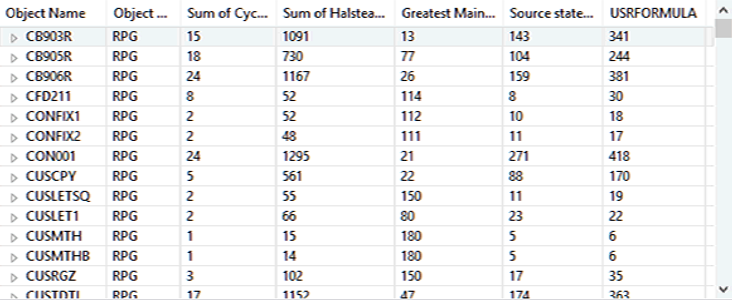


Fig. 5.3.4.B – Specialized Analysis

**View Log**

The View Log icon shows the log related to the Metrics processing.

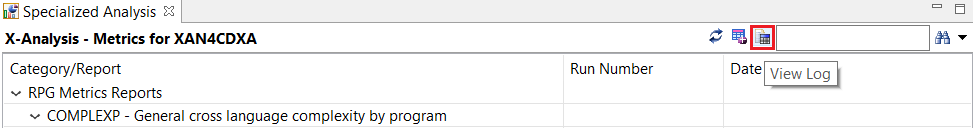


Fig. 5.3.5 – View Log icon

Click the View Log icon to check the log.

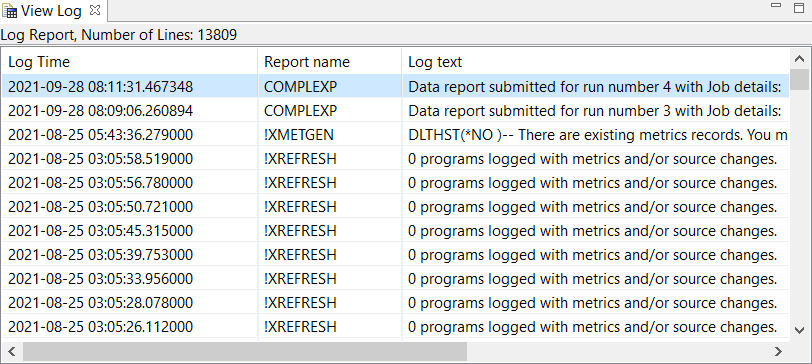


Fig. 5.3.6 – View Log window

Problem Analysis

The quality of analytical procedures improves, if there is prior information of the existent inaccuracies in the data. Issues like object with no source and vice-versa, or source change after the object creation date, or duplicate logical files can hamper advanced analyses.

The Problem Analysis feature warns the user about any such discrepancies in the data. This feature has an expandable Problem Category section, which gives names of the individual files having a particular problem.

To display the Problem Analysis data, right-click on the context menu on the cross-reference node XAN4CDXA and select the Problem Analysis option from Audit Options.

|  |
| --- |
| Fig. 6.1.1 – Audit Options – Problem Analysis option |

The following window will be displayed:

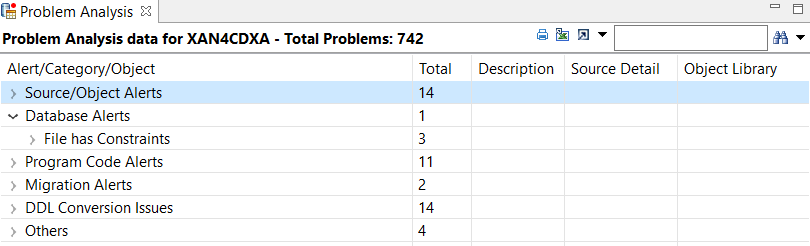


Fig. 6.1.2 – Problem Analysis for XAN4CDXA

Expand the ‘Problem Category’ section to view the objects having problems.

|  |  |
| --- | --- |
|  | In Problem Analysis, the Receiver is too small to hold the result and Divide by Zero categories, display the details in the Additional Details column. Additionally, the Zoom Source points to the respective line within the source editor. |

There is an option to allow Customized Export to Excel for the Problem Analysis data.

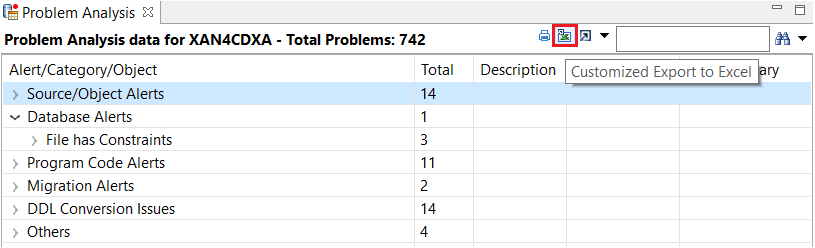


Fig. 6.1.3 – Customized Export to Excel option

Clicking on this icon will invoke the following dialog box:

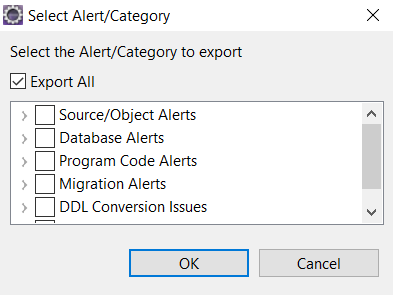


Fig. 6.1.4 – Select Alert/Category dialog box

The user can individually select Categories to be exported to Excel. By default, the Export All box is checked.

The Problem Analysis window also displays the Duplicate Logical Files category. If there is one such record, it indicates a pair of matching LFs. If there are more matches such as these, it means there are as many records. The Referenced By column indicates the PF for the relevant LF.

The Duplicate LF category is shown in the sample screen below:

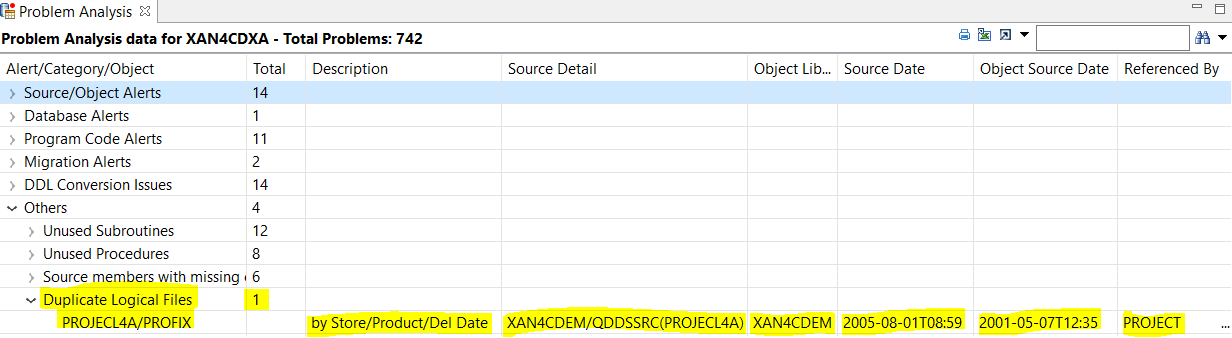


Fig. 6.1.5 – Problem Analysis window – Duplicate Logical Files category

Object Allocation

The Object Allocation window displays information about all the objects, along with the application area names to which they belong. The following screen displays the Object Allocation option:

|  |
| --- |
| Fig. 7.1.1 – Audit Options – Object Allocation option |

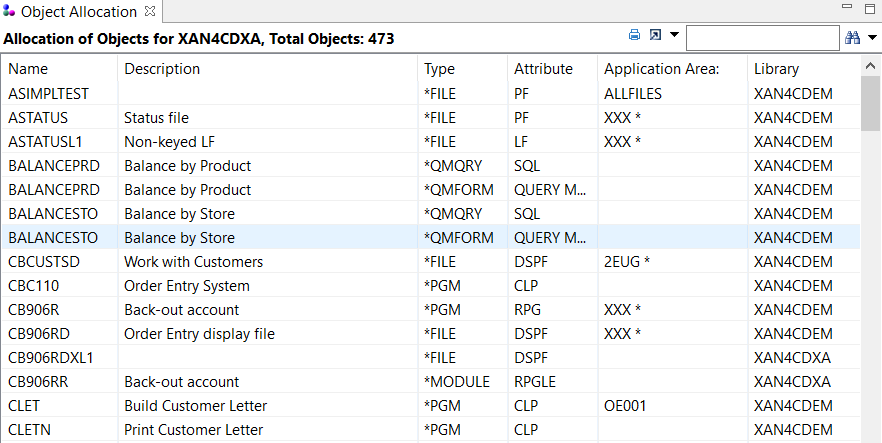


Fig. 7.1.2 – Object Allocation for XAN4CDXA

The above screen lists all objects from the XAN4CDXA application library and provides information about application areas. Note that some rows are blank under the application area column, which means that the object does not belong to any application area. Similarly, note the ‘\*’ sign, which means that the object belongs to multiple application areas.

Database Summary

The Database Summary option gives access to the summarized database report for the entire cross-reference library.

|  |
| --- |
| Fig. 8.1.1 – Audit Options – Database Summary option |

The report contains information related to the files, their unique keys, and other necessary file-related details. When the user clicks the Database Summary option, the following window get displayed.

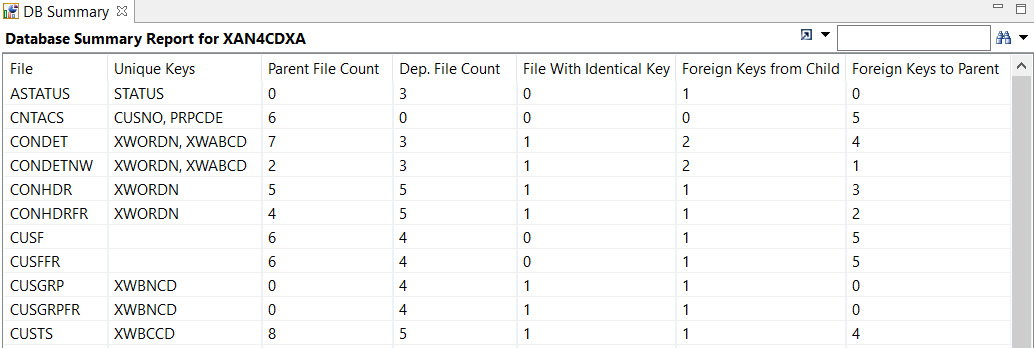


Fig. 8.1.2 – Database Summary Report window

|  |  |
| --- | --- |
| **Column** | **Description** |
| Unique Keys | List of unique keys for the file. |
| Parent File Count | The no. of files that are related to current file as a parent file. |
| Dependent File Count | The no. of files that are related to current file as a dependent file. |
| File With Identical Key | This denotes the no. of files having same set of keys as the current file. |
| Foreign Keys to Parent | This represents the no. of file which have foreign key for that file. |
| Foreign Keys From Child | The no of file where one of the file key is defined as a foreign key. |

|  |  |
| --- | --- |
|  | Parent File count and Dependent File Count can be illustrated on Data Model Diagram of that specific file. |

Summary Report

The Summary Report option generates a structured Metrics Analysis and Problem Analysis report as a PDF / MS Word document.

Select the Summary Report option from the Audit Options sub-menu.

|  |
| --- |
| Fig. 9.1.1 – Audit Options – Summary Report option |

Audit Report

Selecting this option invokes the following Documentation Wizard:

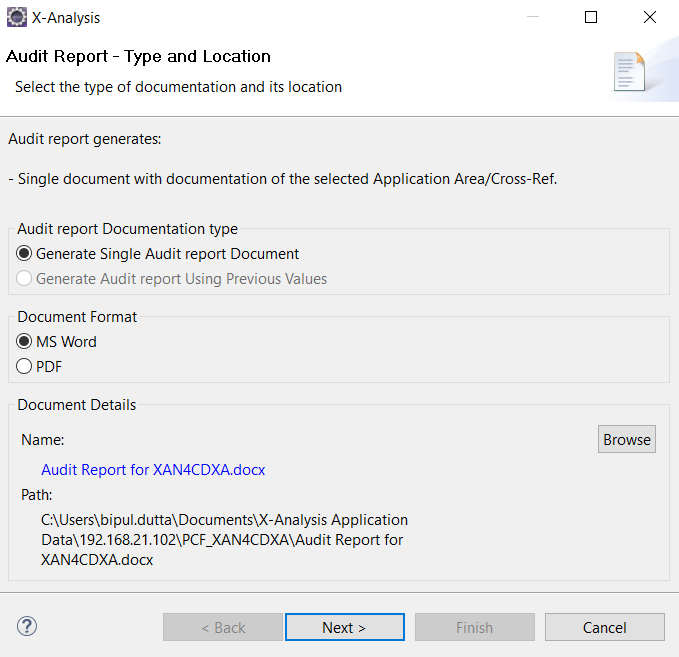


Fig. 9.1.2 – Summary Report – Type and Location

Specify the documentation type and path and click Next.

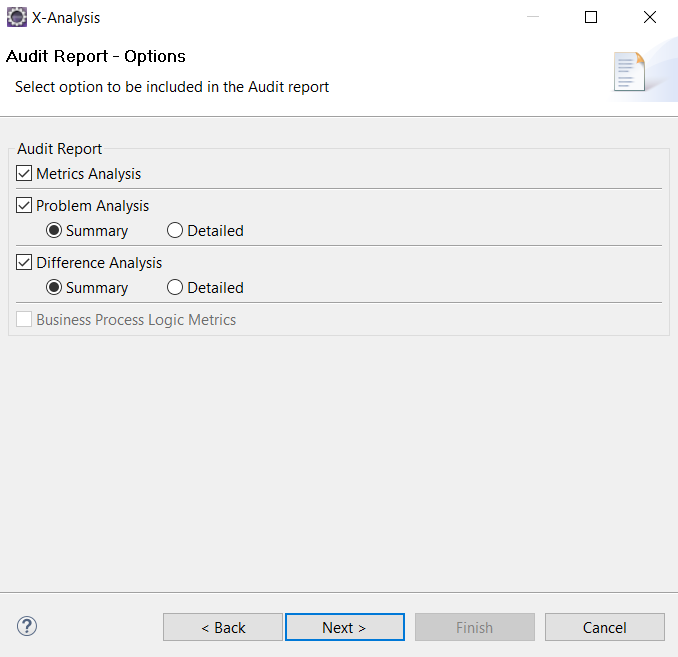


Fig. 9.1.3 – Summary Report – Options

Select the options to be included in the Audit Report and click Next.

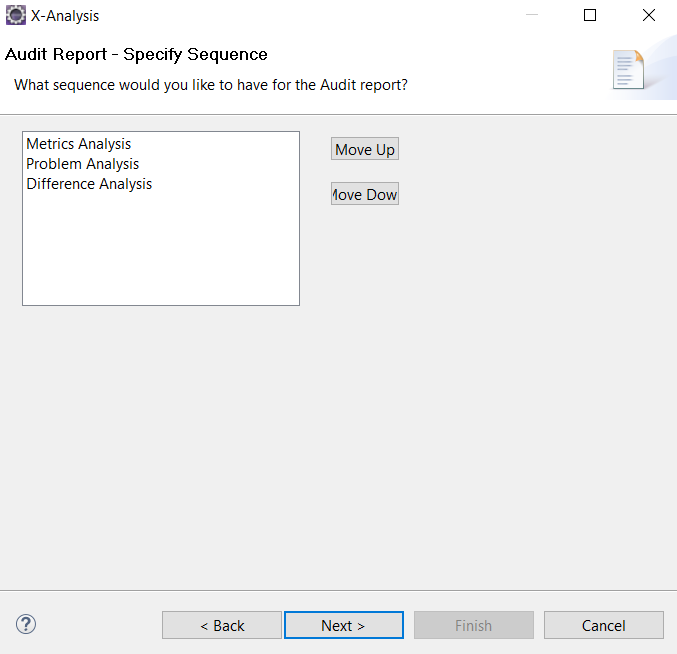


Fig. 9.1.4 – Summary Report – Specify Sequence

Specify the sequence of the contents for the Audit Report and click Next.

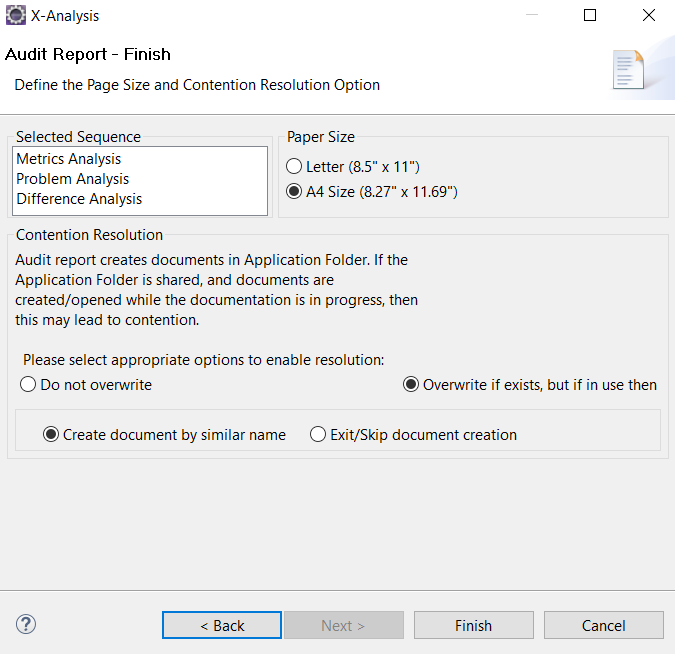


Fig. 9.1.5 – Summary Report – Finish

Choose the required settings for the Audit Report. Click Finish to end the report configuration and generate the report. When the report generation is over, the following dialog box appears:

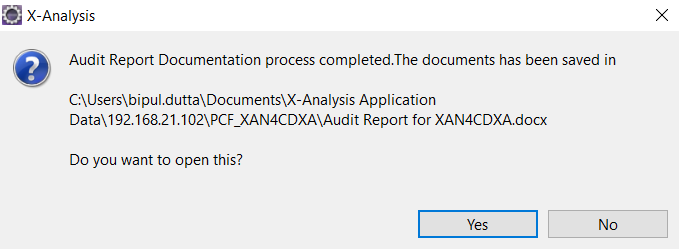


Fig. 9.1.6 – X-Analysis dialog box to open the generated report

Initialize Source Archiving

The Initialize Source Archiving option is available for the cross-reference library and the application areas. The user can run the option independently of metrics.

|  |
| --- |
| Fig. 10.1.1 – Audit Options – Initialize Source Archiving option |

The following dialog box is displayed on selecting this option:

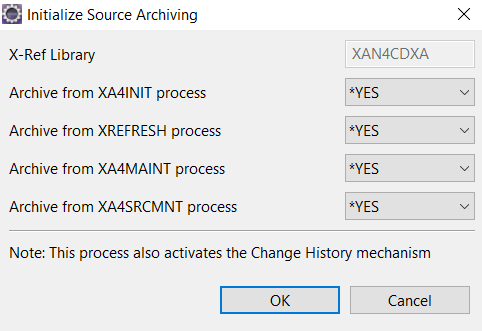


Fig. 10.1.2 – Initialize Source Archiving dialog box

|  |  |
| --- | --- |
|  | The user can activate the Source archiving using the XACVINI command. It sets up a controlling \*DTAARA in the X-Analysis library and makes an initial copy of all the source members.  If the user deactivates the archiving and then reactivates it later, the initial archived copy of the source will not be preserved.  The user can deactivate the Source archiving using the XACVEND command. This optionally clears all the source archives and controlling data. This command runs only on the server. |

**Files**

All archived source versions are recorded in XAACVSRC, whose data is only ever purged when XACVEND DLTHISTO(\*YES) is run (whereas metrics can be purged independently, which clears XMETOBJ). When archiving is active, the date and time of the latest archived source version are recorded in XAACVHDR. Archiving handles all the source types (whereas metrics only records for CL, RPG, and CBL).

Source archiving continues to maintain data in XMETOBJ/XMETOBJH, if necessary. The data in XMETOBJ/XMETOBJH is not necessary to the source archiving process itself but is used by the PC client to locate the change history. If data is written to XMETOBJ by source archiving, the SRCACT field is set to the value 'A'.

**Processing**

Source archiving is invoked from XA4INIT, XREFRESH, XA4MAINT, and XA4SRCMNT, which all call the wrapper program, XRACVMBR.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Commands XA4MAINT and XA4SRCMNT give the same results as the XREFRESH command as they are being called from XREFRESH. But both the commands XA4SRCMNT and XA4MAINT have different functionality.   * XA4MAINT: Command performs the source and object indexing of the object which has been added or changed. * XA4SRCMNT: Command performs the indexing only for the source member which has been added or changed.  |  |  | | --- | --- | |  | Parameters inside the XREFRESH command, that is "Refresh Application Areas" and "Refresh Business Rules" cannot be run from XA4SRCMNT and XA4MAINT commands. |  |  |  | | --- | --- | |  | Commands XA4MAINT and XA4SRCMNT do not support all of the REFRESH features. For this reason, their usage should be complemented by some regular REFRESH executions. | |

The archive processing itself is done in \*srvpgm XACV, in procedure XACV\_ArchiveMember(). This procedure checks the source date and time update against the date and time recorded in XAACVHDR. If there is a difference, archiving takes place that is copy the source, write it out to XAACVSRC, update XAACVHDR, and write it out to XMETOBJ/XMETOBJH, if the appropriate data is not already present in these files.

The wrapper command/program XACVMBR/XRACVMBR first checks whether metrics is active, then does either of the following:

1. If metrics are active, it writes out a record to XMETCHGS, which will cause metrics processing to take place when XRMETCHGS is called, which is the case in XREFRESH, XA4INIT, and XA4SRCMNT.
2. If metrics are not active, it calls XACV\_ArchiveMember() procedure.

If metrics are active, then the source archiving call is made from metrics processing, which will call XACV\_ArchiveMember(). This is so that the metrics will write the appropriate XMETOBJ/XMETOBJH records before source archive processing takes place.

**Purge**

A purge process (XACVPRG command) will move the source to a user-specified library and record the location in XAACVSRC. Purge is by the cut-off date, as compared to the archive timestamp in XAACVSRC. The purge process always leaves one source version in place, even if the timestamp is older than the cut-off date.

|  |  |
| --- | --- |
|  | All the Archiving Source Physical Files will be permanently deleted from X-Reference library. Earlier, these archiving source physical files were getting renamed. |

Generate Metrics and Problem Analysis

Starting from version 13.3.02, a new option named Generate Metrics and Problem Analysis is added to the Audit Options to replace the previously existing options:

* Generate Metrics Analysis and
* Generate Problem Analysis.

|  |  |
| --- | --- |
|  | The order of execution is that first, the Metrics Analysis gets generated, and then the Problem Analysis, which was actually how the former option Generate Metrics Analysis, was behaving. |

The Generate Metrics and Problem Analysis option is shown below:

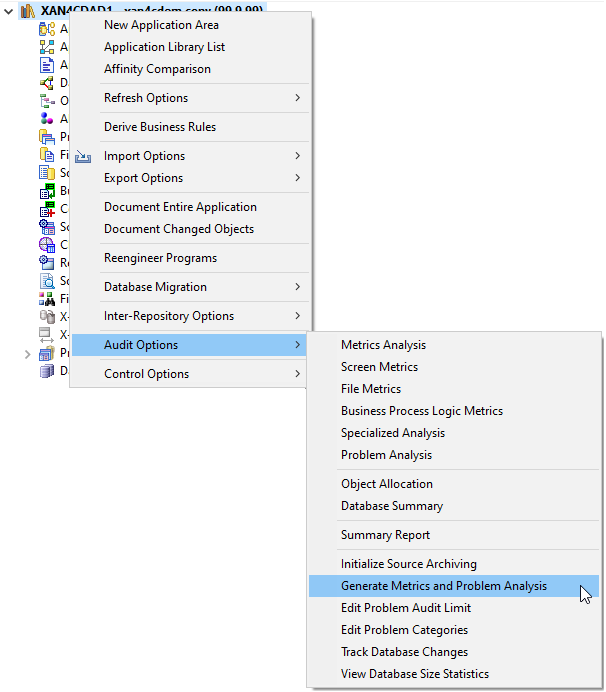


Fig. 11.1.1 – Audit Options – Generate Metrics and Problem Analysis

When calling the Generate Metrics and Problem Analysis, a new dialog gets opened to allow the user to confirm the operation regarding removing or keeping the existing analysis history.

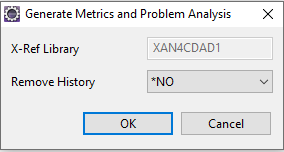


Fig. 11.1.2 – Generate Metrics and Problem Analysis dialog box

Click OK to invoke the batch job command.

Edit Problem Audit Limit

Edit Problem Audit Limit allows the user to decide and set the Problem Audit limit as per requirement. The option is shown below:

|  |
| --- |
| Fig. 12.1.1 – Audit Options – Edit Problem Audit Limit |

Clicking the option invokes the following dialog box.

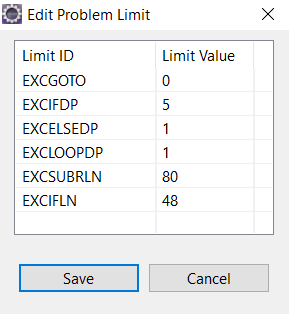


Fig. 12.1.2 – Edit Problem Limit dialog box

The Limit Values given in the above dialog box are user configurable. The values shown are set by default. Assign a new value against the required problem analysis category listed under the first column and click Save to apply the changes. The changed values will get reflected in the related table on the server-side. The subsequent Problem Analysis generation will consider the updated values as the limits to prepare the list of the programs falling under that criteria.

Edit Problem Categories

The user can add a new category or edit/delete a previously existing category by selecting the Edit Problem Categories option. The user can make such modifications for the main category, that is the Problem Category group and the sub-categories. The option is displayed below:

|  |
| --- |
| Fig. 13.1.1 – Audit Options – Edit Problem Categories |

Select the option to invoke the following dialog:

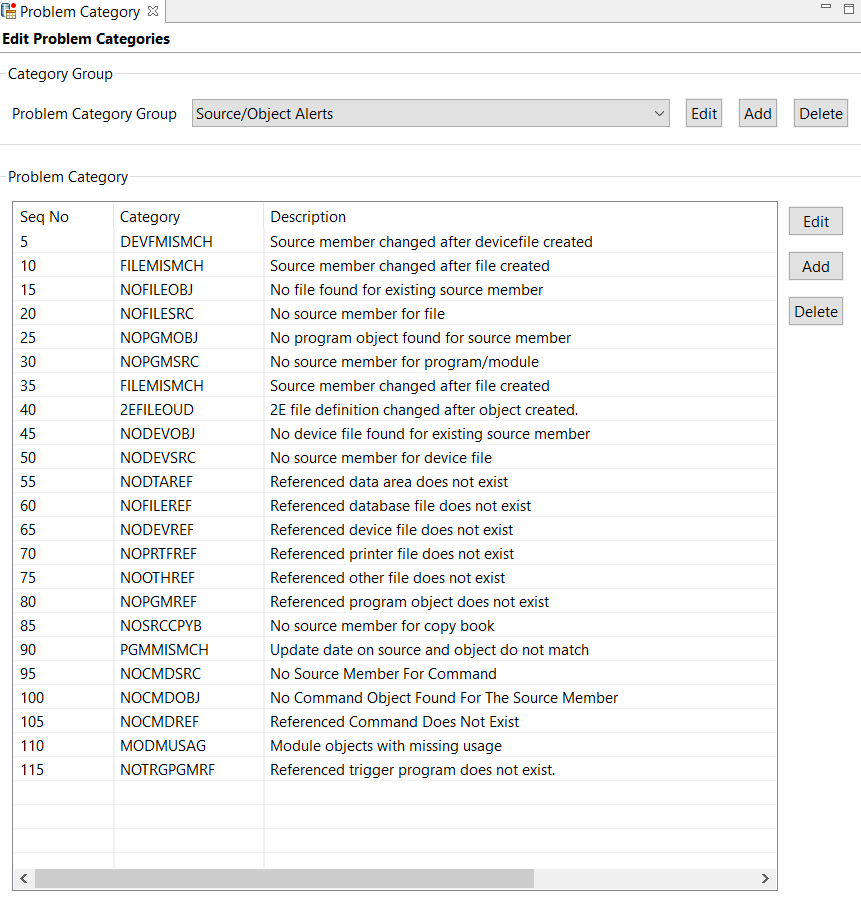


Fig. 13.1.2 – Edit Problem Categories window

Use the Edit, Add, or Delete buttons to make changes to the Problem Category Group. Click Edit to invoke the following dialog box:

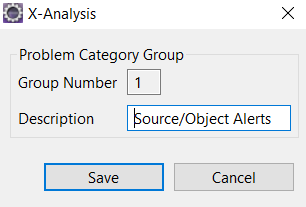


Fig. 13.1.3 – Edit dialog box – Problem Category Group

Edit the Group Number and/or the Description. Click Save.

Click Add to invoke the following dialog box:

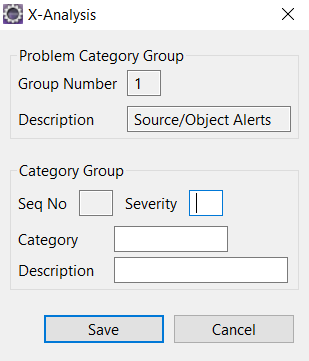


Fig. 13.1.4 – Add dialog box – Problem Category Group

Enter details in the given fields. In the Category Group section, enter the sequence number and the severity scale to be assigned to the new category. The Severity scale indicates the complexity of the problem and is based on the combination of problem report data and additional data. Click Save. The new category will appear in the Edit Problem Categories window.

On selecting the Delete option, a window appears which asks for the user’s confirmation before deleting a specific category.

Similarly, the user can select the Edit and/or Delete options for the displayed sub-categories. By selecting the Add option, the user can add a new category under a pre-defined main category.

Note that the user must make a few changes on the server-side for editing the problem categories on the client-side.

If the user have to add new problem category TRIGGERS with description like "FILE HAS TRIGGERS", follow the steps given below to add problem category in the XPRBCATS file.

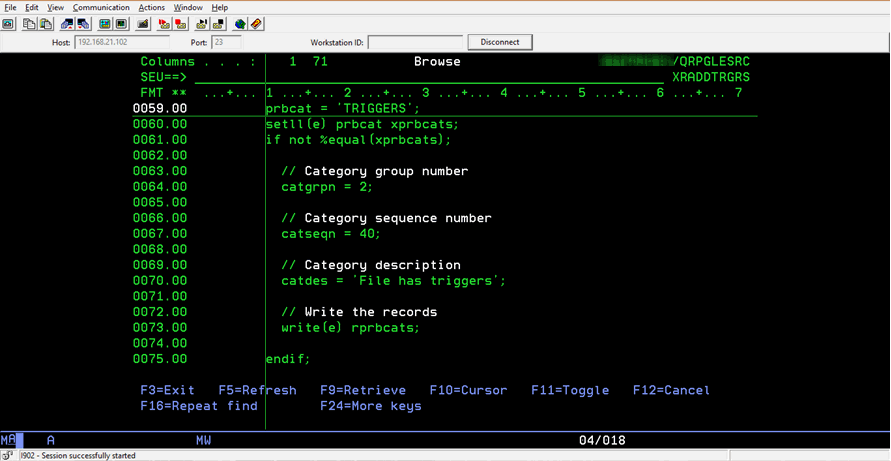


Fig. 13.1.5 – FILE HAS TRIGGERS

The user will get CATGRPN and CATSEQN fields value from X-Analysis while adding new problem category. To write the entry in the XPRBOBJS file for the above category the user have to write their own logic as displayed below:

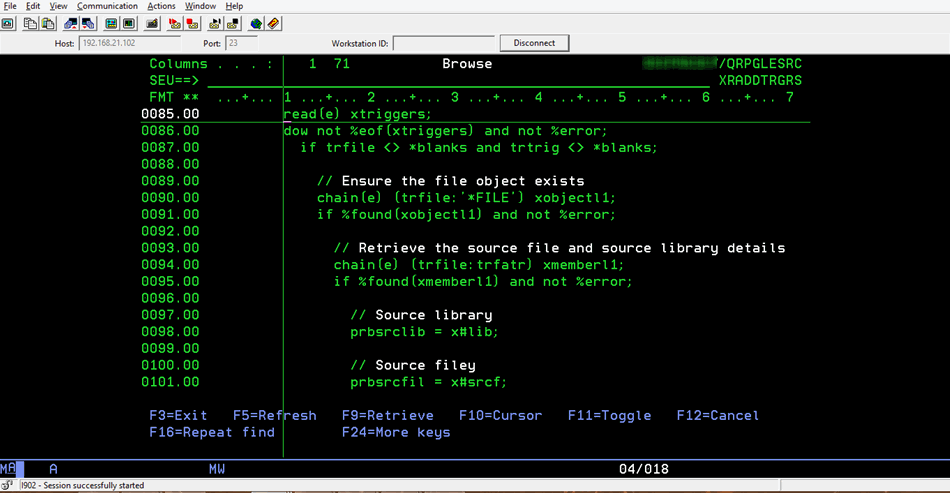


Fig. 13.1.6 – Own Logic

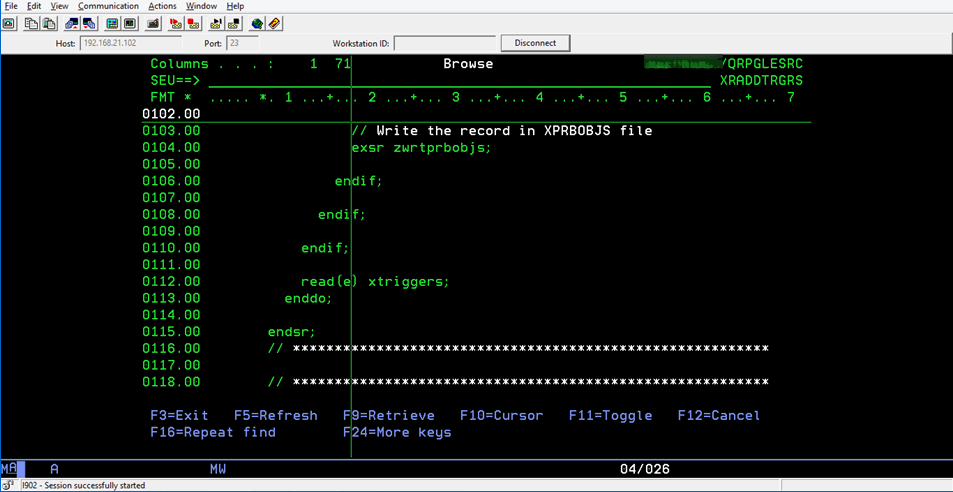


Fig. 13.1.7 – Own Logic

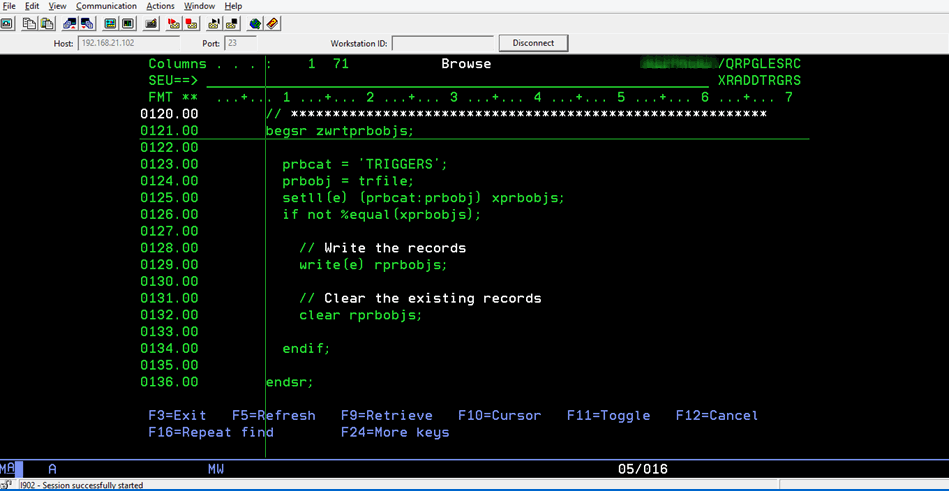


Fig. 13.1.8 – Own Logic

These modifications in the Problem Analysis will be reflected in the Problem Analysis Editor (when taken).

Now, when the user select the Problem Analysis option, the Edited Category can be seen there.

For objects to appear under the new category, a new RPGLE program should be created to identify and populate the objects. Refer [Appendix C - Adding new Problem Analysis categories](#_Ref1919873333) using Service Program for more details.

Track Database Changes

Selecting the Track Database Changes option helps to monitor the changes across all the databases. This editor displays all the PFs and LFs and all the versions of the files. Please note that this option is disabled by default and will get enabled only when the user activates source archiving or generate metrics on the cross-reference library for the first time.

|  |
| --- |
| Fig. 14.1.1 – Audit Options – Track Database Changes option |

Refer to the sample Database Tracking window in the below screen:

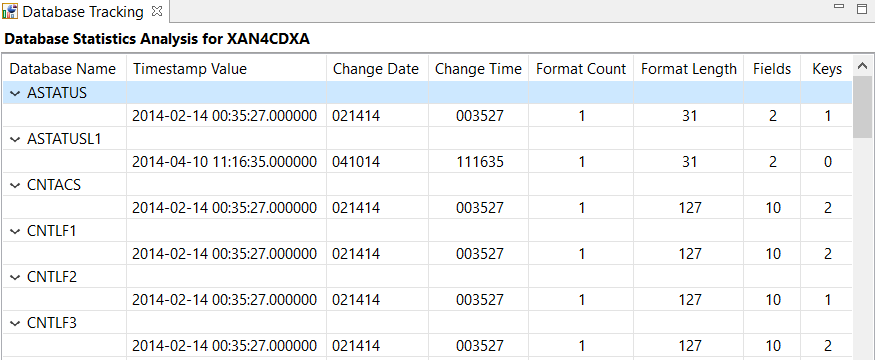


Fig. 14.1.2 – Database Tracking window

This facility to track file changes. This will be allowed when there are more than one instances/version on a file name. To this effect, when the user right-click on a Database name with multiple instances, the user will see the Track File Changes option as shown below:

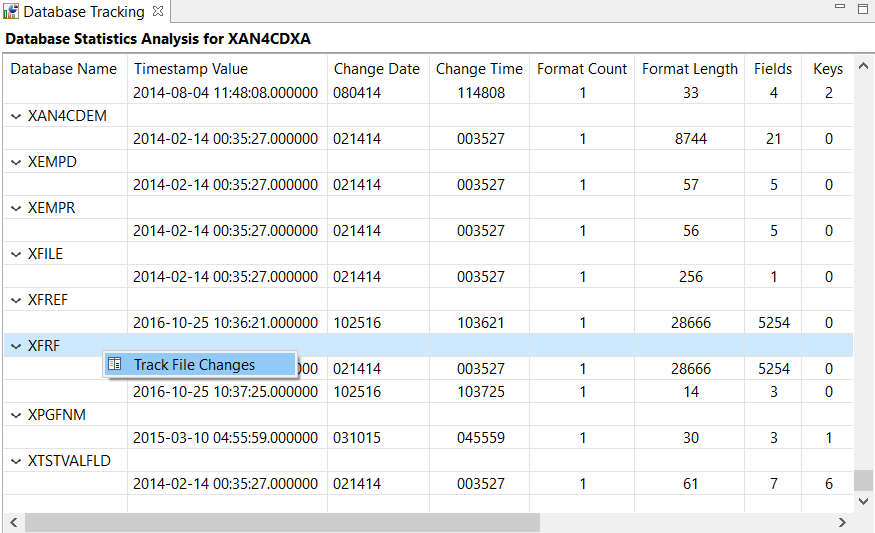


Fig. 14.1.3 – Database Tracking window showing right-click option, Track File Changes

Click the Track File Changes option. The following dialog box will appear:

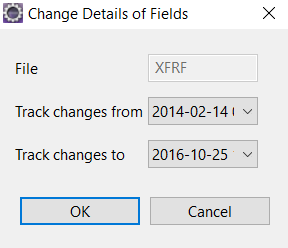


Fig. 14.1.4 – Track File Changes – Change Details of Fields dialog box

As displayed above, this dialog box prompts the user to select the timeline for tracking the changes. Use the drop-down to make the selection and press OK.

The comparison will appear in the lower window. For a sample, refer to the below screen:

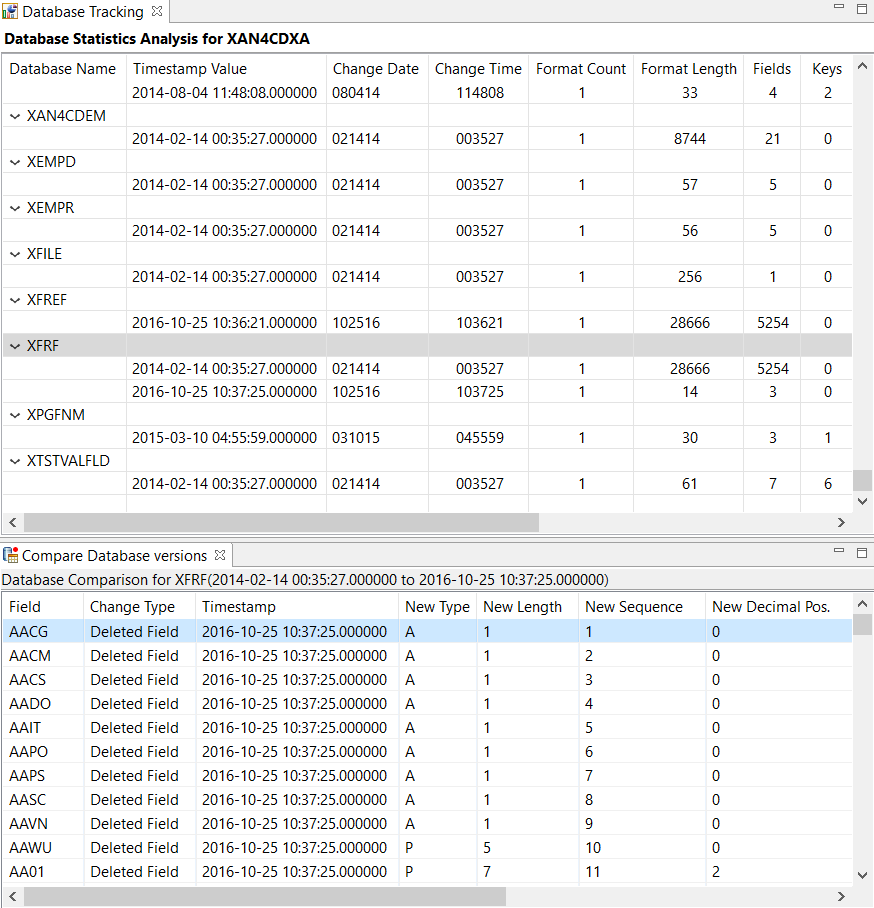


Fig. 14.1.5 – Compare Database versions after selecting the timestamp

The comparison of all the available fields is shown in the view Compare Database versions window. Each modified field will show the changed values for type/length/decimal and the like.

View Database Size Statistics

View Database Size Statistics helps to monitor the growth statistics of a database over a selected period. The current aggregate/total database statistics will be displayed for the database and for individual files on a new editor.

To use this option, execute the below steps:

* Double-click on the Files node for the Object List.
* Select the files required to audit, using the Add to Database Audit option. This option is available only for PF type objects. The following screen shows the option.

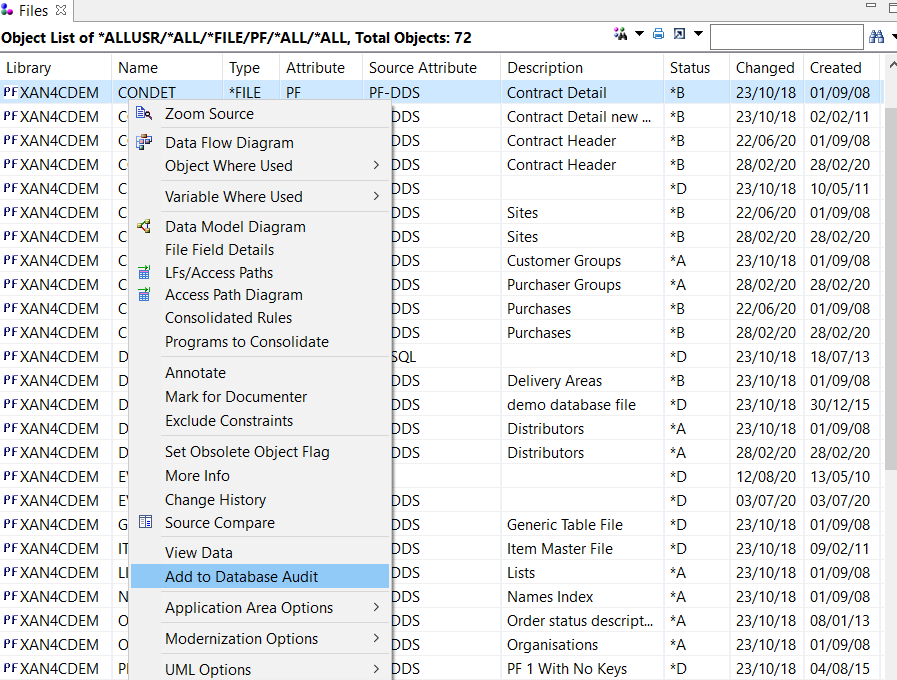


Fig. 15.1.1 – Object List – Add to Database Audit option

After selecting the file, the following window gets displayed, confirming the addition of the file.

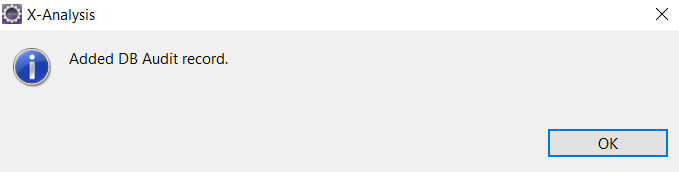


Fig. 15.1.2 – Audit Record

Hereafter, the user has to process the XAOBJXXXXX/XSETUPLOG command from the IBM i screen. The following window display the XSETUPLOG command screen.

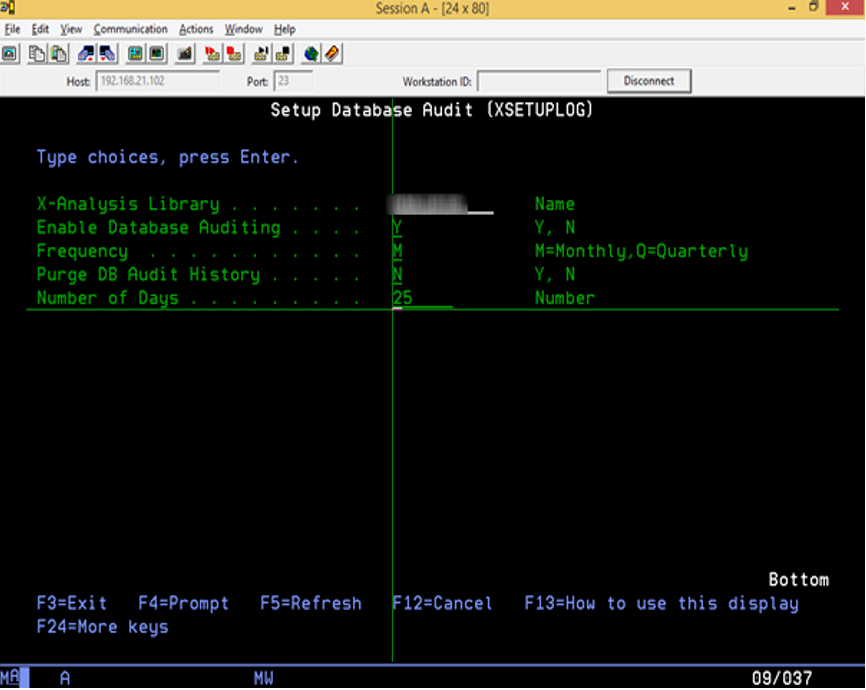


Fig. 15.1.3 – XAOBJXXXXX/XSETUPLOG screen

The Enable Database Auditing is set as N by default. Make sure to change it to Y. Select the other details as required. Press ENTER. Specifying the requirements through this screen will ensure that the selected files are monitored for the database growth from the date they have been added after the user have initialized the X-Ref.

Now select the View Database Size Statistics option from the Audit Options submenu. The database size statistics will be displayed as below:

Enable Database Auditing: This is set as N by default. Make sure to change it to Y.

Frequency: If Initialize/Refresh is performed on the cross-reference library, the XDBAUDLOG captures the last Initialized/Refreshed date of the added files and compares the same with the next Initialization/Refresh date, based on the parameter set as "Monthly/Quarterly". The process gets skipped if the next Initialization/Refresh is run before the set date.

So, if Initialize/Refresh is performed on the cross-reference library for the second time within a month/quarter, then the file XDBAUDLOG does not get populated. The XDBAUDLOG file is populated after the first month or the fourth month from the last initialization date.

Before initializing the cross-reference library for the second time in a month, ensure that the data is same for the XDBAUDLOG file that is File has not been populated.

Purge DB Audit History: If this is set as 'Y' then data will be deleted for before the number of days specified in the Number of Days field.

Number of Days: Suppose a user enters this as 25, then data will be deleted for before 25 days.

|  |  |
| --- | --- |
|  | After executing the XAOBJXXXXX/XSETUPLOG command it is mandatory to perform Refresh/Initialization for changes to take effect.  We can check set up anytime by running the below command: DSPDTAARA <X-Ref Library>/XDBLOGDTA |

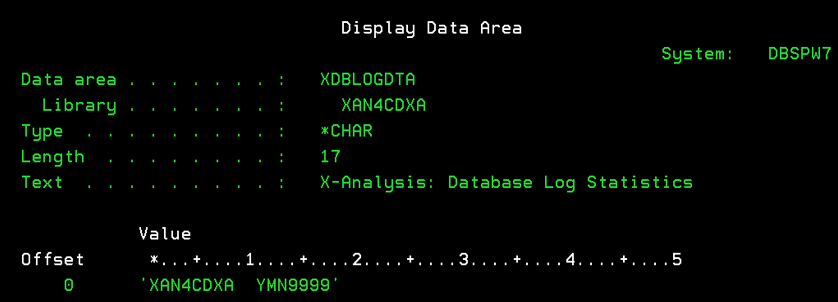


Fig. 15.1.4 – Display Data Area

|  |  |  |
| --- | --- | --- |
| **Position** | **Value** | **Parameter name** |
| 1-10 | cross-ref name | X-Analysis Library |
| 11-11 | Y/N | Enable Database |
| 12-12 | M/Q | Frequency |
| 13-13 | Y/N | Purge DB Audit History |
| 14-17 | Number (1-9999) | Number of Days |

Press ENTER. Specifying the requirements through this screen ensures that the selected files will get monitored for database growth from the date they have been added, after the user initialize the X-Ref.

Now select the View Database Size Statistics option from the Audit Options submenu. The database size statistics will be displayed as below:

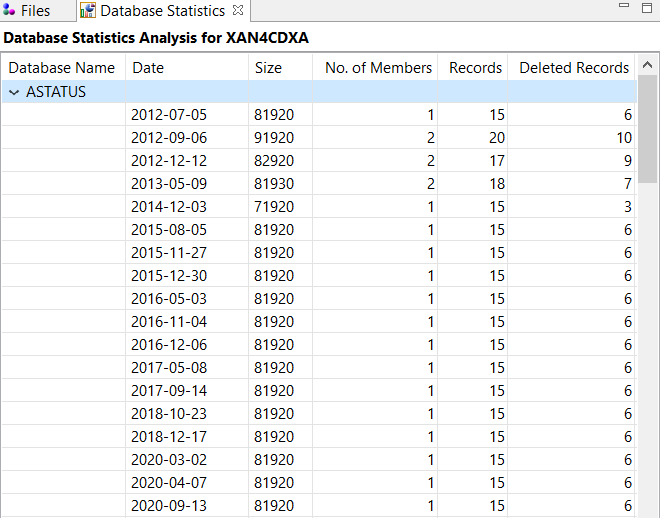


Fig. 15.1.5 – Database Statistics Analysis for XAN4CDXA

Right-click on the Database Name. More options will appear as shown below:

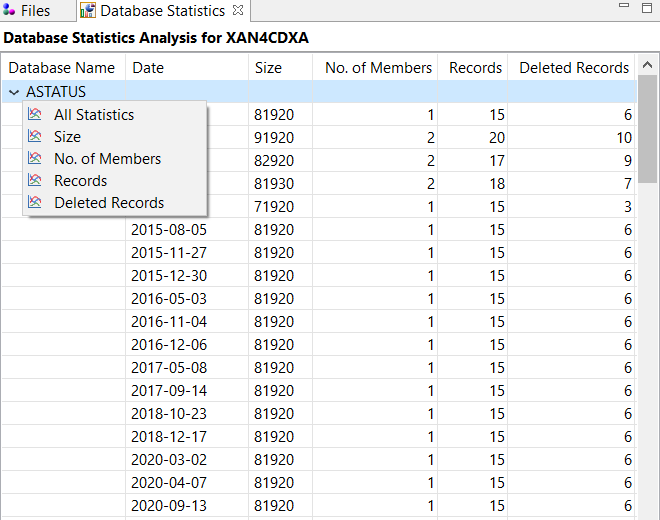


Fig. 15.1.6 – Context menu showing options for viewing Database Statistics

When the user clicks All Statistics, the graph will display the statistics of the entire database over the selected period of time.

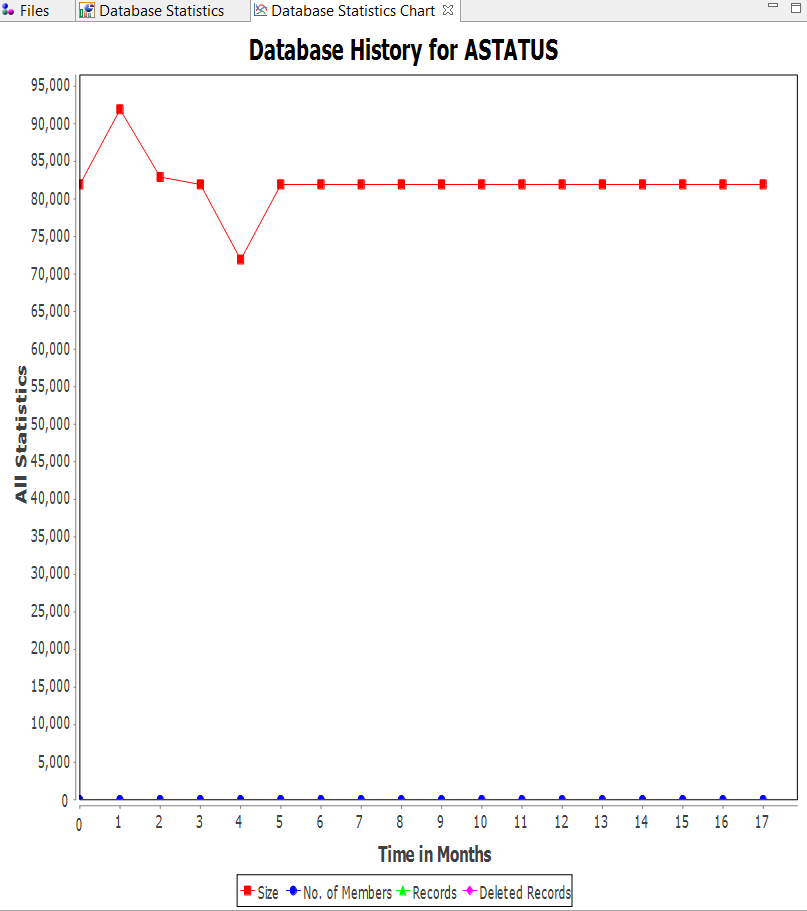


Fig. 15.1.7 – All Statistics window

All Statistics can be further split for individual parameters like Size, or No. of records, as specified in the context menu. When the user selects the Size option, the database size is plotted as follows:

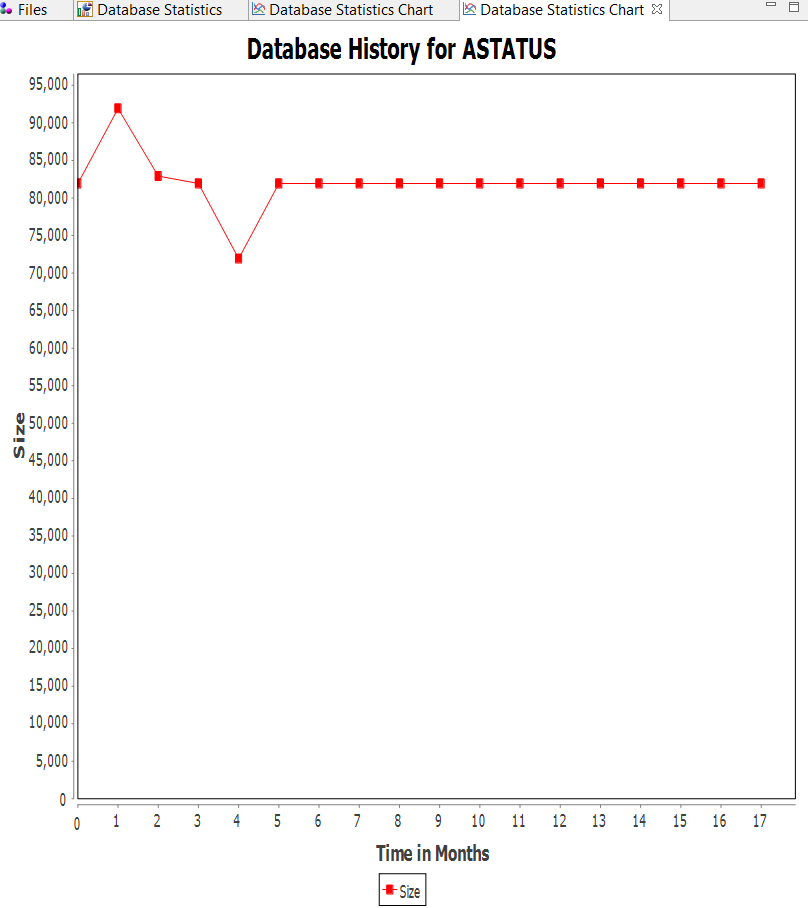


Fig. 15.1.8 – Size - Database Statistics Chart

Inter-Repository Options

The Inter-Repository Options provides different sub-options for comparing database files (across any two cross-reference libraries) and managing linked repositories. The Inter-Repository Options submenu is available on the context menu of the cross-reference library.

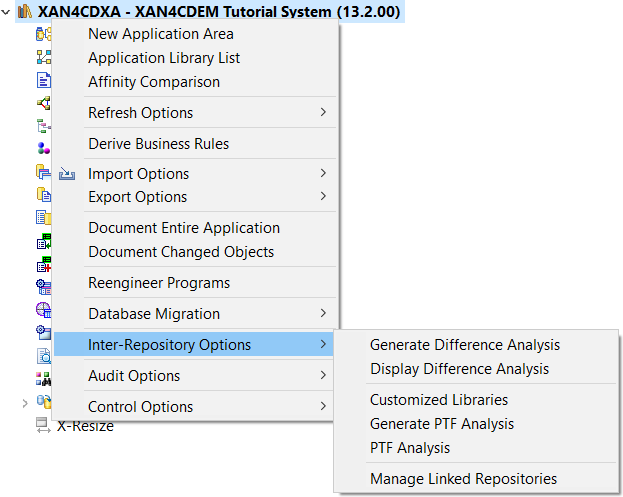


Fig. 17.1.1 – XAN4CDXA context menu – Inter-Repository Options

As displayed in the screen above, Inter-Repository Options contains the following group of options:

* Difference Analysis (Analyses the application database files and reports the difference with the files.)
  + [Generate Difference Analysis](#-1228005375)
  + [Display Difference Analysis](file:///D:/Trunk/XA-Help/Content/XA_Audit/CH_18/dis_diff_ana.htm#Display_Difference_Analysis)
* PTF Analysis (Analyses the base and the customized applications for PTF analysis.)
  + [Customized Libraries](#-889536178)
  + [Generate PTF Analysis](#-503067988)
  + [PTF Analysis](#-600841699)
* Manage Linked Repositories (Allows analysis of one or more IBM i cross-reference library to an existing cross-reference library.)

|  |  |
| --- | --- |
|  | Fresche Solutions supplies two additional data libraries for demonstrating Difference Analysis and PTF Analysis:  XAN4CDEMCU – This library contains objects from XAN4CDEM with simulated changes. XAN4CDEMPT – This library contains simulated PTF for XAN4CDEM. |

When large amounts of data are being analyzed, there could be differences in the data. Users must be informed before re-configuring an application, about the differences. The Inter-Repository Options present on the context menu of an application provides the option for generating differences. The option is explained here using a demo case.

Generate Difference Analysis

The Generate Difference Analysis option submits a batch job which populates data for Difference Analysis. The Generate Difference Analysis expects the cross-reference library name from the user, which is used to compare the cross-reference libraries.

The following dialog box is displayed when the user selects the Generate Difference Analysis option:

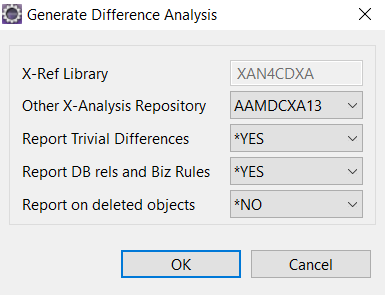


Fig. 17.2.1 – Generate Difference Analysis dialog box

Display Difference Analysis

|  |
| --- |
| The Difference Analysis compares the objects, sources, procedures, subroutines, references, files, fields, relations, and business rules for the selected repository versus the target repository. The difference analysis data is only available when a user has already opted for the Generate Difference Analysis option.  To display the Difference Analysis data, right-click for the context menu on the cross-reference node. Go to the Inter-Repository Options submenu, and then select Display Difference Analysis.  The initial screen displays three headings Changed/Deleted/Added as shown below.  Fig. 17.3.1 – Difference Analysis data  The difference analysis comes with three parameters to incorporate objects.   * Member difference parameter for member source/file field and text differences. * DB relations parameter to include the database relations. * Deleted objects parameter to include the deleted source/object.   The process populates details in the three categories of differences:   * A = Added (Present in this repository but not in the other repository). * D = Deleted (Present only in the other repository). * C = Changed (Compares the XA file data from the current repository to the other repository). |

Customized Libraries

A customized library is one where the customer would store programs taken from the vendor’s library and made modifications. This way a user can retain the original programs from the vendor and have their modified version. The following dialog is displayed when selecting the Customized Libraries option:

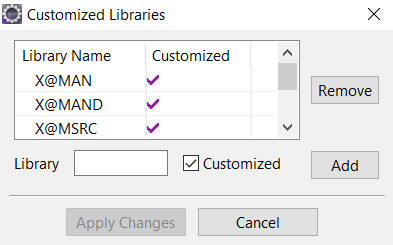


Fig. 17.4.1 – Customized Libraries dialog

The user should provide valid entries for Customized Library (ies) and for Non-Customized Library (ies).

Generate PTF Analysis

The Generate PTF Analysis option submits a batch job which populates data for PTF Analysis.

The Generate PTF Analysis dialog expects the base repository name from the user, which is used to compare the PTF repository with the base repository.

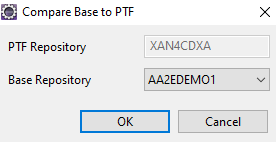


Fig. 17.5.6 – Generate PTF Analysis dialog

PTF Analysis

The PTF Analysis displays the comparison between the PTF repository and the base repository. The PTF Analysis data is available only after the user has opted for the Generate PTF Analysis option.

Note that the PTF Analysis functionality will work on the same version. In case, the users are working on a post V12.X version, then the user must remember to re-create the base repository on the latest version. Also note that, PTF Analysis tracks on the repositories of Initialized version, and not all in the Base Repository drop-down.

**Demo Case – PTF Analysis**

For better understanding of the PTF Analysis, let us create a demo case. Follow the steps to set up the demo case:

1. Create a new X-Analysis application (call it XAN4CDXP) with the following libraries:

Source: XAN4CDEMPT  
Object: XAN4CDEMPT

1. Initialize the new application – XAN4CDXP.
2. To generate the PTF Analysis, the first step is to identify libraries as the customized libraries. Select the X-Analysis application – XAN4CDXC. Go to the Inter-Repository Options, then select Customized Libraries. This invokes the following dialog box:

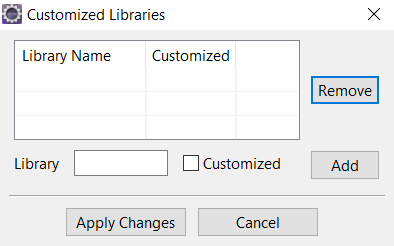


Fig. 17.6.1 – Customized Libraries dialog box

1. Type XAN4CDEMCU into the Library text box and check the box next to Customized. Then, click Add. This adds the XAN4CDEMCU library as the customized library.

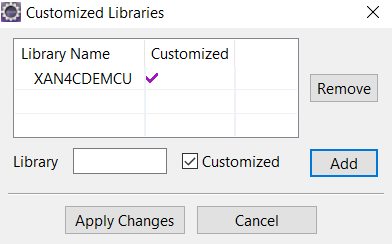


Fig. 17.6.2 – Customized Libraries dialog box

1. Now, add the XAN4CDEM library as the non-customized library. Type XAN4CDEM into the Library text box and un-check the Customized option. Then, click Add. This adds the XAN4CDEM library as a non-customized library.

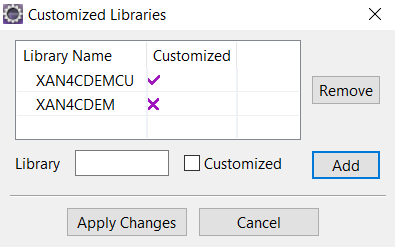


Fig. 17.6.3 – Customized Libraries dialog box

1. The next step is to select the Generate PTF Analysis option. Opt for the context menu on the X-Analysis application – XAN4CDXP. Go to the Inter-Repository Options submenu, and then select the Generate PTF Analysis option. Enter XAN4CDXC as the Base Repository. Then, click OK to submit the batch job.

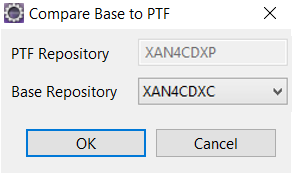


Fig. 17.6.4 – Compare Base to PTF dialog box

1. The final step is to display the PTF Analysis data. Opt for the context menu on the new X-Analysis application, XAN4CDXP. Go to the Inter-Repository Options submenu and select PTF Analysis. The following screen should appear:

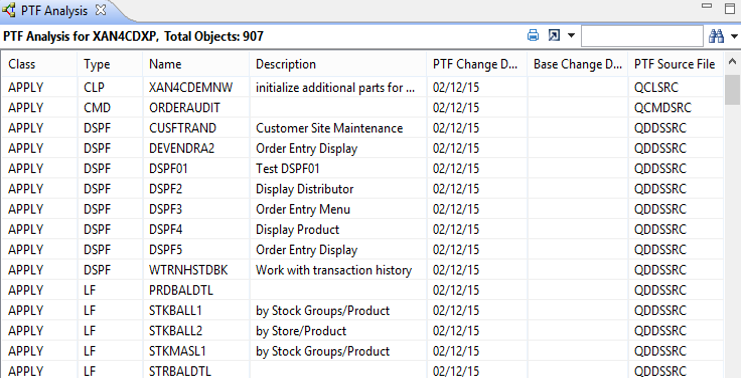


Fig. 17.6.5 – PTF Analysis for XAN4CDXP

The first column of the PTF Analysis displays Class. The Class column can contain any of the following entries:

MODIFIED = The object from the PTF library was found in one of the CUSTOMIZED libraries.  
User Action: The PTF object will have to be reviewed and changes applied in the CUSTOMIZED library; will be manually applied to the object in the PTF library.

NEW = The object from the PTF library was not found in the base repository.  
User Action: The PTF object can be placed in the base library.

APPLY = The object from the PTF library was found in one of the BASE libraries (Vanilla) but not in any of the CUSTOMIZED libraries.  
User Action: The PTF object can overlay the object in the base library.

REFERS = The object from the PTF library refers to one or more objects in one of the CUSTOMIZED libraries. The details are in XPTFROBJ.  
User Action: The PTF object will have to be revised to make sure all customized objects referred to still meet the requirements of this object.

REFERENCED = The object from the PTF library is referenced by an object in one of the CUSTOMIZED libraries. The details are in XPTFROBJ.  
User Action: The CUSTOMIZED objects will have to be reviewed to make sure the PTF object still meets the requirements of that object.

Manage Linked Repositories

This Manage Linked Repositories feature allows to link one or more IBM i cross-reference repository to an existing IBM i repository. The option is available under Inter-Repository Options, on the context menu of the cross-reference library node.

Linking IBM i Repositories

The Manage Linked Repositories option opens the following dialog box:

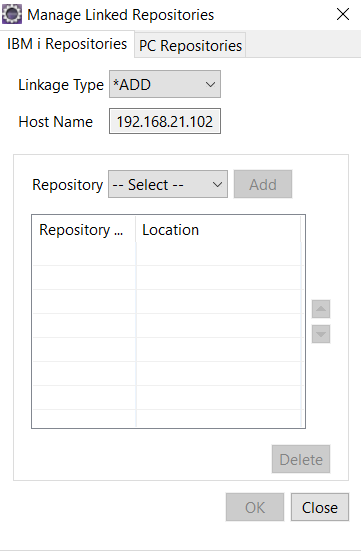


Fig. 17.7.1 – Manage Linked Repositories dialog box

In the above dialog box, Host Name is the field that will show the IP address of the IBM i Repositories server whose repositories can be selected from the Repository Combo.

Through this dialog box a user can add the repository (ies) to the base (working) repository for combined analysis. The dialog box lists all repositories, besides the base repository, available on the X-Analysis server. Select the repository to add.

Linkage Type: Link the repository (ies) in the following two ways:

* \*ADD – In this mode, the cross-reference information from the entire linked repository (ies) is merged with those of the base repository on the X-Analysis lists and diagrams.
* \*REPLACE – Here, ONLY the distinct cross-reference information from the repository (ies) is put on the X-Analysis lists/diagrams. This depends on the sequence of the repositories.

Repository: It lists all the X-Analysis repositories on the machine, besides the one working with. The user can select a repository to link with the base repository.

To link a repository, select the desired repository from the Repository drop-down and press Add. This adds the selected repository to the linked repository list.

To delete the repository from linked repository list, select it and press Delete. This removes the selected repository from the linked repository list. The base repository (in blue) cannot be removed from the list.

Use the arrow buttons to change the sequence of the linked repositories.

Old Linked Repository Object – On selecting the Manage Linked Repositories option, the old users of X-Analysis would see the following information message:

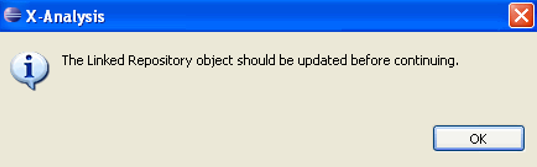


Fig. 17.7.2 – Message informing about an old version of Server Object

Please contact Fresche Solutions for the updated Server Component(s).

Option Data Model Diagram and Data Flow Diagram help in identifying the relations from the linked repository when a user selects a specific file. The below section provides two such scenarios where a user can find such relation of a linked repository.

When a user opts for a specific file in Data Model Diagram, it shows the relations from all the linked repositories. Refer to the screen below that shows relationships from linked repositories when Data Model Diagram gets selected on a specific file.

|  |
| --- |
| Fig. 17.7.3 – Linked Repository of a Specific File in DMD |

Similarly, the option Variable Where Used can also be used for a search across linked repositories. Refer to the screen below where the variable CUSNO can be found in a program called TESTCUSF.

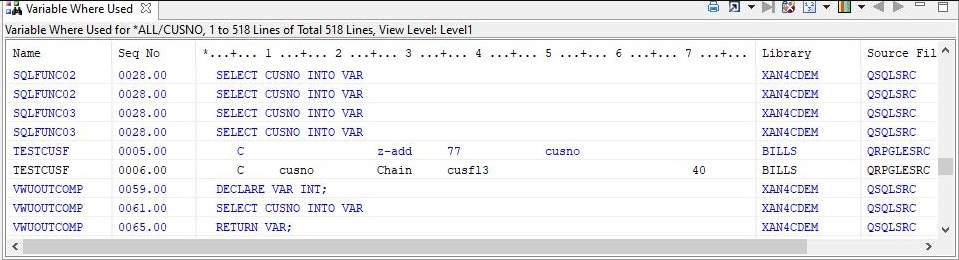


Fig. 17.7.4 – Variable Where Used

Now, If we look at the Data Flow Diagram for TESTCUSF as shown below, it will show that the CUSFL3 file and the TESTCUSF both have golden pins in the object, which means that the entries come from a linked repository.

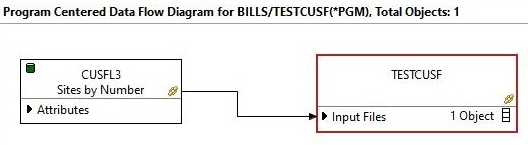


Fig. 17.7.5 – Data Flow Diagram

Appendix A – Code Review feature (TD/OMS Support)

|  |  |
| --- | --- |
|  | The Code Review feature (TD/OMS Support) version 7 support is now available for the X-Analysis customers. |

The Code Review functionality has been added to X-Analysis for harnessing the TD/OMS support. The Code Review functionality, effectively, is the problem analysis execution on the selected object. It helps to view any issues at the source/object level while promoting to the next stage or visualize the problem analysis statistics/history on the selected object.

|  |  |
| --- | --- |
|  | The Code Review functionality is only enabled when the XSCMREVIEW data area is set to ‘Y’. |

The Generate Code Review option is present under the Audit Options submenu as can be seen below:

|  |
| --- |
| Fig. 18.1.1 – Audit Options – Generate Code Review |

When the user clicks the option, the following dialog box is displayed:

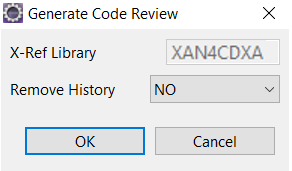


Fig. 18.1.2 – Generate Code Review dialog box

The user can choose to Remove History. The default option is NO. If YES is selected, then all the previous history is deleted or removed. Click OK to begin the Code generation.

|  |
| --- |
| Fig. 18.1.3 – Audit Options – Problem Analysis Request History |

The following screen shows the Problem Analysis history for the selected cross-reference library view.

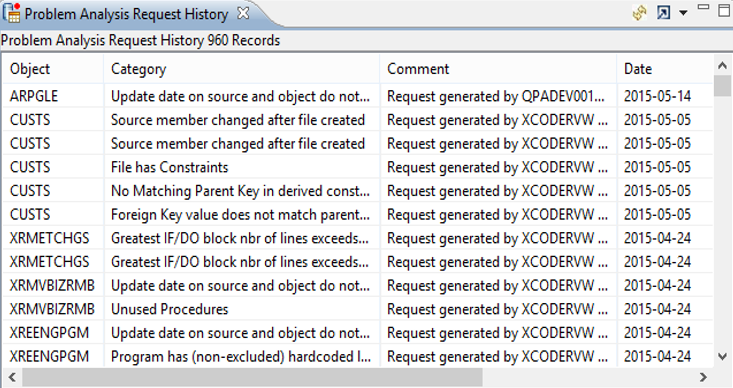


Fig. 18.1.4 – Problem Analysis Request History

The Code Review functionality can be used for the entire cross-reference library and for individual objects as well.

In the following screen, the Display Code Review option has been selected on an individual object, WWCUSTS.

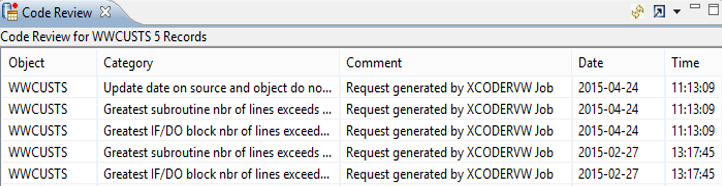


Fig. 18.1.5 – Code Review view for WWCUSTS

Appendix B – Initializing Source Archiving

Perform the steps given below in order to initialize Source Archiving and display Change History window:

1. Run Initialize Source Archiving. The Initialize Source Archiving option is available for the cross-reference library and the application areas. This option is available to run independently of metrics.

The Initialize Source Archiving option is shown below:

|  |
| --- |
| Fig. 19.1.1 – Audit Options – Initialize Source Archiving option |

Now, all the source files will get copied into the Archive Source physical files (name starting with XACV) and the required data areas will get initialized. (for example, M1- original member)

1. Modify the source member (for example M1C1). Subsequently, either initialize the cross-reference or perform XREFRESH on it. Any changes in the source member will be recorded in the database file during the initialization/XREFRESH process.
2. Double click on the Change History option from the X-Analysis client. Any changes made in the source member get displayed in the Change History window.
3. Run the XAOBJXXXXX/XACVEND command with Delete History "\*YES"/ Delete History=\*YES. It will clear all the database file and data areas populated by Initialize Source Archiving. However, it will not clear Archived Source physical file.

|  |  |
| --- | --- |
|  | Archived source physical file would be renamed in X-Ref library. |

1. Modify the source member (for example, M1C1C2). Afterwards, either Initialize the cross-reference or perform XREFRESH. Since archiving has been deactivated, the changes made in the source member will not get recorded in the database file.
2. Double click on the Change History option from the X-Analysis client again. The modified source member will not be displayed because the archiving has been deactivated.
3. Run Initialize Source Archiving. Now, all the source files will get copied into the Archive Source physical files (name starting with XACV).
4. Modify the source member (for example, M1C1C2C3). Afterwards, either Initialize the cross-reference or perform XREFRESH (for example, M1C1C2C3). Since the archiving is activated in the last instruction, the changes made in the source member will be recorded in the database file.
5. Double click on the Change History option from the X-Analysis client. The changes done in the source member will get displayed in the Change History window as the source member changes done in the step 6 becomes the archived source.
6. The changes done in (M1C1C2C3) will be displayed and since the Source archiving was deactivated, the changes executed in the source member (M1C1C2) will not be displayed.

Appendix C – Adding new Problem Analysis Category

The purpose of the Problem Analysis Service Program is to allow the addition of custom problem analysis records to the cross-reference so that they may be retrieved and reported by X-Analysis’s View Problem Analysis.

Prerequisites

The X-Analysis Client allows adding a new Problem Analysis Category.

|  |
| --- |
| * First, enable the Problem Category Editor by setting the XSCMREVIEW data area value of the repository to 'Y' using the IBM i command. CHGDTAARA DTAARA(XREF\_NAME/XSCMREVIEW \*ALL) VALUE('Y') * Next, opt for Edit Problem Categories option as follows:   Fig. 13.1.1 – Audit Options – Edit Problem Categories  Use the Add buttons to add Severity, Category, and Description. |

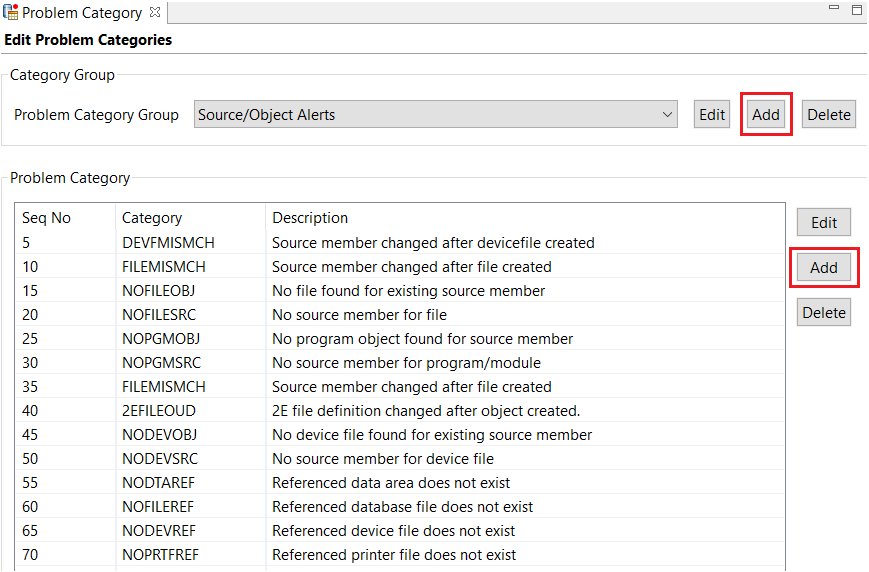


Fig. 20.1.2 – Add Category Editor

Add the following category for use with the sample RPG program provided later in this document.

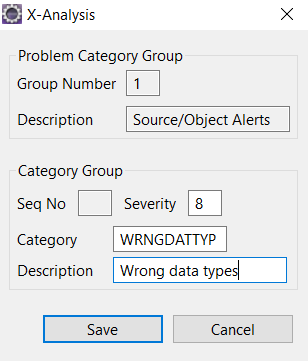


Fig. 20.1.3 – Add Category

Saving the above will add the following entries:  
in XPRBGRPQ:

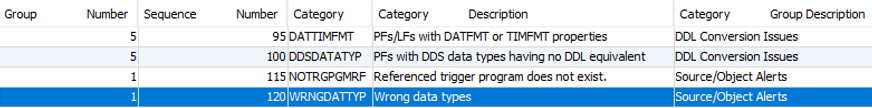


Fig. 20.1.4.A – Problem Category Editor

and in XPRBCATSQ:

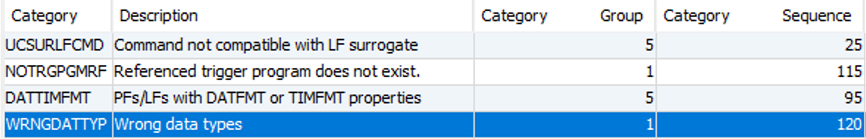


Fig. 20.1.4.B – Problem Category Editor

Programming Guidelines

This section defines the APIs which is used to write problem analysis records to the repository, and also the required copybooks.

In order to use the APIs, the programmer must include the following copybooks in the calling program:

1. XAOBJ13200/XRPRBOBJSI defines the t\_ProblemObjects\_Service structure as follows:

DCL-DS t\_ProblemObjects\_Service qualified template Inz ;  
 category CHAR(10) ;  
 object CHAR(10) ;  
 objectLibrary CHAR(10) ;  
 objectSourceDate DATE ;  
 objectSourceTime TIME ;  
 objectCreateDate DATE ;  
 objectcreateTime TIME ;  
 sourceLibrary CHAR(10) ;  
 sourceFile CHAR(10) ;  
 sourceMember CHAR(10) ;  
 sourceMemberChangeDate DATE ;  
 sourceMemberChangeTime TIME ;  
 source2eFile CHAR(10) ;  
 referenceObjectLibrary CHAR(10) ;  
 referenceObjectName CHAR(10) ;  
 subroutineName CHAR(10) ;  
 numberUses PACKED(9:0) ;  
 lastUsedDate DATE ;  
 numbervariable1 ZONED(5:0) ;  
 numbervariable2 ZONED(5:0) ;  
 numbervariable3 ZONED(5:0) ;  
 additionalInformation CHAR(100) ;  
 End-Ds;

Where

category is the category under which to add the problem object  
object is the problem object name  
objectLibrary is the library in which the problem object exists  
objectSourceDate is the object’s source create date  
objectSourceTime is the object’s source create time  
objectCreateDate is the object’s create date  
objectCreateTime is the object’s create time  
sourceLibrary is the object’s source library  
sourceFile is the object’s source file  
sourceMember is the object’s source member  
sourceMemberChangeDate is the object’s source member change date  
sourceMemberChangeTime is the object’s source member change time  
source2eFile is reserved for future use  
referenceObjectLibrary is the referring object’s library  
referenceObjectName is the referring object’s name  
subroutineName is reserved for future use  
numberUses is reserved for future use  
lastUsedDate is reserved for future use  
numbervariable1 is reserved for future use  
numbervariable2 is reserved for future use  
numbervariable3 is reserved for future use  
additionalInformation is the Additional Details column in Problem Analysis report

1. XAOBJ13200/XRUTILITY and XAOBJ13200/XRUTIL0001A define error handling utilities and data structures.

The format in which error messages are stored and returned is defined in XAOBJ13200/XRUTIL0001A as follows:

ut\_messageFMT DS qualified template   
 msgId 7A varying   
 msgText 1024A varying   
 severity 10I 0   
 help 500A varying   
 forVar 50A varying

where

msgId is a unique message identification number  
msgText is the associated descriptive error message  
severity is the associated severity level  
help is reserved for future use  
forVar is reserved for future use

The APIs used to access the errors are:

uget\_messageCount() returns the number of errors set by the most recent call to any SRVPGM API.

uget\_message(forMessage : msgFormat) retrieves the specific error information defined by forMessage into msgFormat.

uclear\_Messages() resets the internal error counter and clears all error messages.

**SRVPGM APIs used to write problem analysis records**

**add\_ProblemObjects ( data [: origntyp] )**

Adds a record to XPRBOBJSQ table as part of the most recently generated Problem Analysis set of findings. Also, adds the specified category to XPRBCATSQ if it does not already exist.

**Parameters:**

1. data is the record that will be written to XPRBOBJSQ. Its structure is defined in XPRBOBJSI, described above.

Note that this data structure represents a subset of XPRBOBJSQ’s complete record structure. add\_ProblemObjects() populates the remaining fields with default values of blank or 0, depending on the field types.

1. origntyp is an optional parameter indicating the type of problem analysis. For code review problems, set origntyp=XPRBANL. For database-related problems, set origntyp=XPRBANLDDL. The default is XPRBANLDDL.

**validate\_ProblemObjects (data)**

Used as a validation of a record. This procedure also implicitly called by add\_ProblemObjects().

**Parameters:**

1. data is the record that is to be validated. Its structure is the same as described in add\_ProblemObjects().

**Example**

The use of the I/O Service Program is best presented through a working example. It is quite common that dates are stored in file fields of non-DATE storage type. One might want to root out the file fields likely falling in this category.

The following RPG program identifies application files (LFs and PFs) stored in the repository having file fields that possibly represent dates, but are not of the DATE type. It then writes them to the repository problem analysis tables so that they may be reported in X-Analysis Problem Analysis.

Before calling the program, add XAPROD and the repository to the library list:

ADDLIBLE XAPROD  
 ADDLIBLE XA\_REPO POSITION(\*FIRST)

Here is the program code:

\*\*free  
   
 //Identify files with fields possibly representing dates, but not of DATE type.  
   
 Ctl-opt Main(Driver);  
   
 /include QRPGLESRC,XRPRBOBJSI  
 /include QRPGLESRC,XRUTIL001A  
   
 DCL-DS data LikeDS(t\_ProblemObjects\_Service) Inz(\*likeDS);  
   
 DCL-PR Driver extpgm('XRTSTADD');  
 End-PR;  
   
 // ----------------------------------------------------------------  
 // ------------------ M A I N P r o c e d u r e -----------------  
 // ----------------------------------------------------------------  
   
 DCL-PROC Driver;  
   
 // ---------------- Local Variables ----------------------------------------  
 DCL-S errcount PACKED(10:0);  
 DCL-S count PACKED(10:0);  
 DCL-S len PACKED(10:0);  
 DCL-S msg CHAR(40);  
   
 DCL-DS uDsPgmDtls ;  
 objectLib CHAR(10);  
 object CHAR(10);  
 sourceLib CHAR(10);  
 sourceFile CHAR(10);  
 sourceMember CHAR(10);  
 END-DS;  
   
 DCL-DS DS1 likeDs(ut\_messageFMT);  
   
   
 // Set SQL options  
 Exec SQL Set Option  
 DatFmt=\*ISO, TimFmt=\*ISO,  
 Commit=\*NONE, CloSQLCsr=\*ENDMOD;  
   
 Monitor;  
   
 Exec SQL Declare getPrbDtls cursor for  
   
 select distinct l#lib,  
 l#obj,  
 l#slib,  
 l#srcf,  
 l#smbr  
 from xobject o  
 join xdd x on x.vfile=o.l#obj  
 where l#type='\*FILE' and x.vfld like('%DATE%') and x.fdattr != 'L';  
   
 Exec SQL Open getPrbDtls;  
   
 // Fetch the first record.  
 Exec SQL  
 Fetch from getPrbDtls into :uDsPgmDtls;  
   
 If SQLCOD <> 0;  
 dsply ('SQL Error: ' + %char(SQLCOD));  
 Exec SQL Close getPrbDtls;  
 Else;  
   
 // For each record, define problem object and add it to repository  
 data.category = 'WRNGDATTYP';  
   
 \Dow Sqlcod = 0;  
   
 data.objectLibrary = objectLib;  
 data.object = object;  
 data.sourceLibrary = sourceLib;  
 data.sourceFile = sourceFile;  
 data.sourceMember = sourceMember;  
   
 add\_ProblemObjects(data);  
   
 errcount = uget\_messageCount();  
   
 if errcount <> 0;  
 Dsply (%char(errcount) + ' errors!');  
 For count = 1 to errcount by 1;  
 uget\_message(count:DS1);  
 len = %len(%trim(DS1.msgText));  
 msg = %SUBST(DS1.msgText:1:len);  
 dsply msg;  
 Endfor;  
 endif;  
 // Fetch the next record  
 Exec SQL  
 Fetch from getPrbDtls into :uDsPgmDtls;  
   
 Enddo;  
 Endif;  
   
 // Close Cursor  
 Exec SQL Close getPrbDtls;  
   
 On-error;  
 msg = 'Unexpected error';  
 dsply msg;   
 EndMon;  
   
 return;  
   
 END-PROC Driver;

User Guidelines

To call the above program, it must be compiled into a module and bound to the provided service program XRPRBOBJS, as follows:

1. Use the following IBM i command to compile and create a module for the program:

CRTSQLRPGI OBJ(OBJLIB/XRTSTADD) SRCFILE(SRCLIB/QRPGLESRC)  
SRCMBR(SXRTSTADD) OBJTYPE(\*MODULE) TGTRLS(V7R1M0)

where

OBJLIB is the library where the resultant module XRTSTADD will be created.  
SXRTSTADD is the source member name of the above RPG code.  
SRCLIB/QRPGLESRC is the source physical file containing SXRTSTADD.  
V7R1M0 indicates the minimum compatible version of IBM i.

1. Use the following IBM i command to create the program PXRTSTADD, which binds the service program XRPRBOBJS with the module XRTSTADD:

CRTPGM PGM(OBJLIB/PXRTSTADD) MODULE(OBJLIB/XRTSTADD)   
BNDSRVPGM((\*LIBL/XRPRBOBJS)) TGTRLS(V7R1M0)

where

OBJLIB is the library where the resultant program PXRTSTADD will be created.

Note that \*LIBL is specified to dynamically bind to the I/O service program at runtime to the required version specified by the library list ordering.

1. Use XAPROD/XACMD to execute the program as follows. Note that XACMD does not need to be fully qualified as XAPROD should already exist in the library list:

XACMD XRFLIB(XA\_REPO) CMD('CALL PGM(OBJLIB/PXRTSTADD)')

With the support of multiple versions of X-Analysis being allowed on the same partition, XACMD automatically selects the correct version of the product libraries before executing the program.

This command uses two parameters:

* 1. the X-Ref library, and
  2. the command to execute

Based on the specified X-Ref library, the appropriate version of product libraries will be added to the library list before the command is executed.

1. The result will appear in X-Analysis Problem Analysis like the following:

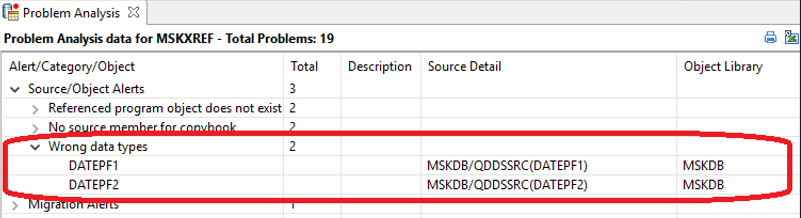


Fig. 20.2.1 – Problem Analysis data

Index

2E, 24

Access Paths, 28

Action Diagram, 24

APIs, 108

application area, 50

Attributes, 11

Audit Options, 7, 8, 13, 22, 27, 30, 32, 46, 50, 52, 54, 59, 63, 66, 68, 73, 78, 97, 102, 105

Audit Report, 30, 55

Batch programs, 12, 20

Business Process Logic, 5, 7, 9, 29

Business Rules, 8, 22, 61, 87

Called Programs, 11

Calling Programs, 12

Client, 6, 60, 70, 103, 105

Client Components, 6

Code Review, 7, 8, 97, 110

Complexity Level, 23

Conditioning Fields, 24

configuration, 37, 58

Create New Report, 40

Creating Programs, 28

Cross-Reference Libraries, 84, 86

cross-reference library, 6, 9, 23, 28, 52, 59, 73, 79, 84, 86, 93, 100, 102

customized, 9, 32, 38, 40, 48, 84, 88, 89

Customized Libraries, 84, 88, 89

Cyclomatic Complexity, 10, 19

Data Flow Diagram, 94

data model, 53, 94

Data Model Diagram, 53, 94

Database Fields, 23

Deleting Programs, 28

Dependent File, 53

Design Level, 8

Device Files, 11, 43

Diagrams, 94

Difference Analysis, 5, 7, 84, 86, 87

DMD, 95

documentation, 11, 55

Edit Problem Categories, 7, 68, 105

Export Options, 17, 26

Export to MS Excel, 17, 26

Export to PDF, 17, 26

Fields, 23, 28, 38, 42, 70, 75, 87, 109

File Metrics, 5, 7, 9, 27

Halstead, 10

IBM, 4, 5, 78, 84, 93, 105, 113

IBM i, 5, 78, 84, 93, 105, 113

Incoming Calls, 23

Initialize, 5, 7, 9, 59, 102

Initialize Source Archiving, 5, 7, 9, 59, 102

Interactive programs, 12, 18

Inter-Repository Options, 84, 87, 89, 93

Level, 8, 23, 41, 97, 109

LFs, 49, 73, 110

libraries, 84, 86, 88, 89, 113

Library List, 110, 113

Maintainability Index, 9

Manage Linked Repositories, 84, 93

Metrics Analysis, 5, 7, 9, 16, 54, 63

Metrics History, 9, 18

Modify Report Definition, 35

Object Allocation, 5, 7, 9, 50

Object List, 77

Outgoing Calls, 23

parameters, 10, 18, 61, 82, 87, 109, 113

Parent File, 53

Preferences, 12, 16, 28

Print, 16, 26

Problem Analysis, 5, 7, 8, 9, 46, 54, 63, 67, 72, 97, 104, 105, 108, 113

Problem Analysis Request History, 100

Problem Category, 7, 46, 68, 105

PTF Analysis, 5, 84, 89

Reading Programs, 28

Refresh Metrics, 40

Remove History, 99

repository, 32, 84, 87, 89, 93, 105, 108

RPG, 10, 29, 60, 106, 110

Run Metrics Report, 35

Screen Metrics, 5, 7, 8, 9, 22, 25

Server, 6, 60, 67, 70, 93

SOURCE, 5, 7, 9, 18, 24, 46, 59, 73, 89, 97, 102, 113

Source Compare, 20

Source Lines, 9

Specialized Analysis, 5, 7, 32, 36, 40

subroutines, 10, 42, 87

Suite of products, 5

Summary Report, 5, 7, 9, 53, 54, 55

Synon, 24

TD/OMS, 7, 8, 97

Total References, 28

Track Database Changes, 5, 7, 73

Units, 14, 23, 27

Updating Programs, 28

Variable Where Used, 95

View Database Size Statistics, 5, 77

View Log, 44

View Report, 36

Work Fields, 23

XACMD, 113

X-Analysis, 4, 5, 8, 9, 24, 28, 58, 60, 71, 79, 89, 93, 97, 103, 104, 105, 110, 113

X-Analysis Client, 103, 105

XAPROD, 110, 113

X-Audit, 4, 5, 7, 8

X-Rules, 8

Zoom Source, 48