

Technology Process and Strategic Tools

**Software Design Description**

**Open Globals Average Age Core Function Design**

**Revision 1.0**

**2015-Dec-11**

***Tom Storey – Process Engineer***

***Note:*** To reveal "hidden" text that will assist in completing this document turn on paragraph markers ( ¶ button).

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Name** | **Comments** |
| 0.1 | 2015-Dec-01 | Tom Storey | Draft in progress |
| 1.0 | 2015-Dec-11 | Tom Storey | Feedback per M. Dameron and complete design. (TBD – Project-level metrics table section) |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1 Overview 1

2 Metric Definition 1

3 Assumptions/Decisions 1

3.1 Project Level vs Customer Level 1

4 Inclusions/Exclusions 2

5 New Import Table - importBpmsGlobals 2

6 Routine to Populate Import Table 3

7 Changes to Stored Procedures that Populate Dimension Tables 3

8 New Globals Daily Table in Data Warehouse 5

9 Routine to Populate Globals Daily Table 6

10 Core Function 6

11 New Project Level Metrics Table – SQM\_QM\_METRICS\_BY\_PROJECT\_DAILY\_SUM EDIT THIS – IT IS CHANGES TO EXISTING TABLE HCL MADE 7

12 Routine to Populate Project Metrics Table 7

13 Nightly Job to Execute Population Routines 7

14 Appendix A – *GetBpmsGlobals* Stored Procedure Code 8

15 Appendix B – *PopulateGlobalsDaily* Stored Procedure Code 9

16 Appendix C – *calculateGlobalAverageAge* Function Code 11

# Overview

The Open Globals Average Age metric (formerly titled, “Global RFSS Aging”) provides the average number of months open global issues in Jira have been open for a Jira project. This metric is reported on the Dashboard, along with a score based on the value.

This design details an approach to pull source data from both the Jira and BPMS tools into the data warehouse and procedures to transform the data to reside in a new project-level metrics table to be consumed by the Dashboard (and perhaps other applications at a later time). It utilizes the same “core function” approach we are implementing for all metrics, so that a central calculation and storage mechanism exists.

# Metric Definition

Average number of months open for non-closed globals in Jira at any given point in time.

Notes:

* For **non-closed** issues, the number of months between the Jira create date and current date.
* Stored as a value with 3 decimal places
* We will also capture the average number of days, although this is not a defined metric at this time.

# Assumptions/Decisions

## Project Level vs Customer Level

* The data warehouse has dimension tables for both Customer and Project. The Project table is fed by the “projects” that are in Jira, which does NOT necessarily correspond to what could be considered a customer. This entity is also commonly referred to as “sites”. **Therefore, the globals-related metrics will reside at the PROJECT level (not the customer level).**
* The Globals Issue Management process in BPMS automatically creates issues in Jira for applicable sites (i.e. projects) when a global is published.
* Validation occurs to ensure the counts of global records in BPMS and Jira balance. A scenario that would throw these counts out of balance would be duplicating global issues in Jira.

# Inclusions/Exclusions

* For all of the following metrics/measures, the following are excluded:
  + Informational rated issues
  + Issues in Jira that are NOT classified as a Global
  + Issues that are closed in Jira
* For an issue to be considered “closed” in Jira, it will be of one of the following statuses:
  + Closed
  + Closed, Duplicate
  + Closed, Not a Problem
  + Closed, Not Reproducible
  + Closed, Passed Verification
  + Closed, Won’t Fix
  + Master Closed

# New Import Table - importBpmsGlobals

A new table will be created in sqmdb to receive a nightly feed of globals data from BPMS. This will later be joined to existing globals data already coming into the data warehouse from Jira, keyed on the Jira Issue Key (ex: “KYSA-151”).

|  |  |  |  |
| --- | --- | --- | --- |
| **Table 1: Table Schema for importBpmsGlobals** | | | |
| **Column** | **Data Type** | **Allow Null** | **Notes** |
| **importBpmsGlobalsId** | **Int** |  | **Auto-incrementing primary key** |
| **importDate** | **datetime** |  |  |
| **jiraIssueKey** | **nvarchar(50)** | **X** | **Ex: “KYSA-151”** |
| **globalNumber** | **nvarchar(50)** | **X** | **Ex: “G0000256”** |
| **rating** | **nvarchar(50)** | **X** | **Ex: “Site-Rated”** |
|  |  |  |  |

# Routine to Populate Import Table

A new stored procedure named **GetBpmsGlobals** will be created within sqmdb that will execute nightly via a nightly job (see Section 12 below) to populate the raw globals data.

The **GetBpmsGlobals** stored procedure will conduct the following steps:

1. Drop the **importBpmsGlobals** table (if it exists)
2. Create the **importBpmsGlobals** table per the specification above
3. Insert rows to the **importBpmsGlobals** table via a query that selects appropriate data from BPMS.

*See Appendix A for code of the* **GetBpmsGlobals***stored procedure.*

# Changes to Stored Procedures that Populate Dimension Tables

There are two similar stored procedures that populate the dimension tables in the data warehouse:

* **PopulateDims** – this stored procedure is called to populate dimension tables if they don’t already exist or are empty. It is therefore rarely executed.
* **UpdateInsertDims** – this stored procedure is called to update existing dimension tables with fresh data. This is therefore executed quite often.

There is a need to add statuses relevant to the Jira globals workflow to the STATUS\_DIM table. The STATUS\_DIM table is related to the TYPE\_DIM table.

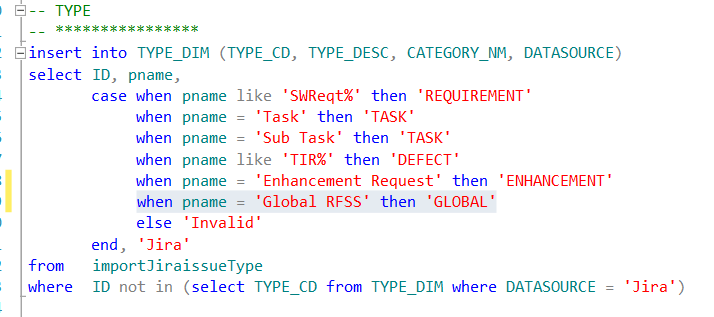
Currently, the above stored procedures only add statuses to the STATUS\_DIM table for the following category names within the TYPES\_DIM table. Others are ignored.

* REQUIREMENT
* TASK
* DEFECT
* ENHANCEMENT

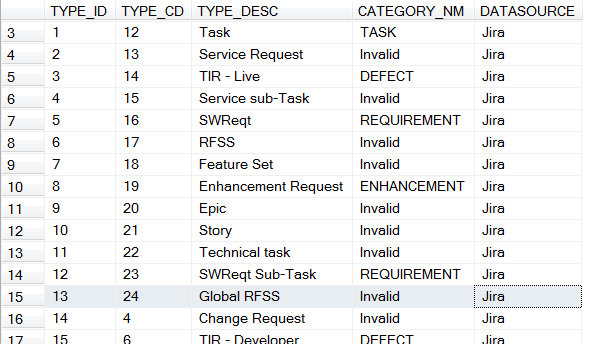
A GLOBAL category needs to be added.

Therefore, the following modification needs to be implemented in both the **PopulateDims** and **UpdateInsertDims** stored procedures:

* Within the section that populates the TYPE\_DIM table, add new logic to create a GLOBAL category name. There is a CASE statement that creates the category name for the CATEGORY\_NM column based on the pname column. See new line of code highlighted below:



* The TYPE\_DIM table then needs to be manually updated for the Global RFSS type to set the CATEGORY\_NM to ‘GLOBAL’ from ‘Invalid’



*UPDATE TYPE\_DIM*

*SET CATEGORY\_NM = ‘GLOBAL’*

*WHERE TYPE\_DESC = ‘Global RFSS’*

* The **UpdateInsertDims** stored procedure should then be executed so the TYPE\_DIM and STATUS\_DIM tables are updated appropriately.
* The **PopulateDims** stored procedure should **NOT** be executed.

# New Globals Daily Table in Data Warehouse

|  |  |  |  |
| --- | --- | --- | --- |
| **Table 2: Table Schema for SQM\_GLOBALS\_DAILY** | | | |
| **Column** | **Data Type** | **Allow Null** | **Notes** |
| **ENTITY\_ID** | **int** |  | **Auto-incrementing primary key** |
| **ENTITY\_DATE\_ID** | **Int** |  | **Int (foreign key to DATE\_DIM table)** |
| **PROJECT\_ID** | **Int** |  | **Associated project in Jira (foreign key to PROJECT\_DIM table)** |
| **CUSTOMER\_ID** | **Int** | **X** | **Populated based on the CUSTOMER\_ID for the particular project in PROJECT\_DIM table.** |
| **JIRA\_PKEY** | **Nvarchar(50)** | **X** | **Ex: “KYSA-151”** |
| **DATE\_CREATED** | **Int** | **X** | **Int (foreign key to DATE\_DIM table)** |
| **DATE\_CLOSED** | **int** | **X** | **Int (foreign key to DATE\_DIM table)** |
| **IS\_ISSUE\_CLOSED** | **Nvarchar(10)** | **X** | **Yes/No** |
| **CURRENT\_STATUS** | **Nvarchar(50)** | **X** | **Jira status of the issue** |
| **GLOBAL\_NUMBER** | **Nvarchar(50)** | **X** | **Ex: “G0000256”** |
| **RATING** | **Nvarchar(50)** | **X** | **Ex: “Site-Rated”** |
| **DESCRIPTION** |  | **X** | **Summary Description of the issue** |
| **OPEN\_DAYS** | **Int** | **X** | **Calculated within SP during population** |
| **OPEN\_MONTHS** | **Decimal(10,3)** | **X** | **Calculated within SP during population** |
|  |  |  |  |
|  |  |  |  |

# Routine to Populate Globals Daily Table

A new stored procedure named **PopulateGlobalsDaily** will be created within sqmdb that will execute nightly via a nightly job (see Section 12 below) to transform the imported data into a daily sum format that includes foreign keys to appropriate dimension tables.

The **PopulateGlobalsDaily** stored procedure will conduct the following steps:

1. Delete any entries for current day if they exist.
2. Derive the foreign keys for related dimension tables, such as DATE\_DIM, PROJECT\_DIM, etc.
3. Determine if the issue is closed based on the statuses in Section 4.
4. Join the importJiraIssueDates table so we can pull the correct closed\_date. The closed date in importJiraIssue is NOT the correct closed date!
5. Calculate the number of days and months each global has been open.
   1. If currently open, this is number of days from create to current date.
   2. If closed, this is number of days from create to closed date.
6. Insert row for each record, including the current date (DATE\_ID) and all other columns.

*See Appendix B for code of the* ***PopulateGlobalsDaily*** *stored procedure.*

# Core Function

A new table-valued function named **calculateGlobalAverageAge** will be created within sqmdb that will execute nightly via a nightly job (see Section 12 below) to calculate and average age and score.

Input parameters:

1. Project (PROJECT\_ID corresponding to the project in the **PROJECT\_DIM** table)
2. Date (DATE\_ID corresponding to the date in the **DATE\_DIM** table)

Function steps:

1. Calculate average of OPEN\_DAYS column for records in **PopulateGlobalsDaily** that match the input parameters. Return -1 if cannot be calculated and store remarks.
2. Calculate average of OPEN\_MONTHS column for records in **PopulateGlobalsDaily** that match the input parameters. Return -1 if cannot be calculated and store remarks.
3. Calculate AVG\_OPEN\_MONTHS\_SCORE column for records in **PopulateGlobalsDaily** that match the input parameters. Return -1 if cannot be calculated and store remarks. Utilize getMetricScore scalar function to retrieve score based on business rules in table and passed values.

Output:

1. countOpenGlobals
2. globalsOpenAverageAgeDaysValue
3. globalsOpenAverageAgeMonthsValue
4. globalsOpenAverageAgeMonthsScore
5. remarks

*See Appendix C for code of the* **calculateGlobalAverageAge***function.*

# New Project Level Metrics Table – SQM\_QM\_METRICS\_BY\_PROJECT\_DAILY\_SUM EDIT THIS – IT IS CHANGES TO EXISTING TABLE HCL MADE

|  |  |  |  |
| --- | --- | --- | --- |
| **Table 3: Table Schema for SQM\_QM\_METRICS\_BY\_PROJECT\_DAILY\_SUM** | | | |
| **Column** | **Data Type** | **Allow Null** | **Notes** |
| **DATE\_ID** | **Int** |  |  |
| **PROJECT\_ID** | **Int** |  |  |
| **AVG\_OPEN\_MONTHS** | **Decimal(10,3)** | **X** |  |
| **AVG\_OPEN\_MONTHS\_SCORE** | **int** | **X** |  |
| **AVG\_OPEN\_DAYS** | **Decimal(10,3)** | **X** |  |
| **REMARKS** | **Nvarchar(max)** | **X** |  |
|  |  |  |  |

# Routine to Populate Project Metrics Table

The existing stored procedure named **PopulateMetricsSumForDate** within sqmdb will be modified to include population of the SQM\_QM\_METRICS\_BY\_PROJECT\_DAILY\_SUM table with records for that day for all projects. This is already executing via a nightly job.

This stored procedure will run the **calculateGlobalAvgAge** core function for each project and pass the current day.

*Code modifications for the* ***PopulateMetricsSumForDate*** *stored procedure will be inserted here during development.*

# Nightly Job to Execute Population Routines

Add the following 2 steps to the existing agent job, **SQMDB Nightly Load:**

* **Get globals data from BPMS** – this step should execute the *GetBpmsGlobals* stored procedure
* **Populate SQM-GLOBALS\_DAILY table** – this step should execute the *PopulateGlobalsDaily* stored procedure.

# Appendix A – *GetBpmsGlobals* Stored Procedure Code

-- =============================================

-- Author: Tom Storey

-- Create date: 2015-Dec-09

-- Description: Imports Globals Data from Bpms into the importBpmsGlobals table.

-- =============================================

CREATE PROCEDURE [dbo].[GetBpmsGlobals]

AS

BEGIN

SET NOCOUNT ON;

--Drop and re-create the importBpmsGlobals table. This is the raw data coming over from the RIBPMDB01

--database.

--Drop the import table if it exists

IF OBJECT\_ID (N'dbo.importBpmsGlobals', N'U') IS NOT NULL

DROP TABLE dbo.importBpmsGlobals

--Create importBpmsGlobals table

CREATE TABLE [dbo].[importBpmsGlobals](

[importBpmsGlobalsId] [int] IDENTITY(1,1) NOT NULL,

[importDate] [date] NULL,

[jiraIssueKey] [nvarchar](50) NULL,

[globalNumber] [nvarchar](50) NULL,

[rating] [nvarchar](50) NULL

) ON [PRIMARY]

INSERT INTO dbo.importBpmsGlobals(

importDate,

jiraIssueKey,

globalNumber,

rating

)

SELECT

GETDATE(),

sae.jiraSiteIssueKey,

mast.globalNumber,

mast.validatedRating as jiraRating

FROM RIBPMDB01.GTKGIM\_Prod.dbo.SiteApplicabilityEvaluation sae left join RIBPMDB01.GTKGIM\_Prod.dbo.MasterGlobalIssue mast

ON sae.globalIssueSystemId = mast.systemId

WHERE mast.validatedRating is not null and jiraSiteIssueKey is not null

# Appendix B – *PopulateGlobalsDaily* Stored Procedure Code

CREATE PROCEDURE [dbo].[PopulateGlobalsDaily]

AS

BEGIN

SET NOCOUNT ON;

--Set variable for dim value of today's date

DECLARE @todaysDate date

DECLARE @todaysDateDimKey int

SET @todaysDate = GETDATE()

SELECT @todaysDateDimKey = DateKey

FROM DATE\_DIM

WHERE DATE = @todaysDate

--Delete any existing rows for today in the SQM\_GLOBALS\_DAILY table

DELETE SQM\_GLOBALS\_DAILY WHERE ENTITY\_DATE\_ID = @todaysDateDimKey

--Populate the set of rows for today

INSERT INTO dbo.SQM\_GLOBALS\_DAILY(

[ENTITY\_DATE\_ID]

,[PROJECT\_ID]

,[CUSTOMER\_ID]

,[JIRA\_PKEY]

,[DATE\_CREATED]

,[DATE\_CLOSED]

,[IS\_ISSUE\_CLOSED]

,[CURRENT\_STATUS]

,[GLOBAL\_NUMBER]

,[RATING]

,[DESCRIPTION]

,[OPEN\_DAYS]

,[OPEN\_MONTHS]

)

SELECT

@todaysDateDimKey,

isnull(project.PROJECT\_ID,-1) as projectDimId,

isnull(project.CUSTOMER\_ID,-1) as customerDimId,

jira.pkey as jiraIssueKey,

dCreated.DateKey as jiraDateCreated,

dClosed.DateKey as jiraDateClosed,

(

CASE

WHEN status.STATUS\_DESC IN ('Closed', 'Closed, Duplicate', 'Closed, Not a Problem', 'Closed, Not Reproducible', 'Closed, Passed Verification', 'Closed, Won''t Fix', 'Master Closed') THEN 'Yes'

ELSE 'No'

END

) as isIssueClosed,

status.STATUS\_DESC as currentStatus,

bpms.globalNumber as globalNumber,

bpms.rating as rating,

jira.SUMMARY as jiraSummary,

(

CASE

WHEN status.STATUS\_DESC IN ('Closed', 'Closed, Duplicate', 'Closed, Not a Problem', 'Closed, Not Reproducible', 'Closed, Passed Verification', 'Closed, Won''t Fix', 'Master Closed') THEN DATEDIFF(day,jira.CREATED,jira.RESOLUTIONDATE)

ELSE DATEDIFF(day,jira.CREATED,GETDATE())

END

) as openDays,

(

CASE

WHEN status.STATUS\_DESC IN ('Closed', 'Closed, Duplicate', 'Closed, Not a Problem', 'Closed, Not Reproducible', 'Closed, Passed Verification', 'Closed, Won''t Fix', 'Master Closed') THEN DATEDIFF(day,jira.CREATED,jira.RESOLUTIONDATE)/ 30.436875E

ELSE DATEDIFF(day,jira.CREATED,GETDATE())/ 30.436875E

END

) as openMonths

FROM importBpmsGlobals bpms

LEFT JOIN importJiraIssue jira ON jira.pkey = bpms.jiraIssueKey

LEFT JOIN PROJECT\_DIM project ON jira.PROJECT = project.PROJECT\_CD

LEFT JOIN STATUS\_DIM status ON jira.issuestatus = status.STATUS\_CD

LEFT JOIN TYPE\_DIM type ON status.TYPE\_ID = type.TYPE\_ID

LEFT JOIN DATE\_DIM dCreated ON CONVERT(date,jira.Created) = dCreated.Date

LEFT JOIN DATE\_DIM dClosed ON CONVERT(date,jira.RESOLUTIONDATE) = dClosed.Date

WHERE type.TYPE\_DESC = 'Global RFSS'

END

# Appendix C – *calculateGlobalAverageAge* Function Code

CREATE FUNCTION [dbo].[calculateGlobalAverageAge]

(

-- PARAMETERS

@projectId int, --DIM value of project

@date int --DIM value of date

)

RETURNS

@resultTable TABLE

(

-- Table columns that will be returned

countOpenGlobals INT,

globalsAverageOpenAgeDaysValue INT,

globalsAverageOpenAgeMonthsValue DECIMAL(10,3),

globalsAverageOpenAgeMonthsScore INT,

remarks VARCHAR(50)

)

AS

BEGIN

DECLARE @countOpenGlobals INT = 0

DECLARE @globalsAverageOpenAgeDaysValue INT = 0

DECLARE @globalsAverageOpenAgeMonthsValue DECIMAL(10,3) = 0.000

DECLARE @globalsAverageOpenAgeMonthsScore INT = 0

DECLARE @remarks VARCHAR(50) = ''

-- Return if both parameters not sent

IF @projectId IS NULL OR @date IS NULL

BEGIN

SET @countOpenGlobals = -1

SET @globalsAverageOpenAgeDaysValue = -1

SET @globalsAverageOpenAgeMonthsValue = -1

SET @globalsAverageOpenAgeMonthsScore = -1

SET @remarks = 'No all parameters sent to calculateGlobalAverageAge function.'

INSERT INTO @resultTable VALUES (@countOpenGlobals,@globalsAverageOpenAgeDaysValue,@globalsAverageOpenAgeMonthsValue,@globalsAverageOpenAgeMonthsScore,@remarks)

RETURN

END

--Return if invalid project

DECLARE @projectCount int

SET @projectCount =

(

SELECT COUNT(\*)

FROM PROJECT\_DIM

WHERE PROJECT\_ID = @projectId

)

IF @projectCount IS NULL OR @projectCount = 0

BEGIN

SET @countOpenGlobals = -1

SET @globalsAverageOpenAgeDaysValue = -1

SET @globalsAverageOpenAgeMonthsValue = -1

SET @globalsAverageOpenAgeMonthsScore = -1

SET @remarks = 'Project does not exist in dimension table.'

INSERT INTO @resultTable VALUES (@countOpenGlobals,@globalsAverageOpenAgeDaysValue,@globalsAverageOpenAgeMonthsValue,@globalsAverageOpenAgeMonthsScore,@remarks)

RETURN

END

--Return if invalid date

DECLARE @dateCount int

SET @dateCount =

(

SELECT COUNT(\*)

FROM SQM\_GLOBALS\_DAILY

WHERE ENTITY\_DATE\_ID = @date

)

IF @dateCount IS NULL OR @dateCount = 0

BEGIN

SET @countOpenGlobals = -1

SET @globalsAverageOpenAgeDaysValue = -1

SET @globalsAverageOpenAgeMonthsValue = -1

SET @globalsAverageOpenAgeMonthsScore = -1

SET @remarks = 'Data does not exist for the date provided.'

INSERT INTO @resultTable VALUES (@countOpenGlobals,@globalsAverageOpenAgeDaysValue,@globalsAverageOpenAgeMonthsValue,@globalsAverageOpenAgeMonthsScore,@remarks)

RETURN

END

--Return if no data exists for the project in the globals table

DECLARE @projectGlobalCount int

SET @projectGlobalCount =

(

SELECT COUNT(\*)

FROM SQM\_GLOBALS\_DAILY

WHERE PROJECT\_ID = @projectId

)

IF @projectGlobalCount IS NULL OR @projectGlobalCount = 0

BEGIN

SET @countOpenGlobals = 0

SET @globalsAverageOpenAgeDaysValue = 0

SET @globalsAverageOpenAgeMonthsValue = 0

SET @globalsAverageOpenAgeMonthsScore = 0

SET @remarks = 'This project has no global records, either open or closed.'

INSERT INTO @resultTable VALUES (@countOpenGlobals,@globalsAverageOpenAgeDaysValue,@globalsAverageOpenAgeMonthsValue,@globalsAverageOpenAgeMonthsScore,@remarks)

RETURN

END

INSERT INTO @resultTable

SELECT COUNT(\*) as countOpenGlobals,

AVG(OPEN\_DAYS) as globalsAverageOpenAgeDaysValue,

AVG(OPEN\_MONTHS) as globalsAverageOpenAgeMonthsValue,

dbo.getMetricScore('GAAM','Project',AVG(OPEN\_MONTHS)) as globalsAverageOpenAgeMonthsScore,

(

CASE

WHEN COUNT(\*) = 0 THEN 'No open globals'

ELSE ''

END

) as remarks

FROM SQM\_GLOBALS\_DAILY

WHERE ENTITY\_DATE\_ID = @date

AND PROJECT\_ID = @projectId

AND CURRENT\_STATUS NOT IN ('Closed', 'Closed, Duplicate', 'Closed, Not a Problem', 'Closed, Not Reproducible', 'Closed, Passed Verification', 'Closed, Won''t Fix', 'Master Closed')

GROUP BY ENTITY\_DATE\_ID, PROJECT\_ID

RETURN

END