Design

Royal Dutch Shell

Eve Program

August 2017



# **DOCUMENT INFORMATION**

**Project Name**: Eve – Release 3

**Prepared by**: Samarendra Bisoyi

**Title**: FM Data File Archiving

**Document Version Number**: 1.1

**Document Version Date**: 05/09/2017

**Reviewed by**: Nabil

**Review Date**: DD/MM/YYYY

**Distribution List**:

|  |  |  |
| --- | --- | --- |
| **From** | **Email** | **Date** |
| XXX | [xxx.yyy@zzz.com](mailto:xxx.yyy@zzz.com) | DD/MM/YYYY |

|  |  |  |  |
| --- | --- | --- | --- |
| **To** | **Email** | **Action\*** | **Due Date** |
| XXX | [xxx.yyy@zzz.com](mailto:xxx.yyy@zzz.com) |  | DD/MM/YYYY |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

*\* Action Types: Approve, Review, Inform, File, Action Required, Attend Meeting, Other (please specify)*

**Version history**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Date** | **Prepared / Modified by** | **Reviewed by** | **Approved by** | **Section and Text Revised** |
| 0.1 | ##/##/#### | XXXX |  |  | DXC template created. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# **TABLE OF CONTENTS**

Contents

[**DOCUMENT INFORMATION** 2](#_Toc492398856)

[**TABLE OF CONTENTS FOR FM DATA FILE PURGING** 3](#_Toc492398857)

[**USE CASE - # 1: Creating a Generic Script for all the Environment:** 5](#_Toc492398858)

[**USE CASE - # 2: Segregate the data file paths based on OU:** 5](#_Toc492398859)

[**USE CASE - # 3: Create a condition to check folders based on OU:** 6](#_Toc492398860)

[**USE CASE - # 4: Create a function to archive and purge files.** 7](#_Toc492398861)

[**USE CASE - # 5: Execution Steps of the script.** 8](#_Toc492398862)

[**TABLE OF CONTENTS FOR FM DB DATA PURGING** 3](#_Toc492398857)

**Introduction:**

This script is prepared for archiving/purging all the input and output data files which has been processed or generated through file manager. The script shall archive/purge data files based on the OU and date range passed to the script.

# **USE CASE - # 1: Creating a Generic Script for all the Environment:**

**Create a generic script to accept 4 parameters.**

1. **Environment ( ex : DEV/SIT/UAT)**
2. **OU( ex: CA/HK/MO)**
3. **Start Date (format : <yyyy-mm-dd>)**
4. **End Date(format : <yyyy-mm-dd>)**

**Allow user to enter the parameters in any case.**

**Then Script will validate the parameters and set the pcard\_profile path according to the environment.**

**# Script to verify the parameters**

if [ $# == 4 ]

then

echo "Performing purging activity in $1 environment .."

ENV=`echo $1| tr '[:lower:]' '[:upper:]'`

if [ $ENV == "MOPS" ]

then

PCARD=/path/to/mops/pcard\_profile

elif [ $ENV == "PROD" ]

then

PCARD=/opt/pwrcard/PROD

exit 1

else

echo "Trying in some other environment... need to modify the script..."

exit 1

fi

else

echo "Parameter need to be passed to the script : $0 <env name> <OU> <Start date : yyyy-mm-dd> <End date

: yyyy-mm-dd>"

exit 1

fi

# **USE CASE - # 2: Segregate the data file paths based on OU:**

**Segregate the data file paths based on ou and type of directories.**

**For example : CA having inbound, outbound and archive directories from which files need to be archived.**

**So first we will create variables each pointing to file path and we will use proper naming convention to identify what type of file path and for which country is this.**

**Then we will create arrays of array to segregate all the file paths based on OU.**

#CA Output Directories

caoutdir1=$OUTBOUND\_DATA\_HOME/CAO01

caoutdir2=$OUTBOUND\_DATA\_HOME/CAO02

…..

#CA Inbound Archived Directory

cainarchdir1=$INBOUND\_ARCHIVE\_HOME/ca/CAI01

cainarchdir2=$INBOUND\_ARCHIVE\_HOME/ca/CAI02

….

#CA Outbound Archived Directory

caoutarchdir1=$OUTBOUND\_ARCHIVE\_HOME/ca/CAO01

caoutarchdir2=$OUTBOUND\_ARCHIVE\_HOME/ca/CAO02

….

#Array of ca folders based on folder type

declare -a caoutdir=("$caoutdir1" "$caoutdir2" …)

declare -a cainarchdir=("$cainarchdir1" "$cainarchdir2" …)

declare -a caoutarchdir=("$caoutarchdir1" "$caoutarchdir2" …)

#Array of all ca folders

declare -a caarr=("${caoutdir[@]}" "${cainarchdir[@]}" "${caoutarchdir[@]}")

# **USE CASE - # 3: Create a condition to check folders based on OU:**

|  |
| --- |
| **Create a condition to check for the 2nd parameter passed to the script i.e. OU.**  **Based on the OU value, it iterate over the array of that particular OU using for loop to check each and every folder whether it’s existing or not. If existing then we need to call a function which will perform the purging and archiving of data files.**  if [ $2 -eq "CA" ];then  for i in "${caarr[@]}"  do  echo "$i"  if [ -d "$i" ]  then  echo "Directory $i exists"  if [ "$(ls -A $i)" ]; then  echo "Take action $i is not Empty"  fun $i  else  echo "$i is Empty"  fi  else  echo "No $i Directory available"  fi  done  fi |

# **USE CASE - # 4: Create a function to archive and purge files.**

**Create a common function to be called each time for each OU values based on the 2nd parameter passed to the script. Pass the folder paths one by one to the function if it exists in the above condition.**

**##Common function for archiving and purging activity**

fun () {

echo "Checking files under : $1"

#arr\_Filenames=($1/\*.\*)

#FileCount=${#arr\_Filenames[@]}

FileCount=`find $1/ -type f -newermt "$StartDate" ! -newermt "$EndDate" | wc -l`

echo "no of file to be archived is $FileCount"

if [ $FileCount != 0 ]

then

files=`find $1/ -type f -newermt "$StartDate" ! -newermt "$EndDate" | rev | cut -d '/' -s -f1 |

rev`

Dir=`echo $1 | rev | cut -d '/' -s -f1,2 | rev`

mkdir -p $Archive\_Dir/$Dir

echo "Archiving Directory : $Archive\_Dir/$Dir"

cd $1

cp -p $files $Archive\_Dir/$Dir

shred -u $files

#done

fi

}

**This function should find files inside each path based on the date range. And it will take the path of that folder and create an archive directory with the same directory structure. Then move all the files to the archived folder.**

# **USE CASE - # 5: Execution Steps of the script.**

**./FM\_DATA\_FILE\_PURGING\_MOPS.sh <env name> <OU> <Start date : yyyy-mm-dd> <End date : yyyy-mm-dd>**

Total 4 no of parameters to be passed to the script.

1. **Environment ( ex : DEV/SIT/UAT)**

**This parameter defines in which environment we are running the script.**

1. **OU( ex: CA/HK/MO)**

**This parameter defines for which country we need to archive or purge files.**

1. **Start Date (format : <yyyy-mm-dd>)**

**Start date is the starting date range to find files which need to be archived/purged.**

1. **End Date(format : <yyyy-mm-dd>)**

**End date is the ending date range to find files which need to be archived/purged.**

**Files are backed up under $Archived\_Folder before purging for the safety purpose.**

**If any file deleted by mistake then it can be recovered under archive folder having same folder structure like the file archived from.**

**USE CASE - # 1: Creating a Generic Script for FM DB data purging:**

**Create a generic script to accept 4 parameters.**

1. **Environment ( ex : DEV/SIT/UAT)**
2. **OU( ex: CA/HK/MO)**
3. **Start Date (format : <yyyy-mm-dd>)**
4. **End Date(format : <yyyy-mm-dd>)**

**Allow user to enter the parameters in any case.**

**Then Script will validate the parameters and set the pcard\_profile path and FM DB credentials according to the environment.**

**Connect to the DB and set the spool on. So that it will write the sql query results into a file.**

**Create the query to remove all FM data based on OU and date range.**

For example :

delete from JOBS where JOBNAME like '%$OU%' and REQUESTDATE between '$Start\_Date' and '$End\_Date';

**That query should use the user passed parameters such as OU, Start Date and End Date.**

* $Start\_Date and $End\_Date should be used date range which you want to purge.
* <Start date> : start date is the date from where we will purge data from table.
* < End date> : end of date . suppose today is the end date then it will purge the data till yesterday.
* Date format example : 20-MAY-17