Architectural Design

Servicing Warehouse

## 

## About the Document

The Architecture document specifies details of how an interface works from end to end. Details on required functionality of the sending, receiving and middleware applications are captured here.

The information in this document will be used as input for Technical Designs in the sending, receiving and middleware applications

Table of Contents

[About the Document 2](#_Toc462148745)

[1. Interface Properties: 3](#_Toc462148746)

[2. High Level Data Flow Design: 4](#_Toc462148747)

[3. Data Profile 5](#_Toc462148748)

[4. Database Structures 5](#_Toc462148749)

[5. SCD Strategy 6](#_Toc462148750)

[a. SCD Indicator: 6](#_Toc462148751)

[6. Audit Strategy 7](#_Toc462148752)

[7. Exception Handling: 7](#_Toc462148753)

[8. Restartability: 7](#_Toc462148754)

# Interface Properties:

|  |  |
| --- | --- |
| **Interface Details** | Servicing Warehouse |
| **Sending System(s)** | * STG\_LSAMS * STG\_GAMECHANGERS * STG\_TMO * STG\_EXTNLDATAFILES * STG\_TKAMS * STG\_PVM |
| **Receiving System(s)** | Staging : SVC\_STG  Warehouse : SW\_DWH |
| **Data Send Frequency** | Daily [Mon – Fri] |
| **Volumes & Sizing** | First Time Load Volume: Approximately between 2.5 and 3 Million Loans  Daily Transactional Volume Average : TBD  Daily Transactional Volume Peak : TBD |
| **SLAs** | TBD |
| **Criticality** | P2 - The data is business critical and will impact reporting |

# High Level Data Flow Design:





# Data Profile

For ETL processing there are a set of daily processing requirements, where changes and additions to source data are extracted and processed through to the system in nightly batches.

# Database Structures

***Source:***

Data for Servicing Warehouse is sourced from Netezza Database from a list of Databases as mentioned below:

* STG\_LSAMS
* STG\_GAMECHANGERS
* STG\_TMO
* STG\_EXTNLDATAFILES
* STG\_TKAMS
* STG\_PVM

Only transactional data is pulled from the source.

***Staging:***

Staging is hosted in SQL Server with a new schema. The staging holds the incremental data required for servicing warehouse.

***Warehouse:***

Warehouse is hosted in SQL Server with a new schema. The warehouse holds historical data [SCD II]



# SCD Strategy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ETL LOGIC** | | | | |
| **Source Extract** | **CDC Logic** | **Load Type** | **Restartability** | **Comments** |
| 1)Source Data from LSAMS will be extracted  2)Only Transactional Data will be extracted from the Source | 1)Source data will be loaded into temp table 2)The Current data in the temp table will undergo a minus operation against the Active records in the warehouse table. Based on the Loan\_key Value ; if it is NULL data from temp table will be loaded into the Staging tables with the Indicator as Y. 3) If the Loan\_key value is NOT NULL then the records have to updated which will be carried out in 3 steps, Step 1 : Insert the Old Values into Staging table with the Indicator as N; Step 2: Insert the New Values into the staging table with the Indicator as Y; Step 3: Delete the Old Values in the warehouse for the same Loan\_key which has the Current Indicator as N in Staging and Y in Warehouse 4) Bulk Insert all the records from Staging into Warehouse. | Delete then  Insert | In case of abort the support team will put the table load on hold and based on the issue will decide to start from the abort instance or initiate a fresh instance | Deleted records will be performed separately which takes care of the records permanently deleted from the Source. |

## SCD Indicator:

|  |  |  |
| --- | --- | --- |
| **Record Type** | **CURR\_IND** | **CURR\_SOR\_IND** |
| Insert | Y | Y |
| Update | N [Old Record] | Y [Old Record] |
| Y [New Record] | Y [New Record] |
| Delete | N | N |

Downstream applications or reporting teams can get the active records based on the Current Source Indicator and Current Indicator value

# Audit Strategy

* **Execution tracking**: High-level ETL execution details are captured into an audit table.
* Every Batch will have an unique Batch Execution Identifier
* Each Batch will be in any 1 of the below 3 status
* Started
* Completed
* Aborted
* If a batch is aborted the same batch id will be reused until it goes into Completed state. In this case for the same day there will be multiple records for the same Batch id with 2 different statuses.

***Sample Data:***

|  |  |
| --- | --- |
| **Batch Id** | **Status** |
| 10 | Aborted |
| 10 | Started or Completed |

# Exception Handling:

Any failures during the batch process will be communicated to the support team, business group and data architects via email

All exceptions will be logged into the audit table as “Aborted”. Upon restarting the same Batch ID which was put into Aborted will be reused for the new run post fixing the issue.

# Restartability:

The datastage batch will be designed in a way to restart from the point of failure. Staging is Truncate and Load hence partial load will be handled.

Warehouse load is through database minus hence partially loaded records will not be reprocessed as they would be captured as Copy records.

**Data Point Links:**

[**Data Model**](http://teams/sites/dataservices/Shared%20Documents/Forms/AllItems.aspx?RootFolder=%2Fsites%2Fdataservices%2FShared%20Documents%2FAll%20Projects%20-%20Central%20Artifacts%2FServicing%20Warehouse%2FETL%2FServicing%20WareHouse%2FData%20Model&InitialTabId=Ribbon%2EDocument&VisibilityContext=WSSTabPersistence)**:**

[**JIRA Link:**](https://nationstar.atlassian.net/browse/DSE-2)

[**STTM Link:**](http://teams/sites/dataservices/Shared%20Documents/Forms/AllItems.aspx?RootFolder=%2Fsites%2Fdataservices%2FShared%20Documents%2FAll%20Projects%20-%20Central%20Artifacts%2FServicing%20Warehouse%2FETL%2FServicing%20WareHouse&InitialTabId=Ribbon%2EDocument&VisibilityContext=WSSTabPersistence)