**Exception Handling & Logging**

**Table Structure:**

# **Logger\_Table**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| APPLICATIONNAME | VARCHAR2(40 BYTE) | Name of the application. |
| LOGDATA | VARCHAR2(500 BYTE) | Stored complete logging details. |
| ACTIVITYNAME | VARCHAR2(40 BYTE) | Logging activity in code. |
| ENVIRONMENT | VARCHAR2(40 BYTE) | Environment like Dev/QA/Stag/Prod. |
| DOMAIN | VARCHAR2(40 BYTE) | Domain where the application is deployed. |
| INTEGRATIONNAME | VARCHAR2(40 BYTE) | Name of the process. |
| LOGLEVEL | VARCHAR2(40 BYTE) | Log level should be Error/Debug. |
| TIMESTAMP | VARCHAR2(40 BYTE) | Current-dateTime of the log. |
| SOURCE | VARCHAR2(40 BYTE) | Name of Source application. |
| DESTINATION | VARCHAR2(40 BYTE) | Name of target application. |
| BUSINESSKEY | VARCHAR2(40 BYTE) | It should be either Key element of the request or processId. |
| TRANSACTIONID | VARCHAR2(40 BYTE) | Unique Identifier of a transaction. |

# **Error\_Table**

# 

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| TRANSACTIONID | VARCHAR2(50 BYTE) | Unique Identifier of a transaction. |
| BUSINESSKEY | VARCHAR2(40 BYTE) | It should be either Key element of the request or processId. |
| RETRYCOUNT | NUMBER | No. of retries required for failed transaction. Part of further enhancement. |
| RETRYFLAG | VARCHAR2(20 BYTE) | True if retry is required, false if not. Part of future enhancement. |
| NOTIFYFLAG | VARCHAR2(20 BYTE) | True if notification is required, false if not. Part of future enhancement. |
| PROCESSNAME | VARCHAR2(50 BYTE) | Name of the processes. |
| APPLICATIONNAME | VARCHAR2(50 BYTE) | Name of the application. |
| ACTIVITYNAME | VARCHAR2(50 BYTE) | Error activity in code. |
| STACKTRACE | CLOB | Stack trace of the error ie. Error details |
| SOURCE | VARCHAR2(30 BYTE) | Name of Source application. |
| DESTINATION | VARCHAR2(30 BYTE) | Name of target application. |
| ENVIRONMENT | VARCHAR2(40 BYTE) | Environment like Dev/QA/Stag/Prod. |
| ERRORTYPE | VARCHAR2(40 BYTE) | Type of error eg. Functional/Technical. |
| DOMAIN | VARCHAR2(40 BYTE) | Domain where the application is deployed. |
| INTEGRATIONNAME | VARCHAR2(30 BYTE) | Name of the process. |
| TIMESTAMP | VARCHAR2(40 BYTE) | Current-dateTime of the logged error. |
| ERROR\_CODE | VARCHAR2(50 BYTE) | Stores error code. |
| ERROR\_MESSAGE | VARCHAR2(1000 BYTE) | Stores error message/details. |

# **Error\_Data**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| TRANSACTIONID | VARCHAR2(40 BYTE) | Unique Identifier of a transaction. |
| INPUTDATA | CLOB | Stored complete request details of an error transaction. |

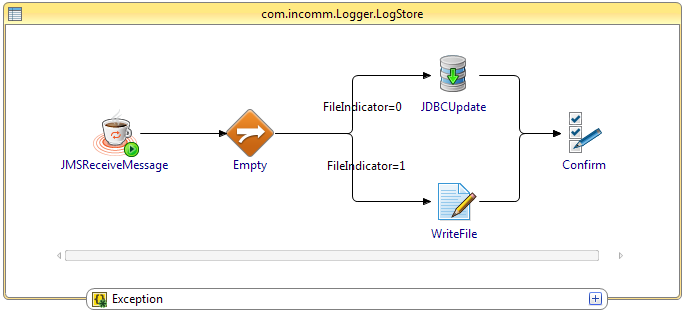
**Application Structure:**

Basically Exception Handling & Logging application consists of two main processes:

1. Logging Process:

This implementation is based on pub-sub model, in which all the application published the logging data to a queue “COM.INCOMM.COM.SHAREDSERVICES.ERRORHANDLING.LOGQUEUE”.

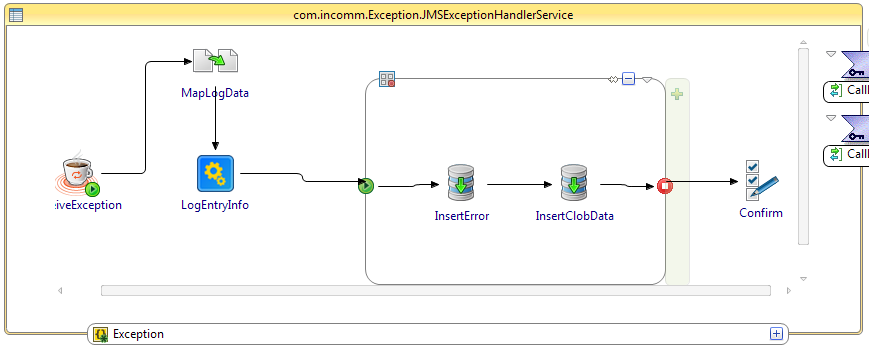
Logging process listens & picks the data from log queue and persist it to the persistence storage (database/file). Current as a part of this application all the data is persisted into database.



**Fig 1:- Logging Process**

1. Exception Process.

This process is similar to logging process. It also follows pub-sub implementation model, in which all the application publishes the error information to a queue “COM.INCOMM.COM.SHAREDSERVICES.ERRORHANDLING.EXCEPTIONQUEUE”. Error process listens & picks the data from error queue, logs the logging information into logger\_table and then persist the error data into error\_table & error\_data tables.



**Fig 2:- Exception Process**