Infosys | Finacle

FINACLE SUPPORT CONNECT

Knowledge Series

April 2021 | Volume 25

Finacle Support brings you this fortnightly knowledge bulletin to augment your problem-solving capability. There is more to it. Every edition is put together with utmost diligence to ensure that best practices and known resolutions are shared. In this edition you will find the following articles:

- **Posting Zero Amount Interest Transactions**
- **Procedure to Enable User Specific Application Logging**
- **Activating Duplicate Check Functionality**

So let's start reading!

Posting Zero Amount Interest Transactions

Product: Finacle Core Banking Version: 10.2.14 onwards

During the interest calculation in some specific scenarios, Finacle Core Banking gives the error **The transaction amount for the transaction ID is less than zero**. This error comes when the calculated interest amount is zero.

A new **COMT** parameter **POST_ZERO_INTEREST_TRANSACTION** has been introduced. This allows the system to post interest transactions with zero amount. The parameter in the **COMT** table should be set as **Y**. This feature is applicable for credit interest-earning accounts (**TDA** and operative accounts such as SBA, CAA, CCA and ODA).

The required **COMT** setups are:

1. **POST_ZERO_INTEREST_TRANSACTION** parameter has to be added and set as **Y** in the **COMT** table. All backend services (**ConfigService**, **CoreSession**, and **Finlistval**) need to be restarted
2. At the scheme Level (**HGSPM**), the **Interest Receivable Flag** under the **Interest** tab should be set as **Y**

Procedure to Enable User-Specific Application Logging

Product: Finacle Online Banking Version: 11.2.x

There are situations where analyzing the logs become difficult, especially in cases where the issue is faced by only a particular user(s). To handle such scenarios, there is a feature in Finacle Online Banking which enables user specific logs. In this, the required logs can be configured and can be captured as a single file for the specific user or users.

The steps to enable **User Specific Logging** are as follows:

1. Configure the below parameters in the **WorkingDirectory/data/LogConfig.xml** or **LogConfig.xml** to enable user specific logging in **FEBA 11.2.x**
   - **AdditionalLoggingRequired:** Value to be set as **Y**
   - **AdditionalLoggingCriteria_USER_ID**: Accepts regular expression patterns
     E.g. **CORPID.*** will enable logging of all user IDs starting with **CORPID**. **TESTUSER** will enable logging for User ID **TESTUSER**. Multiple user IDs can be configured by using a comma separator
   - **AdditionalLoggingLogLevel_USER_ID**: Accepts single log level or a combination
     E.g. **ALL** should be enabled for all levels of log or can specify the individual log levels separated by comma
     Sample:
     <Param name = "AdditionalLoggingRequired" value = "Y"/>

<Param name = "AdditionalLoggingCriteria_USER_ID" value = "CORP1.USER1, CORP2. *, USER2"/>
<Param name = "AdditionalLoggingLogLevel_USER_ID" value = "FATAL, ERROR, DEBUG, SETUP, MESSAGE, WARN"/> **OR** <Param name = "AdditionalLoggingLogLevel_USER_ID" value = "ALL"/>

2. Application restart is required for the changes to be reflected
3. The log file will get generated with the name **$BANK_ID$_ $USER_ID$.log** under the logs folder
   E.g. If the User ID is **USER1** and the Bank ID is **TESTBANK** - the generated log file name will be **TESTBANK_USER1.log**

Activating Duplicate Check Functionality

Product: Finacle Core Banking (Finacle Integrator)

The duplicate check functionality helps to check the same request occurring twice from an external channel via Finacle Integrator. There are two ways in which a duplicate check can be enabled for requests from different channels:

1. **Duplicate Check at Core:** An entry should be made in the **TFS** table for a particular service name with the **duplicate_tran_chk_enabled** as **Y**, which will enable the duplicate check. This will ensure that an entry is made in the **UTT** table at the core database for transactions initiated from the particular service name configured (**APIs**). The duplicate check is performed at the core end by looking into a combination of these four fields in the **UTT** table:
   Request Origination Time (**messageDateTime** tag), Request Type, Request Reference Number (Backend Reference Number), and DCC ID (Channel type)
   Sample TFS table entry for XferTrnAdd and TDAcctAdd APIs –

	SERVICE_NAME	DUPLICATE_TRAN_CHK_ENABLED	LCHG_USER_ID	LCHG_TIME	RCRE_USER_ID	RCRE_TIME	TS_CNT
1	SRV_AddAndPostXferTran	Y	USER1	05-DEC-16	USER1	05-DEC-16	0
2	SRV_OpenTDAcctAndVerify	Y	USER1	05-DEC-16	USER1	05-DEC-16	0

2. **Duplicate Check at FI:** An entry should be made with the **API** name inside the **ServiceMessageLogging.properties** file which is present in the **<FI_PATH>/common/config/fsb**. E.g., **XferTrnAdd = D,Y,Y**. This will ensure that an entry is made in the **FIMASTER** transaction tables for the request raised through that **API**
   The duplicate check is performed at the **FI** end by looking into a combination of these 3 fields in the **FIUSB_TRANSACTION_TABLE**: Req Reference Number (**FE_RQ_REF_NUM**), Channel ID (**ORIGINATING_CHANNEL_ID**), and Request Type (**FIUSB_SRV_REQ_TYPE**)

   Sample : **XferTrnAdd = D,Y,Y** (**D** – Detail Logging along with response structure, **Y** – Duplicate check enabled, **Y** – request can be retried in case of failure)

   TDAcctAdd = H,Y,N (**H** – Only header details logged without response structure, **Y** – Duplicate check enabled, **N** – request can't be retried in case of failure)

In addition to the entry in **ServiceMessageLogging.properties** file, the **TFS** entry can also be added to ensure the duplicate check at the **Core** end. This will be helpful in cases wherein the **FI** request has failed due to the **FI** timeout, whereas the **Core** has processed and committed the request at a later point of time.

In such a case, if the same request is fired again (when retry flag is set as **Y** for that **API**), the system will check the **UTT** table for a duplicate check and if the record exists, then it will fetch the **Transaction ID** (posted at the **Core** database) and sends the response as **Success** with the transaction details, instead of creating a duplicate transaction.

Hope you like this edition. Is there anything that you'd like to see in the forthcoming series? We'd love to hear from you!

Write to us at finaclesupport@edgeverve.com