Reporting and Analytics Aggregates User's Guide

How Do I Troubleshoot Aggregation?

5/4/2025

How Do I Troubleshoot Aggregation?

If you find that the data you expect is not present in the aggregation tables, perform the troubleshooting steps described on this page to ensure proper configuration of your aggregation environment. Correcting these common problems should resolve the majority of issues that you are likely to encounter:

Check for Aggregation Misconfiguration

The following table provides information about errors, and describes configuration checks that you can perform to diagnose and correct configuration problems.

Symptom	Diagnosis
The following error is logged: `SCfg.Err Mapping not found, map: media-code, key: SomeKey`	RAA has encountered a media of unknown type.
The following error is logged: `Agg.SCfg.Err Mapping not found, map: resource-queue, key: someKey`	RAA logs this error when it detects queue configuration on the wrong type of DN or configuration detects a valid resource (such as a queue) for which thresholds are configured, but Genesys Info Mart has not yet added this resource to its RESOURCE_ table.
Not all RAA hierarchies are populated.	Check configuration for disabled hierarchies. If any hierarchy value is specified for the default option in the `[agg-populate-disable]` section of your Genesys Info Mart **Application** object, the aggregation process will not populate that hierarchy. If the appropriate hierarchies are enabled, check that their subhour views (for those hierarchies that have SUBHR views) are able to retrieve data. If they cannot, check the configuration of underlying Genesys Info Mart FACT tables.
Aggregation consumes too much of Info Mart's resources and performance is slowed.	Check how many writers are defined. Although **numberOfWriters** runtime parameter and the **number-of-writers** configuration option accept values up to 16, and the **writer-schedule** configuration option (**writerSchedule** runtime parameter) accepts values up to 10, Genesys Info Mart runs on the same server, and requires resources to do so. Configure these values within the limitations of available processing power. Check database performance, statistics on the tables themselves, and other systems that might compete for the same resources.

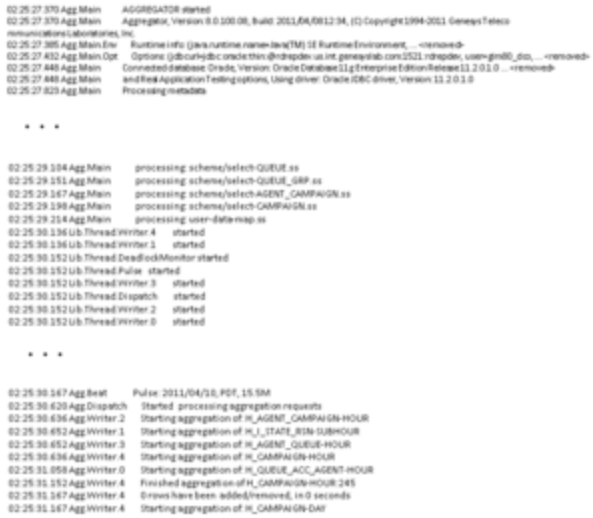
Symptom	Diagnosis
The aggregation process does not start.	Check the value of the **aggregation-engine-class-name** configuration option. If this option is not properly set, the aggregation process will not begin. Check the placement of the agg subdirectory and that the **GIM_ETL_PATHS** shell script points to the aggregation executable: `set GIM_EXT_LIBS=./agg/GIMAgg.jar`. When invoking aggregation in integrated mode, confirm that the aggregation schedule (controlled by the aggregate-schedule configuration option) permits the aggregation process to run. The **run-aggregates** configuration option must also be set to `true`.

Verify that Data Aggregation Has Begun

The aggregation process performs a number of operations before data aggregation begins. Among these operations are Info Mart connection verification; synchronization of tables and columns (if necessary, to ensure that the proper fields exist and are of the correct data type); and processing of the internal component Scheme files that define each aggregation hierarchy, software patches, language files, and user data configuration.

It is possible that the aggregation process will cease before initialization completes for any number of reasons—for example, lack of table space, RDBMS issues, and write-to-db problems, just to name a few. In this circumstance, data aggregation will not commence and an appropriate message, such as the following, will be logged:

```
16:48:39.745 Lib.Thread.AggManager     caught an exception. Monitor: Writer.2 Message:
Writer.2: unable to execute command: Agg
16:48:39.745 Lib.Thread.AggManager     SQLState: 23000
16:48:39.745 Lib.Thread.AggManager     Vendor:   1
16:48:39.745 Lib.Thread.AggManager     java.sql.BatchUpdateException: ORA-00001: unique
constraint (GIM_SG1_1.PK_AG2_I_STATE_RSN_SUBHR) violated
16:48:39.745 Lib.Thread.AggManager     Stack Trace:
```



Excerpt from a Typical Genesys Info Mart Log
(Click to Enlarge)

You can confirm whether data aggregation has begun by viewing the Genesys Info Mart log (by default, named `gim_etl.log`) and looking for "`Thread.Writer ... started`" messages. The log snippet in the Figure **Excerpt from a Typical Genesys Info Mart Log**, for example, shows that five writer threads have been opened. These threads, numbered 0 through 4 in the snippet, correspond to the value defined by the **number-of-writers** configuration option or **numberOfWriters** runtime parameter depending on the mode of aggregation operation (integrated or autonomous).

> ### Tip
>
> The Figure **Excerpt from a Typical Genesys Info Mart Log** shows only aggregation-related log messages. If you run aggregation in integrated mode, other messages will be dispersed throughout the log to notify you about the behavior of other Genesys Info Mart jobs at key junctures.

Check the Content of Source FACT Tables

If data is not being written to aggregate tables:

- Verify that data aggregation has begun and ended for one or more hierarchies.
- Check the content of the source Genesys Info Mart FACT tables to ensure that content exists to be aggregated. Find out which FACT tables support the hierarchy. For information about how to view the SQL for RAA hierarchies, see How Do I View the Aggregation Query?.
- With an appropriate join on INTERACTION_ID to the STG_TRANSFORM_DISCARDS Info Mart table, determine if extraction was complete.

Isolate Aggregation-Related Messages in the Log

Aggregation runs asynchronously with extraction, transformation, and other Genesys Info Mart jobs that share the same processor and memory space. Log entries are directed toward the same output. You can view RAA logs using Genesys Administrator, or alternatively, you can isolate aggregation-related messages from messages written by other Genesys Info Mart jobs, by performing either of the following:

- **If aggregation is running in integrated mode, filter the log:** If Genesys Info Mart log output is directed to a file, run a filter against the log to extract aggregation-related messages. All aggregation-related messages are prefaced with a timestamp of *hh:mm:ss.ddd* format, for example: `2014-05-19 14:28:37,323 DEBUG Agg.DeadlockMonitor 35000 started` The following command creates a new output file, named `gim_agg.log`, that contains aggregation-related messages only:

```
grep “[0–2][0–9]:[0–5][0–9]:[0–5][0–9]\.[0–9][0–9][0–9]”
                gim_etl.log > gim_agg.log
```

  Note, however, that this command does not display log-event messages that are related to aggregation configuration, exceptions, connection and job status, or memory. Log-event messages that are generated by the Genesys Info Mart server have a predictable format. Refer to the Genesys Info Mart section of Framework Combined Log Events Help for further information.

- **Run aggregation in autonomous mode:** If aggregation is operating in integrated mode, disable it, and run it in autonomous mode from the command line. Subsequent output is related exclusively to the aggregation job. Also, issue the **-log** runtime parameter and log file, which directs all output to the specified file (otherwise, output is directed to the console).

Check for Congestion at Peak ETL Periods

By default, Genesys Info Mart maintenance begins daily at 3:00 AM. This is controlled by the values of the maintain-start-time and run-maintain configuration options. For large environments, Genesys recommends that you avoid running the aggregation process in autonomous mode during this period and during high loads.

Run updateAliases for Missing Tenant Data

Whenever the aggregation schema changes or you add tenants to your environment, you should update tenant aliases to modify and/or create new views of tenant data. Otherwise, existing aliases might become unusable, and the subset of queries that are based on the existing tenant views might not retrieve the data that you expect. Schema changes potentially occur with the deployment of hot fixes, upgrade to a new release, migration and, of course, your own database customizations (which Genesys does not support). You can update tenant aliases by running aggregation in autonomous mode and specifying the **-updateAliases** runtime parameter on the command line. For more information about the circumstances under which the update of tenant aliases must be run, and how to configure the accompanying tenant alias file, see What Do I need to Know About Managing Multi-Tenant Environments?. Should you encounter errors while running this alias update, check the log for any of the following and correct the problem:

- The specified tenant account might not exist.
- The account might have insufficient permissions to connect to the database.
- The account might lack permissions to create database objects (views).

Note that the update skips any problematic objects or accounts that it encounters, and proceeds to process the next object or account in the tenant alias file.

Check for Long-Running Interactions

Asynchronous interactions can be long-running—enduring on the order of several days, months, even years. This active interaction state can persist because of technical reasons—Genesys Info Mart might not terminate interactions that are stuck for some reason—or for legitimate business reasons, as in the case in which interactions should be kept active purposefully until a rather time-consuming process completes. Months could pass, for example, before a loan-processing interaction is funded. As described in What is Aggregation and How Do I Enable It?, Genesys Info Mart sends notifications about data that is ready for aggregation. RAA receives these notifications and performs aggregation for the entire length of time in which the interactions were active. For long-running interactions, this activity can generate problems that are manifested as:

- Arithmetic overflow in the Genesys Info Mart log. Most duration fields that RAA populates are measured in number of seconds. The number of elapsed seconds for long-running interactions can extend potentially beyond the field's data type. In this case, RAA logs an error that is similar to the following:

```
Arithmetic overflow error converting expression to data type int
```

- Slowed RDBMS performance. Specifically, the Genesys Info Mart log will show evidence of notifications that are sent about completed long-running interactions in which the interval between the first parameter and the second is huge (for example, in the tens of millions), such as the following:

```
17:59:01.264 Agg.NewData        Got addFactAvailNotification2: 1,267,438,500
 1,367,225,100 -1 INTERACTION_FACT
```

If you encounter either symptom, consider adjusting the value of the days-to-keep-active-facts configuration option to circumvent this option. This option is documented in the *Genesys Info Mart Deployment Guide*.

Check for Incorrect Data Type

If, during aggregation, RAA encounters a string value where it expected an integer, aggregation will fail and log either of the following messages:

- `Conversion failed when converting the varchar value ... to data type int.`
- `ORA-01722: invalid number`

Info Mart stores revenue and satisfaction scores in character format (in the IRF_USER_DATA_GEN_1 table) because that is how the Genesys Info Mart Server receives the data from Interaction Concentrator (ICON). ICON reports all user data as strings, and Genesys Info Mart does not transform predefined user data to INTEGER. During the aggregation process, for certain fields, RAA converts this character data into numeric format and writes the aggregated results to INTEGER fields in the aggregate tables. RAA logs the error that is noted above if RAA encounters user data that it could not convert. To address the error, you must convert all problematic data—not their data type—into data that can be cast into the INTEGER data type. For example, the following value will generate an aggregation error on Oracle:

```
IRF_USER_DATA_GEN_1.REVENUE=”$1,000.00”
```

For RAA purposes, you should change this particular value to exclude the dollar sign, the comma, the decimal point, and the cents:

```
UPDATE IRF_USER_DATA_GEN_1 SET REVENUE=”1000”
   WHERE REVENUE=”$1,000.00”;
```

Upon resolving all problematic data, you must then reaggregate the time period in which aggregation failed. You can also resolve problematic data by setting it to **NULL**. Executing the following SQL statements will correct the error by setting REVENUE and SATISFACTION to NULL wherever these fields do not meet RAA standards.

Oracle Query:

```
UPDATE IRF_USER_DATA_GEN_1 SET REVENUE=NULL
WHERE REVENUE IS NOT NULL
   AND LENGTH(TRIM(TRANSLATE(REVENUE,' +-.0123456789',' ')))
           IS NOT NULL
 AND START_DATE_TIME_KEY IN
           (<values for interval in which aggregation failed>);

UPDATE IRF_USER_DATA_GEN_1 SET SATISFACTION=NULL
WHERE SATISFACTION IS NOT NULL
   AND LENGTH(TRIM(TRANSLATE(SATISFACTION,' +-.0123456789',' ')))
           IS NOT NULL
   AND START_DATE_TIME_KEY IN
           (<values for interval in which aggregation failed>);

COMMIT;
```

Microsoft SQL Server Query:

```
UPDATE IRF_USER_DATA_GEN_1 SET REVENUE=NULL
WHERE REVENUE IS NOT NULL
   AND ISNUMERIC(REVENUE)=0
   AND START_DATE_TIME_KEY IN
           (<values for interval in which aggregation failed>);

UPDATE IRF_USER_DATA_GEN_1 SET SATISFACTION=NULL
WHERE SATISFACTION IS NOT NULL
   AND ISNUMERIC(REVENUE)=0
   AND START_DATE_TIME_KEY IN
           (<values for interval in which aggregation failed>);
```

Alternatively, you can institute procedures whereby the entry of nonnumeric characters is prohibited for the REVENUE and SATISFACTION fields.

Check for connection problems

Use the information in this section to detect and troubleshoot problems with connections, which can be useful both to troubleshoot configuration problems, and for security forensic audits. RAA logs detailed information about the following events:

- Established connections — for example: `Agg.Connection successfully established`. These logs include the connection URL, connection name (if present), connection hash, and user name.
- Failed to establish connection — for example: `Agg.Connection failed to establish`.
- Problems when closing a connection.

You can match up 'established' log entries with the 'closed connection' log events by comparing the connection hash value in the log files.

Example 'connection established' and 'hash closed' log:

```
...
2019-05-20 17:38:37.552 I        Agg.Connection        successfully established
'AGR_NORMALIZER_2' connection hash_1254252690 to database jdbc:postgresql://localhost:5432/
gim as postgres
...
2019-05-20 17:38:37.561 I        Agg.Connection        jdbc connection hash_1254252690 closed
...
2019-05-20 17:38:37.753 I        Agg.Connection        successfully established
'Agg.Writer.0' connection hash_141460585 to database jdbc:postgresql://localhost:5432/gim
....
```

Example 'failed to establish a connection' attempt log:

```
...
2019-05-20 7:50:42.538 W        Agg.Connection        failed to establish 'Temp' connection
to database jdbc:postgresql://localhost:5432/gim as postgres
2019-05-20 17:50:42.538 F        ERROR                Exception happened:
2019-05-20 17:50:42.538 F        ERROR                SQLState: 08001
```

```
2019-05-20 17:50:42.538 F        ERROR               Vendor:   0
2019-05-20 17:50:42.538 F        ERROR               org.postgresql.util.PSQLException:
Connection refused. Check that the hostname and port are correct and that the postmaster is
accepting TCP/IP connections.
....
```

In some cases, a connection instance can be *wrapped* by another instance. In these scenarios, the original connection is closed when the wrapped connection is closed.

Example of a 'wrapped connection' log:

```
2019-05-23 20:46:50.896 I        Agg.Connection        jdbc connection hash_1246350906 was
wrapped by instance hash_1368316340
```