Working with iWD Capture Points

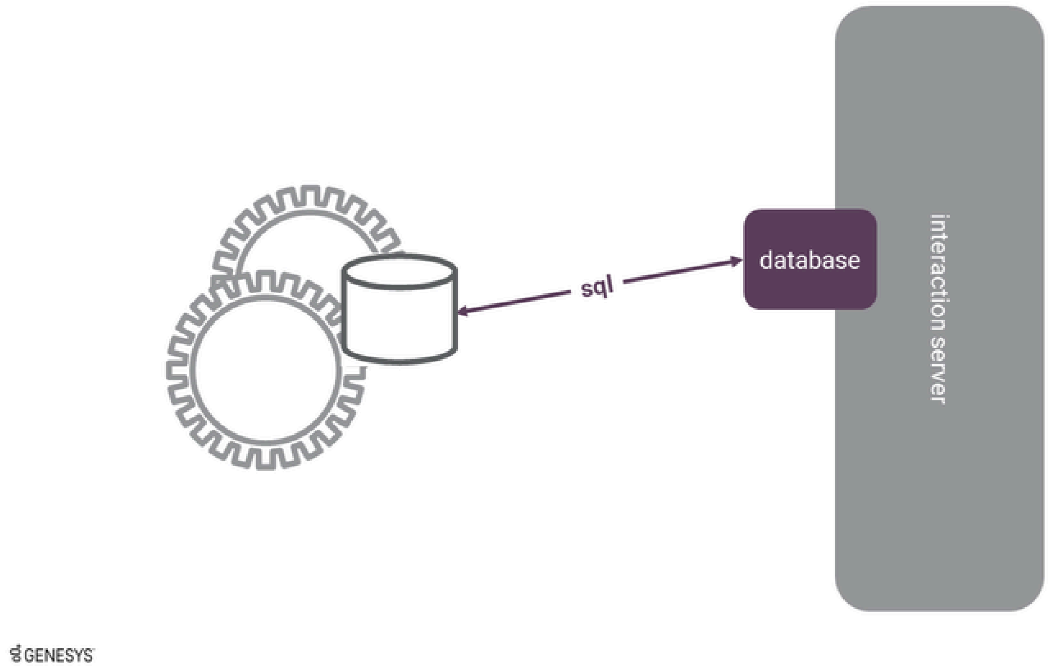
Database Capture Point

12/18/2025

Contents

Database Capture Point

The Database Capture Point provides the ability to create new tasks by directly accessing and querying a source system's database. It also provides the ability to query the source system's database to look for records that have been updated, to then propagate the updates to the corresponding tasks in iWD. Moreover, it is possible to configure database queries that will update specific tables on the source system database when updates to a task's status or other attributes occur in iWD.



The Database Capture Point connects directly to each source system database via ODBC. The configuration of this capture point includes the ODBC connection string or data source name configuration (after setting up a DSN) and the SQL queries that the capture point should execute based on each iWD event. The syntax used for the SQL queries should observe the semantics of the specific database server being used (Microsoft SQL Server, Oracle, and IBM DB2 are supported).

Supported Queries

The Database Capture Point supports the following queries. The tables below show the name of each query as it appears in the capture point configuration options, an explanation of what iWD event corresponds to this query, and an example of the query syntax. The tables are separated into the Inbound Queries, Notification Queries, and Source Update Queries.

Note that when using parameters in a query (for example, "externalid=<external id of the task>"), you write a question mark followed by the name of the parameter as known to the Genesys Interaction Server (which is referred to as an interaction property), in single quotes. The parameter names and interaction property names are case-sensitive.

Inbound Queries

Inbound Query	Corresponding iWD Event	Example
captureQuerySql	This query returns a result set in which each row will be captured as a task in iWD. The result set may contain columns corresponding to task attributes. There must be a field in the table or view that indicates if the task has already been captured; in this example that field is status.	`select caseId "ExternalId", createdTimestamp "ReceivedAt", casePriority "Priority", status "Status" from cases where status='new'`
capturedUpdateSql	This query updates the corresponding database record to reflect that certain data has been successfully captured as a task in iWD. Besides the values available from the corresponding capture query, the InteractionId (a unique value assigned by iWD to each task) is available to this query. This update will be executed for every task captured in the captureQuerySql set	`update cases set caseTrackingId=?'InteractionId', status='submitted' where caseId=?'ExternalId'`
errorUpdateSql	This query updates the corresponding database record to reflect that the associated task has not been captured in iWD. Besides the values available from the corresponding capture query, the additional values ErrorCode (integer) and ErrorDescription (string up to 256 characters) are available to this query.	`update cases set status='error', errorCode=?'ErrorCode', errorDescr=?'ErrorDescription' where caseId=?'ExternalId'`

Notification Queries

For all of the Notification Queries described below, it is possible to use the value of any iWD task attribute to update the corresponding column in the source system database.

Notification Query	Corresponding iWD Event	Example
assignedUpdateSql	This query updates the database to reflect that the associated task has been put in an Assigned status in iWD (i.e. assigned to an employee).	`update cases set status = 'assigned' where caseId=?'ExternalId'`
completedUpdateSql	This query updates the database to reflect that the associated task has been put in a Completed status in iWD.	`update cases set status = 'completed' where caseId=?'ExternalId'`
canceledUpdateSql	This query updates the database to reflect that the associated task has been put in a Canceled status in iWD.	`update cases set status = 'canceled' where caseId=?'ExternalId'`
heldUpdateSql	This query updates the database to reflect that the associated task has been put in a Held status in iWD.	`update cases set status = 'held' where caseId=?'ExternalId'`

Notification Query	Corresponding iWD Event	Example
queuedUpdateSql	This query updates the database to reflect that the associated task has been placed into any queue in the iWD business process, other than the standard queues that are used when tasks are canceled, completed, rejected, or put into an error status.	`update cases set status = 'queued', substatus=?'Queue' where caseId=?'ExternalId'`
errorHeldUpdateSql	This query updates the database to reflect that the associated task has been put in an ErrorHeld status in iWD.	`update cases set status = 'errorheld' where caseId=?'ExternalId'`
rejectedUpdateSql	This query updates the database to reflect that the associated task has put into a Rejected status in iWD.	`update cases set status = 'rejected' where caseId=?'ExternalId'`
restartedUpdateSql	This query updates the database to reflect that the associated task has been restarted in the iWD business process (to be reclassified and reprioritized).	`update cases set status = 'restarted' where caseId=?'ExternalId'`
stoppedUpdateSql	This query updates the database to reflect that the associated task has been stopped in the Genesys Interaction Server. This means that the task gets deleted completely from the iWD system.	`update cases set status = 'stopped' where caseId=?'ExternalId'`
routeRequestedUpdateSql	This query updates the database to reflect that the associated task has been sent to the Genesys Universal Routing Server to request that it be routed to an employee.	`update cases set status = 'routing' where caseId=?'ExternalId'`
updatedUpdateSql	This query updates the database to reflect that the associated task has been updated iWD by some other entity (not this Database Capture Point). For example, the task could have been updated by a user through the iWD Global Task List, by another capture point, by a routing strategy, or by an employee desktop application. Note that since there is only a single query that is executed based on all task updates, you must include all task attributes in this query that could possibly be updated.	`update cases set casePriority=?'Priority', caseDescr=?'Description' [...] where caseId=?'ExternalId'`
resumedUpdateSql	This query updates the corresponding database record to reflect that the associated task has been resumed (i.e. from a Held status).	`update cases set status = 'resumed' where caseId=?'ExternalId'`

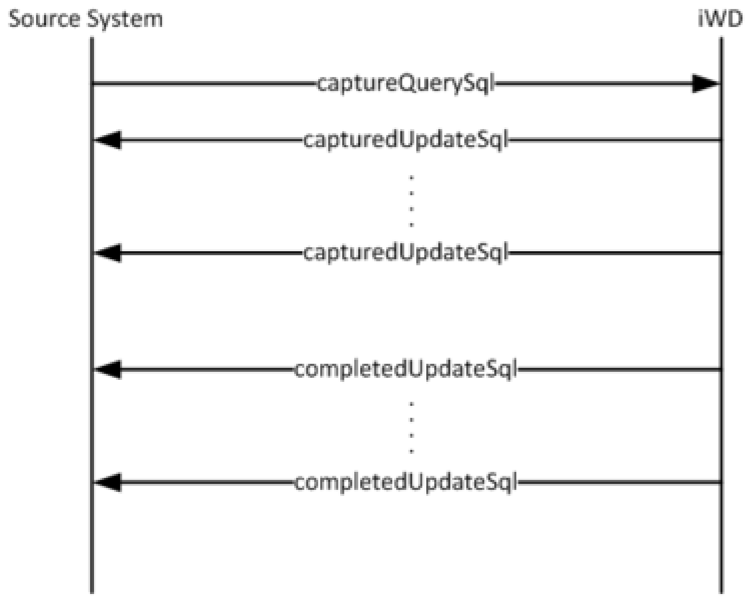
Source Update Queries

Source Update Query	Corresponding iWD Event	Example
sourceUpdateQuerySql	This query returns a set of rows, where each row represents an update request. Each such update request may contain one or more columns that represent iWD task attributes. The name of the column represents the name of the iWD task attribute and the value is the new value of that task attribute. Each row of the result set must	`select caseId "ExternalId", updatedTimestamp "SomeTime", casePriority "Priority" from case_updates where status='new'`

Source Update Query	Corresponding iWD Event	Example
	contain either InteractionId or ExternalId. If both InteractionId and ExternalId are contained in a row, the value of InteractionId will be used to access the interaction, and the value of ExternalId will be treated as one of the attributes to update.	
sourceUpdatedUpdateSql	This update (or delete) query will execute against a special table in the source database to mark a particular update as having been processed.	`update case_updates set status='applied' where caseId=?'ExternalId'`
sourceErrorUpdateSql	This update query is executed when there is an error executing an update request (the one that is returned by sourceUpdateQuerySql). Besides the values available from the corresponding capture query, additional values ErrorCode (integer) and ErrorDescription (string up to 256 characters) are available to this query.	`update case_updates set status='error', errorcode=?'ErrorCode', errordescr=?'ErrorDescription' where caseId=?'ExternalId'`

startupQuerySql

There is one additional query that may be configured, that does not fall into any of the previous three categories. It is called startupQuerySql. This optional query runs once, after the Database Capture Point successfully establishes a connection to the database.



The diagram below provides an example of a simple iWD Database Capture Point flow. The initial captureQuerySql request can return multiple task records. The capturedUpdateSql and completedUpdateSql queries are executed once for each task returned in the captureQuerySql query. More complex system interactions would involve any number of the above queries.

Integration

To use the Database Capture Point, database fields must be added to the source system database tables/views to capture updates from iWD. For example, a field must be added for the capture status as described in the capturedUpdateSql query. This allows iWD to optimize the capturing of new tasks and avoid importing existing tasks repeatedly.

SQL queries must be written that correspond to the capture point configuration parameters described above. Each use case must be tested to ensure that the correct data gets propagated to iWD, and back to the source system database. Batch (.bat) files can be created to quickly execute SQL queries to insert data into database tables on a test database, and other SQL queries to read and output the results.

Further Reading

There is more detailed technical reference information for this capture point in the Database Capture Point section of the eServices Integrated Capture Point Reference Guide.