intelligent Workload Distribution 8.0

Data Mart

Reference Guide

About Genesys

Alcatel-Lucent's Genesys solutions feature leading software that manages customer interactions over phone, Web, and mobile devices. The Genesys software suite handles customer conversations across multiple channels and resources—self-service, assisted-service, and proactive outreach—fulfilling customer requests and optimizing customer care goals while efficiently using resources. Genesys software directs more than 100 million customer interactions every day for 4000 companies and government agencies in 80 countries. These companies and agencies leverage their entire organization, from the contact center to the back office, while dynamically engaging their customers. Go to www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Technical Support website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

Although reasonable effort is made to ensure that the information in this document is complete and accurate at the time of release, Genesys Telecommunications Laboratories, Inc., cannot assume responsibility for any existing errors. Changes and/or corrections to the information contained in this document may be incorporated in future versions.

Your Responsibility for Your System’s Security

You are responsible for the security of your system. Product administration to prevent unauthorized use is your responsibility. Your system administrator should read all documents provided with this product to fully understand the features available that reduce your risk of incurring charges for unlicensed use of Genesys products.

Trademarks

Technical Support from VARs

If you have purchased support from a value-added reseller (VAR), please contact the VAR for technical support.

Technical Support from Genesys

If you have purchased support directly from Genesys, please contact Genesys Technical Support at the regional numbers provided on page 9. For complete contact information and procedures, refer to the ***Genesys Technical Support Guide***.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the ***Genesys Licensing Guide.***

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 80iwd_ref_datamart_12-2011_v8.0.003.01

Table of Contents

Supplements

Index

Preface

Welcome to the *intelligent Workload Distribution 8.0 Data Mart Reference Guide*. This document introduces you to the schemas that make up the intelligent Workload Distribution Data Mart—or iWD Data Mart—to guide you in the design and creation of reports that use the data within the iWD Data Mart.

This document is valid for the 8.0.*x* release(s) of this product.

Note: For versions of this document that have been created for other releases of this product, visit the Genesys Technical Support website, or request the Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com.

This preface provides an overview of this guide, identifies the primary audience, introduces document conventions, and lists related reference information:

- Intended Audience, page 7
- Chapter Summaries, page 8
- Making Comments on This Document, page 8
- Contacting Genesys Technical Support, page 9
- New in This Release, page 9

Intended Audience

This reference guide is intended for reporting and business analysts who want to leverage the data that is contained in the iWD Data Mart to produce reports for business users. It is also intended for IT administrators who would like to gain an understanding of the components that enable iWD Data Mart.

This reference guide assumes that the reader has an understanding of the following:

- Relational database concepts.
- Structured Query Language (SQL) for querying and mining data.
- iWD configuration, iWD Manager

- Data warehouse concepts—including working with star schemas, dimensions, aggregates, and measures.
- Extraction, Transformation, and Loading (ETL) concepts.

Chapter Summaries

In addition to this preface, you will find the following information about iWD Data Mart in this reference guide:

- Chapter 1, "iWD Reporting," on page 15 provides an overview of iWD reporting and the iWD Data Mart.
- Chapter 2, "iWD Data Mart Schema," on page 23 describes the facts, aggregates, dimensions, and views of the iWD Data Mart.
- Chapter 3, "ETL Overview," on page 81 describes the iWD ETL jobs.
- Chapter 4, "Customizing iWD," on page 85, provides the high-level steps that you must follow to have iWD calculate new statistics and aggregates. This chapter also provides one example for how to create the `product_pendingoverdue` statistic.

For information about related resources and about the conventions that are used in this document, see the supplementary material beginning on page 91.

Making Comments on This Document

If you especially like or dislike anything about this document, feel free to e-mail your comments to `Techpubs.webadmin@genesyslab.com`.

You can comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this document. Please limit your comments to the scope of this document only and to the way in which the information is presented. Contact your Genesys Account Representative or Genesys Technical Support if you have suggestions about the product itself.

When you send us comments, you grant Genesys a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Contacting Genesys Technical Support

If you have purchased support directly from Genesys, contact Genesys Technical Support at the following regional numbers:

Region	Telephone	E-mail
North America and Latin America	+888-369-5555 (toll-free) +506-674-6767	support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-127-645-7002	support@genesyslab.co.uk
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Malaysia	1-800-814-472 (toll-free) +61-7-3368-6868 (International)	support@genesyslab.com.au
India	000-800-100-7136 (toll-free) +61-7-3368-6868 (International)	support@genesyslab.com.au
Japan	+81-3-6361-8950	support@genesyslab.co.jp
Before you contact technical support, refer to the *Genesys Technical Support Guide* for complete contact information and procedures.		

New in This Release

Table 1 summarizes the schema changes that occurred to iWD Data Mart tables between the 7.6.1 and 8.0.*x* releases. Where a column was renamed in 8.0.*x,* data types remained the same and therefore, are not listed in the table. Because of its relational structure, changes were also made to the construction and/or names of:

- Views which were based on tables that were altered and
- Indexes (because the names of some primary keys changed).

Please refer to the createdb script deployed during installation for index and view constructions. The upgrade scripts also deployed during installation provide this same information.

The schema provided with the iWD 8.0.002 release supports Genesys Interactive Insights for iWD—a new Genesys product that offers summary and detail reports that cover various aspects of task management within an enterprise.

Also new in this release of this document are guidelines/instructions for customizing statistics and aggregates. Refer to Chapter 4 for this information.

For other types of changes to this release, such as to iWD deployment and functionality, refer to the *intelligent Workload Distribution 8.0 Deployment Guide,* the *intelligent Workload Distribution 8.0 Release Notes,* the Genesys Migration Guide, and other documents within the iWD documentation set.

Table 1: iWD Schema Changes between the 7.6.1 and 8.0.x Releases

Name of Table in 7.6.1	Name of Table in 8.0.0	Name of Column in 7.6.1	Name of Column in 8.0.0
BUSINESS_VALUE	BUSINESS_VALUE	--	BUSINESS_VALUE_RANGE_5_STA
		--	BUSINESS_VALUE_RANGE_5_END
		--	BUSINESS_VALUE_RANGE_10_STA
		--	BUSINESS_VALUE_RANGE_10_END
		--	BUSINESS_VALUE_RANGE_50_STA
		--	BUSINESS_VALUE_RANGE_50_END
		--	BUSINESS_VALUE_RANGE_100_STA
		--	BUSINESS_VALUE_RANGE_100_END
		--	BUSINESS_VALUE_RANGE_500_STA
		--	BUSINESS_VALUE_RANGE_500_END
		--	BUSINESS_VALUE_RANGE_1000_STA
		--	BUSINESS_VALUE_RANGE_1000_END
	ETL_CUSTOM_MAP	CONTRACT_KEY	DEPARTMENT_KEY
ETL_KETTLE_LOG	--	--	--
CONTRACT	DEPARTMENT	CONTRACT_KEY	DEPARTMENT_KEY
		CONTRACT_CONFIG_ID	DEPARTMENT_CONFIG_ID
		CONTRACT_CONFIG_EVENT_ID	DEPARTMENT_CONFIG_EVENT_ID
		CONTRACT_RUNTIME_ID	DEPARTMENT_RUNTIME_ID
		CONTRACT_NAME	DEPARTMENT_NAME
CUSTOM_DIM	CUSTOM_DIM	VALID_FROM	--
		VALID_TO	--
		VERSION	--
CUSTOMER_SEGMENT	CUSTOMER_SEGMENT	CUSTOMER_SEGMENT	CUSTOMER_SEGMENT_NAME
--	DATE_TIME	--	all columns
DISTRIBUTION_POINT	DISTRIBUTION_POINT	SOLUTION_KEY	--
EVENT_DATE	EVENT_DATE	--	EVENT_DATE_STR
MEDIA_CHANNEL	MEDIA_CHANNEL	CHANNEL_NAME	MEDIA_CHANNEL_NAME
METRIC	METRIC	CONTRACT_KEY	DEPARTMENT_KEY
		DISTRIBUTION_POINT_KEY	--

Table 1: iWD Schema Changes between the 7.6.1 and 8.0.x Releases (Continued)

Name of Table in 7.6.1	Name of Table in 8.0.0	Name of Column in 7.6.1	Name of Column in 8.0.0
PRIORITY	PRIORITY	--	PRIORITY_RANGE_5_STA
		--	PRIORITY_RANGE_5_END
		--	PRIORITY_RANGE_10_STA
		--	PRIORITY_RANGE_10_END
		--	PRIORITY_RANGE_50_START
		--	PRIORITY_RANGE_50_END
		--	PRIORITY_RANGE_100_STA
		--	PRIORITY_RANGE_100_END
		--	PRIORITY_RANGE_500_STA
		--	PRIORITY_RANGE_500_END
		--	PRIORITY_RANGE_1000_STA
		--	PRIORITY_RANGE_1000_END
PROCESS	PROCESS	CONTRACT_KEY	DEPARTMENT_KEY
QUEUE_TARGET	QUEUE_TARGET	QUEUE_TYPE	QUEUE_TARGET_TYPE
RESULT_CODE	RESULT_CODE	RESULT_CODE	RESULT_CODE_NAME
SOURCE_TENANT	SOURCE_TENANT	SOURCE_TENANT	SOURCE_TENANT_NAME
TASK_AGENT_FACT	TASK_AGENT_FACT	--	WORK_TIME
		--	DATE_TIME_KEY
TASK_CAPT_FACT	TASK_CAPT_FACT	--	CMPL_TASK_WORK_TIME
		--	CMPL_TASK_ASSIGN_TIME
		--	CMPL_TASK_COMPLETE_TIME
		--	CMPL_TASK_SRC_TIME
		--	CMPL_TASK_PRE_SRC_TIME
		--	DATE_TIME_KEY
TASK_CLASSIF_FACT	TASK_CLASSIF_FACT	--	CMPL_TASK_WORK_TIME
		--	CMPL_TASK_ASSIGN_TIME
		--	CMPL_TASK_COMPLETE_TIME
		--	CMPL_TASK_SRC_TIME
		--	CMPL_TASK_PRE_SRC_TIME
		--	DATE_TIME_KEY

Table 1: iWD Schema Changes between the 7.6.1 and 8.0.x Releases (Continued)

Name of Table in 7.6.1	Name of Table in 8.0.0	Name of Column in 7.6.1	Name of Column in 8.0.0
TASK_QUEUE_FACT	TASK_QUEUE_FACT	--	CMPL_TASK_WORK_TIME
		--	CMPL_TASK_ASSIGN_TIME
		--	CMPL_TASK_COMPLETE_TIME
		--	CMPL_TASK_SRC_TIME
		--	CMPL_TASK_PRE_SRC_TIME
		--	DATE_TIME_KEY
I_TASK_EVENT_FACT	I_TASK_EVENT_FACT	TASK_ID	--
		DISTRIBUTION_ID	INTERACTION_ID
		CONTRACT_KEY	DEPARTMENT_KEY
		--	START_DATE_TIME_KEY
H_TASK_EVENT_FACT	H_TASK_EVENT_FACT	TASK_ID	--
		DISTRIBUTION_ID	INTERACTION_ID
		CONTRACT_KEY	DEPARTMENT_KEY
		--	START_DATE_TIME_KEY
I_TASK_FACT	I_TASK_FACT	CURRENT_DISTRIBUTION_ID	INTERACTION_ID
		CONTRACT_KEY	DEPARTMENT_KEY
		DISTRIBUTE_TIME_FR_CREATED_SEC	--
		DISTRIBUTED_DATE_KEY	--
		DISTRIBUTED_TIME_KEY	--
		DISTRIBUTED_INTERVAL	--
		--	START_DATE_TIME_KEY
H_TASK_FACT	H_TASK_FACT	CURRENT_DISTRIBUTION_ID	INTERACTION_ID
		CONTRACT_KEY	DEPARTMENT_KEY
I_TASK_WORK_FACT	I_TASK_WORK_FACT	CURRENT_DISTRIBUTION_ID	INTERACTION_ID
		DISTRIBUTION_ID	--
		CONTRACT_KEY	DEPARTMENT_KEY
		--	START_DATE_TIME_KEY

Table 1: iWD Schema Changes between the 7.6.1 and 8.0.x Releases (Continued)

Name of Table in 7.6.1	Name of Table in 8.0.0	Name of Column in 7.6.1	Name of Column in 8.0.0
H_TASK_WORK_FACT	H_TASK_WORK_FACT	CURRENT_DISTRIBUTION_ID	INTERACTION_ID
		DISTRIBUTION_ID	--
		CONTRACT_KEY	DEPARTMENT_KEY
		--	START_DATE_TIME_KEY
H_TASK_QUEUE_FACT_15MIN, I_TASK_QUEUE_FACT_15MIN	H_TASK_QUEUE_FACT_15MIN, I_TASK_QUEUE_FACT_15MIN	CMPL_TASK_AVG_DISTRIBUTE_TIME	--
H_TASK_CAPT_FACT_15MIN, I_TASK_CAPT_FACT_15MIN	H_TASK_CAPT_FACT_15MIN, I_TASK_CAPT_FACT_15MIN	CMPL_TASK_AVG_DISTRIBUTE_TIME	--
H_TASK_AGE_FACT I_TASK_AGE_FACT	--	--	--
H_TASK_CLASSIF_FACT I_TASK_CLASSIF_FACT	H_TASK_CLASSIF_FACT I_TASK_CLASSIF_FACT	CMPL_TASK_AVG_DISTRIBUTE_TIME	--
H_TASK_QUEUE_AGE_FACT I_TASK_QUEUE_AGE_FACT	--	--	--

Chapter

1 iWD Reporting

This chapter explains the reporting functionality that is available within intelligent Workload Distribution (iWD). The information in this chapter is organized into the following sections:

What Is intelligent Workload Distribution?

Designed as a business application for business users, Genesys intelligent Workload Distribution works in concert with the Genesys Customer Interaction Management (CIM) Platform—enabling a centralized service-delivery platform, and proactively managing interactions and tasks across all channels and media.

Only with a Global Task List (which is sorted, based on business value) can the enterprise ensure that the right resources—regardless of their location—are proactively receiving the most critical or highest value tasks—regardless of media type—at the right time. Being able to react quickly with new business intelligence and work effectively are key success factors with any enterprise in today's competitive marketplace.

As depicted in Figure 1, iWD supports three main areas:

- Capturing tasks
- Calculating task values
- Distributing tasks

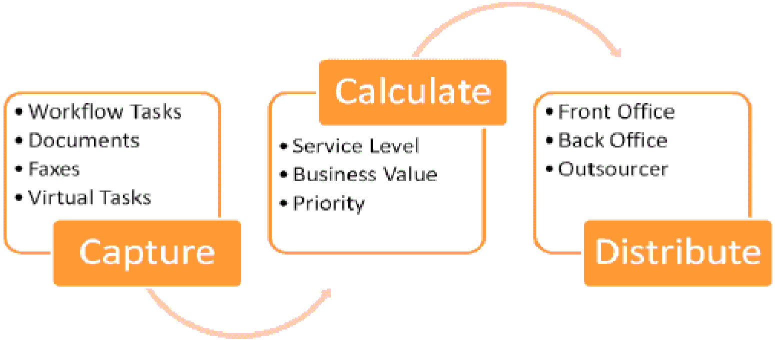


Figure 1: The Realm of Tasks

Capture Tasks

To address the challenge of having tasks stored across multiple enterprise systems, iWD accepts work—in electronic format—from a broad range of applications, such as customer-relationship management, workflow, host systems, and Enterprise Service Bus (ESB) systems.

Calculate Tasks Values

Using the business rules that are configured by users, iWD calculates service-level values such as task due date, business value, and priority. By using these values, iWD orders tasks from most important to least important, and monitors and proactively manages tasks to ensure compliance with service-level objectives that are specific to your business.

Distribute Tasks

iWD distributes tasks to front- or back-office resources, or to external partners like business process outsourcers, working in concert with the Genesys CIM Platform.

iWD Reporting

As customers are presented with an increasing number of choices in the marketplace and hold higher expectations of service quality, the ability to measure the efficiency and effectiveness of customer-service delivery becomes a key component of success. iWD provides harmony to an often-cumbersome reporting process through:

- *Cradle-to-grave* reporting from the time that a task enters the contact center until its completion.

- Consolidated reporting across the various systems that are involved in customer-service delivery: fax servers, workflow, customer-relationship management, and Genesys Customer Interaction Management.
- Reporting that is based on business context—with business process, customer segment, and product independent of channel, instead of being limited to interactions, queues, channels, and workflows.

The key to achieving the desired business results is having access to actionable business intelligence. Genesys iWD offers comprehensive reporting, providing management insight into business operation. It provides key indicators of performance both in real time—through current-day statistics—and on an historical basis. The historical metrics are provided based on aggregates and measures that are populated by scheduled ETL processes, which extract data from the Interaction Server Event Log database and load it into the iWD Data Mart. Figure 2 provides a functional overview of iWD's reporting components. Third-party services can reference extended statistics from the GTL_STAT table to display data in dashboards or within CCPulse+.

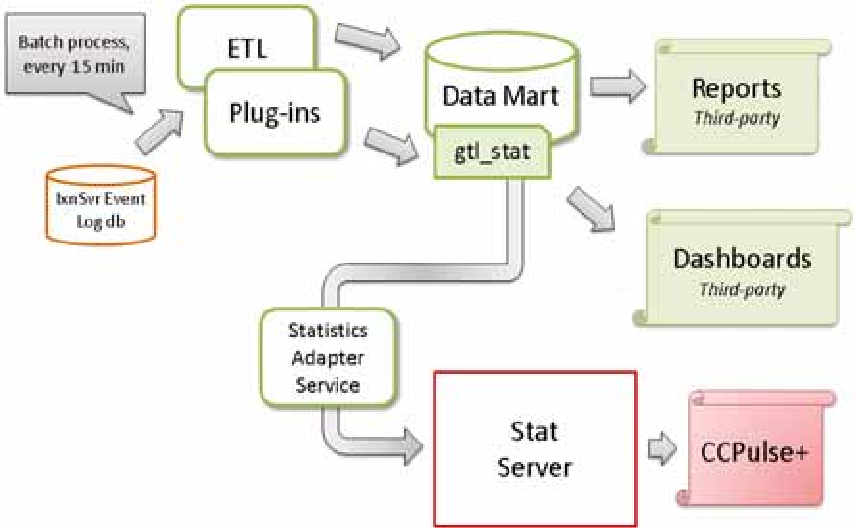


Figure 2: iWD Reporting Overview

Note: Each iWD tenant solution requires its own Data Mart.

Similar to the layout of other data warehouses, iWD Data Mart consists of the following database objects:

- Fine-grained fact tables—Store all attributes that are associated with tasks (I_TASK_FACT/H_TASK_FACT tables), work-related events (I_TASK_WORK_FACT/H_TASK_WORK_FACT tables), when the task was assigned to one or more agents; and a full audit history of the task (I_TASK_EVENT_FACT/H_TASK_EVENT_FACT tables).
- Aggregated fact tables—Describe tasks in an iWD-oriented context across the various stages, or the iWD life cycle of the task—from capture and classification to distribution to agent.

- Dimensions—Describe task attributes that are common across the fact tables in iWD Data Mart, such as iWD business process, priority, business value, and date and time. Fact tables link to these dimensions through keys.
- Measures—Represent numerical values—such as totals, durations, averages, minimums, and maximums—that are stored in aggregated fact tables across intraday and historical intervals. For example, the total number of completed tasks by 15-minute interval by an iWD process and business value would be captured within the `I_TASK_CAPT_FACT_15MIN` intraday table.

When they are connected to existing enterprise data marts, including Genesys Info Mart, analysts gain access to comprehensive views of the entire customer experience. Analytical reporting leverages existing business-intelligence tools, such as those that are provided by Pentaho (which is an open-source product suite for business intelligence) or through a host of commercial products from Cognos or Business Objects Crystal Reports.

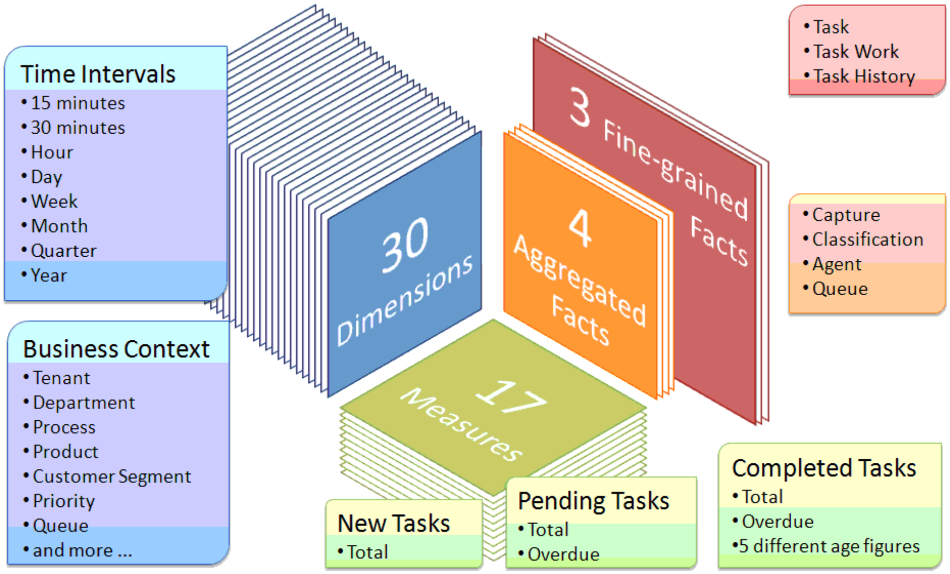


Figure 3: iWD Data Mart—Dimensions, Measures, and Facts

iWD Extended Statistics

Extended statistics are written to the GTL_STAT iWD Data Mart table to avail iWD data to other Genesys products. These statistics are generated by Kettle plug-ins—Kettle, being the component of Pentaho that is responsible for ETL processes; plug-ins are user-designed Kettle scripts for generating statistics. The aggregate_stats job, described on page 82, reads the stats.properties file and executes all of the plug-ins listed therein, providing them each with four parameters:

- TENANT_RUNTIME_ID
- SOLUTION_RUNTIME_ID
- STAT_SERVICE_ID
- INTERVAL_KEY, the key of the last aggregated date and time interval

Using these parameters, the plug-in can query fine-grained facts, aggregated facts, and dimensions to create the requested statistics and write them to GTL_STAT. Table 2 shows the structure of this table.

The Statistics Adapter Service, shown in Figure 2, processes the statistics in the GTL_STAT table and writes statistical parameters (stat types and filters) to the configuration of the indicated Stat Server application in Configuration Server. The iWD Stat Server Java Extension [BPR_iWD_Extension.jar] for business-priority routing (BPR) also processes statistics from GTL_STAT on behalf of Stat Server to avail statistical data to other Genesys applications. To accomplish this, you must both configure the Statistics Adapter Service in iWD with information to connect to a specific Stat Server application and configure Stat Server to use BPR_iWD_Extension.jar. CCPulse+, in turn, requests iWD statistics from Stat Server and reads the stat types and filters from Stat Server configuration.

The properties of the Statistics Adapter Service are described in the *intelligent Workload Distribution Deployment Guide*. Instructions for configuring Java Extensions within Stat Server are described in the *Framework 8.0 Stat Server Deployment Guide*.

Table 2: The GTL_STAT Table

Column	Data Type	Description
ID	int	Primary key of this table.
TENANTID	varchar(16)	iWD Manager ID of the tenant.
SOLUTIONID	varchar(16)	iWD Manager ID of the solution.
STATSERVICEID	varchar(16)	iWD Manager ID of the Extended Statistics Service that is used to submit statistics to Genesys.
DIMENSIONTYPE	varchar(32)	Dimension type (for example, SLT–solution, DPT–department, and PRC–process).

Table 2: The GTL_STAT Table (Continued)

Column	Data Type	Description
DIMENSIONID	varchar(64)	Dimension ID. Used to generate Genesys statistics filters.
MEASUREID	varchar(64)	Measure ID. Used to generate Genesys statistical types.
MEASUREVALUE	int	Value of the measure.

iWD Task Attributes

At the heart of iWD reporting is the set of task attributes that describe aspects of a task and its association within business context, such as iWD business process. Task attributes can be classified into one of three categories:

- Core attributes
- Extended attributes
- Custom attributes

Understanding these attributes will help you understand the measures and aggregates stored in the iWD Data Mart. Incorrect use or interpretation of these values can negatively impact reporting outcomes.

Core Attributes

Core attributes describe the fundamentals of a task. These attributes are used in assembling tasks in the Global Task List, based on the business value and priority that are defined within iWD. Core attributes are either set automatically by iWD (through iWD business rules), or they are provided by the source system (through the Capture Adapter interface). The following are some iWD core task attributes:

- `activationDateTime`
- `assignedDateTime`
- `assignedToUser`
- `businessValue`
- `captureId`
- `category`
- `completedDateTime`
- `dueDateTime`
- `expirationDateTime`
- `heldDateTime`
- `interactionid`
- `mediaChannel`
- `priority`
- `queue`
- `queueTarget`
- `queueType`

Extended Attributes

Extended attributes provide additional context about a task—enabling you to tailor the service-level agreement (SLA) rules for managing tasks on the Global Task List—and can aid in customizing current-day and historical reporting. For example, use of several capture dates allow an organization to measure performance against the date and time at which an order or loan application was received by the originating system or was submitted by the customer via a web form. The following are some extended task attributes:

- `customerId`
- `customerSegment`
- `productSubtype`
- `productType`
- `requestedAgent`
- `requestedSkill`
- `resultCode`
- `sourceCreatedDateTime`
- `sourceDueDateTime`
- `sourceFirstCreatedDateTime`
- `sourceProcessSubtype`
- `sourceProcessType`
- `sourceTenant`

Custom Attributes

In addition to the core and extended attributes, iWD enables you to customize additional task details through iWD custom attributes. *Custom attributes* are defined as key-value pairs that are provided by the source system. For example, a web form can contain several fields that might not be mapped to a core or extended attribute. They can instead be mapped to a custom attribute.

When custom attributes are submitted via an iWD Capture Adapter—such as iWD Web Service, XML File Capture, or WebSphere MQ—iWD stores the values in the Genesys Interaction Server database as user data. Custom attributes can also be mapped to custom dimensions in iWD Data Mart by defining mapping properties on the ETL configuration in iWD Manager—again, further extending the level of tailored reporting that is enabled by iWD Data Mart. Refer to the “Configuration” chapter in the *intelligent Workload Distribution 8.0 Deployment Guide* for information about configuring this mapping.

Chapter

2 iWD Data Mart Schema

The Genesys iWD Data Mart is a relational database designed around a star-schema model. This particular type of multi-dimensional model is simplistic requiring relatively simple queries using joins and conditions that involve only one fact table and a single level of dimension tables in order to build reports.

This chapter describes the objects of iWD Data Mart. It contains the following sections:

Overview

The fact tables in iWD Data Mart store the primary information about a task—its core, extended, and custom attributes—while the dimension tables provide structured labeling of the attributes that are common to many of the facts in the associated fact tables categorizing each attribute into non-overlapping regions. This combination of one fact table and its set of dimensions creates a star schema as shown in Figure 4.

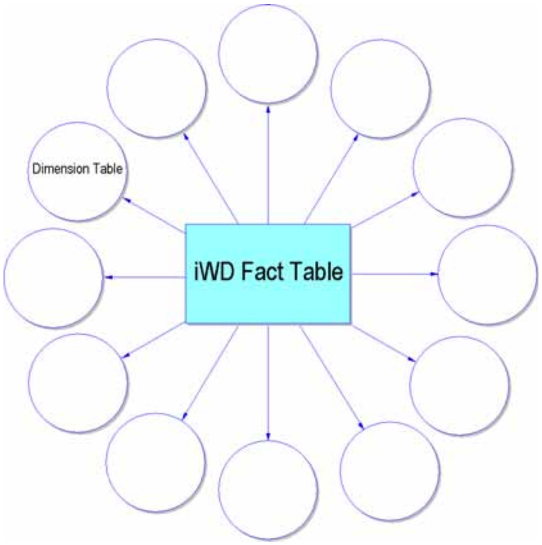


Figure 4: Star Schema

Each fact table holds two types of fields:

- Foreign keys that join each of the table's records to appropriate values in dimension tables.
- Measures that provide individual facts about the tasks, such as the number of seconds that an agent worked on task.

The iWD Data Mart houses three types of facts:

- Intraday—Fine-grained (or raw) and aggregated data for the current day, used for near-real-time dashboards and operational reports.
- Historical—Historical facts and aggregated data, used for historical and analytical reporting.
- Blended—Database views that combine intraday and historical facts.

This separation of intraday from historical data enables faster generation of real-time reports and dashboards. iWD moves intraday task data to the appropriate historical fact table when both of the following conditions are met:

- The day's data (all 15-minute intervals [96 total (4x24)]) has been fully aggregated.
- The task reaches its "final" state.

A final state is achieved when the task reaches a `Completed`, `Canceled`, or `Rejected` status in iWD. Until such time, the task remains in the intraday tables, ineligible for advancement to the historical fact tables, even if the task's duration spans more than 24 hours.

Figure 5 provides a bus matrix that illustrates how iWD Data Mart fact tables and dimension tables interrelate.

(Rotate page counterclockwise to read.)

Queue	Classification	Capture	Agent	iWD Aggregates	
			✓	Agent	Dimensions
		✓		Business Value	
	✓			Category	
				Customer	
	✓			Customer Segment	
✓	✓	✓	✓	Interval Date	
✓	✓	✓	✓	Interval Time	
	✓			Media Channel	
✓				Priority	
	✓			Product	
✓			✓	Queue	
			✓	Result Code	
	✓			Source Process	
	✓			Source Tenant	
		✓		Capture Point	
✓	✓	✓	✓	Custom Dim	
✓	✓	✓	✓	Process	
	✓	✓		Total New Tasks	Measures
✓	✓	✓		Total Completed Tasks	
✓	✓	✓		Total Completed Overdue	
✓	✓	✓		Total Pending Tasks	
✓	✓	✓		Total Overdue Tasks	
			✓	Task Work Count	
✓	✓	✓	✓	Avg Task Work Time	
✓	✓	✓		Avg Assign Time	
✓	✓	✓		Avg Complete Time	
✓	✓	✓		Avg Source Create Time	
✓	✓	✓		Avg Source First Create Time	
			✓	Min Work Time	
			✓	Max Work Time	
✓				Total Entered Tasks	
✓				Total Exited Tasks	

Figure 5: Overview of iWD Data Mart Aggregates, Dimensions, and Facts

General Notes About the Tables and Schemas

iWD Tables

- iWD Data Mart is supported on the following RDBMSs:
  - Microsoft SQL
  - MySQL
  - Oracle

  Refer to the *Supported Operating Environment Reference Manual* for the exact supported version. Field data types, however, are only presented for MySQL in the table descriptions that follow.
- All date and time keys are recorded in the time zone that is configured for the Kettle ETL Service. By default, this time zone is Universal Coordinated Time (UTC) (when no time zone is configured).

This Document's Tables

- PK in the column headers of the tables stands for Primary Key.

Star Schemas

- The `ETL_AUDIT` dimension, which logs execution details about the Data Mart jobs that populate all fact tables, is part of all star schemas, but, for simplicity, is excluded from all illustrations in this document.
- A few of the dimensions, such as `EVENT_DATE`/`EVENT_TIME` and `BUSINESS_VALUE`/`BUSINESS_RULE`, in the illustrations depicting start schemas share the same physical space within the graphic. In actuality though, they represent two separate and independent dimensions. This space sharing was necessary only to simplify the illustrations.
- The star schema illustrations depict only one join between facts and dimensions where more than one join may exist. Refer to the table and field descriptions for a more accurate assessment of table interrelationships.

iWD Data Mart Core Fact Tables

The iWD Data Mart contains the following fine-grained (or core) fact tables:

- I_TASK_FACT and its historical counterpart, H_TASK_FACT. (See page 27.)
- I_TASK_WORK_FACT and H_TASK_WORK_FACT (page 33).
- I_TASK_EVENT_FACT and H_TASK_EVENT_FACT (page 36).

This section illustrates the star schemas, or subject areas, that support each core table and provides a data dictionary of each that lists the data types of each column for the supported relational database management systems (RDBMSs).

TASK_FACT Tables

A *task* describes a definite piece of work from the perspective of the customer. Each iWD task record results in a single fact being written to the I_TASK_FACT or H_TASK_FACT tables. In the iWD Data Mart, you can access a task fact through the following database objects:

- I_TASK_FACT—Intraday data table
- H_TASK_FACT—Historical data table
- TASK_FACT—Blended view of historical and intraday data

Figure 6 shows the dimensions that support the iWD task fact tables and view.

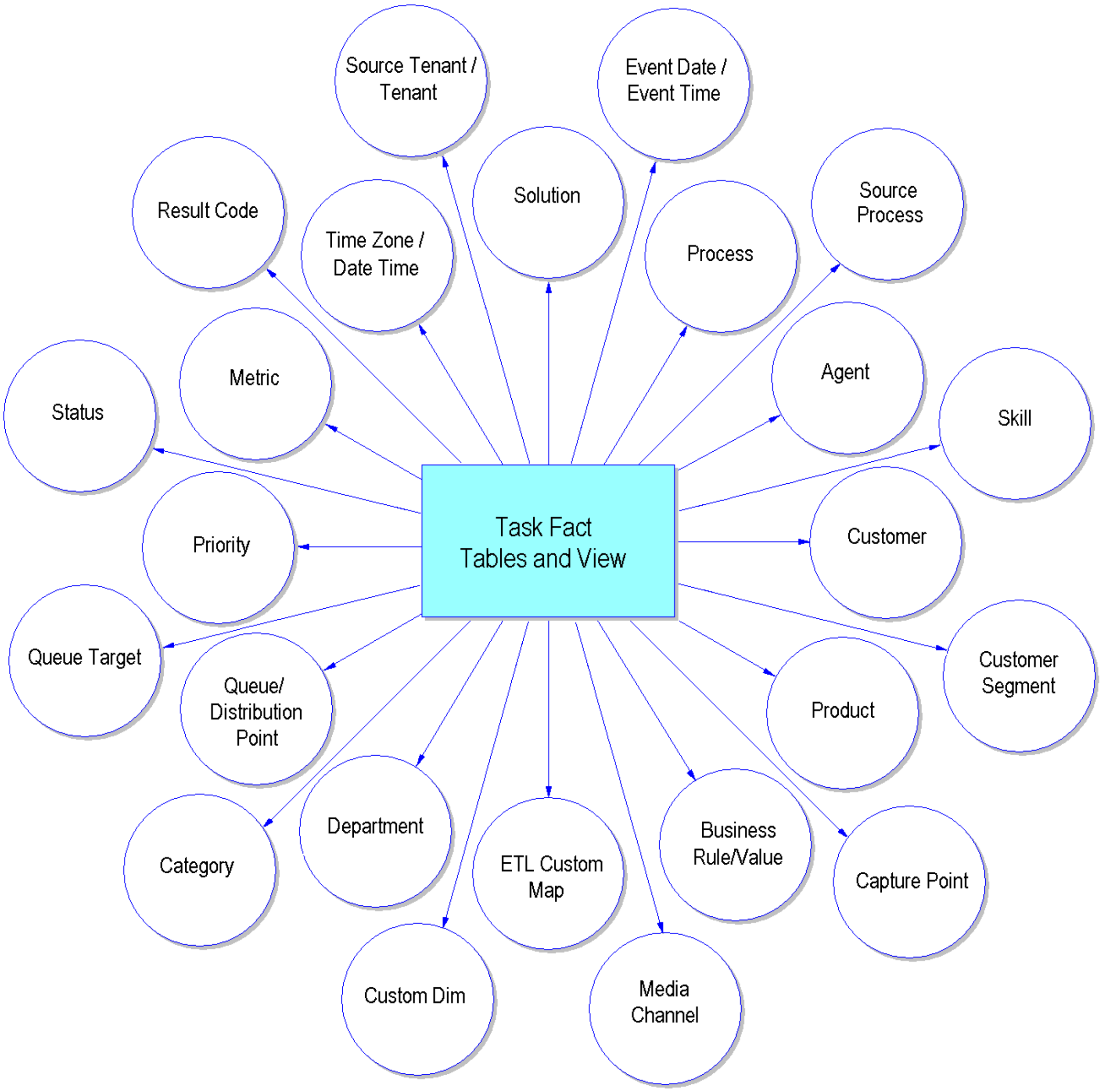


Figure 6: The TASK_FACT Star Schema

Table 3 describes the columns of the `TASK_FACT` intraday and historical tables.

Table 3: The I_/H_TASK_FACT Tables

Field	Data Type	Description
INTERACTION_ID	varchar(64)	The Interaction ID—unique within a single Interaction Server database. This field, together with `SOLUTION_KEY`, serves as the primary key of this table.
SOLUTION_KEY	int	Key to the `SOLUTION` dimension, describing the solution instance of the task (as configured in iWD Manager)—for example, Production versus Test. A solution is assigned as soon as a task is created in the Interaction Server database. A tenant can have more than one solution instance. This field, together with `INTERACTITON_ID`, serves as the primary key of this table.

Table 3: The I_/H_TASK_FACT Tables (Continued)

Field	Data Type	Description
LAST_TASK_EVENT_ID	int	Unique identifier for the last event that is associated with the task.
CAPTURE_ID	varchar(64)	ID of the task capture, as stored or referenced in the originating system—for example, work item ID. This field is a core task attribute.
TENANT_KEY	int	Key to the TENANT dimension, describing the tenant of the task (as configured in iWD Manager). A tenant is assigned as soon as a task is created in the Interaction Server database.
DEPARTMENT_KEY	int	Key to the DEPARTMENT dimension, identifying the department that is associated with the task.
PROCESS_KEY	int	Key to the PROCESS dimension, identifying the parent iWD business process of the task.
CAPTURE_POINT_KEY	int	Key to the CAPTURE_POINT dimension, identifying the capture point that captured the task.
CURRENT_DISTRIBUTION_POINT_KEY	int	Key to DISTRIBUTION_POINT dimension, identifying the distribution point that distributed the task.
CURRENT_QUEUE_KEY	int	Key to the QUEUE dimension, identifying the queue in which the task resides and queue type.
CURRENT_QUEUE_TARGET_KEY	int	Key to the QUEUE_TARGET dimension, identifying the agent, agent group, place, or place group to which the task was assigned.
SOURCE_FIRST_CREATED_DATE_KEY	int	Key to the EVENT_DATE dimension. This field is reserved for the DTM (Driver Test Manager) from the first system that captured the task. **Note:** iWD provides for task-flow scenarios that involve two source DTMs, where two systems were involved in the origination of a task—for example, fax server and workflow.
SOURCE_FIRST_CREATED_TIME_KEY	int	Key to the EVENT_TIME dimension, identifying the time at which the first source system captured the task.
SOURCE_CREATED_DATE_KEY	int	Key to the EVENT_DATE dimension, identifying the date on which the second source system captured the task. The second source system is the DTM that submitted the task to iWD.
SOURCE_CREATED_TIME_KEY	int	Key to the EVENT_TIME dimension, identifying the time at which the second source system captured the task.
SOURCE_DUE_DATE_KEY	int	Key to the EVENT_DATE dimension, identifying the date when task is due in source system.
SOURCE_DUE_TIME_KEY	int	Key to the EVENT_TIME dimension, identifying the time when task is due in source system.
CREATED_DATE_KEY	int	Key to the EVENT_DATE dimension, describing the date that the iWD task was created. Additional created date and time stamps are provided for in the extended attributes to report not only on iWD capture date and time, but also on the originating system—for example, workflow capture date and time. Refer to the SOURCE_CREATED and SOURCE_FIRST_CREATED date and time keys.

Table 3: The I_/H_TASK_FACT Tables (Continued)

Field	Data Type	Description
CREATED_TIME_KEY	int	Key to the EVENT_TIME dimension, describing the time at which the iWD task was created.
ACTIVATION_DATE_KEY	int	Key to the EVENT_DATE dimension, describing the iWD task activation date. This is the date on which the task becomes active; before this date, the task remains in the iWD_Captured queue and will not be prioritized or delivered to agents.
ACTIVATION_TIME_KEY	int	Key to the EVENT_TIME dimension, describing the activation time for the task. Activation time is the moment at which the task becomes active. Before this time, the task remains in the iWD_Captured queue and will not be prioritized or delivered to agents.
DUE_DATE_KEY	int	Key to the EVENT_DATE dimension, describing the due date of the task, as set by either iWD rules or the source system.
DUE_TIME_KEY	int	Key to the EVENT_TIME dimension, describing the time at which the task is due, as set by either iWD rules or the source system.
COMPLETED_DATE_KEY	int	Key to the EVENT_DATE dimension, describing the task completion date.
COMPLETED_TIME_KEY	int	Key to the EVENT_TIME dimension, describing the task completion time.
ASSIGNED_DATE_KEY	int	Key to the EVENT_DATE dimension, describing the date on which withe task was assigned to an agent.
ASSIGNED_TIME_KEY	int	Key to the EVENT_TIME dimension, describing the time at which the task was assigned to an agent.
MEDIA_CHANNEL_KEY	int	Key to the MEDIA_CHANNEL dimension, describing the channel through which the task was received—for example, fax. This value can be set in iWD rules or by the source system that submitted the task.
CATEGORY_KEY	int	Key to CATEGORY dimension, further describing the task, such as a follow-up.
BUSINESS_VALUE_KEY	int	Key to the BUSINESS_VALUE dimension. Business value is assigned by using iWD rules during the classification phase of the task.
CURRENT_PRIORITY_KEY	int	Key to the PRIORITY dimension. As with business value, initial priorities should be assigned during classification. The priority of a task can change over time. For example, as the task gets closer to its due-date, rules can be configured to re prioritize the task proactively. The value that is stored in this field represents the current priority of the task. Historical priority values are stored in the H_TASK_EVENT_FACT table.
CURRENT_STATUS_KEY	int	Key to the STATUS dimension, describing the current status of the task.
LAST_ASSIGNED_ AGENT_KEY	int	Key to the AGENT dimension, identifying the last agent who was assigned to the task.
LAST_RESULT_CODE_KEY	int	Key to the RESULT_CODE dimension. This value often represents the wrap code from a soft phone, the result code from a routing strategy, or the result code from the source system.

Table 3: The I_/H_TASK_FACT Tables (Continued)

Field	Data Type	Description
CUSTOMER_KEY	int	Key to the CUSTOMER dimension. Often used as the customer ID from the source system. This ID can be utilized to retrieve further customer details from a Customer Relationship Management (CRM) data warehouse or other customer data repository.
CUSTOMER_SEGMENT_KEY	int	Key to the CUSTOMER_SEGMENT dimension, describing the customer to whom the task is associated. The customer segment is received from the source system as an extended iWD task attribute—for example, gold, silver, or bronze.
PRODUCT_KEY	int	Key to the PRODUCT dimension, describing the product to which the task is related—for example, a product name or product type, such as a loan or Internet Digital Subscriber Line (DSL). The product can be further defined by using product subtypes, such as residential loan or home DSL.
SOURCE_TENANT_KEY	int	Key to the SOURCE_TENANT dimension, describing the tenant who submitted the task. It can be important in a multi-tenant or service-bureau environment.
SOURCE_PROCESS_KEY	int	Key to the SOURCE_PROCESS dimension. Source process includes the type and subtype that describe the source process that is associated with the task—for example, Order and DSL Order.
REQUESTED_SKILL_KEY	int	Key to the SKILL dimension, identifying the agent skill that was requested by the iWD rule.
REQUESTED_AGENT_KEY	int	Key to the AGENT dimension, identifying the agent who was requested by the iWD rule.
CUSTOM_DIM_KEY	int	Key to the CUSTOM_DIM dimension containing five additional attributes (beyond those that are listed below) that can be used to describe a task.
CUSTOM_ATTRIBUTE1	varchar(255)	Custom attributes describe a task. A total of 10 customer attributes can be mapped to the task, with 5 additional in the CUSTOM_DIM dimension. If more than 10 task attributes exist, only the first 10 are mapped; the ones that remain are not mapped.
CUSTOM_ATTRIBUTE2		
CUSTOM_ATTRIBUTE3		
CUSTOM_ATTRIBUTE4		
CUSTOM_ATTRIBUTE5		
CUSTOM_ATTRIBUTE6		
CUSTOM_ATTRIBUTE7		
CUSTOM_ATTRIBUTE8		
CUSTOM_ATTRIBUTE9		
CUSTOM_ATTRIBUTE10		
SRC_CRT_TIME_FR_FIRST_CRTD_SEC	int	Calculated time value, in seconds, that counts the time elapsed from task capture from the first system to the source system—for example, fax server to workflow system.
CRT_TIME_FR_SRC_CRTD_SEC	int	Calculated time value, in seconds, from the time of task creation in the source system—for example, workflow—to the time that it was created in iWD.
ACTIVATE_TIME_FROM_CREATED_SEC	int	Calculated value, in seconds, that counts the elapsed time from the time that the task was submitted to iWD until it was activated.

Table 3: The I_/H_TASK_FACT Tables (Continued)

Field	Data Type	Description
ASSIGN_TIME_FROM_ CREATED_SEC	int	Calculated value, in seconds, that counts the elapsed time from when the task was created in iWD until it was assigned to an agent.
COMPLETE_TIME_FROM_ CREATED_SEC	int	Calculated value, in seconds, that counts the elapsed time from when the task was created in iWD until it was completed by the agent.
TOTAL_HELD_TIME_SEC	int	Calculated value, in seconds, that counts the total time that a task was held in iWD. This is an iWD "hold" action via the Web Service Capture API or through iWD Manager user interface and not a hold event from a soft phone or desktop application.
TOTAL_WORK_TIME_SEC	int	Calculated value, in seconds, that counts the time that a task was assigned to an agent until the task was completed by the agent. A task may have multiple work times as noted in TASK_WORK_FACT. This is the total sum.
CREATED_INTERVAL	int	Time interval that is derived from the CREATED_DATE_KEY and CREATED_TIME_KEY fields. Used for ETL scripts.
COMPLETED_INTERVAL	int	Time interval that is derived from the COMPLETED_DATE_KEY and COMPLETED_TIME_KEY fields. Used for ETL scripts.
DUE_TS	int	Timestamp for the iWD task due date and time. Used for ETL scripts.
COMPLETED_TS	int	Timestamp for the iWD task completed date and time. Used for ETL scripts.
ACTIVATION_INTERVAL	int	Time interval that is derived from the ACTIVATION_DATE_KEY and ACTIVATION_TIME_KEY fields. Used for ETL scripts.
ASSIGNED_INTERVAL	int	Time interval that is derived from the ASSIGNED_DATE_KEY and ASSIGN_TIME_KEY fields. Used for ETL scripts.
DUE_INTERVAL	int	Time interval that is derived from the DUE_DATE_KEY and DUE_TIME_KEY fields. Used for ETL scripts.
SOURCE_CREATED_ INTERVAL	int	Time interval that is derived from the SOURCE_CREATED_DATE_ KEY and SOURCE_CREATED_TIME_KEY fields. Used for ETL scripts.
SOURCE_FIRST_ CREATED_INTERVAL	int	Time interval that is derived from the SOURCE_FIRST_ CREATED_DATE_KEY and SOURCE_FIRST_CREATED_TIME_KEY fields. Used for ETL scripts.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this task fact.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this task fact.
TIMEZONE_KEY	int	Key to the TIMEZONE dimension, identifying the time zone of the timestamp when the task was created.
START_DATE_TIME_KEY	int	Key to the DATE_TIME table, identifying the 15-minute interval in which this record was created.

TASK_WORK_FACT Tables

A *task work fact* describes a task from the perspective of enterprise resources. Each time a task is assigned to an agent, iWD records a new task work fact. In the Data Mart, you can access a task work fact through the following objects:

- `I_TASK_WORK_FACT`—Intraday data table
- `H_TASK_WORK_FACT`—Historical data table
- `TASK_WORK_FACT`—Blended view of historical and intraday data

Figure 7 depicts the dimensions that support the iWD task work fact tables and view.

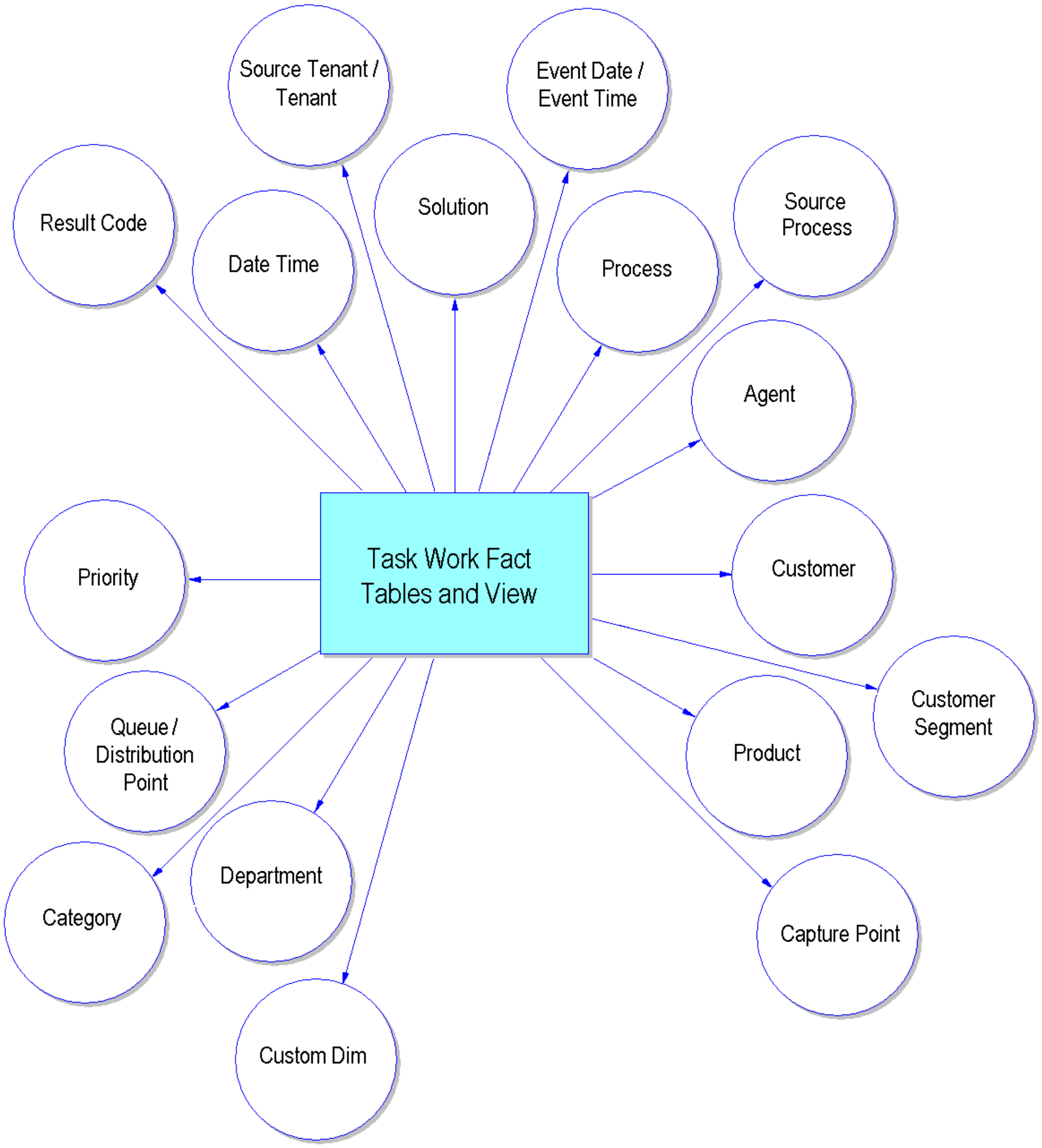


Figure 7: The TASK_WORK_FACT Star Schema

Table 4 describes the columns of the TASK_WORK_FACT intraday and historical tables.

Table 4: The I_/H_TASK_WORK_FACT Tables

Field	Data Type	Description
ASSIGN_TASK_EVENT_ID	int	ID, taken from the Interaction Server event log, that corresponds to the event where the task was assigned to agent. This field, together with SOLUTION_KEY, forms the primary key of this table.
SOLUTION_KEY	int	Key to the SOLUTION dimension. A solution is assigned as soon as a task is created in the Interaction Server database. A tenant can have more than one solution instance. This field, together with ASSIGN_TASK_EVENT_ID, forms the primary key of this table.
INTERACTION_ID	varchar(64)	Interaction ID. This field is unique within a single Interaction Server database.
FINISH_TASK_EVENT_ID	int	ID, taken from the Interaction Server event log, that corresponds to the event where an agent finished working on the task.
IS_ABANDON	int	Indicates whether a task was abandoned: • 0 indicates that the task was not abandoned (status finished). • 1 indicates that the task was abandoned.
CAPTURE_ID	varchar(64)	Capture ID for the task, assigned by the source system. This field is a core task attribute.
TENANT_KEY	int	Key to the TENANT dimension, describing the parent iWD tenant of the task.
DEPARTMENT_KEY	int	Key to the DEPARTMENT dimension, describing the parent iWD department of the task.
PROCESS_KEY	int	Key to the PROCESS dimension, describing the parent iWD process of the task.
CAPTURE_POINT_KEY	int	Key to the CAPTURE_POINT dimension, describing the parent iWD capture point of the task—for example, capture point name = XML File Capture).
DISTRIBUTION_POINT_KEY	int	Key to the DISTRIBUTION_POINT dimension describing the task's parent iWD distribution point (for example, name = Toronto Call Center).
QUEUE_KEY	int	Key to the QUEUE dimension.
ASSIGN_DATE_KEY	int	Key to the EVENT_DATE for when the task was assigned to the agent.
ASSIGN_TIME_KEY	int	Key to the EVENT_TIME for when the task was assigned to the agent.
FINISH_DATE_KEY	int	Key to the EVENT_DATE for when the task was finished by the agent.
FINISH_TIME_KEY	int	Key to the EVENT_TIME for when the task was finished by the agent.
MEDIA_CHANNEL_KEY	int	Key to the MEDIA_CHANNEL dimension, describing the channel through which the task was received—for example, fax. This value can be set in iWD rules or by the system that is submitting the task.

Table 4: The I_/H_TASK_WORK_FACT Tables (Continued)

Field	Data Type	Description
CATEGORY_KEY	int	Key to the BUSINESS_VALUE dimension, describing the category that is associated with the task.
BUSINESS_VALUE_KEY	int	Key to the BUSINESS_VALUE dimension.
PRIORITY_KEY	int	Key to the PRIORITY dimension.
ASSIGNED_AGENT_KEY	int	Key to the AGENT dimension, storing the agent ID for the agent who received the task. This key can be used to retrieve further agent information from Genesys Info Mart, such as Agent Skill, or other employee data from EDW.
RESULT_CODE_KEY	int	Key to the RESULT_CODE dimension.
CUSTOMER_KEY	int	Key to the CUSTOMER dimension, storing the unique value that identifies the customer. Can be used to retrieve further details about the customer from other enterprise data repositories.
CUSTOMER_SEGMENT_KEY	int	Key to the CUSTOMER_SEGMENT dimension, describing the segment for the customer— for example, gold, silver, or bronze).
PRODUCT_KEY	int	Key to the PRODUCT dimension, describing the product type (Internet) and subtype (DSL) that are associated with the task.
SOURCE_TENANT_KEY	int	Key to the SOURCE_TENANT dimension, describing the source tenant (where the source system is part of a multi-tenant environment).
SOURCE_PROCESS_KEY	int	Key to the SOURCE_PROCESS dimension, describing the source process—for example, Order.
CUSTOM_DIM_KEY	int	Key to the CUSTOM_DIM dimension containing five additional attributes (beyond those that are listed below) that can be used to describe a task
CUSTOM_ATTRIBUTE1	varchar(255)	Custom attributes describe a task. A total of 10 customer attributes can be mapped to the task, with 5 additional in the CUSTOM_DIM dimension.
CUSTOM_ATTRIBUTE2		
CUSTOM_ATTRIBUTE3		
CUSTOM_ATTRIBUTE4		
CUSTOM_ATTRIBUTE5		
CUSTOM_ATTRIBUTE6		
CUSTOM_ATTRIBUTE7		
CUSTOM_ATTRIBUTE8		
CUSTOM_ATTRIBUTE9		
CUSTOM_ATTRIBUTE10		
WORK_TIME_SEC	int	Calculated value, in seconds, where the work time is the time from agent complete to agent assigned.
FINISH_INTERVAL	int	Time interval that is derived from the FINISH_DATE_KEY and FINISH_TIME_KEY fields. Used for ETL scripts.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this task work fact.

Table 4: The I_/H_TASK_WORK_FACT Tables (Continued)

Field	Data Type	Description
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this task work fact.
START_DATE_TIME_KEY	int	Key to the DATE_TIME table, identifying the 15-minute interval in which this record was created.

TASK_EVENT_FACT Tables

A *task event* provides detailed audit information about a task. Each task update, deletion, or creation in iWD Manager generates an audit event in the Interaction Server database. When a rule is applied in iWD, for example, an audit event record stores the date and time at which the rule was applied, the name of the rule, and its result (such as an increase in priority from 100 to 110) to the I_TASK_EVENT_FACT table. These events can be seen in iWD Manager runtime view. Each runtime audit event results in a single task event fact.

In the iWD Data Mart, you access task event facts through the following database objects:

- I_TASK_EVENT_FACT—Intraday data table
- H_TASK_EVENT_FACT—Historical data table
- TASK_EVENT_FACT—Blended view of historical and intraday data

Figure 8 shows the dimensions that support iWD task event fact tables and view.

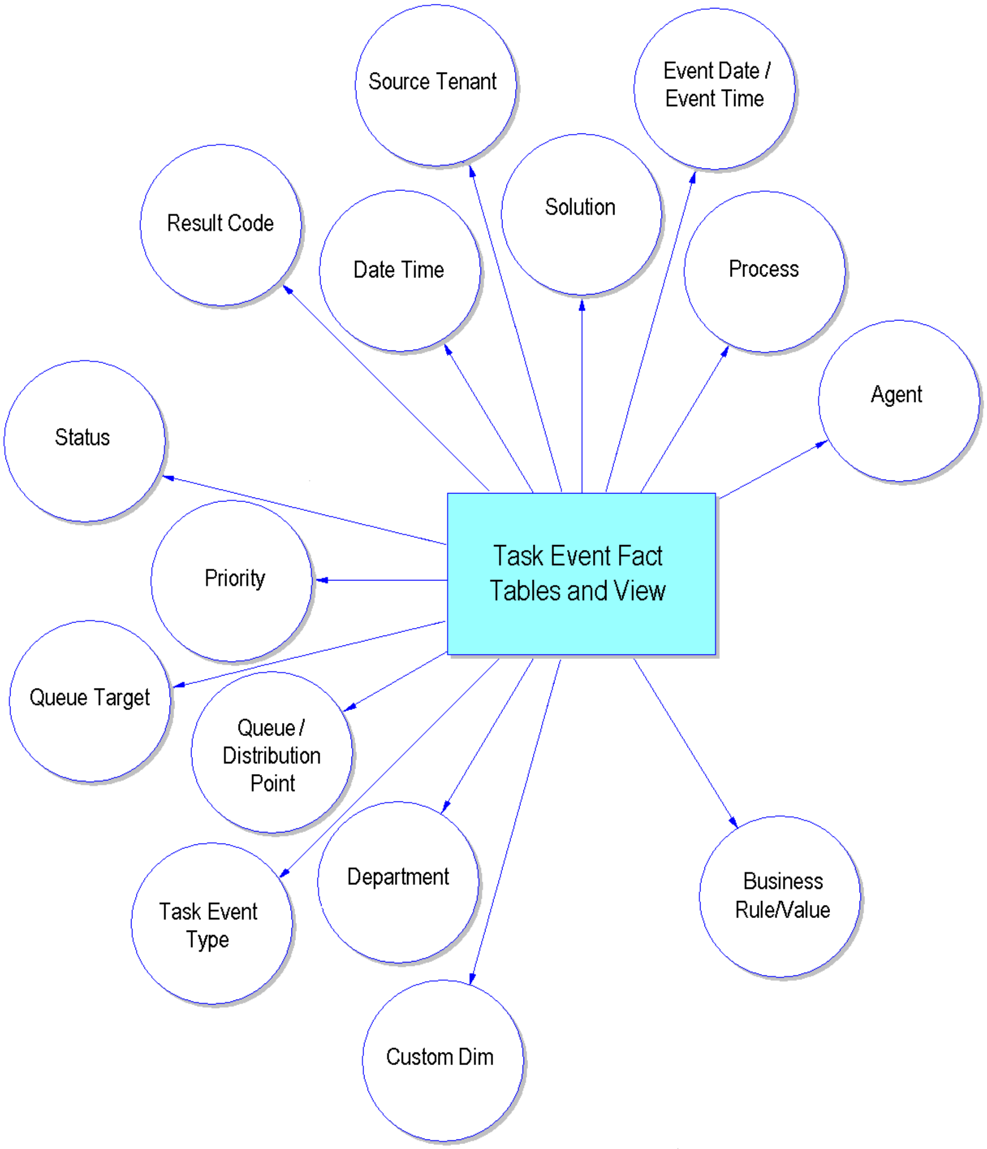


Figure 8: The TASK_EVENT_FACT Star Schema

Table 5 describes the columns of the TASK_EVENT_FACT intraday and historical tables.

Table 5: The I_/H_TASK_EVENT_FACT Tables

Field	Data Type	Description
TASK_EVENT_ID	int	Unique ID for the event. This field, together with SOLUTION_KEY, forms the primary key of this table.
SOLUTION_KEY	int	Key to the SOLUTION dimension, describing the solution instance of the task (as configured in iWD Manager)—for example, Production versus Test. A solution is assigned as soon as a task is created in the Interaction Server database. A tenant can have more than one solution instance. This field, together with TASK_EVENT_ID, forms the primary key of this table.
CAPTURE_ID	varchar(64)	Capture ID for the task, assigned by the source system. This field is a core task attribute.
INTERACTION_ID	varchar(64)	Interaction ID. This field is unique within a single Interaction Server database.
TASK_EVENT_TYPE_KEY	int	Key to the TASK_EVENT_TYPE dimension. Event types are iWD event types, such as RULE_APPLIED. You can view event types in the task history in iWD Manager.
DISTRIBUTION_POINT_KEY	int	Key to the DISTRIBUTION_POINT dimension, identifying the task's parent iWD distribution point (for example, name = Toronto Call Center).
EVENT_DATE_KEY	int	Key to the EVENT_DATE dimension, identifying the date on which the event occurred.
EVENT_TIME_KEY	int	Key to the EVENT_TIME dimension, identifying the time at which the event occurred.
STATUS_KEY	int	Key to the STATUS dimension.
ACTIVATION_DATE_KEY	int	Key to the EVENT_DATE dimension, describing the iWD task activation date. Tasks can be submitted and not acted upon until this date.
ACTIVATION_TIME_KEY	int	Key to the EVENT_TIME dimension, describing the iWD activation time of the iWD task.
DUE_DATE_KEY	int	Key to the EVENT_DATE dimension, describing the task due date (as set by iWD rules or by the source system).
DUE_TIME_KEY	int	Key to the EVENT_TIME dimension, describing task due time.
BUSINESS_VALUE_KEY	int	Key to the BUSINESS_VALUE dimension.
PRIORITY_KEY	int	Key to the PRIORITY dimension.
RULE_KEY	int	Key to the BUSINESS_RULE dimension, describing the business rule that generated the event record. This field might not apply to all events and, therefore, might not be populated for all records in this table.
ASSIGNED_AGENT_KEY	int	Key to the AGENT dimension, describing the agent to whom the task was assigned when the event record resulted in a task assignment to an agent.
RESULT_CODE_KEY	int	Key to the RESULT_CODE dimension when the event is an update of a result code from the agent or source system.

Table 5: The I_/H_TASK_EVENT_FACT Tables (Continued)

Field	Data Type	Description
DEPARTMENT_KEY	int	Key to the DEPARTMENT dimension, describing the iWD department.
PROCESS_KEY	int	Key to the PROCESS dimension, describing the iWD process.
ENTERED_QUEUE_KEY	int	Key to the QUEUE dimension, describing the queue into which the task entered (for the DISTRIBUTE_WORKBIN and DISTRIBUTE_QUEUE event types).
ENTERED_QUEUE_TARGET_KEY	int	Key to the QUEUE_TARGET dimension, describing the agent, agent group, place, or place group to which the task was assigned in a new (entered) workbin (for the DISTRIBUTE_WORKBIN and DISTRIBUTE_QUEUE event types).
EXITED_QUEUE_KEY	int	Key to the QUEUE dimension, describing the queue from which the task exited (for DISTRIBUTE_WORKBIN and DISTRIBUTE_QUEUE event types).
EXITED_QUEUE_TARGET_KEY	int	Key to the QUEUE_TARGET dimension, describing the agent, agent group, place, or place group to which the task was assigned in a previous (exit) workbin (for the DISTRIBUTE_ WORKBIN and DISTRIBUTE_QUEUE event types).
CUSTOM_DIM_KEY	int	Key to the CUSTOM_DIM dimension,
WORK_TIME_SEC	int	Number of seconds that the agent worked on task
HELD_TIME_SEC	int	Number of seconds that the task was in HELD state
EVENT_INTERVAL	int	Time interval that is derived from the EVENT_DATE_KEY and EVENT_TIME_KEY fields. Used for ETL scripts.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this task event fact.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this task event fact.
START_DATE_TIME_KEY	int	Key to the DATE_TIME table, identifying the 15-minute interval in which this record was created.

iWD Data Mart Aggregate Fact Tables

Aggregate facts are aggregated representations of the core facts that were described in previous section. There are three main purposes for aggregated facts:

- Simplified data queries
- Increased query performance
- Decreased database size (granular core facts do not need to be stored for an extended period of time)

Each aggregated fact in iWD Data Mart is an aggregation of two dimensions, one of which is always a time interval. iWD Data Mart directly aggregates facts for two time intervals:15 minutes (intraday and historical) and day (historical).

In addition, those aggregation levels represented in Figure 9 are supported via database views.

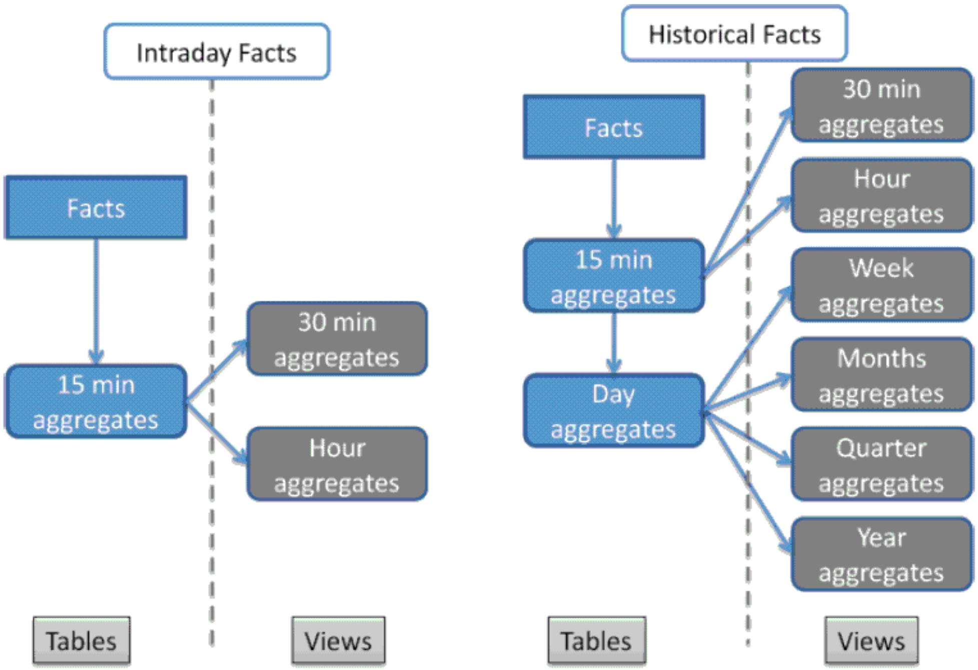


Figure 9: Aggregation Levels

Each aggregated table or view in the Data Mart is postfixed with a time interval: `_15MIN`, `_30MIN`, `_HOUR`, `_DAY`, `_WEEK`, `_MONTH`, `_QUARTER`, or `_YEAR`.

Similar to the core facts, intraday aggregations are prefixed with `I_` and historical aggregations are prefixed with `H_`. Blended aggregations are available only for 15-minute, 30-minute, and hourly time intervals, as shown in Figure 10. Table 6 lists all of the possible aggregation tables and views per single aggregation subject area.

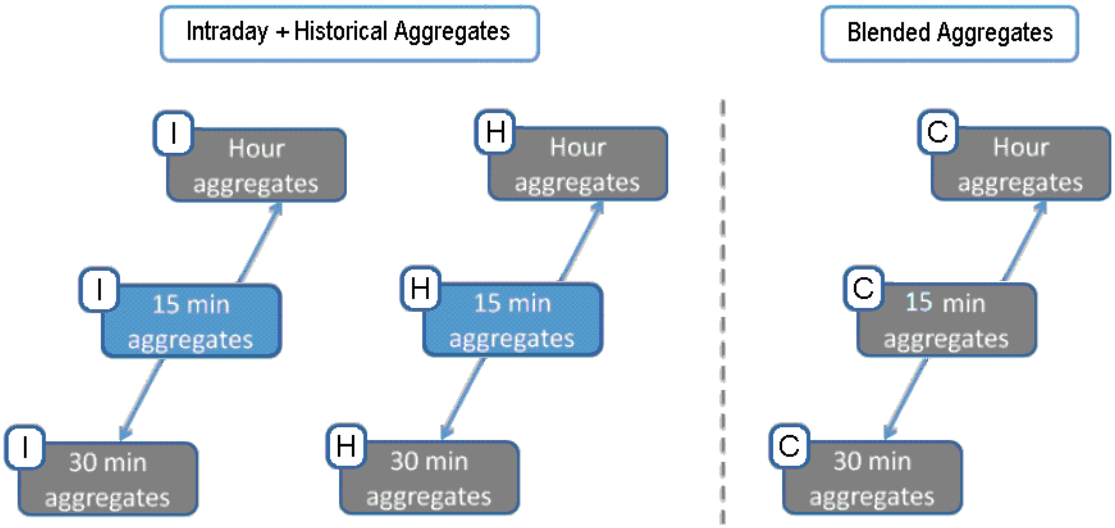


Figure 10: Blended Aggregation Levels

Table 6: Aggregation Tables/Views per Subject Area

Name	Aggregate Type	Type
I_<*subj_area*>_15MIN	Intraday 15-min aggregation	Table
I_<*subj_area*>_30MIN	Intraday 30-min aggregation	View
I_<*subj_area*>_HOUR	Intraday hourly aggregation	View
H_<*subj_area*>_15MIN	Historical 15-min aggregation	Table
H_<*subj_area*>_DAY	Historical daily aggregation	Table
H_<*subj_area*>_WEEK	Historical weekly aggregation	View
H_<*subj_area*>_MONTH	Historical monthly aggregation	View
H_<*subj_area*>_QUARTER	Historical quarterly aggregation	View
H_<*subj_area*>_YEAR	Historical yearly aggregation	View
<*subj_area*>_15MIN	Blended 15-min aggregation	View
<*subj_area*>_30MIN	Blended 30-min aggregation	View
<*subj_area*>_HOUR	Blended hour aggregation	View

The iWD Data Mart provides aggregate tables and views for the following subject areas:

- TASK_AGENT_FACT (described beginning on page 42)
- TASK_CAPT_FACT (page 44)
- TASK_CLASSIF_FACT (page 47)
- TASK_QUEUE_FACT (page 50)

Example

So, for example, the complete set of database tables and views that are provided for the TASK_CAPT_FACT subject area are the following:

Tables

- I_TASK_CAPT_FACT_15MIN
- H_TASK_CAPT_FACT_15MIN
- H_TASK_CAPT_FACT_DAY

Views

- I_TASK_CAPT_FACT_30MIN
- I_TASK_CAPT_FACT_HOUR
- H_TASK_CAPT_FACT_WEEK
- H_TASK_CAPT_FACT_MONTH
- H_TASK_CAPT_FACT_QUARTER
- H_TASK_CAPT_FACT_YEAR
- TASK_CAPT_FACT_15MIN
- TASK_CAPT_FACT_30MIN
- TASK_CAPT_FACT_HOUR

TASK_AGENT_FACT Aggregate

The *task agent aggregate* provides measures which are grouped by task process, the queue from which tasks were distributed, the result code, and the agent who was assigned the task over the different time intervals. Data is aggregated from the I_TASK_WORK_FACT and H_TASK_WORK_FACT tables. This table's values reflect an aggregate of the number of times that an agent has worked on tasks which could differ from the number of tasks that were actually completed (for those tasks that were not assigned to agents).

Figure 11 shows the dimensions that support the iWD task agent aggregate tables and views. Table 7 shows the structure of the TASK_AGENT_FACT aggregate tables.

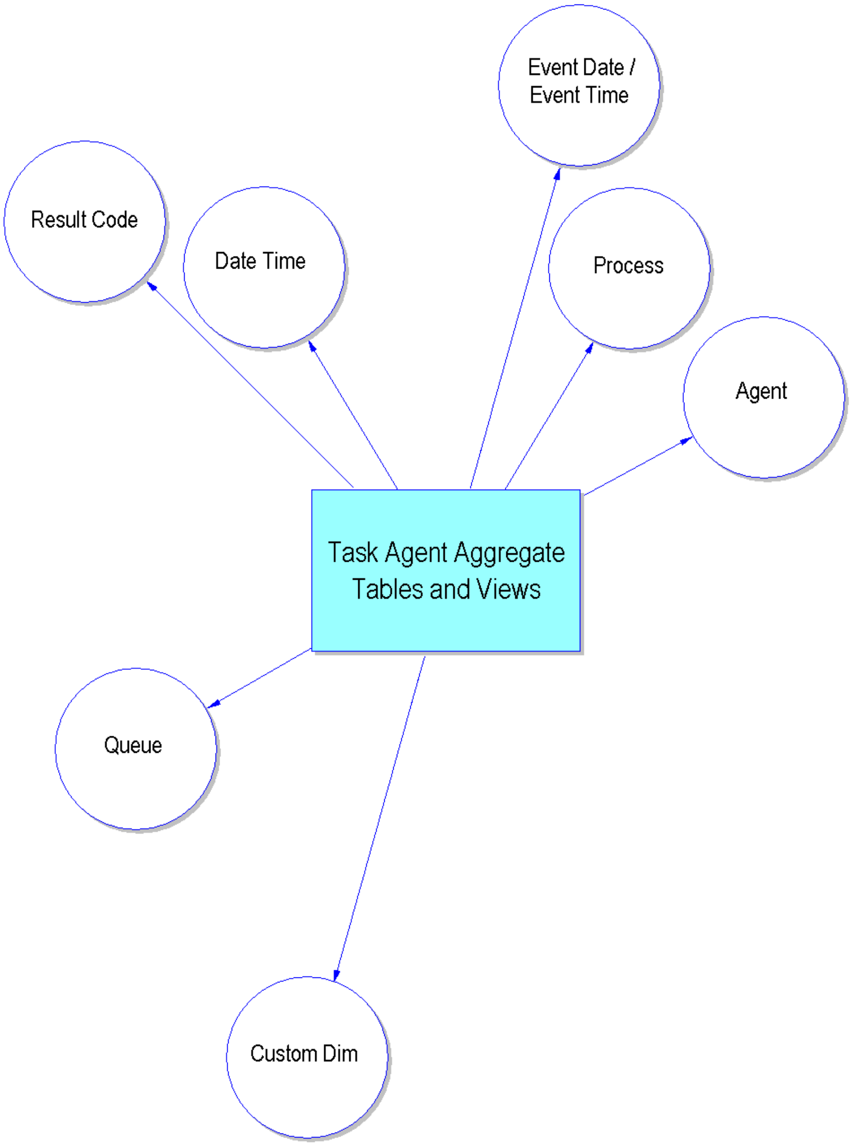


Figure 11: The TASK_AGENT_FACT Star Schema

Table 7: The TASK_AGENT_FACT Aggregate Tables

Field	Data Type	PK	Description
INTERVAL_DATE_KEY	int	✓	Key to the EVENT_DATE dimension, indicating the start date of time interval.
INTERVAL_TIME_KEY	int	✓	Key to the EVENT_TIME dimension, indicating the start time of the time interval. This field is present only for sub-day aggregation levels—for example, 15 minutes, 30 minutes, and hourly.
AGENT_KEY	int	✓	Key to the AGENT dimension.
PROCESS_KEY	int	✓	Key to the PROCESS dimension.
QUEUE_KEY	int	✓	Key to the QUEUE dimension
CUSTOM_DIM_KEY	int	✓	Key to the CUSTOM_DIM dimension, identifying custom attributes of the task agent fact.
RESULT_CODE_KEY	int	✓	Key to the RESULT_CODE dimension.
INTERVAL_KEY	int		Technical field that is derived from the INTERVAL_DATE_KEY and INTERVAL_TIME_KEY fields for ETL internal use. $\frac{(\mathrm{IDateKey} \times 1440) + \mathrm{ITimeKey}}{15}$ This field is present only for sub-day aggregation levels.
TASK_WORK_COUNT	int		Number of tasks that the agent has handled during the given time interval.
AVG_WORK_TIME	int		Average amount of time, in seconds, that the agent spent working on a task (finished – assigned).
MIN_WORK_TIME	int		Least amount of time, in seconds, that the agent spent working on a task.
MAX_WORK_TIME	int		Most amount of time, in seconds, that the agent spent working on a task.
WORK_TIME	int		Total time, in seconds, that the agent spent working on a task (finished – assigned).
DATE_TIME_KEY	int		Key to the DATE_TIME dimension

TASK_CAPT_FACT Aggregate

The *task capture aggregate* provides measures that are grouped by task process, business value, and capture point over different time intervals. Data is aggregated from the I_TASK_FACT and H_TASK_FACT tables. Figure 12 shows the dimensions that support the iWD task capture aggregate tables and views. Table 8 shows the structure of the TASK_CAPT_FACT aggregate tables.

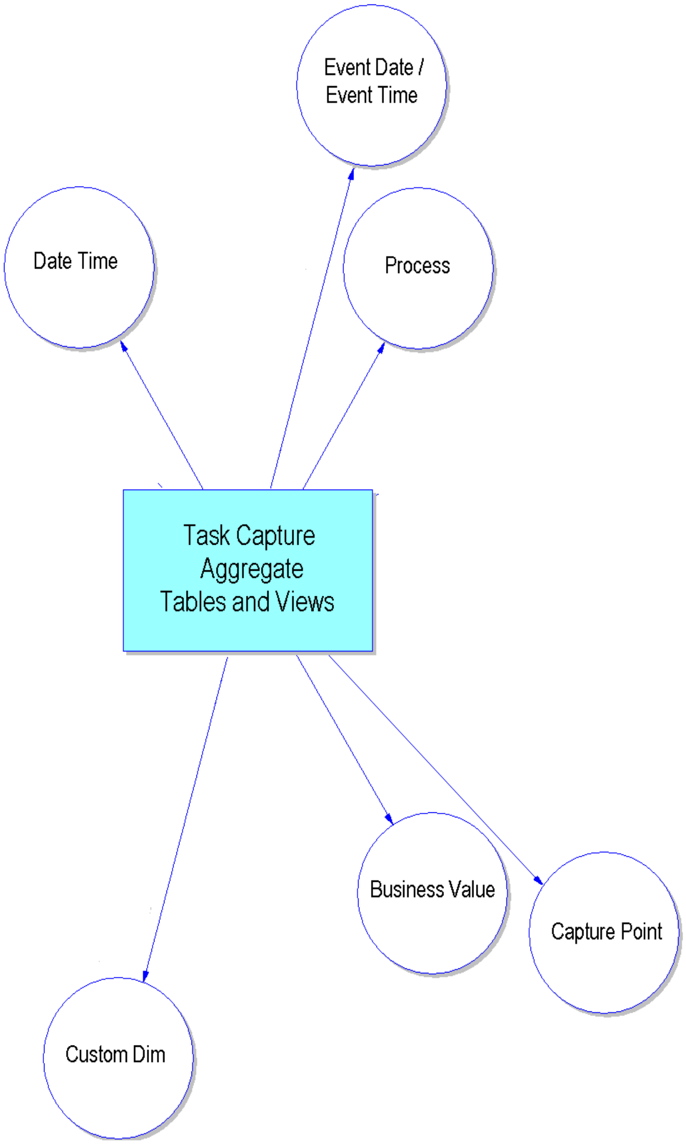


Figure 12: The TASK_CAPT_FACT Star Schema

Table 8: The TASK_CAPT_FACT Aggregate Tables

Field	Data Type	PK	Description
INTERVAL_DATE_KEY	int	✓	Key to the EVENT_DATE dimension, indicating the start date of the time interval. In combination with INTERVAL_TIME_KEY, represents the time aggregation axis.
INTERVAL_TIME_KEY	int	✓	Key to the EVENT_TIME dimension, indicating the start time of the time interval. This field is present only for sub-day aggregation levels—for example, 15 minutes, 30 minutes, and hourly.
PROCESS_KEY	int	✓	Key to the PROCESS dimension.
CAPTURE_POINT_KEY	int	✓	Key to the CAPTURE_POINT dimension.
CUSTOM_DIM_KEY	int	✓	Key to the CUSTOM_DIM dimension, identifying custom attributes of the task capture fact.
BUSINESS_VALUE_KEY	int	✓	Key to BUSINESS_VALUE dimension.
INTERVAL_KEY	int		Technical field that is derived from the INTERVAL_DATE_KEY and INTERVAL_TIME_KEY fields for ETL internal use. $\frac{(\text{IDateKey} \times 1440) + \text{ITimeKey}}{15}$ This field is present only for sub-day aggregation levels.
NEW_TASK_COUNT	int		Number of new tasks that were submitted to iWD during the given time interval. A task is counted only after it has been classified (where status is at least Queued).
CMPL_TASK_COUNT	int		Number of tasks that were completed during the given time interval.
COMPLETED_ OVERDUE_TASK_COUNT	int		Number of tasks that were completed during the given time interval that had been overdue; that is, the service-level agreement (SLA) for the task expired, or the due date and time were not met.
CMPL_TASK_ AVG_WORK_TIME	int		Average agent work time, in seconds, for completed tasks during the given time interval (from finished date and time stamp – assigned date and time stamp for a task). See task history in iWD Manager for examples.
CMPL_TASK_ AVG_ASSIGN_TIME	int		Average time, in seconds, before a task was assigned for the first time. This is calculated as the average of from the task assigned timestamp – task creation timestamp for completed tasks during the given time interval. This measure reflects how long, on average, tasks were in backlog before they were assigned to an agent.
CMPL_TASK_ AVG_COMPLETE_TIME	int		Average time, in seconds, before a task was completed. This is derived from the task completed timestamp – created timestamp for completed tasks during the given time interval. Similar to CMPL_TASK_AVG_ASSIGN_TIME, this measure reflects how long a task was in backlog, but it also includes work time.

Table 8: The TASK_CAPT_FACT Aggregate Tables (Continued)

Field	Data Type	PK	Description
CMPL_TASK_ AVG_SRC_TIME	int		Average time, in seconds, that a task spent in the preceding system before it was submitted to and created within iWD. This is calculated as the average of (iWD – source system) creation timestamp for completed tasks during the given time interval. The creation timestamp from the source system is an extended attribute (sourceCreatedDateTime) that must be provided by the source system.
CMPL_TASK_ AVG_PRE_SRC_TIME	int		Average pre-source system time (source created – first created), in seconds, for completed tasks during the given time interval. Similar to CMPL_TASK_AVG_SRC_TIME except that the beginning time stamp is that of the system before the preceding system to iWD—for example, fax server to workflow to iWD, fax server is the source that was first created.
TOTAL_ PENDING_TASK_COUNT	int		Current number of pending tasks (where the status is Queued, Assigned, or Held) at the end of the given time interval.
TOTAL_ OVERDUE_TASK_COUNT	int		Current number of pending tasks that are overdue at the end of the given time interval. A task is considered overdue when the SLA due date and time has expired.
CMPL_TASK_ WORK_TIME			Total agent work time, in seconds, for completed tasks during the given time interval (from finished date and time stamp – assigned date and time stamp for a task). See task history in iWD Manager for examples.
CMPL_TASK_ ASSIGN_TIME			Total time, in seconds, before a task was assigned for the first time. This is calculated as the task-assigned timestamp – task-creation timestamp for completed tasks during the given time interval. This measure reflects how long tasks were backlogged before they were assigned to agents.
CMPL_TASK_ COMPLETE_TIME			Total time, in seconds, before task were completed. This is derived from the task completed timestamp – created timestamp for completed tasks during the given time interval. Similar to CMPL_TASK_AVG_ASSIGN_TIME, this measure reflects how long tasks were backlogged, but it also includes work time.
CMPL_TASK_SRC_TIME			Total time, in seconds, that a task spent in the preceding system before it was submitted to and created within iWD. This is calculated as the (iWD – source system) creation timestamp for completed tasks during the given time interval. The creation timestamp from the source system is an extended attribute (sourceCreatedDateTime) that must be provided by the source system.
CMPL_TASK_ PRE_SRC_TIME			Total pre-source system time (source created – first created), in seconds, for completed tasks during the given time interval. Similar to CMPL_TASK_AVG_SRC_TIME except that the beginning time stamp is that of the system before the preceding system to iWD—for example, fax server to workflow to iWD, fax server is the source that was first created.
DATE_TIME_KEY			Key to DATE_TIME dimension.

TASK_CLASSIF_FACT Aggregate

The *task classification aggregate* provides measures that are grouped by task process, media, category, customer segment, product, source process, and source tenant over different time intervals. Data is aggregated from the I_TASK_FACT table. Figure 13 shows the dimensions that support the iWD task classification aggregate tables and views. Table 9 shows the structure of the TASK_CLASSIF_FACT aggregate tables.

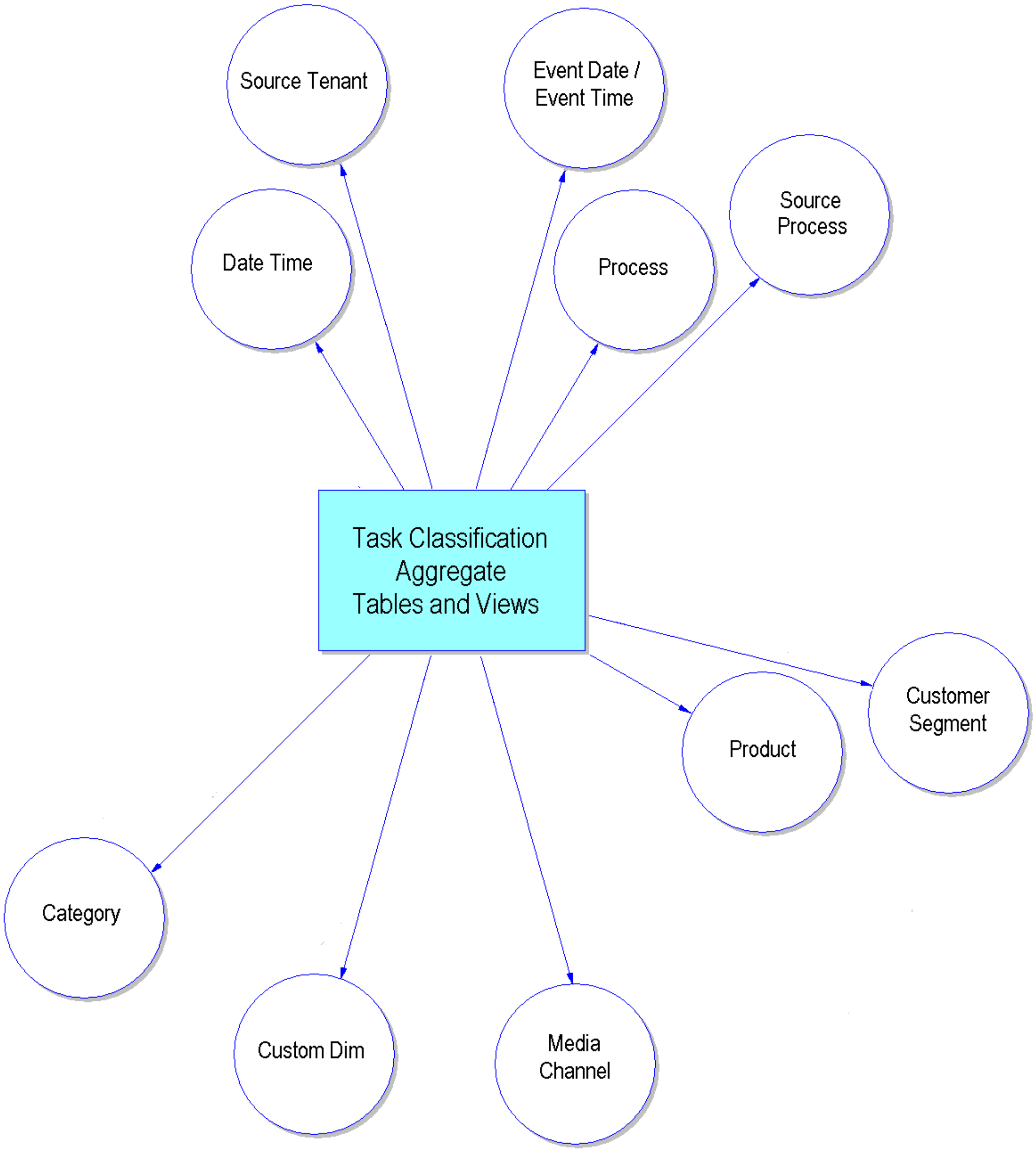


Figure 13: The TASK_CLASSIF_FACT Star Schema

Table 9: The TASK_CLASSIF_FACT Aggregate Tables

Field	Data Type	PK	Description
INTERVAL_DATE_KEY	int	✓	Key to the EVENT_DATE dimension, indicating the start date of the time interval. In combination with INTERVAL_TIME_KEY, represents the time aggregation axis.
INTERVAL_TIME_KEY	int	✓	Key to the EVENT_TIME dimension, indicating the start time of the time interval. This field is present only for sub-day aggregation levels—for example, 15 minutes, 30 minutes, and hourly.
PROCESS_KEY	int	✓	Key to the PROCESS dimension.
MEDIA_CHANNEL_KEY	int	✓	Key to the MEDIA_CHANNEL dimension, describing the channel through which the task was received—for example, fax. This value can be set in iWD rules or by the system that is submitting the task.
CATEGORY_KEY	int	✓	Key to the CATEGORY dimension.
CUSTOMER_SEGMENT_ KEY	int	✓	Key to the CUSTOMER_SEGMENT dimension.
PRODUCT_KEY	int	✓	Key to the PRODUCT dimension.
SOURCE_PROCESS_KEY	int	✓	Key to the SOURCE_PROCESS dimension.
SOURCE_TENANT_KEY	int	✓	Key to the SOURCE_TENANT dimension.
CUSTOM_DIM_KEY	int	✓	Key to the CUSTOM_DIM dimension, identifying custom attributes of the task classification fact.
INTERVAL_KEY	int		Technical field that is derived from the INTERVAL_DATE_KEY and INTERVAL_TIME_KEY fields for ETL internal use. $\frac{(\text{IDateKey} \times 1440) + \text{ITimeKey}}{15}$ This field is present only for sub-day aggregation levels.
NEW_TASK_COUNT	int		Number of new tasks that were submitted to iWD during the given time interval. The task is counted only after it has been classified (where status is at least Queued).
CMPL_TASK_COUNT	int		Number of tasks that have been completed during the given time interval.
COMPLETED_ OVERDUE_TASK_COUNT	int		Number of tasks that have been completed during the given time interval that had been overdue.
CMPL_TASK_ AVG_WORK_TIME	int		Average agent work time (finished – assigned), in seconds, for completed tasks during the given time interval.
CMPL_TASK_ AVG_ASSIGN_TIME	int		Average time, in seconds, before a task was assigned for the first time. This is calculated as the average of (task assigned – task creation) timestamp for completed tasks during the given time interval. This measure reflects how long, on average, tasks were in backlog before they were assigned to an agent.
CMPL_TASK_ AVG_COMPLETE_TIME	int		Average time, in seconds, before a task was completed. This is calculated as the average of (task completed – task creation) timestamp for completed tasks during the given time interval. Similar to CMPL_TASK_AVG_ASSIGN_TIME, this measure reflects how long a task was in backlog, but it also includes work time.

Table 9: The TASK_CLASSIF_FACT Aggregate Tables (Continued)

Field	Data Type	PK	Description
CMPL_TASK_ AVG_SRC_TIME	int		Average time, in seconds, that a task spent in the preceding system before it was submitted to and created within iWD. This is calculated as the average of (iWD – source system) creation timestamp for completed tasks during the given time interval. The creation timestamp from the source system is an extended attribute (sourceCreatedDateTime) that must be provided by the source system.
CMPL_TASK_ AVG_PRE_SRC_TIME	int		Average pre-source system time (source created – first created), in seconds, for completed tasks during the given time interval.
TOTAL_ PENDING_TASK_COUNT	int		Current number of pending tasks (where the status is Queued, Assigned, or Held) at the end of the given time interval.
TOTAL_ OVERDUE_TASK_COUNT	int		Current number of pending tasks that are overdue at the end of the given time interval. A task is considered overdue when the SLA due date and time has been missed.
CMPL_TASK_WORK_TIME	int		Total agent work time (finished – assigned), in seconds, for completed tasks during the given time interval.
CMPL_TASK_ ASSIGN_TIME	int		Total time, in seconds, before a task was assigned for the first time. This is calculated as the (task assigned – task creation) timestamp for completed tasks during the given time interval. This measure reflects how long tasks were backlogged before they were assigned to an agent.
CMPL_TASK_ COMPLETE_TIME	int		Total time, in seconds, before a task was completed. This is calculated as the (task completed – task creation) timestamp for completed tasks during the given time interval. Similar to CMPL_TASK_AVG_ASSIGN_TIME, this measure reflects how long tasks were backlogged, but it also includes work time.
CMPL_TASK_SRC_TIME	int		Total time, in seconds, that tasks spent in the preceding system before they were submitted to and created within iWD. This is calculated as the (iWD – source system) creation timestamp for completed tasks during the given time interval. The creation timestamp from the source system is an extended attribute (sourceCreatedDateTime) that must be provided by the source system.
CMPL_TASK_ PRE_SRC_TIME	int		Total pre-source system time (source created – first created), in seconds, for completed tasks during the given time interval.
DATE_TIME_KEY	int		Key to the DATE_TIME dimension.

TASK_QUEUE_FACT Aggregate

The *task queue aggregate* provides measures that are grouped by business process, priority, and queue over different time intervals. Data is aggregated from the I_TASK_FACT and I_TASK_EVENT_FACT tables. Figure 14 shows the dimensions that support the iWD task queue aggregate tables and views. Table 10 shows the structure of the TASK_QUEUE_FACT aggregate tables.

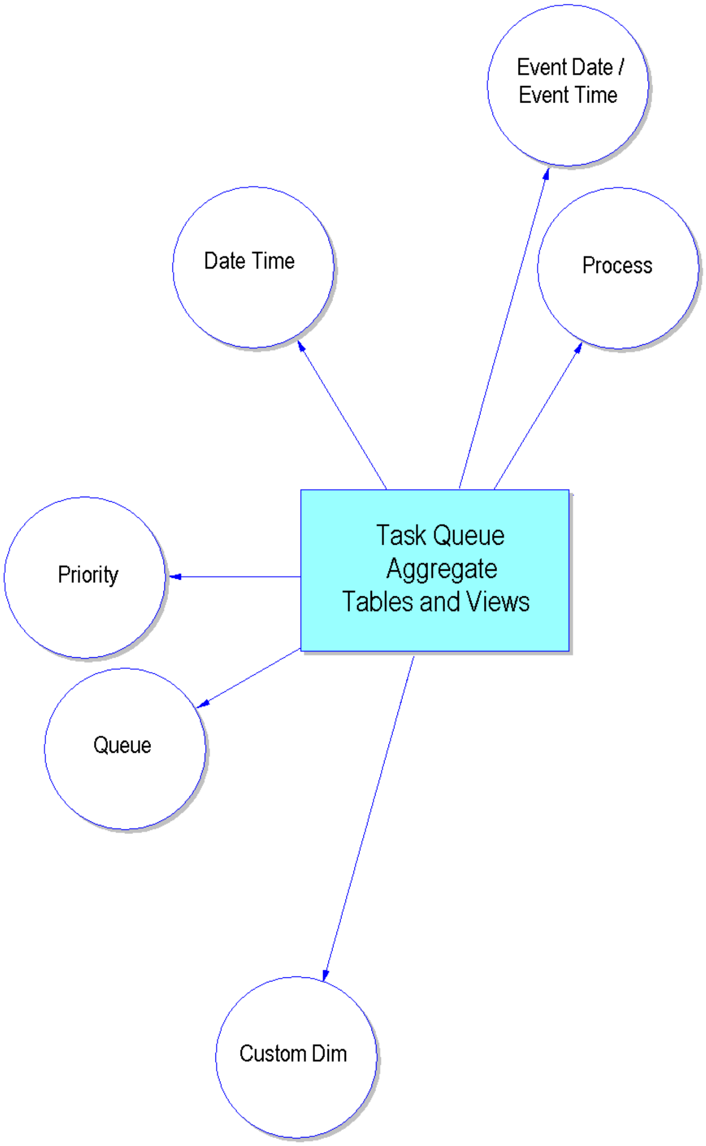


Figure 14: The TASK_QUEUE_FACT Star Schema

Table 10: The TASK_QUEUE_FACT Aggregate Tables

Field	Data Type	PK	Description
INTERVAL_DATE_KEY	int	✓	Key to the EVENT_DATE dimension
INTERVAL_TIME_KEY	int	✓	Key to the EVENT_TIME dimension

Table 10: The TASK_QUEUE_FACT Aggregate Tables (Continued)

Field	Data Type	PK	Description
QUEUE_KEY	int	✓	Key to the QUEUE dimension.
PROCESS_KEY	int	✓	Key to the PROCESS dimension.
CUSTOM_DIM_KEY	int	✓	Key to the CUSTOM_DIM dimension, identifying custom attributes of the task queue fact.
PRIORITY_KEY	int	✓	Key to the PRIORITY dimension.
INTERVAL_KEY	int		Technical field that is derived from the INTERVAL_DATE_KEY and INTERVAL_TIME_KEY fields for ETL internal use. $\frac{(\text{IDateKey} \times 1440) + \text{ITimeKey}}{15}$
ENTERED_TASK_COUNT	int		Number of tasks that entered the queue or workbin during the given time interval.
EXITED_TASK_COUNT	int		Number of tasks that exited the queue or workbin during the given time interval.
CMPL_TASK_COUNT	int		Number of tasks that were completed during the given time interval.
COMPLETED_ OVERDUE_TASK_COUNT	int		Number of completed tasks that were overdue during the given time interval.
CMPL_TASK_ AVG_WORK_TIME	int		Average amount of time, in seconds, that agents worked on tasks that were completed during the given time interval.
CMPL_TASK_ AVG_ASSIGN_TIME	int		Average time, in seconds, before task were assigned for the first time. This is calculated as the average of from the task assigned timestamp minus the task creation timestamp for completed tasks during the given time interval. This measure reflects how long, on average, tasks were backlogged before they were assigned to an agent.
CMPL_TASK_ AVG_COMPLETE_TIME	int		Average time, in seconds, that tasks were completed within iWD during the given time interval.
CMPL_TASK_ AVG_SRC_TIME	int		Average time, in seconds, that tasks spent in the preceding system before they were submitted to and created within iWD. This is calculated as the average of (iWD – source system) creation timestamp for completed tasks during the given time interval. The creation timestamp from the source system is an extended attribute (sourceCreatedDateTime) that must be provided by the source system.
CMPL_TASK_ AVG_PRE_SRC_TIME	int		Average pre-source system time (source created – first created), in seconds, for completed tasks during the given time interval.
TOTAL_ PENDING_TASK_COUNT	int		Current number of pending tasks (where the status is Queued, Assigned, or Held) at the end of the given time interval.
TOTAL_ OVERDUE_TASK_COUNT	int		Current number of pending tasks that are overdue tasks at the end of the given time interval. A task is considered overdue when the SLA due date and time has been missed.
DATE_TIME_KEY	int		Key to the DATE_TIME dimension.
CMPL_TASK_WORK_TIME	int		Total time, in seconds, that agents worked on tasks that were completed during the given time interval.

Table 10: The TASK_QUEUE_FACT Aggregate Tables (Continued)

Field	Data Type	PK	Description
CMPL_TASK_ ASSIGN_TIME	int		Total time, in seconds, before a task was assigned for the first time. This is calculated as the task assigned timestamp minus the task creation timestamp for completed tasks during the given time interval. This measure reflects how long tasks were backlogged before they were assigned to an agent.
CMPL_TASK_ COMPLETE_TIME	int		Total time, in seconds, that tasks were completed within iWD during the given time interval.
CMPL_TASK_SRC_TIME	int		Total time, in seconds, that tasks spent in the preceding system before they were submitted to and created within iWD. This is calculated as the (iWD – source system) creation timestamp for completed tasks during the given time interval. The creation timestamp from the source system is an extended attribute (`sourceCreatedDateTime`) that must be provided by the source system.
CMPL_TASK_ PRE_SRC_TIME	int		Total pre-source system time (source created – first created), in seconds, for completed tasks during the given time interval.

iWD Data Mart Dimension Tables

Dimensions contain static or slowly changing information—as well as information that is used in lookups against the fact tables—and provide the basis for OLAP/cube queries. Dimensions in the iWD Data Mart are populated from one of three sources:

- Static values, such as time zone and date/time dimensions
- iWD Configuration, as defined in iWD Manager
- iWD task data

The iWD Data Mart defines the following dimensions:

```
AGE
AGENT
BUSINESS_RULE
BUSINESS_VALUE
CAPTURE_POINT
CATEGORY
CUSTOM_DIM
CUSTOMER
CUSTOMER_SEGMENT
DATE_TIME
DEPARTMENT
DISTRIBUTION_POINT
EVENT_DATE
EVENT_TIME
MEDIA_CHANNEL
METRIC
PRIORITY
PROCESS
PRODUCT
QUEUE
QUEUE_TARGET
RESULT_CODE
SKILL
SOLUION
SOURCE_PROCESS
SOURCE_TENANT
STATUS
TASK_EVENT_TYPE
TENANT
TIMEZONE
```

The following dimensions store core task attributes:

```
AGENT
BUSINESS_VALUE
CAPTURE_POINT
CATEGORY
DEPARTMENT
DISTRIBUTION_POINT
EVENT_DATE
EVENT_TIME
MEDIA_CHANNEL
PRIORITY
PROCESS
SOLUTION
TENANT
```

And, the following dimensions store extended task attributes:

CUSTOMER	PRODUCT	SOURCE_PROCESS
CUSTOMER_SEGMENT	RESULT_CODE	SOURCE_TENANT

In addition, one core attribute (captureId) and several extended task attributes (such as requestedAgent, requestedSkill, sourceCreatedDateTime, source DueDateTime, sourceFirstCreatedDateTime) are stored directly in task FACT tables.

This section describes the structure of each dimension.

AGE Dimension

The AGE dimension is a static dimension that contains age ranges to define the age of a task. Table 11 describes the fields of this dimension.

Table 11: The AGE Dimension

Field	Data Type	Description
AGE_KEY	int	Primary key of this table.
AGE_MINUTES	int	Age, in minutes, at the beginning of the interval.
AGE_RANGE_15MIN	varchar(64)	15–minute range (0–15 minutes, 15–30 minutes, and so on).
AGE_RANGE_1HOUR	varchar(64)	1-hour range (0–1 hour, 1–2 hours, and so on).
AGE_RANGE_4HOUR	varchar(64)	4-hour range (0–4 hours, 4–8 hours, and so on).
AGE_RANGE_8HOUR	varchar(64)	8-hour range (0–8 hours, 8–16 hours, 16–24 hours).
AGE_RANGE_1DAY	varchar(64)	Days (0–1 day, 1–2 days, and so on).
AGE_RANGE_WEEK	varchar(64)	Weeks (0–1 weeks, 1–2 weeks, and so on).

AGENT Dimension

The AGENT dimension, which is populated from task information, stores agent keys and IDs that are captured from the source system. This dimension is a core iWD attribute. Table 12 describes its fields.

Table 12: The AGENT Dimension

Field	Data Type	Description
AGENT_KEY	int	Primary key of this table.
AGENT_ID	varchar(255)	ID of the agent as captured by the source system.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

BUSINESS_RULE Dimension

Populated from iWD Manager configuration, the BUSINESS_RULE dimension contains information about rules. Table 13 describes its fields.

Table 13: The BUSINESS_RULE Dimension

Field	Data Type	Description
BUSINESS_RULE_KEY	int	Primary key of this table.
BUSINESS_RULE_CONFIG_ID	int	iWD Manager ID for the rule.
BUSINESS_RULE_CONFIG_EVENT_ID	int	ID of the event that created or updated the rule record.
BUSINESS_RULE_NAME	varchar(255)	Descriptive name of the rule.
BUSINESS_RULE_RUNTIME_ID	varchar(255)	iWD Manager runtime ID of the rule.
TENANT_KEY	int	Key to the TENANT dimension, identifying the tenant with whom the rule is associated.
SOLUTION_KEY	int	Key to the SOLUTION dimension, identifying the solution with which the rule is associated. A solution is assigned as soon as a task is created in the Interaction Server database. A tenant can have more than one solution instance.
DEPARTMENT_KEY	int	Key to the DEPARTMENT dimension, identifying the department with which the rule is associated.
PROCESS_KEY	int	Key to the PROCESS dimension, identifying the process with which the rule is associated.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.
VALID_FROM_DATE_KEY	int	Date key for the “valid from” rule attribute.
VALID_FROM_TIME_KEY	int	Time key for the “valid from” rule attribute.
VALID_TO_DATE_KEY	int	Date key for the “valid to” rule attribute.
VALID_TO_TIME_KEY	int	Time key for the “valid to” rule attribute.
VALID_FROM	datetime	The date, in YYYY-M-D HH:MM:SS F format, from which this business rule is valid.
VALID_TO	datetime	The date, in YYYY-M-D HH:MM:SS F format, until which this business rule is valid.
VERSION	int	Version of the record.

BUSINESS_VALUE Dimension

BUSINESS_VALUE is a static dimension that contains ranges to define the business value assigned to a task. You can query this dimension to determine how many tasks fall within a particular range; for example:

```
SELECT COUNT(INTERACTION_ID) CNT, BUSINESS_VALUE_RANGE_100 RNG FROM
BUSINESS_VALUE BV
```

```
LEFT JOIN I_TASK_FACT I ON
I.BUSINESS_VALUE_KEY = BV.BUSINESS_VALUE_KEY
WHERE BV.BUSINESS_VALUE_KEY < 1000
GROUP BY BUSINESS_VALUE_RANGE_100
ORDER BY BV.BUSINESS_VALUE_KEY
```

Table 14 describes the fields of this dimension.

Table 14: The BUSINESS_VALUE Dimension

Field	Data Type	Description
BUSINESS_VALUE_KEY	int	Primary key of this table.
BUSINESS_VALUE_RANGE_5	varchar(32)	Values in the business value granularity of 5—that is, “1–5”, “6–10”, and so on.
BUSINESS_VALUE_RANGE_10	varchar(32)	Values in the business value granularity of 10—that is, “1–10”, “11–20”, and so on.
BUSINESS_VALUE_RANGE_50	varchar(32)	Values in the business value granularity of 50—that is, “1–50”, “51–100”, and so on.
BUSINESS_VALUE_RANGE_100	varchar(32)	Values in the business value granularity of 100—that is, “1–100”, “101–200”, and so on.
BUSINESS_VALUE_RANGE_500	varchar(32)	Values in the business value granularity of 500—that is, “1–500”, “501–1000”, and so on.
BUSINESS_VALUE_RANGE_1000	varchar(32)	Values in the business value granularity of 1000—that is, “1–1000”, “1001–2000”, and so on, with a maximum value of 50000.
BUSINESS_VALUE_RANGE_5_STA	int	Values that mark the start of each BUSINESS_VALUE_RANGE_5 range. Values step by 5—for example, 1, 6, 11, and so forth.
BUSINESS_VALUE_RANGE_5_END	int	Values that mark the end of each BUSINESS_VALUE_RANGE_5 range. Values step by 5—for example, 5, 10, 15, and so forth.
BUSINESS_VALUE_RANGE_10_STA	int	Values that mark the start of each BUSINESS_VALUE_RANGE_10 range. Values step by 10—for example, 1, 11, 21, and so forth.
BUSINESS_VALUE_RANGE_10_END	int	Values that mark the end of each BUSINESS_VALUE_RANGE_10 range. Values step by 10—for example, 10, 20, 30, and so forth.
BUSINESS_VALUE_RANGE_50_STA	int	Values that mark the start of each BUSINESS_VALUE_RANGE_50 range. Values step by 50—for example, 1, 51, 101, and so forth.
BUSINESS_VALUE_RANGE_50_END	int	Values that mark the end of each BUSINESS_VALUE_RANGE_50 range. Values step by 50—for example, 50, 100, 150, and so forth.
BUSINESS_VALUE_RANGE_100_STA	int	Values that mark the start of each BUSINESS_VALUE_RANGE_100 range. Values step by 100—for example, 1, 101, 201, and so forth.
BUSINESS_VALUE_RANGE_100_END	int	Values that mark the end of each BUSINESS_VALUE_RANGE_100 range. Values step by 100—for example, 100, 200, 300, and so forth.

Table 14: The BUSINESS_VALUE Dimension (Continued)

Field	Data Type	Description
BUSINESS_VALUE_RANGE_500_STA	int	Values that mark the start of each BUSINESS_VALUE_RANGE_500 range. Values step by 500—for example, 1, 501, 1001, and so forth.
BUSINESS_VALUE_RANGE_500_END	int	Values that mark the end of each BUSINESS_VALUE_RANGE_500 range. Values step by 500—for example, 500, 1000, 1500, and so forth.
BUSINESS_VALUE_RANGE_1000_STA	int	Values that mark the start of each BUSINESS_VALUE_RANGE_1000 range. Values step by 1000—for example, 1, 1001, 2001, and so forth.
BUSINESS_VALUE_RANGE_1000_END	int	Values that mark the end of each BUSINESS_VALUE_RANGE_1000 range. Values step by 1000—for example, 1000, 2000, 3000, and so forth.

CAPTURE_POINT Dimension

Tasks are captured through a capture point that is configured in iWD Manager. Capture point information is stored in the CAPTURE_POINT dimension which is populated from task information. Table 15 describes the fields of this dimension.

Table 15: The CAPTURE_POINT Dimension

Field	Data Type	Description
CAPTURE_POINT_KEY	int	Primary key of this table.
CAPTURE_POINT_CONFIG_ID	int	iWD Manager ID for the capture point.
CAPTURE_POINT_CONFIG_EVENT_ID	int	Event that created or updated the capture point record.
CAPTURE_POINT_RUNTIME_ID	varchar(16)	iWD Manager runtime ID for the capture point.
TENANT_KEY	int	Key to the TENANT dimension, identifying the tenant with whom the capture point is associated.
SOLUTION_KEY	int	Key to the SOLUTION dimension.
CAPTURE_POINT_NAME	varchar(255)	Descriptive name of the capture point.
CAPTURE_POINT_TYPE	varchar(255)	Type of capture point, such as Web Service Capture Point, Database Capture Point, and XML File Capture Point.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.
VALID_FROM	datetime	The date from which this capture point is valid.
VALID_TO	datetime	The date until which this capture point is valid.
VERSION	int	Version of the record.

CATEGORY Dimension

The CATEGORY dimension, which is populated from task information, contains a list of categories that further describe a task, such as a specific type of refund. This dimension is an extended iWD attribute that can be set by the source system. Table 16 describes the fields of this table.

Table 16: The CATEGORY Dimension

Field	Data Type	Description
CATEGORY_KEY	int	Primary key of this table.
CATEGORY_NAME	varchar(255)	Descriptive name of the category.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

CUSTOM_DIM Dimension

A custom attribute is represented by a string value. Each row in the CUSTOM_DIM dimension describes a distinct combination of the custom attributes of a task. Table 17 describes this table's fields.

Table 17: The CUSTOM_DIM Dimension

Field	Data Type	Description
CUSTOM_DIM_KEY	int	Primary key of this table.
CUSTOM_DIM_ATTRIBUTE1	varchar(255)	Custom attribute that further classifies a task.
CUSTOM_DIM_ATTRIBUTE2	varchar(255)	
CUSTOM_DIM_ATTRIBUTE3	varchar(255)	
CUSTOM_DIM_ATTRIBUTE4	varchar(128)	
CUSTOM_DIM_ATTRIBUTE5	varchar(128)	
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

CUSTOMER Dimension

Tasks are typically associated with a customer. The CUSTOMER dimension holds the unique identifier—either customer ID or account ID—and is populated from task information. This dimension is an extended iWD attribute, and its fields are described in Table 18.

Table 18: The CUSTOMER Dimension

Field	Data Type	Description
CUSTOMER_KEY	int	Primary key of this table.
CUSTOMER_ID	varchar(64)	Unique ID assigned to the customer, provided by the source system.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

CUSTOMER_SEGMENT Dimension

A *customer segment* is an extended iWD attribute that further describes a customer. Customer segments are set by the source system that is submitting the task. A segment often represents the value of the client to the enterprise—for example, gold, silver, or bronze. The table that stores this information is CUSTOMER_SEGMENT and is described in Table 19.

Table 19: The CUSTOMER_SEGMENT Dimension

Field	Data Type	Description
CUSTOMER_SEGMENT_KEY	int	Primary key of this table.
CUSTOMER_SEGMENT_NAME	varchar(64)	Descriptive name of the customer segment.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

DATE_TIME Dimension

The DATE_TIME dimension allows facts to be described by attributes of calendar date and 15-minute time interval. It is described in Table 20.

Table 20: The DATE_TIME Dimension

Field	Data Type	Description
DATE_TIME_KEY	int	The primary key of this table. It is used to join a particular 15-minute interval in this table to the fact and aggregate tables. This field increases monotonically to facilitate the calculation of time interval ranges and is equal to the UTC-equivalent time at which the time interval started.
DATE_TIME_30MIN_KEY	int	The surrogate key that is used to join a particular 30-minute interval in this table to the fact and aggregate tables. Two rows in this table share the same value, which is the DATE_TIME_KEY that represents the start of the 30-minute interval.

Table 20: The DATE_TIME Dimension (Continued)

Field	Data Type	Description
DATE_TIME_HOUR_KEY	int	The surrogate key that is used to join a particular hour in this table to the fact and aggregate tables. Four rows in this table share the same value, which is the DATE_TIME_KEY that represents the start of the hour interval.
DATE_TIME_DAY_KEY	int	The surrogate key that is used to join a particular day in this table to the fact and aggregate tables. Ninety-six rows in this table share the same value, which is the DATE_TIME_KEY that represents the start of the day interval.
DATE_TIME_WEEK_KEY	int	The surrogate key that is used to join a particular week in this table to the fact and aggregate tables. Multiple rows in this table share the same value, which is the DATE_TIME_KEY that represents the start of the week interval.
DATE_TIME_MONTH_KEY	int	The surrogate key that is used to join a particular month in this table to the fact and aggregate tables. Multiple rows in this table share the same value, which is the DATE_TIME_KEY that represents the start of the month interval.
DATE_TIME_QUARTER_KEY	int	The surrogate key that is used to join a particular quarter in this table to the fact and aggregate tables. Multiple rows in this table share the same value, which is the DATE_TIME_KEY that represents the start of the quarter interval.
DATE_TIME_YEAR_KEY	int	The surrogate key that is used to join a particular year in this table to the fact and aggregate tables. Multiple rows in this table share the same value, which is the DATE_TIME_KEY that represents the start of the year interval.
DATE_TIME_NEXT_KEY	int	Points to the next record of this table. This value is DATE_TIME_KEY+1.
DATE_TIME_NEXT_30MIN_KEY	int	Points to the DATE_TIME_30MIN_KEY record that represents the next 30-minute period.
DATE_TIME_NEXT_HOUR_KEY	int	Points to the DATE_TIME_HOUR_KEY record that represents the next hour.
DATE_TIME_NEXT_DAY_KEY	int	Points to the DATE_TIME_DAY_KEY record that represents the next calendar day.
DATE_TIME_NEXT_WEEK_KEY	int	Points to the DATE_TIME_WEEK_KEY record that represents the next calendar week.
DATE_TIME_ NEXT_MONTH_KEY	int	Points to the DATE_TIME_MONTH_KEY record that represents the next calendar month.
DATE_TIME_ NEXT_QUARTER_KEY	int	Points to the DATE_TIME_QUARTER_KEY record that represents the next calendar quarter.
DATE_TIME_NEXT_YEAR_KEY	int	Points to the DATE_TIME_YEAR_KEY record that represents the next year.
CREATE_AUDIT_KEY	int	The surrogate key used to join to the CTL_AUDIT_LOG dimension. Specifies the lineage for data creation. This value can be useful for aggregation, enterprise application integration (EAI), and ETL tools—that is, applications that need to identify newly added data.

Table 20: The DATE_TIME Dimension (Continued)

Field	Data Type	Description
UPDATE_AUDIT_KEY	int	The surrogate key used to join to the CTL_AUDIT_LOG dimension. Specifies the lineage for data update. This value can be useful for aggregation, enterprise application integration (EAI), and ETL tools--that is, applications that need to identify recently modified data.
CAL_DATE	datetime	The date/time data type for a calendar date that is specific for this RDBMS.
CAL_DAY_NAME	varchar(32)	The calendar day name—for example, "Sunday".
CAL_MONTH_NAME	varchar(32)	The calendar month name—for example, "January".
CAL_DAY_NUM_IN_WEEK	smallint	The day number in a week. By default, the values start with 1 for Sunday and end with 7 for Saturday.
CAL_DAY_NUM_IN_MONTH	smallint	The day number in the calendar month, starting with 1 and ending with 28, 29, 30, or 31, depending on the month.
CAL_DAY_NUM_IN_YEAR	smallint	The day number in the calendar year, starting with 1 for January 1 and ending with 365 or 366 for December 31.
CAL_LAST_DAY_IN_WEEK	tinyint	The indicator for the last day of the calendar week: 0 = No, 1 = Yes. For example, this value may be 0 for Wednesday records and 1 for Saturday records.
CAL_LAST_DAY_IN_MONTH	tinyint	The indicator for the last day of the calendar month: 0 = No, 1 = Yes. For example, this value is set to 0 for January 16 and 1 for January 31.
CAL_WEEK_NUM_IN_YEAR	smallint	The week number in the calendar year, starting with 1 and ending with 53. The first week begins on the first day of the calendar year and may contain fewer than seven days. Likewise, the last week, ending with the last day of the year, may contain fewer than seven days.
WEEK_YEAR	smallint	The year number for the week to which this day belongs.
CAL_WEEK_START_DATE	date	The start date of the calendar week to which this date belongs. All dates in the same calendar week share the same calendar week start date. For example, this value is March 6, 2011 for all dates between March 6, 2011 and March 12, 2011.
CAL_WEEK_END_DATE	date	The end date of the calendar week to which this date belongs. All dates in the same calendar week share the same calendar week end date. For example, this value is March 6, 2011 for all dates between March 6, 2011 and March 12, 2011.
CAL_MONTH_NUM_IN_YEAR	smallint	The month number in the calendar year, starting with 1 for January and ending with 12 for December.
CAL_QUARTER_NUM_IN_YEAR	smallint	The number of the quarter in the calendar year, starting with 1 for the first quarter (January 1 through March 31) and ending with 4 for the fourth quarter (October 1 through December 31).
CAL_HALF_NUM_IN_YEAR	smallint	The number of the half of the calendar year, starting with 1 for January 1 through June 30 and ending with 2 for July 1 through December 31.
CAL_YEAR_NUM	smallint	The Gregorian calendar year, expressed as a four-digit integer—for example, 2011.

Table 20: The DATE_TIME Dimension (Continued)

Field	Data Type	Description
CAL_HOUR_NUM_IN_DAY	smallint	The hour of the day, expressed as an integer from 1–12. This field is intended to be used in conjunction with the AMPM_ INDICATOR field.
CAL_HOUR_24_NUM_IN_DAY	smallint	The hour of the day, as an integer from 00 to 23.
CAL_MINUTE_NUM_IN_HOUR	smallint	The 15-minute number of the hour. This value is one of the following: 0: for 0 <= min < 15 15: for 15 <= min < 30 30: for 30 <= min < 45 45: for 45 <= min < 60
CAL_30MINUTE_NUM_IN_HOUR	smallint	The 30-minute number of the hour. This value is one of the following: 0: for 0 <= min < 30 30: for 30 <= min < 60
LABEL_YYYY	varchar(32)	The current date expressed as a string in YYYY format, where YYYY represents a four-digit year. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011".
LABEL_YYYY_QQ	varchar(32)	The current date, expressed as a string in YYYY QQ format, where QQ represents the number of the quarter (1–4), followed by the letter "Q", which is not localizable. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011 1Q".
LABEL_YYYY_MM	varchar(32)	The current date, expressed as a string in YYYY-MM format, where MM represents the two-digit month. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01".
LABEL_YYYY_WE	varchar(32)	The current date, expressed as a string in YYYY-Www format, where Www represents the two-digit week number of the year, preceded by the letter "W". This field is useful when it is used as a label in report headers. For example, with simple week numbering, the label that this field stores for January 30, 2011, at 15:45 is "2011-W05" (January 30, 2011 fell in the fifth week of the year).
LABEL_YYYY_WE_D	varchar(32)	The current date expressed as a string in YYYY-Www-D format, where Www represents the two-digit week number of the year, preceded by the letter "W", and "D" represents the day number in the week. This field is useful when used as a label in report headers. For example, with simple week numbering, the label that this field stores for January 30, 2011, at 15:45 is "2011-05-1" (January 30, 2011 fell in the fifth week of the year, and Sunday is the first day of the week).

Table 20: The DATE_TIME Dimension (Continued)

Field	Data Type	Description
LABEL_YYYY_MM_DD	varchar(32)	The current date, expressed as a string in YYYY-MM-DD format, where DD represents the two-digit day of the month. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01-30".
LABEL_YYYY_MM_DD_HH	varchar(32)	The current date, expressed as a string in YYYY-MM-DD HH format, where hour (HH) values range from 01 to 12. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 3, 2011, at 15:45 is "2011-01-30 03".
LABEL_YYYY_MM_DD_HH24	varchar(32)	The current date, expressed as a string in YYYY-MM-DD HH format where hour (HH) values range from 01 to 24. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01-30 15".
LABEL_YYYY_MM_DD_HH_30MI	varchar(32)	The current date, expressed as a string in YYYY-MM-DD HH:mm format, where hour (HH) values range from 01 to 12 and mm represents the closest 30-minute period that is less than or equal to the actual minute. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01-30 03:30".
LABEL_YYYY_MM_DD_ HH24_30MI	varchar(32)	The current date, expressed as a string in YYYY-MM-DD HH:mm format, where hour (HH) values range from 01 to 24 and mm represents the closest 30-minute period that is less than or equal to the actual minute. This field is useful when it is used as a label in report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01-30 15:30".
LABEL_YYYY_MM_DD_HH_MI	varchar(32)	The current date, expressed as a string in YYYY-MM-DD HH:mm format, where hour (HH) values range from 01 to 12 and mm represents the actual minute. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01-30 03:45".
LABEL_YYYY_MM_DD_HH24_MI	varchar(32)	The current date, expressed as a string in YYYY-MM-DD HH:mm format, where hour (HH) values range from 01 to 24 and mm represents the actual minute. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01-30 15:45".
LABEL_YYYY_MM_DD_ HH_15INT	varchar(32)	The current date, expressed as a string in YYYY-MM-DD 15INT format, where 15INT represents the 15-minute interval within the day. Hour values range from 01 to 12. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "2011-01-30 03:45-04:00".

Table 20: The DATE_TIME Dimension (Continued)

Field	Data Type	Description
LABEL_YYYY_MM_DD_ HH24_15INT	varchar(32)	The current date, expressed as a string in `YYYY-MM-DD 15INT` format, where `15INT` represents the 15-minute interval within the day and includes the hour, in a range from `01` to `24`. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "`2011-01-30 15:45-16:00`".
LABEL_YYYY_MM_DD_ HH_30INT	varchar(32)	The current date, expressed as a string in `YYYY-MM-DD 30INT` format, where `30INT` represents the 30-minute interval within the day and includes the hour, in a range from `01` to `12`. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "`2011-01-30 03:30-04:00`".
LABEL_YYYY_MM_DD_ HH24_30INT	varchar(32)	The current date, expressed as a string in `YYYY-MM-DD 30INT` format, where `30INT` represents the 30-minute interval within the day and includes the hour, in a range from `01` to `24`. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "`2011-01-30 15:30-16:00`".
LABEL_QQ	varchar(32)	A string representation of the current date, expressed in `QQ` format, where `QQ` represents the number of the quarter (1–4), followed by the letter "`Q`", which is not localizable. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "`1Q`".
LABEL_MM	varchar(32)	A string representation of the current date, expressed in `MM` format, where `MM` represents the two-digit month. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "`01`".
LABEL_WE	varchar(32)	A string representation of the current date, expressed in `Www` format, where `Www` represents the two-digit week number of the year, preceded by the letter "`W`". This field is useful when it is used as a label for report headers. For example, with simple week numbering, the label that this field stores for January 30, 2011, at 15:45 is "`W05`". (January 30, 2011 falls in the fifth week of the year.)
LABEL_DD	varchar(32)	A string representation of the current date, expressed in `DD` format, where `DD` represents the two-digit day of the month. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "`30`".
LABEL_HH	varchar(32)	A string representation of the current date, expressed in `HH` format, where hour (`HH`) values range from `01` to `12`. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "`03`".

Table 20: The DATE_TIME Dimension (Continued)

Field	Data Type	Description
LABEL_HH24	varchar(32)	A string representation of the current date, expressed in HH format, where hour (HH) values range from 01 to 24. This field is useful when it is used as a label for report headers. For example, the label that this field stores for January 30, 2011, at 15:45 is "15".
LABEL_30MI	varchar(32)	A string representation of the current date, expressed in mm format, where mm represents the closest 30-minute period that is less than or equal to the actual minute. For example, the label that this field stores for January 30, 2011, at 15:45 is "30".
LABEL_MI	varchar(32)	A string representation of the current date, expressed in mm format, where mm represents the actual minute. For example, the label that this field stores for January 30, 2011, at 15:45 is "45".
LABEL_TZ	varchar(32)	A string representation of the time zone designator, as defined in ISO 8601 standard. For the time zone in which the UTC offset is equal zero, the letter "Z" is stored as the time zone designator. The zone designator for other time zones is specified by the offset from UTC in the format ±HH:<mm>, where HH represents hours and mm represents minutes, if applicable. For example, if the time that is being described is one hour ahead of UTC, the stored value would be "+01".
AMPM_INDICATOR	varchar(4)	Indicates the period between midnight and noon ("AM") or between noon and midnight ("PM").
RUNNING_YEAR_NUM	int	The running year number, starting with 1 for the year that is populated as the first year in this calendar. By default, the calendar starts with the year that precedes the DATE_TIME table initialization. For example, if the iWD Data Mart is initiated in year 2011, this field stores the value of 2 for rows that are generated for 2011 dates.
RUNNING_QUARTER_NUM	int	The running quarter number, starting with 1 as the first quarter of the first year that is populated for this calendar. Running values do not reset at the beginning of each year, so that this value is 1–4, respectively, for the four quarters of the first populated year (for example, 2011); 5–8, respectively, for the four quarters of the second populated year (in this example, 2011); and so forth.
RUNNING_MONTH_NUM	int	The running month number, starting with 1 as the first month of the first year that is populated for this calendar. Running values do not reset at the beginning of each year, so that this value is 1–12, respectively, for the 12 months of the first populated year (for example, 2011); 13–24, respectively, for the 12 months of the second populated year (in this example, 2012); and so forth.

Table 20: The DATE_TIME Dimension (Continued)

Field	Data Type	Description
RUNNING_WEEK_NUM	int	The running week number, starting with 1 as the first week of the first year that is populated for this calendar. Running values do not reset at the beginning of each year, so that, this value is 1–53, respectively, for the 53 weeks of the first populated year (for example, 2011); 54–107, respectively, for the 53 weeks of the second populated year (in this example, 2012); and so forth.
RUNNING_DAY_NUM	int	The running day number, starting with 1 as the first day of the first year that is populated for this calendar. Running values do not reset at the beginning of each year, so that this value is 1–365, respectively, for the 365 days of the first populated year (for example, 2011); 366–730, respectively, for the 365 days of the second populated year (in this example, 2012); and so forth.
RUNNING_HOUR_NUM	int	The running hour number, starting with 1 as the first hour of the first day of the first year that is populated for this calendar. Running hours do not reset at the beginning of each day, so that this value is 1–24, respectively, for the 24 hours of the first populated day (for example, 1/1/2011); 25–48, respectively, for the 24 hours of the second populated day (in this example, 1/2/2011); and so forth.
RUNNING_30MIN_NUM	int	The running 30-minute number, starting with 1 as the first 30-minute interval of the first hour of the first day of the first year that is populated for this calendar. Running 30-minute periods do not reset at the beginning of each hour, so that this value is 1–2, respectively, for the two 30-minute intervals of the first hour of 1/1/2011, if 2011 is the first year populated for this calendar; 3–4, respectively, for the two 30-minute intervals in the second hour of this day; and so forth.

DEPARTMENT Dimension

All tasks are associated with a department that is configured in iWD Manager. The DEPARTMENT dimension contains the fields that are described in Table 21.

Table 21: The DEPARTMENT Dimension

Field	Data Type	Description
DEPARTMENT_KEY	int	Primary key of this table.
DEPARTMENT_CONFIG_ID	int	iWD Manager ID for the department.
DEPARTMENT_CONFIG_EVENT_ID	int	ID of the event that created or updated this department record.
DEPARTMENT_RUNTIME_ID	varchar(16)	iWD Manager runtime ID for the department.
TENANT_KEY	int	Key to the TENANT dimension, identifying the tenant with whom the process is associated.
SOLUTION_KEY	int	Key to the SOLUTION dimension, identifying the solution with which the process is associated.

Table 21: The DEPARTMENT Dimension (Continued)

Field	Data Type	Description
DEPARTMENT_NAME	varchar(255)	Name of the department.
CUSTOM_DIM_KEY	int	Key to the CUSTOM_DIM dimension, identifying custom attributes that apply to the department.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.
VALID_FROM_DATE_KEY	int	Date key for the “valid from” process attribute.
VALID_FROM_TIME_KEY	int	Time key for the “valid from” process attribute.
VALID_TO_DATE_KEY	int	Date key for the “valid to” process attribute.
VALID_TO_TIME_KEY	int	Time key for the “valid to” process attribute.
VALID_FROM	datetime	For the version of the record, the date from which it is valid.
VALID_TO	datetime	For the version of the record, the date to which it is valid.
VERSION	int	Version of the record.

DISTRIBUTION_POINT Dimension

In iWD releases prior to 8.0, this dimension stored key information about distribution points, which were configured in iWD Manager and were the objects responsible for sending captured tasks on for completion. Beginning with release 8.0, this table is provided only for backward compatibility. The DISTRIBUTION_POINT dimension contains the fields listed in Table 22.

Table 22: The DISTRIBUTION_POINT Table

Field	Data Type	Description
DISTRIBUTION_POINT_KEY	int	The primary key of this table.
DISTRIBUTION_POINT_CONFIG_ID	int	iWD Manager ID for the distribution point.
DISTRIBUTION_POINT_CFG_EVT_ID	int	Event that created/updated the distribution point record.
DISTRIBUTION_POINT_RUNTIME_ID	varchar(16)	iWD Manager runtime ID for the distribution point.
TENANT_KEY	int	Key to the TENANT dimension, identifying the tenant with whom the distribution point is associated.
DISTRIBUTION_POINT_NAME	varchar(255)	Descriptive name of the distribution point.
DISTRIBUTION_POINT_TYPE	varchar(255)	Type of distribution point.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension to identify the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension to identify the ETL job that last updated this record.

Table 22: The DISTRIBUTION_POINT Table (Continued)

Field	Data Type	Description
VALID_FROM	datetime	For the version of the record, the date that it is valid from
VALID_TO	datetime	For the version of the record, the date that it is valid to
VERSION	int	Version of the record

EVENT_DATE Dimension

The `EVENT_DATE` dimension is a static dimension that stores information about dates. The ETL initialization script populates this table determining date values from your system's locale settings. Table 23 describes its fields.

Table 23: The EVENT_DATE Dimension

Field	Data Type	Description
EVENT_DATE_KEY	int	Primary key of this table.
EVENT_DATE	datetime	Date of the event in "`YYYY-M-D 0:0:0.0`" format.
DAY_NAME	varchar(32)	Text name of the day ("`Monday`", "`Tuesday`", and so on).
DAY_NUM_IN_WEEK	int	Day of the week (`1`–`7`).
DAY_NUM_IN_MONTH	int	Day of the month (`1`–`31`).
DAY_NUM_IN_YEAR	int	Day of the year (`1`–`366`).
WEEK_NUM_IN_YEAR	int	Week number of the year (`1`–`53`).
WEEK_START_DATE	datetime	Date of the first day of the week in "`YYYY-M-D 0:0:0. 0`" format.
WEEK_END_DATE	datetime	Date of the last day of the week in "`YYYY-M-D 0:0:0. 0`" format.
MONTH_NAME	varchar(32)	Text name of the month—for example, "`January`".
MONTH_NUM_IN_YEAR	int	Month of the year (`1`–`12`).
QUARTER_NUM_IN_YEAR	int	Quarter (`1`–`4`).
YEAR_NUM	int	Year—for example, `2011`.
EVENT_DATE_STR	varchar(10)	

EVENT_TIME Dimension

The `EVENT_TIME` dimension, like the `EVENT_DATE` dimension, is a static that stores time information with minute-level granularity. Table 24 describes its fields.

Table 24: The EVENT_TIME Dimension

Field	Data Type	Description
EVENT_TIME_KEY	int	Primary key of this table.
EVENT_TIME_24	varchar(6)	24-hour time, stored in `HHMM` format—for example, "`2301`" (for `11:01 PM`).

Table 24: The EVENT_TIME Dimension (Continued)

Field	Data Type	Description
EVENT_TIME_12	varchar(8)	12-hour time, including AM or PM—for example, “1101PM”.
TIME_INTERVAL_15MIN	int	The 15-minute interval in which the time falls (1–96).
TIME_INTERVAL_30MIN	int	The 30-minute interval in which the time falls (1–48).
TIME_INTERVAL_60MIN	int	The 60-minute interval in which the time falls (1–24).

MEDIA_CHANNEL Dimension

The MEDIA_CHANNEL dimension describes the type of media or media channel—for example, webform, fax— through which a task is received. This table is populated from task information and is a core iWD attribute. Table 25 describes the fields of this dimension.

Table 25: The MEDIA_CHANNEL Dimension

Field	Data Type	Description
MEDIA_CHANNEL_KEY	int	Primary key of this table.
MEDIA_CHANNEL_NAME	varchar(255)	Name of the media channel.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

METRIC Dimension

Metrics are values that can be associated with a department or process. Metrics are set in iWD and are used to associate a value for evaluation against various attributes for reporting purposes (that is, “Cost per Task” set to $0.50). The table name is METRIC; it contains the following fields:

Table 26: The METRIC Dimension

Field	Data Type	Description
METRIC_KEY	int	Primary key of this table.
METRIC_CONFIG_ID	int	iWD Manager ID for the metric.
METRIC_CONFIG_EVENT_ID	int	ID of the event that created or updated the metric record.
METRIC_NAME	varchar(255)	Name of the metric as set in iWD.
METRIC_DESCRIPTION	varchar(255)	Description of what the metric represents as set in iWD.
METRIC_VALUE	varchar(255)	Metric value. It is incumbent on the report writer to perform any type conversion on this field.
TENANT_KEY	int	Key to the TENANT dimension.
SOLUTION_KEY	int	Key to the SOLUTION dimension.

Table 26: The METRIC Dimension (Continued)

Field	Data Type	Description
DEPARTMENT_KEY	int	Key to the DEPARTMENT dimension.
PROCESS_KEY	int	Key to the PROCESS dimension.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.
VALID_FROM	datetime	The date, in YYYY-M-D HH:MM:SS F format, from which this metric is valid.
VALID_TO	datetime	The date, in YYYY-M-D HH:MM:SS F format, until which this metric is valid.
VERSION	int	Version of the record.

PRIORITY Dimension

iWD arranges a task list in order of priority based on business rules that are configured for capture points, departments, and processes within departments. This prioritization is stored in the PRIORITY dimension, which is a static dimension. Table 27 describes the fields of this dimension.

Table 27: The PRIORITY Dimension

Field	Data Type	Description
PRIORITY_KEY	int	Primary key of this table.
PRIORITY_RANGE_5	varchar(32)	Values in the priority granularity of 5—that is “1–5”, “6–10”, and so on.
PRIORITY_RANGE_10	varchar(32)	Values in the priority granularity of 10—that is “1–10”, “11–20”, and so on.
PRIORITY_RANGE_50	varchar(32)	Values in the priority granularity of 50—that is “1–50”, “51–100”, and so on.
PRIORITY_RANGE_100	varchar(32)	Values in the priority granularity of 100—that is “1–100”, “101–200”, and so on.
PRIORITY_RANGE_500	varchar(32)	Values in the priority granularity of 500—that is “1–500”, “501–1000”, and so on.
PRIORITY_RANGE_1000	varchar(32)	Values in the priority granularity of 1000—that is “1–1000”, “1001–2000”, and so on, with a maximum value of 50000.
PRIORITY_RANGE_5_START	int	Values that mark the start of each PRIORITY_RANGE_5 range. Values step by 5—for example, 1, 6, 11, and so forth.
PRIORITY_RANGE_5_END	int	Values that mark the end of each PRIORITY_RANGE_5 range. Values step by 5—for example, 5, 10, 15, and so forth.
PRIORITY_RANGE_10_START	int	Values that mark the start of each PRIORITY_RANGE_10 range. Values step by 10—for example, 1, 11, 21, and so forth.

Table 27: The PRIORITY Dimension (Continued)

Field	Data Type	Description
PRIORITY_RANGE_10_END	int	Values that mark the end of each PRIORITY_RANGE_10 range. Values step by 10—for example, 10, 20, 30, and so forth.
PRIORITY_RANGE_50_START	int	Values that mark the start of each PRIORITY_RANGE_50 range. Values step by 50—for example, 1, 51, 101, and so forth.
PRIORITY_RANGE_50_END	int	Values that mark the end of each PRIORITY_RANGE_50 range. Values step by 50—for example, 50, 100, 150, and so forth.
PRIORITY_RANGE_100_START	int	Values that mark the start of each PRIORITY_RANGE_100 range. Values step by 100—for example, 1, 101, 201, and so forth.
PRIORITY_RANGE_100_END	int	Values that mark the end of each PRIORITY_RANGE_100 range. Values step by 100—for example, 100, 200, 300, and so forth.
PRIORITY_RANGE_500_START	int	Values that mark the start of each PRIORITY_RANGE_500 range. Values step by 500—for example, 1, 501, 1001, and so forth.
PRIORITY_RANGE_500_END	int	Values that mark the end of each PRIORITY_RANGE_500 range. Values step by 500—for example, 500, 1000, 1500, and so forth.
PRIORITY_RANGE_1000_START	int	Values that mark the start of each PRIORITY_RANGE_1000 range. Values step by 1000—for example, 1, 1001, 2001, and so forth.
PRIORITY_RANGE_1000_END	int	Values that mark the end of each PRIORITY_RANGE_1000 range. Values step by 1000—for example, 1000, 2000, 3000, and so forth.

PROCESS Dimension

All tasks are associated with a business process. *Processes* are associated with a department in iWD. Populated from iWD Manager configurations, the PROCESS dimension contains information about iWD processes. Table 28 describes this table.

Table 28: The PROCESS Dimension

Field	Data Type	Description
PROCESS_KEY	int	Primary key of this table.
PROCESS_CONFIG_ID	int	iWD Manager ID for the process.
PROCESS_CONFIG_EVENT_ID	int	Event that created or updated the process record.
PROCESS_RUNTIME_ID	varchar(16)	iWD Manager runtime ID for the process.
TENANT_KEY	int	Key to the TENANT dimension, identifying the tenant with whom the process is associated.

Table 28: The PROCESS Dimension (Continued)

Field	Data Type	Description
SOLUTION_KEY	int	Key to the SOLUTION dimension, identifying the solution with which the process is associated.
DEPARTMENT_KEY	int	Key to the DEPARTMENT dimension, identifying the department with which the process is associated.
PROCESS_NAME	varchar(255)	Descriptive name of the process.
CUSTOM_DIM_KEY	int	Key to the CUSTOM_DIM dimension, identifying custom attributes that apply to the process.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.
VALID_FROM_DATE_KEY	int	Date key for the VALID_FROM process attribute.
VALID_FROM_TIME_KEY	int	Time key for the VALID_FROM process attribute.
VALID_TO_DATE_KEY	int	Date key for the VALID_TO process attribute.
VALID_TO_TIME_KEY	int	Time key for the VALID_TO process attribute.
VALID_FROM	datetime	The date, in YYYY-M-D HH:MM:SS F format, from which this process is valid.
VALID_TO	datetime	The date, in YYYY-M-D HH:MM:SS F format, until which this process is valid.
VERSION	int	Version of the record.

PRODUCT Dimension

In some cases, a task may be associated with a specific product—for example, an order for a widget—in which the widget is the product. The PRODUCT dimension includes fields to capture the type and subtype—for example, North American widget versus European widget—of a product. This dimension is an extended attribute that can be set by the source system. Table 29 describes the fields of this dimension.

Table 29: The PRODUCT Dimension

Field	Data Type	Description
PRODUCT_KEY	int	Primary key of this table.
PRODUCT_TYPE	varchar(64)	Type of the product.
PRODUCT_SUBTYPE	varchar(64)	Subtype of the product.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

QUEUE Dimension

The QUEUE dimension describes queues and workbins in the Genesys Interaction Server. This dimension contains the following fields:

Table 30: The QUEUE Dimension

Field	Data Type	Description
QUEUE_KEY	int	Primary key of this table.
QUEUE_TYPE	varchar(255)	Type of the distribution queue; one of the following: • InteractionQueue • AgentWorkbin • AgentGroupWorkbin • PlaceWorkbin • PlaceGroupWorkbin
QUEUE_NAME	varchar(255)	Descriptive name of the Interaction queue or workbin.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

QUEUE_TARGET Dimension

A *queue target* represents the destination object where a task was distributed. The QUEUE_TARGET dimension stores this information, and Table 31 describes its fields.

Table 31: The QUEUE_TARGET Dimension

Field	Data Type	Description
QUEUE_TARGET_KEY	int	Primary key of this table.
QUEUE_TARGET_TYPE	varchar(255)	Type of the queue; one of the following: • InteractionQueue • AgentWorkbin • AgentGroupWorkbin • PlaceWorkbin • PlaceGroupWorkbin
QUEUE_TARGET_NAME	varchar(255)	Workbin name of the agent, agent group, place, or place group or the name of the interaction queue.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

RESULT_CODE Dimension

A *result code* is a system- or agent-assigned value that is set upon the completion of a task. Result codes are often included in reports to enable further understanding of the disposition of a task—for example, the outcome of a routing strategy, a result code from the source system, or the invocation/termination of after-call work. The RESULT_CODE dimension is populated from the iWD extended attribute, resultCode. Table 32 describes this dimension's fields.

Table 32: The RESULT_CODE Dimension

Field	Data Type	Description
RESULT_CODE_KEY	int	Primary key of this table.
RESULT_CODE_NAME	varchar(64)	Descriptive name of the result code.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

SKILL Dimension

The SKILL dimension describes only those agent skills existing within Configuration Server that were requested by iWD rules.

Table 33: The SKILL Dimension

Field	Data Type	Description
SKILL_KEY	int	Primary key of this table.
SKILL_ID	varchar(255)	ID of the skill.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

SOLUTION Dimension

Each tenant in iWD can have one or more solutions. A solution can be configured in iWD for testing a new iWD configuration, independent of a production solution. Solution information is stored in the SOLUTION dimension which is populated from iWD Manager configuration. Many fact tables in the iWD Data Mart include a SOLUTION_KEY column to join to this dimension.

Table 34: The SOLUTION Dimension

Field	Data Type	Description
SOLUTION_KEY	int	Primary key of this table.
SOLUTION_CONFIG_ID	int	iWD Manager ID for the solution.
SOLUTION_CONFIG_EVENT_ID	int	Event that created or updated the solution record.
SOLUTION_RUNTIME_ID	varchar(16)	iWD Manager runtime ID for the solution.
TENANT_KEY	int	Key to the TENANT dimension, identifying the tenant with whom the solution is associated
SOLUTION_NAME	varchar(255)	Descriptive name of the solution.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.
VALID_FROM	datetime	The date, in YYYY-M-D HH:MM:SS F format, from which this solution is valid.
VALID_TO	datetime	The date, in YYYY-M-D HH:MM:SS F format, until which this solution is valid.
VERSION	int	Version of the record.

SOURCE_PROCESS Dimension

Tasks may be associated with a larger process—say, in a workflow system. As an extended attribute, the SOURCE_PROCESS dimension includes a process type and subtype and can be set by the source system. Table 35 describes the fields of this dimension.

Table 35: The SOURCE_PROCESS Dimension

Field	Data Type	Description
SOURCE_PROCESS_KEY	int	Primary key of this table.
SOURCE_PROCESS_TYPE	varchar(64)	Name of the source-system process—for example, Order.
SOURCE_PROCESS_SUBTYPE	varchar(64)	Subtype of the process—for example, Activation.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

SOURCE_TENANT Dimension

In cases in which the source system is part of a multi-tenant system, the source_process and source_product values might not be unique across tenants. The SOURCE_TENANT dimension provides the ability to define the source of the task further. As an extended attribute, SOURCE_TENANT can be populated by the source system that is submitting the task. The table name is SOURCE_TENANT; it contains the following fields:

Table 36: The SOURCE_TENANT Dimension

Field	Data Type	Description
SOURCE_TENANT_KEY	int	Primary key of this table.
SOURCE_TENANT_NAME	varchar(64)	Name of the tenant from the source system.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that last updated this record.

STATUS Dimension

The STATUS dimension is a static dimension that stores the status of an iWD task. Table 37 shows the structure of this dimension.

Table 37: The STATUS Dimension

Field	Data Type	Description
STATUS_KEY	int	Primary key of this table.
STATUS_NAME	varchar(16)	Name of the status of a task. One of the following values: • new—Newly created task awaiting processing. • rejected—Task was rejected during processing. This can occur when a task is assigned to an expired process or closed department. • newheld[a] • captured—Task has been classified by iWD, but not yet prioritized. • queued—Task has been processed and prioritized at least once. • distributed[a] • canceled—Task has been canceled. • completed—Task has been completed. • errorheld—Error occurred during task classification or prioritization. Error details are stored in the “error” custom extended task attribute. When resumed, iWD attempts to process the task again. • held—Task is in a held state (either by user action or the system) and will not be reprioritized until the task is resumed. • assigned—Task has been assigned to an agent.
IS_FINAL	int	Indicates task finality: • 0 indicates a task status other than Completed, Canceled, or Rejected. • 1 indicates a task status of Completed, Canceled, or Rejected.
IS_HELD	int	Indicates whether a task was held: • 0 indicates a task status other than NewHeld, ErrorHeld, or Held. • 1 indicates a task status of NewHeld, ErrorHeld, or Held.

a. This value is retained only for backward compatibility. iWD 8.0 does not generate this value.

TASK_EVENT_TYPE Dimension

The TASK_EVENT_TYPE dimension is a static dimension that stores various iWD task event types. Table 38 describes its fields.

Table 38: The TASK_EVENT_TYPE Dimension

Field	Data Type	Description
TASK_EVENT_TYPE_KEY	int	Primary key of this table
TASK_EVENT_TYPE	varchar(64)	iWD task event type name. One of the following: UPDATE_NEW CLASSIFY NEW RESTART RESUME_NEW HOLD_NEW ERROR_NEW ERROR REJECT CANCEL_NEW CANCEL UPDATE PRIORITIZE REPRIORITIZE RESUME DISTRIBUTE HOLD CANCEL_HELD COMPLETE ASSIGN FINISH FINISH_RETURN CLASSIFY_START PRIORITIZE_START QUEUE REPRIORITIZE_START RULE_APPLIED SYNC_COMPLETE RESTART_HELD DISTRIBUTE_QUEUE STOPPED UPDATE_ERROR DISTRIBUTE_WORKBIN UPDATE_COMPLETED

TENANT Dimension

The TENANT dimension describes the iWD tenant, and the values are populated from the values set up in iWD Manager. The tenant forms part of the ownership chain for a task. Table 39 describes the fields of the TENANT table.

Table 39: The TENANT Dimension

Field	Data Type	Description
TENANT_KEY	int	Primary key of this table.
TENANT_CONFIG_ID	int	iWD Manager ID for the tenant
TENANT_CONFIG_EVENT_ID	int	Event that created or updated the tenant record.
TENANT_RUNTIME_ID	varchar(255)	iWD Manager runtime ID for the tenant
TENANT_NAME	varchar(255)	Descriptive name of the tenant
CUSTOM_DIM_KEY	int	Key to the CUSTOM_DIM dimension, identifying custom attributes of the tenant.
CREATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that created this record.
UPDATED_ETL_AUDIT_KEY	int	Key to the ETL_AUDIT dimension, identifying the ETL job that updated this record.
VALID_FROM	datetime	The date, in YYYY-M-D HH:MM:SS F format, from which this tenant is valid.

Table 39: The TENANT Dimension (Continued)

Field	Data Type	Description
VALID_TO	datetime	The date, in YYYY-M-D HH:MM:SS F format, until which this tenant is valid.
VERSION	int	Version of the record.

TIMEZONE Dimension

The TIMEZONE dimension is populated during iWD Data Mart initialization. It stores the name and ID of nearly 600 time zones. Table 40 describes the fields of this table.

Table 40: The TIMEZONE Dimension

Field	Data Type	Description
TIMEZONE_KEY	int	Primary key of this table
TIMEZONE_ID	varchar(48)	The iWD-assigned ID for this time zone. This assignment coincides with Java time zone IDs.
TIMEZONE_NAME	varchar(64)	The name of the time zone.

iWD Data Mart System Tables

iWD Data Mart includes two system tables:

- ETL_AUDIT, described on page 77
- ETL_CUSTOM_MAP, described on page 78

ETL_AUDIT

The ETL_AUDIT table records information about every iWD Data Mart ETL job that has been executed. Various ETL jobs use the information stored in this table to determine when task processing ended from the last run of the job and thus, where to resume processing. Table 41 describes the fields of this table.

Table 41: The ETL_AUDIT Table

Field	Data Type	Description
ETL_AUDIT_KEY	int	Primary key of this table.
ETL_AUDIT_TIME	datetime	Date and time when the job finished running.
BATCH_ID	int	Batch number, for jobs that process data in batches (such as GTL_DM_load_intraday). Each batch job has its own ETL audit record.

Table 41: The ETL_AUDIT Table (Continued)

Field	Data Type	Description
DATA_SOURCE_TYPE	varchar(16)	Primary source of the job data; either config, task, or datamart. (Data from the Interaction Server Events Log database is classified as task data source type.)
DATA_SOURCE_NAME	varchar(255)	Database ID (equals database JDBC URL).
PROCESS_NAME	varchar(255)	The name of the job that is processing the records (such as, load_config).
FIRST_EXTRACTED_EVENT_ID	int	ID that starts the range of event IDs that are processed
LAST_EXTRACTED_EVENT_ID	int	ID that ends the range of event IDs that are processed.
BATCH_LAST_EVENT_ID	int	ID of the last event in the batch. Used to set the starting point for the next batch.
LAST_INTERVAL_DATE_KEY	int	Last date interval that is processed by the aggregate ETL scripts.
LAST_INTERVAL_TIME_KEY	int	Last time interval that is processed by the aggregate ETL scripts.

ETL_CUSTOM_MAP

The ETL_CUSTOM_MAP table is used to configure mapping of task custom attributes to attributes in the I_/H_TASK_FACT and I_/H_TASK_WORK_FACT tables, as well as the CUSTOM_DIM dimension. Table 42 describes it fields.

Table 42: The ETL_CUSTOM_MAP Table

Field	Data Type	Description
ETL_CUSTOM_MAP_KEY	int	Unique mapping record key.
SOLUTION_KEY	int	Key to the SOLUTION dimension identifying the solution for which this mapping is valid. If NULL, the custom mapping applies to all solutions.
PROCESS_KEY	int	For which process this mapping is valid. If null applies to all processes.
DEPARTMENT_KEY	int	Key to the DEPARTMENT dimension.
TYPE	varchar(30)	Indicates where this custom attribute has been configured. One of the following: • taskattribute • taskdimension • tenant • department • process
KEY_NAME	varchar(255)	Custom attribute key name in the Interaction Server database.
IS_DIMENSION	int	Indicates whether this attribute should be mapped to the CUSTOM_DIM dimension instead of to a fact table. • 0 for taskattribute types • 1 for other than taskattribute types
CUSTOM_ATTRIBUTE_INDEX	int	Custom attribute index. If IS_DIMENSION is false (0), 1–10; otherwise 1–5.

iWD Data Mart Views

iWD Data Mart provides a number of read-only views of iWD Data Mart tables, including views of:

- Dimensions
- Intraday fact tables
- Historical fact tables
- Blended fact tables (union of intraday and historical facts)
- Intraday aggregates (hour level)
- Historical aggregates (hour, week, month, quarter, year levels)
- Blended aggregate (union of intraday and historical hour aggregates)

Business users and reporting analysts who create reports by using iWD Data Mart data are isolated by the views in iWD Data Mart from changes that might occur in the underlying iWD Data Mart tables.

The views listed in Table 43 are created when the iWD Data Mart database is initialized:

Table 43: iWD Data Mart Views

View Name	Based on ...
DATE_INTERVAL_WEEK	EVENT_DATE
DATE_INTERVAL_MONTH	EVENT_DATE
DATE_INTERVAL_QUARTER	EVENT_DATE
DATE_INTERVAL_YEAR	EVENT_DATE
TIME_INTERVAL_15MIN	EVENT_TIME
TIME_INTERVAL_30MIN	TIME_INTERVAL_15MIN
TIME_INTERVAL_60MIN	TIME_INTERVAL_15MIN
PROCESS_CURRENT_VERSION	PROCESS, DEPARTMENT, SOLUTION, TENANT
PROCESS_CURRENT	PROCESS, DEPARTMENT, SOLUTION, TENANT, PROCESS_CURRENT_VERSION
TASK_EVENT_FACT	H_TASK_EVENT_FACT and I_TASK_EVENT_FACT
TASK_FACT	H_TASK_FACT and I_TASK_FACT
TASK_WORK_FACT	H_TASK_WORK_FACT and I_TASK_WORK_FACT

Refer also to Table 6 on page 41 for the aggregate views that are created for each subject area.

Chapter

3 ETL Overview

This chapter explains the process by which information is extracted from the Interaction Server database, the Interaction Server Event Log database, and the iWD configuration database; and how this data is transformed and loaded into the various iWD Data Mart star schemas.

The information in this chapter is organized into the following sections:

What Is ETL?

The data in iWD Data Mart is made available through a process that is called Extract, Transform and Load—or ETL, for short. The system that is used to create, configure, and execute the ETL process is Kettle, which is part of the Pentaho reporting suite. Kettle top-level objects are known as jobs. *Jobs* are a sequence of steps that are executed according to success/failure criteria. One of the steps that is used by iWD reporting is to transform steps.

iWD Data Mart ETL Jobs

Table 44 describes the preconfigured ETL jobs that are responsible for creating and populating the iWD Data Mart and for calculating the various aggregates and measures that are described in Chapter 2 of this reference guide. Jobs names are those displayed in iWD Manager.

Table 44: Preconfigured ETL Jobs

Job Function	Attribute	Description
Initialize iWD Data	Job Name	Initialize
	Function	Initializes the necessary data structures and populates static dimensions, such as the `DATE_TIME` and `AGE` dimension.
	Schedule	Runs once.
Load Configuration	Job Name	Load Configuration
	Function	Loads updates from iWD configuration tables into dimension tables.
	Schedule	Configurable through service properties; typically, runs on a 15-minute cycle, but not more frequently than a 15-minute cycle.
Load Intraday	Job Name	Load Intraday
	Function	Loads updates from tables in the Interaction Server database and event log into the core fact tables.
	Schedule	Configurable through Service Properties in iWD Manager; recommended that it be scheduled to run after the Load Configuration job ends through the Job Dependency scheduling option.
Aggregate Intraday	Job Name	Aggregate Intraday
	Function	Aggregates data that previously was loaded into fact tables by the Load Intraday job into the aggregation tables.
	Schedule	It is recommended that this job be scheduled immediately after the Load Intraday job has completed—typically, running every 15 minutes. The frequency of running this aggregate job does not have any bearing on the current 15-minute aggregate that is being populated.
Aggregate Statistics	Job Name	Aggregate Statistics
	Function	Generate extended statistics by executing statistics plug-ins.
	Schedule	It is recommended that this job be scheduled immediately following completion of the Aggregate Intraday job, because most of the statistics plug-ins user aggregated facts.

Table 44: Preconfigured ETL Jobs (Continued)

Job Function	Attribute	Description
Load Historical	Job Name	Load Historical
	Function	Moves noncurrent data from the intraday fact tables to their corresponding historical fact tables. (“Noncurrent” refers to other than today’s data [CREATED_DATE_KEY < today].) **Note:** Tasks must also be finalized (having reached Completed, Canceled, or Rejected state) before they can be moved from the intraday fact tables.
	Schedule	Runs daily through the schedule that is defined in Service Properties.
Aggregate Historical	Job Name	Aggregate Historical
	Function	Aggregates data from historical fact tables into 15-minute, hourly, daily, and monthly aggregation tables.
	Schedule	Runs once a day, after the Load Historical job.
Maintain iWD	Job Name	Maintain
	Function	Cleanses historical tables—that is, removes expired facts from these tables, based on rules that are defined on the ETL Service property in iWD Manager. This job also adds to DATE_TIME ensuring that next-year values are present in this table.
	Schedule	Runs once a day, after the Aggregate Historical job.
Prune events and interactions	Job Name	Prune
	Function	Removes event logs and interactions from the databases that are loaded in the Data Mart when tasks/interactions have reached their final state.
	Schedule	Runs once a day, after the Maintain job.

Scheduling ETL Jobs

You can schedule ETL jobs to run:

- On a recurring basis by using a cron expression.
- Manually.
- Upon the successful completion of a dependent service.

For more information on the configuration of scheduling ETL jobs, please refer to the *intelligent Workload Distribution 8.0 Deployment Guide*.

Chapter

4 Customizing iWD

You can customize iWD both to create new statistics and to aggregate them. This chapter lists the general guidelines for how to accomplish this and provides one specific example.

This chapter contains the following sections:

Composition of Genesys-Provided Statistics and Aggregates

Each statistic is represented by a transformation script in `aggregate_stats\stats` subfolder of the iWD ETL package. Each aggregate is represented by a transformation script in the `plugins` subfolder. Either can define more than one statistic. As deployed during installation, the `aggregate_stats\stats` subfolder contains the following five sample scripts that define five statistics:

- `department_activeheld.ktr`
- `department_newcompleted.ktr`
- `process_activeheld.ktr`
- `process_pendingoverdue.ktr`
- `solution_newcompleted.ktr`

And, the `\plugins` directory provides a `plugins.properties` file and four aggregate subfolders corresponding to the four aggregates described in this document (beginning on page 42):

- `\agent`
- `\capt`
- `\classif`
- `\queue`

Guidelines for Creating New Statistics and Aggregates

Customizing Statistics

The recommended approach for creating custom statistics is the following:

1. Copy an existing sample transformation script from the `aggregate_stats\stats` subdirectory and name it appropriately.
2. Modify the script as appropriate using Kettle ETL Designer.
3. Enable the new statistic by adding it to the `aggregate_stats\stats.properties` file.

Customizing Aggregates

The recommended approach for creating custom aggregates is the following:

1. Create a new subdirectory in the `\plugins` directory that is copied from an existing aggregate. Name this subdirectory appropriately.
2. Modify the aggregate transformations as appropriate using Kettle ETL Designer.
3. Enable the new aggregate by adding it to the `plugins\plugins.properties` file.

Example

This example creates a new statistic transformation script, `product_pendingoverdue.ktr,` that calculates how many pending and overdue tasks there are for a particular product. We start by copying the `process_pendingoverdue` transformation script, which calculates similar statistics for the `PROCESS` dimension. This script references the `TASK_CLASSIF_FACT` subject area (see page 47) which supports a join to the `PRODUCT` dimension.

1. In the `aggregate_stats\stats` subdirectory, copy the `process_pendingoverdue.ktr` transformation script and rename it:

   ```
   product_pendingoverdue.ktr
   ```

2. Open this script in Kettle ETL Designer.

3. Within the Transformation window, right-click and select `Transformation Settings` from the context menu (shown in Figure 15) to open the `Transformation properties` dialog box.

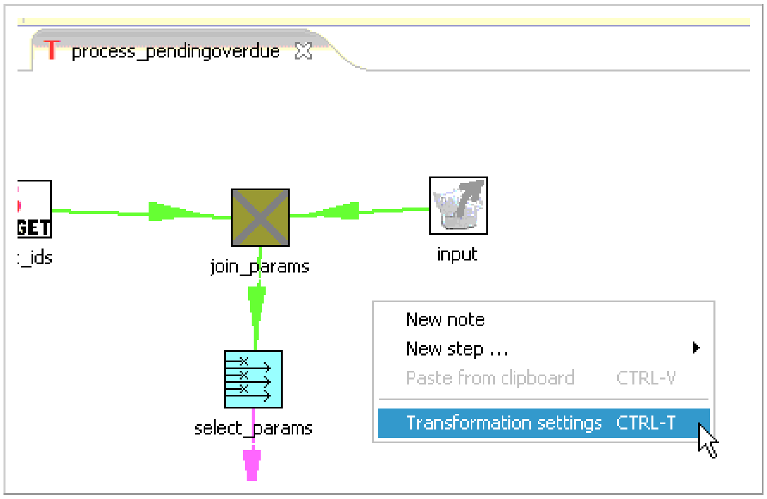
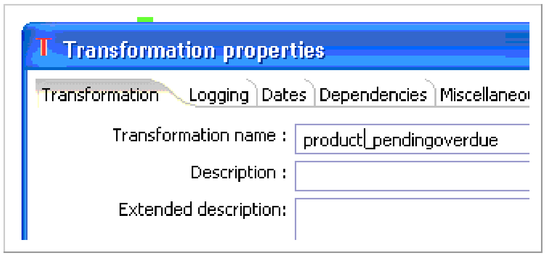


Figure 15: Setting Transformation Properties

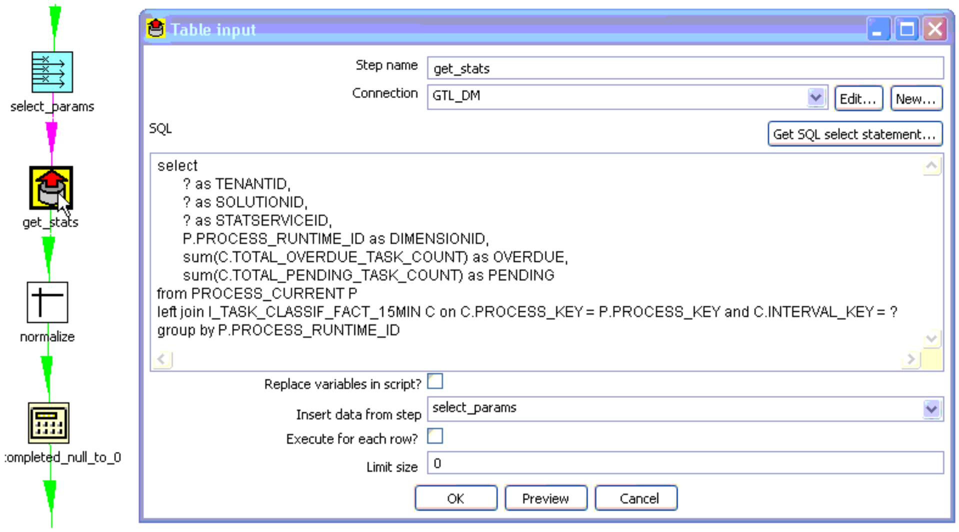
4. Rename the transformation appropriately and click `OK`. Figure 16 uses the name `product_pendingoverdue`.



Transformation name : product_pendingoverdue
Description :
Extended description:

Figure 16: Renaming the Transformation

5. Double-click the `get_stats` step to open the `Table input` dialog box shown in Figure 17. We must next update the logic for calculating this statistic.



Step name: get_stats

Connection: GTL_DM [Edit...] [New...]

SQL [Get SQL select statement...]

```
select
     ? as TENANTID,
     ? as SOLUTIONID,
     ? as STATSERVICEID,
     P.PROCESS_RUNTIME_ID as DIMENSIONID,
     sum(C.TOTAL_OVERDUE_TASK_COUNT) as OVERDUE,
     sum(C.TOTAL_PENDING_TASK_COUNT) as PENDING
from PROCESS_CURRENT P
left join I_TASK_CLASSIF_FACT_15MIN C on C.PROCESS_KEY = P.PROCESS_KEY and C.INTERVAL_KEY = ?
group by P.PROCESS_RUNTIME_ID
```

Replace variables in script? □

Insert data from step: select_params

Execute for each row? □

Limit size: 0

[OK] [Preview] [Cancel]

Figure 17: Modifying the Statistic's Definition

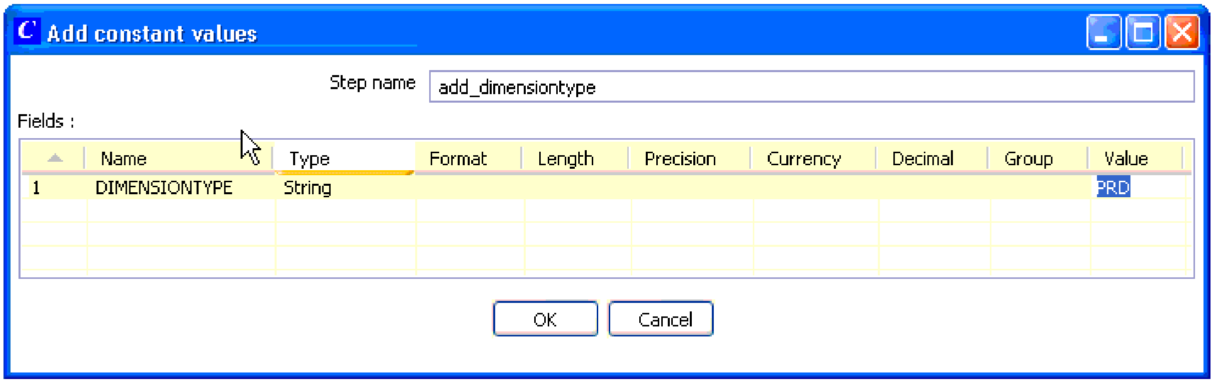
6. Replace the SQL statement with the following and click `OK`:

```
SELECT   ?                               AS TENANTID     ,
         ?                               AS SOLUTIONID   ,
         ?                               AS STATSERVICEID,
         P.PRODUCT_TYPE                  AS DIMENSIONID  ,
         SUM(C.TOTAL_OVERDUE_TASK_COUNT) AS OVERDUE      ,
         SUM(C.TOTAL_PENDING_TASK_COUNT) AS PENDING
FROM     PRODUCT P
         LEFT JOIN I_TASK_CLASSIF_FACT_15MIN C
         ON       C.PRODUCT_KEY  = P.PRODUCT_KEY
         AND      C.INTERVAL_KEY = ?
GROUP BY P.PRODUCT_TYPE
```

We use a left join on `I_TASK_CLASSIF_FACT_15MIN` instead of an inner join because we want to retrieve data about all products—whether or not they have tasks associated with them.

7. Double-click the `add_dimensiontype` step to open the `Add constant values` dialog box.

8. Set the value of `DIMENSIONTYPE` to some string and click `OK`. Figure 18 sets this string to `PRD`.



Step name: add_dimensiontype

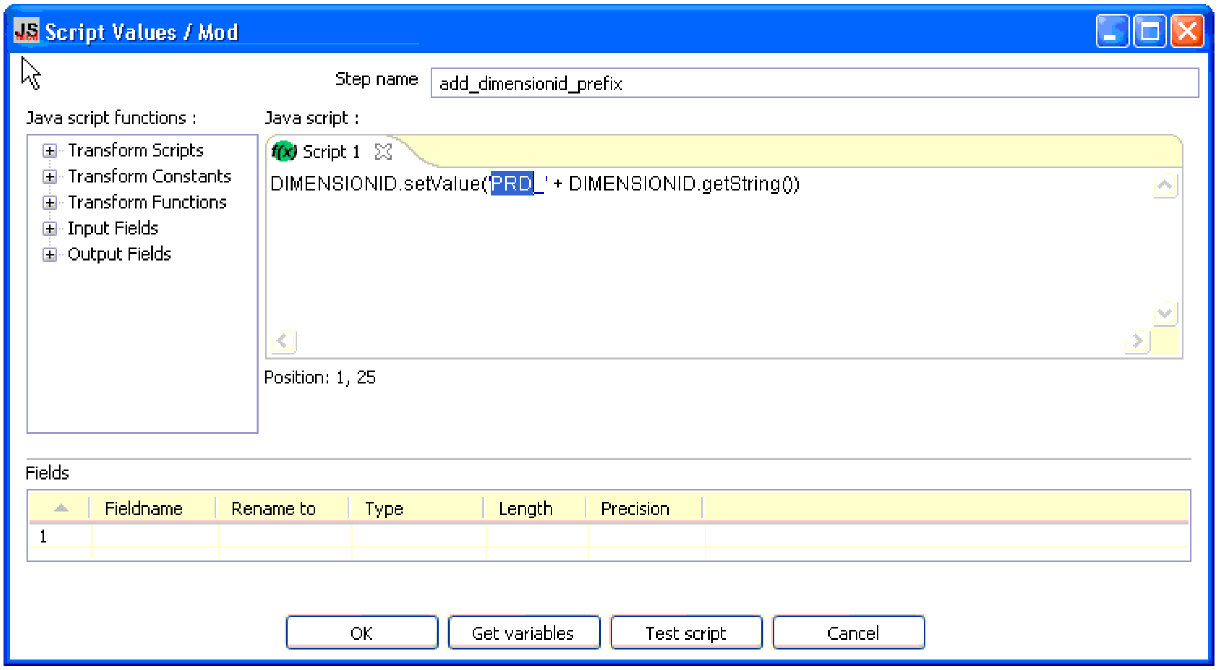
Fields :

	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value
1	DIMENSIONTYPE	String							PRD

Figure 18: Resetting the DIMENSIONTYPE String

9. Double-click the `add_dimensionid_prefix` step to open the `Script Values` dialog box.

10. Change the script to use the dimension-type string assigned in Step 8, as shown in Figure 19, and click `OK`.



Step name: add_dimensionid_prefix

Java script :

```
DIMENSIONID.setValue('PRD_' + DIMENSIONID.getString())
```

Position: 1, 25

Fields

	Fieldname	Rename to	Type	Length	Precision
1					

Figure 19: Redefining DIMENSIONID

11. Close the Designer saving all work.

12. Stop iWD Data Mart runtime node.

13. In the `aggregate_stats\stats` subdirectory, using an ASCII editor, edit the `stats.properties` file to enable the statistic. Add the line highlighted below and then save and close the file:

```
${KETTLE_REPOS_DIR}\aggregate_stats\stats\department_activeheld.ktr
${KETTLE_REPOS_DIR}\aggregate_stats\stats\department_newcompleted.ktr
${KETTLE_REPOS_DIR}\aggregate_stats\stats\process_activeheld.ktr
${KETTLE_REPOS_DIR}\aggregate_stats\stats\process_pendingoverdue.ktr
${KETTLE_REPOS_DIR}\aggregate_stats\stats\solution_newcompleted.ktr
${KETTLE_REPOS_DIR}\aggregate_stats\stats\product_pendingoverdue.ktr
```

This new `product_pendingoverdue` script is now ready for ETL to calculate overdue and pending tasks on its next run.

Activating Aggregate Plugins

Similar to activating statistics (described in Step 13 above), you activate a plug-in by appending it to the `plugins.properties` file, which is located in the `\plugins` subfolder. By default, only the task classification aggregate is enabled in this file—its contents consist of the following single line:

```
${KETTLE_REPOS_DIR}\plugins\classif,task_classif_fact
```

To enable additional plugins, perform the following steps:

1. Stop iWD Data Mart runtime node.
2. Using an ASCII editor, open `plugins.properties`, add a row for each plug-in that you want to enable, and save.
3. Run the iWD database upgrade command-line utility, `dbup_dm.cmd`, to see the effect of these changes. (Refer to the “Creating the iWD Data Mart Database” procedure in the *intelligent Workload Distribution 8.0 Deployment Guide* for information about this utility.)

For example, to add the task agent aggregate plug-in, add the highlighted line below to the file:

```
${KETTLE_REPOS_DIR}\plugins\classif,task_classif_fact
${KETTLE_REPOS_DIR}\plugins\agent,task_agent_fact
```

Genesys also provides transformation scripts for task capture aggregates and task queue aggregates that you must enable in order to use.

Supplements

Related Documentation Resources

The following resources provide additional information that is relevant to this software. Consult these additional resources as necessary.

Kettle

Genesys iWD Data Mart uses Kettle, which is a third-party ETL product from Pentaho, a leader in open-source business intelligence. For more information on Kettle, please visit `www.pentaho.com`.

intelligent Workload Distribution

You are encouraged to read the following iWD product documentation to gain an understanding of the concepts in iWD—particularly, an emphasis on the various task states that are used heavily in reporting and in the creation of many iWD Data Mart measures:

- *intelligent Workload Distribution 8.0 iWD Manager Help*
- *intelligent Workload Distribution 8.0 Deployment Guide*

In addition, the `https://sites.google.com/a/iwdlab.com/iwd8/` website provides a few examples illustrating how to create custom statistics and aggregates.

Framework

The *Framework 8.0 Stat Server Deployment Guide* includes instructions on how to configure Java Extensions.

Genesys Interactive Insights for iWD

The *Genesys Interactive Insights 8.0 Universe Guide* includes descriptions of the reports and the supporting universe elements that summarize the routing, categorization, and handling of tasks throughout an enterprise.

Genesys Info Mart

Because comprehensive tasks and interaction reports can be made available by leveraging data from iWD Data Mart and Genesys Info Mart, you are encouraged to review the product and reference guide documentation—specifically, the *Genesys Info Mart User's Guide* and the *Genesys Info Mart Reference Guide* for your RDBMS.

Genesys

- The *Genesys Technical Publications Glossary* ships on the Genesys Documentation Library DVD and provides a comprehensive list of the Genesys and computer-telephony integration (CTI) terminology and acronyms that are used in this document.
- The *Genesys Migration Guide* ships on the Genesys Documentation Library DVD and provides documented migration strategies for Genesys product releases. Contact Genesys Technical Support for more information.
- The Release Notes and Product Advisories for this product are available on the Genesys Technical Support website at http://genesyslab.com/support.

Information about supported hardware and third-party software is available on the Genesys Technical Support website in the following documents:

- *Genesys Supported Operating Environment Reference Manual*
- *Genesys Supported Media Interfaces Reference Manual*

Consult these additional resources, as necessary:

- The *Genesys Hardware Sizing Guide*, which provides information about Genesys hardware sizing guidelines for the Genesys releases.
- The *Genesys Interoperability Guide*, which provides information on the compatibility of Genesys products with various Configuration Layer Environments; Interoperability of Reporting Templates and Solutions; and G*plus* Adapters Interoperability.
- The *Genesys Licensing Guide*, which introduces you to the concepts, terminology, and procedures that are relevant to the Genesys licensing system.

For additional system-wide planning tools and information, see the release-specific listings of System Level Documents on the Genesys Technical Support website (these are accessible from the system level documents by release tab in the Knowledge Base Browse Documents Section).

Genesys product documentation is available on the:

- Genesys Technical Support website at http://genesyslab.com/support.
- Genesys Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com.

Document Conventions

This document uses certain stylistic and typographical conventions—introduced here—that serve as shorthands for particular kinds of information.

Document Version Number

A version number appears at the bottom of the inside front cover of this document. Version numbers change as new information is added to this document. The following is a sample version number:

80iwd_ref_datamart_03-2011_v8.0.003.00

You will need this number when you are talking with Genesys Technical Support about this product.

Screen Captures Used in This Document

Screen captures from the product graphical user interface (GUI), as used in this document, may sometimes contain minor spelling, capitalization, or grammatical errors. The text that accompanies and explains the screen captures corrects such errors, *except* when such a correction would prevent you from installing, configuring, or successfully using the product. For example, if the name of an option contains a usage error, the name would be presented exactly as it appears in the product GUI; the error would not be corrected in any accompanying text.

Type Styles

Table 45 describes and illustrates the type conventions that are used in this document.

Table 45: Type Styles

Type Style	Used For	Examples
Italic	• Document titles • Emphasis • Definitions of (or first references to) unfamiliar terms • Mathematical variables Also used to indicate placeholder text within code samples or commands, in the special case in which angle brackets are a required part of the syntax (see the note about angle brackets on page 94).	Please consult the *Genesys Migration Guide* for more information. Do *not* use this value for this option. A *customary and usual* practice is one that is widely accepted and used within a particular industry or profession. The formula $x + 1 = 7$, where x stands for . . .

Table 45: Type Styles (Continued)

Type Style	Used For	Examples
Monospace font (Looks like `teletype` or `typewriter text`)	All programming identifiers and GUI elements. This convention includes: • The *names* of directories, files, folders, configuration objects, paths, scripts, dialog boxes, options, fields, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages. • The values of options. • Logical arguments and command syntax. • Code samples. Also used for any text that users must manually enter during a configuration or installation procedure, or on a command line.	Select the `Show variables on screen` check box. In the `Operand` text box, enter your formula. Click `OK` to exit the `Properties` dialog box. T-Server distributes the error messages in `EventError` events. If you select `true` for the `inbound-bsns-calls` option, all established inbound calls on a local agent are considered business calls. Enter `exit` on the command line.
Square brackets ([ ])	A particular parameter or value that is optional within a logical argument, a command, or some programming syntax. That is, the presence of the parameter or value is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information.	`smcp_server -host [/flags]`
Angle brackets (< >)	A placeholder for a value that the user must specify. This might be a DN or a port number specific to your enterprise. **Note:** In some cases, angle brackets are required characters in code syntax (for example, in XML schemas). In these cases, italic text is used for placeholder values.	`smcp_server -host <confighost>`

Index