Performance Optimization for Informatica Data Services (9.5.0 - 9.6.1 Hotfix 3)
Abstract

You can optimize data service performance by enabling performance features and tuning Data Integration Service properties that benefit data service requests. This article describes the data services architecture, performance features, and the steps to increase data service performance.

Supported Versions

- Data Services 9.5.0 - 9.6.1 HotFix 3

Table of Contents

Overview ............................................................................................................. 3
Performance Optimization New Features and Enhancements since 9.1.0 HotFix 2. .................................................................... 3
  9.6.1 HotFix 1 .......................................................................................... 3
  9.6.0 ...................................................................................................... 3
  9.5.1 HotFix 1 .......................................................................................... 4
  9.5.0 ...................................................................................................... 4
Data Services Architecture.................................................................................. 4
Mapping Optimization......................................................................................... 5
  Early Projection Optimization Method. .......................................................... 6
  Early Selection Optimization Method. ............................................................ 6
  Predicate Optimization Method. ..................................................................... 7
  Pushdown Optimization Method. .................................................................... 8
Cost-Based Optimization Method. ..................................................................... 8
Semi-Join Optimization Method. ....................................................................... 9
Branch Pruning Optimization Method. ............................................................ 10
Push-Into Optimization. .................................................................................. 10
SQL Data Service Optimization Level Configuration. ...................................... 10
Web Service Optimization Level Configuration. .............................................. 12
Data Object Caching.......................................................................................... 13
  Enabling Data Object Caching. ..................................................................... 14
Result Set Caching. .......................................................................................... 16
  Enabling Result Set Caching for an SQL Data Service. .................................. 16
  Enabling Result Set Caching for a Web Service. ........................................... 16
Transformation Cache. .................................................................................... 17
Data Integration Service Optimization.............................................................. 18
  SQL Data Service Property for Memory and Concurrent Requests. .................. 19
  Web Service Property to Configure an Active DTM Instance. ......................... 20
  Web Services Properties for Memory and Concurrent Requests ..................... 20
  Web Service Log Management. ..................................................................... 21
Example Data Integration Service Configuration for Concurrent Web Service Requests . 22
Overview

Informatica Data Services provides the ability to access and transform data through an SQL data service or a web service. An SQL data service is a virtual database that end users can query. A web service is a collection of operations that provide access to data integration functionality. The Data Integration Service provides end-users access to data services. You can optimize data service performance by enabling performance features and tuning Data Integration Service properties that benefit each type of data service request.

Use the following performance optimization techniques to get the best performance results based on your requirements:

- Mapping optimization
- Result set caching
- Data object caching
- Transformation caching
- Data Integration Service tuning

Note: Components outside of the Informatica product such as network and database performance can also affect data services performance results.

Performance Optimization New Features and Enhancements since 9.1.0 HotFix 2

This section describes the new features and enhancements related to performance optimization from 9.5.0 to 9.6.1 HotFix 2.

9.6.1 HotFix 1

Branch pruning optimization

The Data Integration Service can apply the branch pruning optimization method. When the Data Integration Service applies the branch pruning method, it removes transformations that do not contribute any rows to the target in a mapping.

The Developer tool enables the branch pruning optimization method by default when you choose the normal or full optimizer level. You can disable branch pruning if the optimization does not increase performance by setting the optimizer level to minimal or none.

Pushdown optimization

The Data Integration Service can push transformation logic to Netezza sources that use native drivers.

9.6.0

Pushdown optimization

- The Data Integration Service can push expression, aggregator, operator, union, sorter, and filter functions to Greenplum sources when the connection type is ODBC.
- The Data Integration Service can push transformation logic to SAP HANA and Teradata sources when the connection type is ODBC.
In the Developer tool you can enable a mapping to perform the following optimizations:

- Push a custom SQL query to a relational data object.
- Push operations such as Union, Union All, Intersect, Intersect All, Minus, Minus All, and Distinct to a relational data object.
- Perform early selection and push queries that contain the SQL keyword LIMIT to a relational data object.
- Push a Union transformation to a relational data object.
- Push Filter, Expression, Union, Sorter, and Aggregator transformations to a Hive relational object.

### 9.5.1 HotFix 1

**Pushdown optimization**

You can push transformation logic for an Aggregator transformation and a Sorter transformation to a relational source database.

### 9.5.0

**Early selection and push-into optimization**

You can configure early selection and push-into optimization with the Java transformation, Web Service Consumer transformation, and the SQL transformation.

**Database optimizer**

You can add hints to a source SQL query to pass instructions to a database optimizer. The optimizer uses the hints to choose a query run plan to access the source. The source database must be Oracle, Sybase, IBM DB2, or Microsoft SQL Server.

**Cost-based optimization**

The Data Integration Service cost-based optimization method can use profiling statistics or database statistics to estimate the number or rows that pass through a transformation.

Previously, the Data Integration Service could use database statistics to estimate the number of rows that pass through a transformation.

**Data Integration Service**

- Configure the **Temporary Directories** property to specify the location where the Result Set Cache Manager stores cache files when there is not enough space for the cache in memory.
  
  Previously, you configured the **Storage Directory** property to specify the location where the Result Set Cache Manager stored cache files when there was not enough space for the cache in memory.

- You can configure the optimizer level in Data Integration Service application properties for an SQL data service or a web service. The optimizer level determines which optimization methods the Data Integration Service applies to the SQL data service query or to the web service request at run time.

### Data Services Architecture

When you create an SQL data service or a web service, the Developer tool stores the objects in the Model repository. To make a data service accessible to end users, you deploy the data service to the Data Integration Service. When you deploy a data service, you add the object to an application and deploy the application to the Data Integration Service. The Data Integration Service performs the data transformation processes for SQL data services and web services.
The following figure shows how clients connect to data services:

An external client sends SQL queries to access data in virtual tables of an SQL data service, execute virtual stored procedures, and access metadata. A web service client sends requests to run a web service operation to read, transform, or write data.

When the Data Integration Service receives an SQL query from an external client, the SQL Service Module of the Data Integration Service starts a Data Transformation Manager (DTM) process to run the query. The Data Integration Service returns the query results to the external client.

When the Data Integration Service receives a web service request message, the Web Service Module of the Data Integration Service starts a DTM process to complete the request. The Data Integration Service sends the results in a web service response message to the web service client.

**Mapping Optimization**

Use mapping optimization to increase the performance of SQL data service queries and web service requests. The Data Integration Service optimizes mappings based on the mapping optimizer level that you configure.

The optimizer level determines which optimization methods the Data Integration Service applies to the mapping at run time. You can choose one of the following mapping optimizer levels:

- **None**
  - The Data Integration Service does not apply any optimization.

- **Minimal**
  - The Data Integration Service applies the early projection optimization method.

- **Normal**
  - The Data Integration Service applies the early projection, early selection, branch pruning, push-into, pushdown, and predicate optimization methods. Normal is the default optimization level.

- **Full**
  - The Data Integration Service applies the cost-based, early projection, early selection, branch pruning, predicate, push-into, pushdown, and semi-join optimization methods.

By default, the Data Integration Service applies the Normal optimizer level to mappings.

When you configure an optimization level, consider the following guidelines:

- The normal optimizer level can benefit most mappings.
• The full optimizer level benefits mappings that contain a Joiner transformation. It is especially beneficial when you join data from tables that are different sizes.

• The minimal or none optimizer level is useful when you verify data output or debug mapping logic.

You can configure the optimizer level in the Developer tool data viewer configuration setting, in the Administrator tool Data Integration Service properties, in the SQL data service connection, or in the header of the web service request.

**Early Projection Optimization Method**

When the Data Integration Service applies the early projection optimization method, it identifies unused ports and removes the links between those ports.

Early projection improves performance by reducing the amount of data that the Data Integration Service moves across transformations. When the Data Integration Service processes a mapping, it moves the data from all connected ports in a mapping from one transformation to another. In large, complex mappings, or in mappings that use nested mapplets, some ports might not supply data to the target. The Data Integration Service identifies the ports that do not supply data to the target. After the Data Integration Service identifies unused ports, it removes the links between all unused ports from the mapping.

The Data Integration Service does not remove all links. For example, it does not remove the following links:

• Links connected to a transformation that has side effects.

• Links connected to transformations that call an ABORT() or ERROR() function, send email, or call a stored procedure.

If the Data Integration Service determines that all ports in a transformation are unused, it removes all transformation links except the link to the port with the least data. The Data Integration Service does not remove the unused transformation from the mapping.

The Developer tool enables this optimization method by default.

**Early Selection Optimization Method**

When the Data Integration Service applies the early selection optimization method, it splits, moves, or removes the Filter transformations in a mapping. It moves filters up the mapping closer to the source.

The Data Integration Service might split a Filter transformation if the filter condition is a conjunction. For example, the Data Integration Service might split the filter condition "A>100 AND B<50" into two simpler conditions, "A>100" and "B<50." When the Data Integration Service splits a filter, it moves the simplified filters up the mapping pipeline, closer to the source. The Data Integration Service moves the filters up the pipeline separately when it splits the filter.

The early selection optimization method is enabled by default when you choose the normal or full optimizer level in the Developer tool. The Data Integration Service ignores early selection optimization if a transformation that appears before the Filter transformation has side effects. The Data Integration Service cannot determine if the SQL transformation, Web Service Consumer transformation, and Java transformation have side effects. You can configure early selection optimization for these transformations if they do not have side effects.

You can disable early selection if the optimization does not increase performance. The Data Integration Service enables this optimization method by default.
**Predicate Optimization Method**

When the Data Integration Service applies the predicate optimization method, it examines the predicate expressions that a mapping generates. It determines whether it can simplify or rewrite the expressions to increase mapping performance.

When the Data Integration Service runs a mapping, it generates queries against the mapping sources and performs operations on the query results based on the mapping logic and the transformations within the mapping. The queries and operations often include predicate expressions. Predicate expressions represent the conditions that the data must satisfy. The filter and join conditions in Filter and Joiner transformations are examples of predicate expressions.

With the predicate optimization method, the Data Integration Service also attempts to apply predicate expressions as early as possible in the mapping to improve mapping performance.

The Data Integration Service infers relationships from existing predicate expressions and creates new predicate expressions. For example, a mapping contains a Joiner transformation with the join condition "$A=B" and a Filter transformation with the filter condition "$A>5." The Data Integration Service might be able to add "$B>5" to the join condition.

The Data Integration Service applies the predicate optimization method with the early selection optimization method when it can apply both methods to a mapping. For example, when the Data Integration Service creates new filter conditions through the predicate optimization method, it also attempts to move them upstream in the mapping through the early selection method. Applying both optimization methods improves mapping performance when compared to applying either method alone.

The Data Integration Service applies the predicate optimization method if the application increases performance. The Data Integration Service does not apply this method if the application changes the mapping results or it decreases the mapping performance.

**Predicate Optimization Rules and Guidelines**

When the Data Integration Service rewrites a predicate expression, it applies mathematical logic to the expression to optimize it.

The Data Integration Service might perform any or all of the following actions:

- Identifies equivalent variables across predicate expressions in the mapping and generates simplified expressions based on the equivalencies.
- Identifies redundant predicates across predicate expressions in the mapping and removes them.
- Extracts subexpressions from disjunctive clauses and generates multiple, simplified expressions based on the subexpressions.
- Normalizes a predicate expression.
- Applies predicate expressions as early as possible in the mapping.

The Data Integration Service might not apply predicate optimization to a mapping when the mapping contains transformations with a datatype mismatch between connected ports.

The Data Integration Service might not apply predicate optimization to a transformation when any of the following conditions are true:

- The transformation contains explicit default values for connected ports.
- The transformation has side effects.
- The transformation does not allow predicates to be moved. For example, a transformation that has side effects might have this restriction.

The Developer tool enables the predicate optimization method by default.


Pushdown Optimization Method

When the Data Integration Service applies pushdown optimization, it pushes transformation logic to the source database. The Data Integration Service translates the transformation logic into SQL queries and sends the SQL queries to the database. The source database runs the SQL queries to process the transformations.

Pushdown optimization increases mapping performance when the source database can process transformation logic faster than the Data Integration Service. The Data Integration Service also reads less data from the source.

The amount of transformation logic that the Data Integration Service pushes to the source database depends on the database, the transformation logic, and the mapping configuration. The Data Integration Service processes all transformation logic that it cannot push to a database.

The Data Integration Service applies pushdown optimization to a mapping when you select the normal or full optimizer level. When you enable pushdown optimization, the Data Integration Service analyzes the optimized mapping from the source to the target or until it reaches a downstream transformation that it cannot push to the source database. The Data Integration Service generates and executes a SELECT statement for each source that has transformation logic pushed down. Then, it reads the results of this SQL query and processes the remaining transformations in the mapping.

Pushdown Optimization Rules and Guidelines

The Data Integration Service can push transformation logic to the source database.

The following rules and guidelines apply to pushdown optimization:

- The Data Integration Service can push Joiner transformation logic to the source database if the sources are in the same database management system and they use identical connections.
- The Data integration Service cannot push transformation logic to a source that has a binary data type.

Cost-Based Optimization Method

With cost-based optimization, the Data Integration Service evaluates a mapping, generates semantically equivalent mappings, and runs the mapping with the best possible performance. Cost-based optimization reduces run time for mappings that perform adjacent, sorted, unsorted, and inner-join operations.

Semantically equivalent mappings are mappings that perform identical functions and produce the same results. To generate semantically equivalent mappings, the Data Integration Service divides the original mapping into fragments. The Data Integration Service then determines which mapping fragments it can optimize.

During optimization, the Data Integration Service might add, remove, or reorder transformations within a fragment. The Data Integration Service verifies that the optimized fragments produce the same results as the original fragments and forms alternate mappings that use the optimized fragments.

The Data Integration Service generates all or almost all of the mappings that are semantically equivalent to the original mapping. It uses profiling statistics or database statistics to compute the cost for the original mapping and each alternate mapping. Then, it identifies the mapping that runs most quickly. The Data Integration Service performs a validation check on the best alternate mapping to ensure that it is valid and produces the same results as the original mapping.

The Data Integration Service caches the best alternate mapping in memory. When you run a mapping, the Data Integration Service retrieves the alternate mapping and runs it instead of the original mapping.

The Developer tool does not enable this method by default.
**Semi-Join Optimization Method**

The semi-join optimization method attempts to reduce the amount of data extracted from the source by modifying join operations in the mapping.

The Data Integration Service applies this method to a Joiner transformation when one input group has many more rows than the other and when the larger group has many rows with no match in the smaller group based on the join condition. The Data Integration Service attempts to decrease the size of the data set of one join operand by reading the rows from the smaller group, finding the matching rows in the larger group, and then performing the join operation. Decreasing the size of the data set improves mapping performance because the Data Integration Service no longer reads unnecessary rows from the larger group source. The Data Integration Service moves the join condition to the larger group source and reads only the rows that match the smaller group.

Before applying this optimization method, the Data Integration Service performs analyses to determine whether semi-join optimization is possible and likely to be worthwhile. If the analyses determine that this method is likely to increase performance, the Data Integration Service applies it to the mapping. The Data Integration Service then reanalyzes the mapping to determine whether there are additional opportunities for semi-join optimization. It performs additional optimizations if appropriate. The Data Integration Service does not apply semi-join optimization unless the analyses determine that there is a high probability for improved performance.

The Developer tool does not enable this method by default.

**Semi-Join Optimization Requirements for Increased Performance**

The semi-join optimization method does not always increase performance. The following factors affect mapping performance with semi-join optimization:

- The Joiner transformation master source must have significantly fewer rows than the detail source.
- The detail source must be large enough to justify the optimization. When the Data Integration Service applies semi-join optimization, the method adds some overhead time to mapping processing. If the detail source is small, the time required to apply the semi-join method might exceed the time required to process all rows in the detail source.
- The Data Integration Service must be able to obtain source row count statistics for a Joiner transformation in order to accurately compare the time requirements of the regular join operation against the semi-join operation.

**Semi-Join Optimization Rules and Guidelines**

The Data Integration Service can apply semi-join optimization to a Joiner transformation if the transformation meets the following requirements:

- The join type must be normal, master outer, or detail outer. The joiner transformation cannot perform a full outer join.
- The detail pipeline must originate from a relational source.
- The join condition must be a valid sort-merge-join condition. That is, each clause must be an equality of one master port and one detail port. If there are multiple clauses, they must be joined by AND.
- If the mapping does not use target-based commits, the Joiner transformation scope must be All Input.
- The master and detail pipelines cannot share any transformation.
- The mapping cannot contain a branch between the detail source and the Joiner transformation.
**Branch Pruning Optimization Method**

The Data Integration Service can apply the branch pruning optimization method to transformations that do not contribute any rows to the target in a mapping.

The Data Integration Service might remove a Filter transformation if the filter condition evaluates to FALSE for the data rows. For example, a mapping has two Filter transformations that filter data from two relational sources. A Filter transformation has the filter condition `Country=US`, and the other Filter transformation has the filter condition `Country=Canada`. A Union transformation joins the two relational sources and has the filter condition `Country=US`. The Data Integration Service might remove the Filter transformation with the filter condition `Country=Canada` from the mapping.

The Developer tool enables the branch pruning optimization method by default when you choose the normal or full optimizer level. You can disable branch pruning if the optimization does not increase performance by setting the optimizer level to minimal or none.

**Push-Into Optimization**

With Push-into optimization, the Data Integration Service moves the Filter transformation logic to the transformation immediately upstream of the Filter transformation in the mapping. Push-into optimization increases performance by reducing the number of rows that pass through the mapping.

The Data Integration Service does not move filter logic into another transformation if the transformation has side effects. The SQL transformation, Java transformation, and the Web Service Consumer transformation can perform different functions. The Developer tool cannot determine whether these transformations have side effects or not. You can indicate that the SQL transformation or Web Service Consumer transformation do not have side effects when you configure the transformation properties. You can configure the Java transformation for push-into optimization.

**SQL Data Service Optimization Level Configuration**

Configure the optimizer level when you want the SQL data service to use an optimizer level other than the normal optimizer level. By default, each SQL data service uses the normal optimizer level.

To understand how the mapping optimizer level creates an optimized query for an SQL data service, view the query plan for an SQL Data Service. When you view the query plan, the Developer tool displays a graphical representation of the optimized query based on the optimizer level and a graphical representation of the original query.

You can use one or more of the following methods to configure the optimizer level for an SQL data service:

- Configure the optimizer level for data preview of an SQL data service before you deploy it to a Data Integration Service.
- Configure the optimizer level for deployed SQL data services that run on a specific Data Integration Service.
- Configure the optimizer level in the connection string of queries that you run against a deployed SQL data services.

**Configuring the SQL Data Service Optimizer Level for Data Preview**

Configure the optimizer level that the Data Integration Service uses to execute SQL queries when you preview the output of a SQL data service.

1. In the Developer tool, click **Run > Open Run Dialog**.
   The **Run** dialog box appears.
2. Click **Data Viewer Configuration**.
3. Click the **New** button.
4. Enter a name for the data viewer configuration.
5. Click the **Advanced** tab.
6. Select an optimizer level.
7. Click **Apply**.
8. Click **Close**

The Developer tool creates the data viewer configuration.

### Configuring the Optimizer Level for Deployed SQL Data Services

Configure the optimizer level that the Data Integration Services uses to execute SQL queries against a deployed SQL data service. You can choose to override the optimizer level for a single query by configuring the optimizer level in the SQL data service connection.

1. In the Administrator tool, select a Data Integration Service.
2. Click the **Applications** view.
3. Expand the application that contains the SQL data service for which you want to configure the optimizer level.
4. Select the SQL data service and edit the following property:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization Level</td>
<td>The optimizer level that the Data Integration Service applies to the object. Enter the numeric value that is associated with the optimizer level that you want to configure. You can enter one of the following numeric values:</td>
</tr>
<tr>
<td></td>
<td>- 0. The Data Integration Service does not apply optimization.</td>
</tr>
<tr>
<td></td>
<td>- 1. The Data Integration Service applies the early projection optimization method.</td>
</tr>
<tr>
<td></td>
<td>- 2. The Data Integration Service applies the early projection, early selection, push-into, pushdown, and predicate optimization methods.</td>
</tr>
<tr>
<td></td>
<td>- 3. The Data Integration Service applies the cost-based, early projection, early selection, push-into, pushdown, predicate, and semi-join optimization methods.</td>
</tr>
</tbody>
</table>

5. To override optimizer level that the Data Integration Services uses to execute a query, append the following entry to the JDBC URL or ODBC connection string: `SQLDataServiceOptions.optimizeLevel=<numeric_optimizer_level>`. 

### SQL Data Service Query Plan

When you view the query plan for an SQL data service, you view the graphical representation of the original query and the graphical representation of the optimized query. The graphical representation describes how the Data Integration Service processes the query. It includes the transformations and the order which the Data Integration Services processes each transformation.

The Developer tool uses the optimizer level that you set in the Developer tool to generate the optimized query. The optimized query displays the query as the Data Integration Service runs it.

For example, you want to query the CUSTOMERS virtual table in an SQL data service. In the **Data Viewer** view, you choose the default data viewer configuration settings, which sets the optimizer level for the query to normal.

You enter the following query in the **Data Viewer** view:

```sql
select * from CUSTOMERS where CUSTOMER_ID > 150000 order by LAST_NAME
```

When you view the SQL query plan, the Developer tool displays the following graphical representation of the query:
The non-optimized view displays the query that you enter. The Developer tool displays the WHERE clause as a Filter transformation and the ORDER BY clause as a Sorter transformation. The Developer tool uses the pass-through Expression transformation to rename ports.

When you view the optimized query, the Developer tool displays the following graphical representation of the query:

The optimized view displays the query that the Data Integration Service runs. Because the optimizer level is normal, the Data Integration Service pushes the filter condition to the source data object. Pushing the filter condition increases query performance because it reduces the number of rows that the Data Integration Service reads from the source data object. Similar to the non-optimized query, the Developer tool displays the ORDER BY clause as a Sorter transformation. It uses pass-through Expression transformations to enforce the datatypes that you specify in the logical transformations.

**Viewing an SQL Query Plan**

Display the SQL query plan to view a mapping-like representation of the SQL query you enter when you preview virtual table data.

1. Open an SQL data service that contains at least one virtual table.
2. Click the **Data Viewer** view.
3. Enter an SQL query in the **Input** window.
4. Optionally, select a data viewer configuration that contains the optimizer level you want to apply to the query.
5. Click **Show Query Plan**.
   - The Developer tool displays the SQL query plan for the query as you entered it on the **Non-Optimized** tab.
6. To view the optimized query, click the **Optimized** tab.
   - The Developer tool displays the optimized SQL query plan.

**Web Service Optimization Level Configuration**

Configure the optimizer level when you want the web service to use an optimizer level other than the normal optimizer level. By default, each web service uses the normal optimizer level.

You can use one or more of the following methods to configure the optimizer level for a web service:

- Configure the optimizer level for data preview of a web service before you deploy it to a Data Integration Service.
- Configure the optimizer level for deployed web services that run on a specific Data Integration Service.
- Configure the optimizer level in the header of the web service request.
Configuring the Web Service Optimizer Level for Data Preview

Configure the optimizer level that the Data Integration Services uses to preview the output of a web service.

1. In the Developer tool, click Run > Open Run Dialog.
   The Run dialog box appears.
2. Click Web Service Configuration.
3. Click the New button.
4. Enter a name for the web service configuration.
5. Click the Advanced tab.
6. Select an optimizer level.
7. Click Apply.
8. Click Close
   The Developer tool creates the web service configuration.

When you run the data viewer to preview the output of an operation mapping, select the web service configuration that includes the optimizer level that you want to use.

Configuring the Optimizer Level for Deployed Web Services

Configure the optimizer level that the Data Integration Services uses to run a deployed web service. You can choose to override the optimizer level for a single request by configuring the optimizer level in the HTTP header of the web service SOAP request.

1. In the Administrator tool, select a Data Integration Service.
2. Click the Applications view.
3. Expand the application that contains the web service for which you want to configure the optimizer level.
4. Select the web service and edit the following property:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization Level</td>
<td>The optimizer level that the Data Integration Service applies to the object. Enter the numeric value that is associated with the optimizer level that you want to configure. You can enter one of the following numeric values:</td>
</tr>
<tr>
<td></td>
<td>- 0. The Data Integration Service does not apply optimization.</td>
</tr>
<tr>
<td></td>
<td>- 1. The Data Integration Service applies the early projection optimization method.</td>
</tr>
<tr>
<td></td>
<td>- 2. The Data Integration Service applies the early projection, early selection, push-into, pushdown, and predicate optimization methods.</td>
</tr>
<tr>
<td></td>
<td>- 3. The Data Integration Service applies the cost-based, early projection, early selection, push-into, pushdown, predicate, and semi-join optimization methods.</td>
</tr>
</tbody>
</table>

5. To override the web service optimization level for a web service request, include the following entry in the HTTP header of the web service SOAP request: WebServiceOptions.optimizeLevel=\<numeric_optimizer_level>.

Data Object Caching

The Data Integration Service can use cached data for data objects. Users that access data objects frequently might want to enable caching for data objects.

You can increase mapping and query performance for end users by caching the logical data objects and virtual tables. If the application contains an SQL data service, you can cache logical data objects and virtual tables. If the application
contains a web service, you can cache logical data objects. When you enable data object caching, the Data Integration Service accesses pre-built logical data objects and virtual tables. When the Data Integration Service can access pre-built data objects, performance increases for SQL queries or web service requests.

The Data Integration Service can use the following databases to store data object cache tables:

- IBM DB2
- Microsoft SQL Server
- Oracle

To optimize data object cache performance, consider the following techniques:

- Define primary keys and foreign keys for logical data objects. When the Data Integration Service generates cache for logical data objects with keys, it creates indexes. The indexes can increase the performance of queries on the cache database.
- Cache logical data objects that you join in a mapping. When you join cached logical data objects, the Data Integration Service can pushdown the Joiner transformation logic to the cache database even when the source data originates from different databases.

For more information about data object caching, see the Informatica How-To Library article "Increasing the Performance of Deployed Objects Through Data Object Caching."

**Enabling Data Object Caching**

To configure data object caching, configure the cache properties for the Data Integration Service and the application that contains the logical data object and virtual tables.

1. In the Administrator tool, select a Data Integration Service.
2. Click the **Properties** view.
3. Edit the following Logical Data Object/Virtual Table Cache properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache Removal Time</td>
<td>The number of milliseconds that the Data Integration Service waits before cleaning up cache storage after a refresh. Default is 3,600,000.</td>
</tr>
<tr>
<td>Cache Connection</td>
<td>The database connection name for the database that stores the data object cache. Select a valid connection object name.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Maximum Concurrent Refresh Requests</td>
<td>Maximum number of cache refreshes that can occur at the same time. Limit the concurrent cache refreshes to maintain system resources.</td>
</tr>
<tr>
<td>Enable Nested LDO Cache</td>
<td>Indicates that the Data Integration Service can use cache data for a logical data object used as a source or a lookup in another logical data object during a cache refresh. If false, the Data Integration Service accesses the source resources even if you enabled caching for the logical data object used as a source or a lookup. For example, logical data object LDO3 joins data from logical data objects LDO1 and LDO2. A developer creates a mapping that uses LDO3 as the input and includes the mapping in an application. You enable caching for LDO1, LDO2, and LDO3. If you enable nested logical data object caching, the Data Integration Service uses cache data for LDO1 and LDO2 when it refreshes the cache table for LDO3. If you do not enable nested logical data object caching, the Data Integration Service accesses the source resources for LDO1 and LDO2 when it refreshes the cache table for LDO3. Default is False.</td>
</tr>
</tbody>
</table>

4. Click OK.
5. Click the Applications view.
6. Select the application that contains the logical data object or virtual table for which you want to enable or disable caching.
7. Stop the application.
8. Expand the application that contains the data object for which you want to enable or disable caching.
9. Select the logical data object or virtual table.
10. Edit the following logical data object or virtual table properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Caching</td>
<td>Cache the logical data object or virtual table.</td>
</tr>
<tr>
<td>Cache Refresh Period</td>
<td>Number of minutes between cache refreshes.</td>
</tr>
<tr>
<td>Cache Table Name</td>
<td>The name of the user-managed table from which the Data Integration Service accesses the logical data object or the virtual table cache. A user-managed cache table is a table in the data object cache database that you create, populate, and manually refresh when needed. If you specify a cache table name, the Data Object Cache Manager does not manage the cache for the object and ignores the cache refresh period. If you do not specify a cache table name, the Data Object Cache Manager manages the cache for the object.</td>
</tr>
</tbody>
</table>

11. Click OK.
12. Restart the application.
Result Set Caching

Result set caching enables the Data Integration Service to use cached results for SQL data service queries and web service requests. Users that run identical queries in a short period of time may want to use result set caching to decrease the runtime of identical queries.

When you configure result set caching, the Data Integration Service caches the results of the DTM process associated with each SQL data service query and web service request. The Data Integration Service caches the results for the expiration period that you configure. When a client makes the same query or request before the cache expires, the Data Integration Service returns the cached results.

The Data Integration Service stores the result set cache in memory and in cache files on disk. When the Data Integration Service caches data, it first writes the cache to memory and then writes the overflow cache to cache files on disk. Data Integration Service allocates cache memory based on the result set cache Maximum Total Memory Size property. Data Integration Service allocates disk space based on the result set cache Maximum Total Disk Size property. When the amount of cache exceeds the cache memory and disk space allocation, the Data Integration Service does not store additional cache.

When you enable the Data Integration Service to use cached results, data service performance increases. However, to further improve the data service processing time for identical queries and requests, allocate enough space to store the cache in memory. When you configure the amount of cache memory to be equal to or greater than what is required to cache the results, you increase performance by reducing the system I/O overhead. When the Data Integration Service writes cache files to disk, the data service processing time increases due to system I/O overhead.

Enabling Result Set Caching for an SQL Data Service

To use cached results for identical SQL data service queries, configure the Data Integration Service to use result set caching.

1. In the Administrator tool, select a Data Integration Service.
2. Click the Process view to configure the following result set cache properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Name Prefix</td>
<td>The prefix for the names of all result set cache files stored on disk. Default is RSCACHE.</td>
</tr>
<tr>
<td>Enable Encryption</td>
<td>Indicates whether result set cache files are encrypted using 128-bit AES encryption. Valid values are true or false. Default is true.</td>
</tr>
</tbody>
</table>

3. Click the Application view and then click the SQL data service to configure the following property:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result Set Cache Expiration Period</td>
<td>The number of milliseconds that the result set cache is available for use. If set to -1, the cache never expires. If set to 0, result set caching is disabled. Changes to the expiration period do not apply to existing caches. If you want all caches to use the same expiration period, purge the result set cache after you change the expiration period. Default is 0.</td>
</tr>
</tbody>
</table>

Enabling Result Set Caching for a Web Service

To use cached results for identical web service requests, configure the Data Integration Service to use result set caching.

1. In the Administrator tool, select a Data Integration Service.
2. Click the **Process** view to configure the following result set cache properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Name Prefix</td>
<td>The prefix for the names of all result set cache files stored on disk. Default is RSCACHE.</td>
</tr>
<tr>
<td>Enable Encryption</td>
<td>Indicates whether result set cache files are encrypted using 128-bit AES encryption. Valid values are true or false. Default is true.</td>
</tr>
</tbody>
</table>

3. Click the **Application** view, click the web service, and then click the operation to configure the following property:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result Set Cache Expiration Period</td>
<td>The number of milliseconds that the result set cache is available for use. If set to -1, the cache never expires. If set to 0, result set caching is disabled. Changes to the expiration period do not apply to existing caches. If you want all caches to use the same expiration period, purge the result set cache after you change the expiration period. Default is 0.</td>
</tr>
</tbody>
</table>

4. If you want the Data Integration Service to cache the results by user, enable WS-Security in the web service properties.

**Transformation Cache**

The Data Integration Service stores cache in memory and in cache files on disk. When the Data Integration Service caches data, it first writes the cache to memory and then writes the overflow cache to cache files to disk.

Consider the following solutions for transformation cache bottlenecks:
- **Configure the transformations to allocate enough space to store the cache in memory.**
  
  To improve the processing time for mappings that contain the Aggregator, Joiner, Lookup, Rank, or Sorter transformation, configure the transformations to allocate enough space to store the cache in memory. When you configure the amount of cache memory to be equal to or greater than what is required to cache the data and index, you increase performance by reducing the system I/O overhead. When the Data Integration Service writes cache files to disk, the processing time increases due to system I/O overhead.

- **Configure the transformation cache properties.**
  
  To increase performance, consider configuring the transformation cache properties for the Aggregator, Joiner, Lookup, Rank, or Sorter transformations.

When you configure numeric values for transformation cache properties, the value that you configure is equal to the amount of memory that the Data Integration Service can use to store data cache and index cache. When you configure the transformation cache properties to use the Auto setting, the Data Integration Service allocates cache memory based on the Data Integration Service process Maximum Memory Size and Maximum Session Memory properties.

When the maximum memory size is 0, the Data Integration Service divides the maximum session memory allocation among transformations that use the Auto setting. When the maximum memory size value is greater than zero, the Data Integration Service allocates the required amount of cache memory for each transformation, until the memory requirement exceeds the maximum memory size value.
Data Integration Service Optimization

Tune the Data Integration Service process to improve service performance. You can configure the Data Integration Service process properties for memory. You can configure each web service and SQL data service that runs on a Data Integration Service to handle concurrent requests.

Consider the following solutions for Data Integration Service bottlenecks:

**Configure the Maximum Heap Size property for the Data Integration Service process.**

The Data Integration Service can consume large amounts of memory while processing SQL data services and web services.

Use the Administrator tool to configure the Maximum Heap Size property to a larger value in the Advanced Properties for the Data Integration Service process.

**Configure the web service DTM Keep Alive Time property for the Data Integration Service.**

The Data Integration Service consumes system resources to spawn a DTM instance for each web service request. Configure the Data Integration Service to use one DTM instance to process more than one web service request.

Use the Administrator tool to configure the web service DTM Keep Alive Time property for the Data Integration Service.

**Configure the execution options in the Data Integration process properties and the web service and SQL data service properties for concurrent requests.**

The Data Integration Service, each SQL data service, and each web service that runs on the Data Integration Service consumes system and memory resources for each concurrent request.

To configure the number of concurrent request that the Data Integration Service, each SQL data service, and each web service can accept, configure the Data Integration Service process properties and the web service properties.

Use the Administrator tool to configure the following options and properties for the Data Integration Service, web service and SQL data service:

- Configure the execution options for the Data Integration Service process.
- Configure the Maximum # of Concurrent Connections property for each SQL data service for the Data Integration Service process.
- Configure the Maximum Backlog Request and the Maximum Concurrent Requests properties for each web service in the HTTP configuration properties for the Data Integration Service process.

**Turn off the web service trace level.**

The number of web service log files that the Data Integration Service writes and maintains can decrease performance.

Use the Administrator tool to configure the web service trace level to reduce the amount of web service runtime log files that the Data Integration Service stores on disk.
**SQL Data Service Property for Memory and Concurrent Requests**

To configure how the Data Integration Service manages concurrent requests for SQL data service queries, tune the Data Integration Service properties in the Administrator tool.

The following table describes the maximum heap size property that you can configure for the Data Integration Service process:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Maximum Heap Size       | Amount of RAM allocated to the Java Virtual Machine (JVM) that runs the Data Integration Service. Use this property to increase the performance. Append one of the following letters to the value to specify the units:  
    - b for bytes  
    - k for kilobytes  
    - m for megabytes  
    - g for gigabytes  
    Default is 512 megabytes.  
  **Note:** Consider increasing the maximum heap size when the Data Integration Services needs to process large amounts of data. |

The following table describes the SQL properties for the Data Integration Service process:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum # of Concurrent Connections</td>
<td>Limits the number of database connections that the Data Integration Service can make for SQL data services. Default is 100.</td>
</tr>
</tbody>
</table>

The following table describes the execution options for the Data Integration Service process:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Maximum Execution Pool Size  | The maximum number of requests that the Data Integration Service can run concurrently. Requests include data previews, mappings, profiling jobs, SQL queries, and web service requests. Default is 10.  
  **Note:** When you increase the pool size value, the DTM process of the Data Integration Service uses more hardware resources such as CPU, memory, and system I/O. Set this value based on the resources available on your machine and your data service requirements. For example, consider the number of CPUs on the machine that hosts the Data Integration Service and the amount of memory that is available to the Data Integration Service. |
| Temporary Directories        | Location of temporary directories for Data Integration Service process on the node. Default is <Informatica Services Installation Directory>/tomcat/bin/disTemp. Add a second path to this value to provide a dedicated directory for temporary files created in profile operations. Use a semicolon to separate the paths. Do not use a space after the semicolon. |
Web Service Property to Configure an Active DTM Instance

To increase performance, you can configure the Data Integration Service to keep a DTM instance active so that it can process more than one web service request. You can configure the DTM Keep Alive Time property for the Data Integration Service in the Administrator tool.

The following table describes the DTM Keep Alive Time property:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTM Keep Alive Time</td>
<td>Number of milliseconds that the DTM instance stays open after it completes the last request. Web service requests that are issued against the same operation can reuse the open instance. Use the keep alive time to increase performance when the time required to process the request is small compared to the initialization time for the DTM instance. If the request fails, the DTM instance terminates. Default is 5000. <strong>Note:</strong> The ability to use an existing DTM instance increases performance. The DIS requires additional resources to start a DTM instance for each request. Keeping the DTM active consumes memory. Therefore, users should consider the memory consumption when configuring this option.</td>
</tr>
</tbody>
</table>

Web Services Properties for Memory and Concurrent Requests

To increase performance, configure concurrency and memory properties for the Data Integration process and each web service in the Administrator tool.

The following table describes the HTTP properties for the Data Integration Service Process:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Backlog Request</td>
<td>Maximum number of HTTP or HTTPS connections that can wait in a queue for this Data Integration Service process. Default is 100. You must restart the Data Integration Service to apply your changes.</td>
</tr>
<tr>
<td>Maximum Concurrent Requests</td>
<td>Maximum number of HTTP or HTTPS connections that can be made to this Data Integration Service process. Default is 200. You must restart the Data Integration Service to apply your changes. <strong>Note:</strong> For a web service, this property impacts the number of web service requests that the Data Integration Services accepts before it sends the requests to the Data Integration Service backlog.</td>
</tr>
</tbody>
</table>
The following table describes the maximum heap size property that you can configure for the Data Integration Service process:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Heap Size</td>
<td>Amount of RAM allocated to the Java Virtual Machine (JVM) that runs the Data Integration Service. Use this property to increase the performance. Append one of the following letters to the value to specify the units: - b for bytes - k for kilobytes - m for megabytes - g for gigabytes Default is 512 megabytes. <strong>Note:</strong> Consider increasing the maximum heap size when the Data Integration Services needs to process large amounts of data.</td>
</tr>
</tbody>
</table>

The following table describes the execution options for the Data Integration Service process:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Execution Pool Size</td>
<td>The maximum number of requests that the Data Integration Service can run concurrently. Requests include data previews, mappings, profiling jobs, SQL queries, and web service requests. Default is 10. <strong>Note:</strong> When you increase the pool size value, the DTM process of the Data Integration Service uses more hardware resources such as CPU, memory, and system I/O. Set this value based on the resources available on your machine and your data service requirements. For example, consider the number of CPUs on the machine that hosts the Data Integration Service and the amount of memory that is available to the Data Integration Service.</td>
</tr>
<tr>
<td>Temporary Directories</td>
<td>Location of temporary directories for Data Integration Service process on the node. Default is &lt;Informatica Services Installation Directory&gt;/tomcat/bin/disTemp. Add a second path to this value to provide a dedicated directory for temporary files created in profile operations. Use a semicolon to separate the paths. Do not use a space after the semicolon.</td>
</tr>
<tr>
<td>Maximum Memory Size</td>
<td>The maximum amount of memory, in bytes, that the Data Integration Service can allocate for running requests. If you do not want to limit the amount of memory the Data Integration Service can allocate, set this threshold to 0. When you set this threshold to a value greater than 0, the Data Integration Service uses it to calculate the maximum total memory allowed for running all requests concurrently. The Data Integration Service calculates the maximum total memory as follows: Maximum Memory Size + Maximum Heap Size + memory required for loading program components Default is 512,000,000. <strong>Note:</strong> If you run profiles or data quality mappings, set this threshold to 0.</td>
</tr>
<tr>
<td>Maximum Session Size</td>
<td>The maximum amount of memory, in bytes, that the Data Integration Service can allocate for any request. For optimal memory utilization, set this threshold to a value that exceeds the Maximum Memory Size divided by the Maximum Execution Pool Size. The Data Integration Service uses this threshold even if you set Maximum Memory Size to 0 bytes. Default is 50,000,000.</td>
</tr>
</tbody>
</table>

**Web Service Log Management**

System I/O performance can decrease when the Data Integration Service writes and maintains a large number of log files. The Data Integration Service generates web service run-time logs based on the trace level that you configure. Consider managing the number of log files that the Data Integration Service writes and maintains.

Consider the following solutions for web service log bottlenecks:
Set the web service trace level to off.

When you configure web service properties for a deployed web service, you can specify the log trace level. The trace level determines the types of logs that the Data Integration Service writes to the run-time log location. The default web service trace level is INFO. When the trace level is set to FINEST or ALL, performance can decrease because the Data Integration Service writes more logs to the log file. The performance impact of setting the trace level to FINEST or ALL is the greatest when the web service uses HTTPS and WS-Security.

Archive log files that are no longer required.

System I/O is affected by storing too many log files. By default, the Data Integration Services writes the web service run-time logs in the following directory: `<InformaticaInstallationDir>/tomcat/bin/disLogs/ws`

**Note:** If you delete the ws folder to empty the logs, you must re-create the ws folder. Stop the Data Integration Service before you delete and re-create the ws folder.

**Example Data Integration Service Configuration for Concurrent Web Service Requests**

When you configure how the Data Integration Service processes concurrent web services requests, verify that the value for the maximum number of concurrent requests is the same for the web service and the Data Integration Service process.

For example, in the following configuration the Data Integration Service accepts 200 concurrent HTTP requests but only 10 web service concurrent requests:

<table>
<thead>
<tr>
<th>Property Type</th>
<th>Property Name</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Integration Service Process</td>
<td>Maximum Concurrent Requests</td>
<td>200</td>
</tr>
<tr>
<td>Data Integration Service Process</td>
<td>Maximum Backlog Request</td>
<td>500</td>
</tr>
<tr>
<td>Data Integration Service Process</td>
<td>Maximum Execution Pool Size</td>
<td>100</td>
</tr>
<tr>
<td>Web Service</td>
<td>Maximum Concurrent Request</td>
<td>10</td>
</tr>
</tbody>
</table>

When the Data Integration Service receives 20 web service requests, 10 web service requests fail because the web service can only accept 10 concurrent requests.

To avoid web service requests failing when the web service reaches its maximum number of concurrent requests, configure the same maximum value for the Data Integration Service process and the web service. When the number of requests sent to the Data Integration Service exceeds the maximum concurrent requests value, the additional requests remain in the backlog until the Data Integration Service process is available to process the requests.

**Author**

Informatica Documentation Team