TIBCO Data Virtualization®

SAP Adapter Guide

Version 8.6.0
Last Updated: April 8, 2022

Copyright © 2002-2022. TIBCO Software Inc. All Rights Reserved.

Contents

Using the SAP for TDV Adapter 3

About SAP Data Sources 3
SAP Basic tab 6
SAP Advanced tab 6
Configuring TDV Logging Level when Using SAP for TDV 8
SQL Support for SAP 9
TDV SQL Support for SAP Tables 17
TDV SQL Support for SAP ABAP Queries 21
TDV SQL Support for SAP Dates 24
Security for SAP with TDV 25
Pooled and Cluster Tables 27
SAP Global Properties 29

TIBCO Product Documentation and Support Services 37

How to Access TIBCO Documentation 37
Release Version Support 38
How to Contact TIBCO Support 38
How to Join TIBCO Community 38

Legal and Third-Party Notices 39

Using the SAP for TDV Adapter

This topic describes how to create and introspect an SAP data source.

- About SAP Data Sources, page 3
- SAP Basic tab, page 6
- SAP Advanced tab, page 6
- Configuring TDV Logging Level when Using SAP for TDV, page 8
- SQL Support for SAP, page 9
  - TDV SQL Support for SAP Functions, page 9
  - TDV SQL Support for SAP Tables, page 17
  - TDV SQL Support for SAP ABAP Queries, page 21
  - TDV SQL Support for SAP Dates, page 24
- Security for SAP with TDV, page 25
- Pooled and Cluster Tables, page 27
- SAP Global Properties, page 29

About SAP Data Sources

For installation requirements, see the *TDV Installation and Upgrade Guide*.

See the TDV User Guide Chapter "Configuring Advanced Adapters" section "Installing the SAP Java Connector Library" for more details on where to download SAP JCo connection library and how to install it.

The SAP data source provides access to four types of resources: ABAP Queries, Functions, InfoSets, and Tables:

- ABAP Queries are organized by query area and functional area.
- You can browse functions using two hierarchies: BAPI and RFC. The BAPI hierarchy organizes functions are organized as in the SAP Business Object Repository. The RFC hierarchy organizes functions by SAP Development Class and Function Group.
- InfoSets are organized by query area.
- Tables are organized by the first two characters of their name.

Note: If the SAP system is Unicode, RFC and BAPI resources support Unicode. However, such support for table resources requires SAP Note 758278.

Application Views for SAP

The organization of the views and folders mirrors that of SAP's Business Object Repository (BOR). Field names are aliased to provide human-readable names. In some cases several objects in BOR are synthesized in a single query to produce a more detailed, unified view of the data.

A common pattern employed in application views is that of List and Details. Views with names ending in "List" return a minimal set of columns that serve to uniquely identify an object. The identifiers are passed as arguments into a corresponding view, its name ending in "Details," to produce a more extensive set of columns.

Filter Data from Application Views

There are two ways to filter data from SAP using application views: using a filter provided by SAP itself, or by filtering within TDV. Both methods are used by application views depending on the view and the capabilities of SAP.

SAP Functions

In this document, the word "function" means any RFC-enabled function in SAP. The resources located in the BAPI and RFC folders are all functions for the purposes of query processing.

SAP functions are represented as table resources in TDV to aid interoperability with SQL, but the best way to think of an SAP function is as a stored procedure, with inputs and outputs. Invoking it is a discrete event. SAP functions can have scalar, structure, and table-type parameters. In TDV these are mapped to a single virtual table representing the SAP function. To call SAP functions within SQL, TDV maps the SQL WHERE clause to SAP function inputs, and SAP function outputs are mapped to relational columns.

SAP Tables

SAP tables resemble ordinary relational database tables, with one or more primary keys. SAP foreign keys are not visible in TDV. TDV interprets SQL table queries and translates them into SAP API calls. For joins. TDV pulls necessary data from SAP and processes the query outside of SAP, slowing TDV and SAP performance.

ABAP Queries

ABAP queries are views defined within SAP. An ABAP query in TDV consists of columns with no primary key defined. Columns with names starting with an underscore character are input parameters to the query, and can be used to filter data.

- InfoSets in TDV perform identically to ABAP queries, except that they are not divided by functional area, and during introspection an InfoSet query (ABAP query) is generated in SAP by TDV.

Use in a load balanced environment

If you are using TDV and SAP in a load balanced environment, then you must specify these fields when defining the SAP data source:

- On the Basic tab, specify Client, User, Password, and Language. Application Server, SAP Router String, and System Number are not applicable.
- On the Advanced tab, specify Message Server, System ID, and Logon Group.

When setting up your SAP data source you can define pass-through login information. The operations you can and cannot perform in pass-through mode depend on whether or not Save Password is checked.

Save password ?	Operations you can perform	Operations you cannot perform
Yes	Introspection. You do not have to resupply the password.	N/A
No	• Query/update/insert/delete operations. You need to resupply the original login credentials for the current session. • Re-introspection, Add/Remove data source resources. You will be prompted to resupply the password that was used when the data source was originally introspected.	Schedule reintrospection.

SAP Basic tab

Field	Description
Application Server	Name of the machine hosting SAP or the host machine's IP address. For load-balanced configurations of SAP, leave this property empty.
SAP Router String	Routing entry. If an SAP Router is used to connect to the Application Server, enter its routing string here. Sample: /H/saprouter/H/194.117.106.130/S/3297/H/
System Number	Two-digit system or gateway service number of the SAP system. For load-balanced configurations of SAP, leave this property empty.
Client	Three-digit client number of the SAP system.
User and Password	Valid user name and password to SAP.
Save Password	This option works in combination with the Pass-through Login option. By default, this option is disabled and not editable. It becomes editable when you select the Pass-through Login option. For more details, see the *TDV User Guide*.
Pass-through Login	Works in combination with the Save Password option. By default, this mode is disabled ("non-pass-through mode") and the password is saved. Refer to the description for the Save Password option (above). If you select the Enabled option, the Save Password option becomes editable.
Language	The SAP logon language. Default is EN (English).

SAP Advanced tab

Field	Description
Maximum connections in pool	SAP JCo parameter specifying the maximum number of simultaneous connections in an SAP connection pool.

Field	Description
Maximum idle connections in pool	SAP JCo parameter specifying the maximum number of simultaneous idle connections in an SAP connection pool kept open by the destination. A value of 0 has the effect that there is no connection pooling. That is, connections will be closed after each request.
Table row count	Maximum number of records to retrieve from SAP for any query against a Table resource. The default is 250,000. If queries are taking too long to run, setting this value to 1,000 can help troubleshoot the issue and prevent "runaway queries." Setting this value to 0 means there is no restriction on the number of rows returned. TDV must be restarted for a change to table row count to take effect.
Function thread pool size	Maximum number of threads available to invoke SAP functions in parallel for a single query. Setting this property to a value greater than 0 enables TDV to make function calls in parallel to SAP. This can improve performance when many function calls are required to process a SQL query. For example, if a table containing 10 customer IDs is joined with an SAP function to retrieve customer details by ID, the function will be invoked 10 times. Setting this property to 5 would result in 5 of the function calls being executed simultaneously. The default is 0, which means this feature is disabled and function calls are processed serially.
ABAP Query DB access limit	Maximum number of database accesses SAP allows during execution of queries against ABAP Queries and InfoSets. This does not correlate to number of rows returned, because database access patterns vary significantly between ABAP Queries. In general, reducing the database access limit results in fewer rows returned and provides faster response times. Valid values range from 1 to 99,999,999. The default is 1,000.
Table read function	An SAP function to be used for accessing table resources. Default is RFC_READ_TABLE.
Table row length	Length of data field returned by table read function, in bytes. Default is 512.
Table decimal fix	If checked, SAP Note 758278 or equivalent has been applied to SAP such that the table read function encodes decimal values properly. Default is unchecked.

Field	Description
Table maximum options	Maximum number of filter options that can be sent to SAP when querying a table resource. Default is 500. Filter options are created from the WHERE clause for SQL queries, so this setting is important when tuning queries containing semijoins. The number of options allowed by SAP depends on its version and configuration. Too many options typically results in a core dump error from SAP.
Load-balance configuration options	
Message Server	Host name of the SAP Message Server.
Message Server Port	Port of the SAP Message Server.
System ID	Logical name of the SAP system, also known as R3Name.
Logon Group	Represents a group of SAP Application Servers.

Configuring TDV Logging Level when Using SAP for TDV

The logging level that you set for TDV can have an impact on your disk space. Because of the way SAP and TDV interact, if you set the TDV logging level to the highest level (debug), log files are written to the SAP Adapter's root directory. The files are created for all successful and unsuccessful actions. Capturing this much information can cause your system to run out of space.

To set the logging level

1. Locate the TDV log4j.properties file.
2. Make sure that the following line is commented out (that is, have a hash sign at the beginning):

```
#log4j.logger.com.compositesw.cdms.ds.sap=DEBUG
```

3. Locate and open the product.properties file.
4. Make sure that sap.debug is set to false.

5. Locate the following properties in the product.properties file. Typical installations use the default values of these properties.

Property	Description
sap.debug.freespaceLimitInMB =100	If the usable file system space goes beyond the configured limit, no new debug log files are written. This should prevent file system capacity exhaustion.
sap.debug.cleanUpAtStart=false	When set to true, the first attempt to write one of the debug files, triggers an attempt to clear earlier log files from that directory.

SQL Support for SAP

This section describes TDV SQL support for SAP—SAP resource types (functions, tables, and ABAP queries) and their behavior within TDV SQL queries.

- TDV SQL Support for SAP Functions, page 9
- TDV SQL Support for SAP Tables, page 17
- TDV SQL Support for SAP ABAP Queries, page 21
- TDV SQL Support for SAP Dates, page 24

TDV SQL Support for SAP Functions

The following section describes how SAP functions work within TDV. It is divided into five parts:

- Introspection and SAP Functions, page 9
- Capabilities and SAP Functions, page 11
- Parameter Mapping and SAP Functions, page 13
- Result Mapping and SAP Functions, page 15
- Joins and SAP Functions, page 17

Introspection and SAP Functions

Introspection examines SAP for its list of functions, allows the user to select functions, and translates the function metadata to relational tables that TDV understands. The two sections below explain:

- Resource Hierarchy, page 10
- Metadata Mapping, page 10

Resource Hierarchy

The introspector contains two folders for SAP functions: BAPIs and RFCs. Each folder shows the same set of functions at the leaf nodes, but the hierarchy differs. BAPI hierarchy consists of Applications, Object Types, and Object Methods, which are the functions themselves. RFC hierarchy contains Development Classes, Function Groups, and the functions themselves.

To use an SAP function in TDV it must be RFC-enabled, and the user specified in the SAP data source configuration must be authorized to invoke RFCs. SAP transaction SE37 can be used to determine if a function is RFC-enabled (a "Remote-Enabled Module").

Metadata Mapping

To call an SAP function, the following information is needed:

- Function name
- Import parameters (collection of named input fields)
- Export parameters (collection of named output fields)
- Table parameters (collection of named tabular structures)

In TDV, all import, export, and table parameters become columns in a single, named virtual table that represents the SAP function. The following naming convention is used to map fields of an SAP function to columns of a TDV table:

- — Class—Import or Export. Omitted if the field is a table parameter.
- — Structure—Name of the structure containing the value. Omitted if the value is scalar. Table structure name if the field is a table parameter.
- — Field—Required name of the field.

Each data type in SAP is mapped to a TDV SQL-based type. TDV is limited to types supported in the underlying SAP JCo library.

The following table lists the SAP data types, whether they are supported in TDV, and their corresponding TDV type.

SAP Data Type	ABAP Type	Supported	TDV Data Type
CHAR, UNIT, CUKY, CLNT, LANG, LCHR, VARC	C	Yes	VARCHAR
DATS	D	Yes	DATE
FLTP	F	Yes	DOUBLE
STRING	g	Yes	VARCHAR
INT4, INT2, INT1	I	Yes	BIGINT
NUMC	N	Yes	VARCHAR
CURR, QUAN, DEC, PREC	P	Yes	DECIMAL, NUMERIC
TIMS	T	Yes	TIME
LRAW, RAW	X	Yes	VARCHAR
XSTRING	y	Not supported. Cannot be used in RFC-callable functions.	

Documentation of SAP function and its columns is retrieved from SAP and stored in the Annotation field in TDV.

Capabilities and SAP Functions

Data-source-specific capabilities determine how a SQL statement is to be divided between SAP, TDV, and other data sources that might be referenced in the query. Where a capability is lacking in a data source but required to run a query, TDV typically compensates by performing the work itself.

The following table lists commonly used capabilities and whether they are supported in queries against SAP functions. Pushed indicates whether the capability is supported directly on SAP. For efficient queries, it is best to minimize use of capabilities that cannot be pushed.

Capability	Supported?	Pushed?	Notes
CASE	Yes	No	
DELETE	No	No	Every SAP function handles deleting data differently. Consult SAP documentation before constructing SELECT statements with DELETE.
DISTINCT	Yes	No	
Filter	Yes, with special usage	Yes	All filters are mapped to parameters. (See Parameter Mapping and SAP Functions, page 13.) SAP does not filter output parameters, but by default they are pushed anyway. Workaround: wrap queries to SAP functions.
Filter–BETWEEN	Yes, with special usage	No	See Notes on Filter.
Filter–IN	Yes, with special behavior	Yes	See Parameter Mapping and SAP Functions, page 13.
Filter–LIKE	Yes, with special usage	No	See Notes on Filter.
Functions–aggregate	Yes	No	
Functions–CAST	Yes	Yes	
Functions–others	Yes	No	
GROUP BY	Yes	No	
INSERT	No	No	Every SAP function handles inserting data differently. Consult SAP documentation before constructing SELECT statements with INSERT.
Join	Yes	No	

Capability	Supported?	Pushed?	Notes
ORDER BY	Yes	No	
Subquery	Yes	No	
Transactions	No	No	INSERT/UPDATE/DELETE not supported.
UPDATE	No	No	Every SAP function handles updating data differently. Consult SAP documentation before constructing SELECT statements with UPDATE.

Parameter Mapping and SAP Functions

Parameter mapping is the process of applying SQL filter expressions to the SAP function call. The SQL filter becomes a series of arguments to the SAP function. How this is done depends on the number and types of parameters:

- No Parameters, page 13
- Import Parameters, page 13
- Table Parameters, page 14
- Multiple Parameter Sets, page 14

No Parameters

The following is a valid SQL statement:

```
SELECT * FROM /Shared/DataServicesForSAP_4_6/Sources/SAP/BAPI_CUSTOMER_GETLIST
```

This invokes the SAP function BAPI_CUSTOMER_GETLIST, and all of the values in its import parameters, export parameters, and table parameters are returned. The SAP system on which to execute the function is specified by the data source named SAP in the resource tree folder:

Shared > DataServicesForSAP_4_6 > Sources

Import Parameters

Import parameters are scalar or structure values that represent function input. Import parameters may or may not be required for the function to succeed, depending on the implementation of the function in SAP.

Include a WHERE clause that assigns a value to each import parameter. For example:

```
SELECT * FROM BAPI_COMPANYCODE_GETDETAIL WHERE "Import COMPCODE" = '1000'
```

Values must be provided using the "=" operator. Comparison operators such as "<" and ">" have no meaning and are ignored.

The IN operator is supported, but its behavior is special. See Multiple Parameter Sets, page 14 for more information.

Note: Import parameters are often used to capture object identifiers, for example Customer ID. Some SAP functions require that identifiers be padded to their full length with leading zeros. For example, SAP might not recognize the value '1000' for a field of type VARCHAR(10) unless it is padded to '0000001000'.

Table Parameters

To provide a table parameter with a single row of data, follow the instructions described in Import Parameters, page 13. For example, this SQL invokes the function BAPI_CUSTOMER_GETLIST with one row of data in the table CUSTOMERRANGE:

```
SELECT * FROM BAPI_CUSTOMER_GETCONTACTLIST WHERE "CUSTOMERRANGE SIGN" = 'I' AND
"CUSTOMERRANGE OPTION" = 'BT' AND "CUSTOMERRANGE LOW" = '0000000000' AND
"CUSTOMERRANGE HIGH" = '9999999999'
```

To provide multiple rows of data, use an IN clause for each column. An IN clause on a table parameter is interpreted as a set of input rows for a given column.

Multiple Parameter Sets

An SAP function can be invoked multiple times with a single SQL statement. This can be helpful when a function returns detail on a single object but the query must return detail data for multiple customers in a single result set.

To support this, parameters passed in a WHERE clause are decomposed into a set of invocations of the SAP function, where each invocation is a set of input values. The behavior of the WHERE clause with SAP functions is described below.

- OR designates a new invocation of the function.
- NOT is ignored.
- An IN clause on an import parameter is expanded and treated as a series of OR expressions.
- NOT IN is treated like IN.

- An IN clause on a table parameter is preserved as a set of input rows for a given column. This enables table parameters to be populated with many rows of data, using an IN clause for each column.

The four examples below illustrate different combinations of the rules and their results. The SQL statement is listed first, and then a breakdown of the function invocations that would result.

1. Single invocation, imports (2 fields)

```
SELECT * FROM FOO WHERE "Imports A" = 1 AND "Imports B" = 2
```

Results in an invocation to FOO with Imports A=1, B=2.

2. Multiple invocations, imports (2 fields)

```
SELECT * FROM FOO WHERE "Imports A" = 1 AND "Imports B" = 2 OR
"Imports B" = 3 AND "Imports A" IN (3, 4)
```

Results in three invocations of FOO with the following values:

```
Imports A=1, Imports B=2
Imports A=3, Imports B=3
Imports A=4, Imports B=3
```

3. Single invocation, imports and a table (1 column, 1 row)

```
SELECT * FROM FOO WHERE "Imports A" = 1 AND "FOO_TABLE B" = 2
```

Results in one invocation of FOO with the following values:

```
Imports A=1, FOO_TABLE (row 1): B=2
```

4. Multiple invocations, imports, and a table (2 columns, 2 rows)

```
SELECT * FROM FOO WHERE "Imports A" = 1 AND "FOO_TABLE B" IN (3, 4) AND "FOO_TABLE C" IN (5, 6)
OR "Imports A" = 2 AND "FOO_TABLE B" IN (7, 8) AND "FOO_TABLE C" IN (9, 10)
```

Results in two invocations of FOO, with the following values:

```
Imports A=1, FOO_TABLE (row 1): B=3, C=5 (row 2): B=4, C=6
Imports A=2, FOO_TABLE (row 1): B=7, C=9 (row 2): B=8, C=10
```

Result Mapping and SAP Functions

Result mapping is the process of translating data returned by an SAP function call into a standard database result set with rows and columns. Because data structures within SAP functions are more complex than rows and columns, this is not always a one-to-one mapping.

- Import Parameters, page 16
- Export Parameters, page 16
- Table Parameters, page 16

Import Parameters

Import parameters provided in filters are returned unmodified in each row of the result set.

Export Parameters

Export parameters are returned unmodified in each row of the result set.

Table Parameters

An SAP function can contain many table parameters, which are like export parameters with multiple rows. Every table parameter belongs to a named table structure. An SAP function can contain table parameters from multiple, independent table structures, without a common key to relate them. You can simultaneously select table parameters from more than one of these independent table structures.

When a query selects table parameters from more than one table structure, the table structure containing the greatest number of rows is designated as the primary table structure. Values from other table structures are iterated over in lock-step with the primary table structure. When a table structure has no further rows but the primary table structure contains more rows, null values are output.

For example, table structure A contains table parameters A1 and A2 with three rows, and table structure B contains table parameters B1 with one row:

```
SELECT "A A1", "A A2", "B B1"
```

This statement generates the following results:

```
"A A1" "A A2" "B B1"
------ ------ ------
row1-1 row1-2 row1-1
row2-1 row2-2 <NULL>
row3-1 row3-2 <NULL>
```

The behavior is different when table structures and single-row collections (import and export parameters) are combined. These collections always return a single row, so they are repeated over the number of rows returned by the primary table structure. For example, you can have a statement with export parameters E E1 and E E2 and a table structure T containing table parameter T1 with three rows:

```
SELECT "Export E E1", "Export E E2", "T T1"
```

This query returns the following results:

```
"Export E E1" "Export E E2" "T T1"
-------------- ------------- ------
val-e1        val-e2        row1
val-e1        val-e2        row2
val-e1        val-e2        row3
```

Joins and SAP Functions

This section describes how to join SAP functions with each other and with other resources. You can use a semijoin to combine the output of an SAP function with another SAP function or any TDV resource. Semijoin uses the results of one query as inputs to another query. As described Multiple Parameter Sets, page 14, every unique input set provided to the SAP function resource causes a function call that adds one or more rows to the result. It may be desirable to control the ordering of the function invocation, and thus the rows in the result, by adding NESTEDLOOP as a query hint; for example, {OPTION SEMIJOIN, NESTEDLOOP}.

If SAP function invocation fails, export parameter values are typically available to diagnose the problem. But from the perspective of the TDV join, the right-side resource still has rows to return, and so NULL values may be injected into the result despite the failure of the function.

With a relational data source, an extra condition would be added to the JOIN expression, so that rows containing an error message would be omitted. Because all filters are pushed to SAP, and SAP only allows filtering function by function, and only via specific table and import parameters, you may need to have TDV take over filtering the results to omit error rows. One way to do this is by writing a SQL script to contain the query against the SAP function, and then a view with which to filter out the NULL rows.

A good idea is to test the SAP function in SAP GUI to understand how it operates, and then test it in isolation in a simple TDV view.

TDV SQL Support for SAP Tables

This section describes how SAP tables work within TDV. It is divided into three parts:

- Introspection and SAP Tables, page 17
- Capabilities and SAP Tables, page 20
- Joins and SAP Tables, page 21

Introspection and SAP Tables

The introspection process examines SAP for its list of tables, allows the user to select tables, and translates the SAP table metadata into relational tables for TDV. The important considerations are:

- Resource Hierarchy, page 18
- Metadata Mapping, page 10

Resource Hierarchy

The SAP Adapter provides logical SAP tables from the SAP data dictionary as resources in TDV. Typically, any table that can be accessed via the SAP function RFC_READ_TABLE can be queried from TDV. Exceptions to this rule are discussed in Metadata Mapping, page 10.

SAP tables are introspected by drilling down into a two-level hierarchy of folders. SAP tables are organized into folders based on the first two characters of their name. For example, the SAP table named VBAK is located in V > VB.

Metadata Mapping

When translating metadata from SAP tables to TDV table resources, each column of the SAP table becomes a column in TDV with the same name. Primary key columns in the SAP table become primary key columns in TDV.

Each data type in SAP is mapped to a TDV SQL-based type that is reported in the FIELDS table returned by RFC_READ_TABLE or YRFC_READ_TABLE. TDV reads the table and uses the field lengths listed there.

Note: TDV recommends using the YRFC_READ_TABLE if it is implemented on your SAP system.

The following table lists the SAP data types, whether they are supported in TDV, and their corresponding TDV type.

SAP Data Type Name	ABAP Type	Supported	TDV Data Type
CHAR, UNIT, CUKY, CLNT, LANG, LCHR, VARC	C	Yes	VARCHAR
DATS	D	Yes	DATE
FLTP	F	No – see below	
STRING, XSTRING	g, y	No – see below	
INT4, INT2, INT1	I	Yes	BIGINT
NUMC	N	Yes	VARCHAR
CURR, QUAN, DEC, PREC	P	Yes – see below	DECIMAL, NUMERIC
TIMS	T	Yes	TIME

SAP Data Type Name	ABAP Type	Supported	TDV Data Type
LRAW, RAW	X	Yes – see below	VARCHAR

In addition to the columns, descriptive text is retrieved from SAP and stored in the Annotation field of TDV.

Several limitations occur if you introspect with RFC_READ_TABL:

- By default, columns greater than 512 bytes cannot be read, and are automatically ignored, when introspecting SAP tables.
- In SAP versions 5.0, and 6.0, accessing tables containing a floating point (FLTP) data type are not supported. These tables appear in the introspector but an exception is thrown when they are used. (Floating point values are supported if you use the YRFC_READ_TABLE.)
- RAW and LRAW columns (ABAP Type X) are not encoded properly, and may appear truncated when selected.
- Tables containing columns of STRING or RAWSTRING data type cannot be accessed. These tables appear in the introspector, but an exception is thrown when they are used.
- Some data in decimal columns (known as P (ABAP), Packed, BCD (Binary Coded Decimal), and DEC) appears as DECIMAL with native type Px.y, where x is the number of digits and y is the number of decimal places. DECIMAL(9,5) truncates positive numbers greater than 999.99999 and negative numbers less than -99.99999 because they exceed the nine characters allocated for the field. (YRFC_READ_TABLE can resolve this limitation.)

  To avoid returning silently incorrect data, the SAP Adapter raises an error when it encounters rows that contain truncated values.

TDV recommends using YRFC_READ_TABLE for your SAP data source whenever possible.

SAP note 758278 provides an implementation of YRFC_READ_TABLE that addresses decimal issues. To apply this fix, install the YRFC_READ_TABLE function module on SAP, then modify the following three advanced data source properties:

- Table read function: YRFC_READ_TABLE
- Table row length: 4010
- Table decimal fix: Checked

Capabilities and SAP Tables

Capabilities determine how a SQL statement is divided for execution among SAP, TDV, and other data sources that may be referenced in the query. Capabilities are a system of classifying SQL elements to account for the unique features and limitations of data sources. Where a capability is lacking in a data source but required to run a query, TDV performs the work itself.

The following table lists commonly used capabilities and how they apply to SAP tables. limitations For efficient queries, minimize use of non-push capabilities.

Capability	Supported in SAP?	Pushed to SAP?	Notes
Filter	Yes	Yes	Filter comparing two columns of same table is not allowed by SAP.
Filter–BETWEEN	Yes	Yes	
Filter–IN	Yes	Yes	
Filter-LIKE	Yes	Yes	
Functions–CAST	Yes	Yes	
CASE	Yes	No	
DISTINCT	Yes	No	
Functions–aggregate	Yes	No	
Functions–others	Yes	No	
GROUP BY	Yes	No	
Join	Yes	No	
ORDER BY	Yes	No	
Subquery	Yes	No	
UNION	Yes	No	
DELETE	No	No	SAP tables are read-only.
INSERT	No	No	SAP tables are read-only.

Capability	Supported in SAP?	Pushed to SAP?	Notes
Transactions	No	No	INSERT/UPDATE/DELETE not supported.
UPDATE	No	No	SAP tables are read-only.

Joins and SAP Tables

This section describes how to join SAP tables with each other and with other resources. Joins cannot be pushed to SAP; instead, every row of the joined tables must be fetched (a procedure known as a table scan). The technology used to communicate to SAP from TDV is not optimized for working with large data sets, so table scans should be avoided.

A semijoin is the best way to reduce the number of SAP rows that TDV retrieves and processes. To force a semijoin to occur in a query, add it immediately before the tables to be joined. For example:

```
SELECT * FROM A INNER { option semijoin } JOIN B ON A.K = B.K
```

Values of A.K are collected and passed in a query to B as the filter.

```
SELECT * FROM A
SELECT * FROM B WHERE K IN ({values of A.K from previous query})
```

Many rows in A can result in lengthy queries against B. If queries against B are large, TDV automatically breaks them into smaller queries, executes them separately, and reassembles a unified result set.

Put the larger table on the right side of the join whenever possible. When running a new query for the first time, activate the Execution Plan tab in Studio and click Execute and Show Statistics. Examine each node's row count and query after processing has begun, to make sure filters are pushed down to SAP.

If the interaction between TDV and SAP is still unclear and performance is poor, enable debug logging as described in the *TDV Installation and Upgrade Guide*.

TDV SQL Support for SAP ABAP Queries

The following sections describes how SAP ABAP queries work within TDV:

- Introspection and ABAP Queries, page 22
- Capabilities and ABAP Queries, page 23

Introspection and ABAP Queries

Introspection examines SAP for its list of ABAP queries and InfoSets, allows the user to select them, and translates their metadata to relational tables that TDV understands. The resource hierarchy and metadata mapping of ABAP queries is explained here.

- Resource Hierarchy, page 22
- Metadata Mapping, page 22

Resource Hierarchy

The SAP Adapter provides ABAP Queries and InfoSets as resources in TDV. In general, any ABAP query that can be executed via transaction SQ01 and output using the SAP List Viewer can be accessed through TDV.

ABAP queries are introspected by expanding the ABAP Queries folder within an SAP data source. The first level of folders separates ABAP queries into Global (cross-client) or Standard (client-specific). The second level of folders separates ABAP queries by their functional areas.

Infosets are introspected by expanding the InfoSets folder within an SAP data source. In general, any InfoSet in transaction SQ01 that can be used in an InfoSet query can be accessed through TDV.

Infosets are introspected by expanding the InfoSets folder within an SAP data source. The first level of folders separates InfoSets into Global (cross-client) or Standard (client-specific).

Metadata Mapping

Translating metadata from ABAP queries to TDV ABAP query resources is straightforward. Each output field of the ABAP query becomes a column in TDV with the same name. Selection fields in the ABAP query also become columns in TDV, but their name is prefaced with an underscore character to indicate their role as specifiers of filter constraints to SAP.

Each data type in SAP is mapped to a TDV SQL-based type. The following table lists the SAP data types found in ABAP queries, whether they are supported in TDV, and their corresponding TDV type.

SQL Data Type Name	ABAP Type	Supported in TDV?	TDV Data Type
CHAR, UNIT, CUKY, CLNT, LANG, LCHR, VARC	C	Yes	VARCHAR
DATS	D	Yes	DATE
FLTP	F	Yes	DOUBLE
STRING, XSTRING	g, y	No – not supported by ABAP queries	
INT4, INT2, INT1	I	Yes	BIGINT
NUMC	N	Yes	VARCHAR
CURR, QUAN, DEC, PREC	P	Yes	DECIMAL, NUMERIC
TIMS	T	Yes	TIME
LRAW, RAW	X	No – not supported by ABAP queries	

Capabilities and ABAP Queries

Capabilities determine how a SQL statement will be divided between SAP, TDV, and other data sources that may be referenced in the query.

The following table lists commonly used capabilities and how they apply to SAP ABAP queries. Supported indicates whether or not the SQL capability is supported in queries against ABAP queries. Pushed indicates whether the capability is supported directly on SAP. For efficient queries, minimize use of non-push capabilities.

Capability	Supported?	Pushed?	Notes
CASE	Yes	No	

Capability	Supported?	Pushed?	Notes
DELETE	No	No	ABAP queries and InfoSets are read-only.
DISTINCT	Yes	No	
Filter	Yes	Yes	
Filter–BETWEEN	Yes	Yes	
Filter–IN	Yes	Yes	
Filter–LIKE	Yes	No	
Functions–aggregate	Yes	No	
Functions–CAST	Yes	Yes	
Functions–others	Yes	No	
GROUP BY	Yes	No	
INSERT	No	No	ABAP queries and InfoSets are read-only.
Join	Yes	No	
ORDER BY	Yes	No	
Subquery	Yes	No	
Transactions	No	No	ABAP queries and InfoSets are read-only.
UNION	Yes	No	
UPDATE	No	No	ABAP queries and InfoSets are read-only.

TDV SQL Support for SAP Dates

By default, the SAP application data service throws an exception and aborts a query if erroneous dates are returned from the SAP system.

To override SAP date behavior

1. Uncomment the following line in the product.properties file:
sap.handle.erroneous.date=USE_DEFAULT sap.handle.erroneous.date.default=99991231

2. To enable the properties, edit product.properties at:
<TDV _install_dir>\apps\dlm\app_ds_sap\conf\product.properties
3. The date format is YYYYMMDD. If no default date is specified, the date will be 19000101. If the specified default date is invalid, an exception is generated.

Security for SAP with TDV

This topic describes TDV support for SAP security features. It assumes knowledge of SAP's security infrastructure.

- Required Authorizations, page 25
- Troubleshooting Security-Related Errors, page 26
- Custom Security on SAP Tables, page 27

Required Authorizations

The following authorizations are required:

- To log in to SAP from TDV and introspect BAPIs and RFCs, the following authorization object is required:

  Class: AAAB (Cross-application Authorization Objects, Object: S_RFC (Authorization Check for RFC Access

  This object controls access to RFCs by function group. The most restricted configuration of S_RFC is to allow access to only the function groups required by TDV, then add new groups as necessary.
- To log in to SAP from TDV, the following function groups are required:

  SYST (System interface)

  SYSU (RFC resource administration)

  SDIFRUNTIME (Interfaces for Type Runtime Objects)

  RFC1 (RFC utilities)
- To introspect SAP metadata, the following function groups are required:

  SEM5 (Generic Browser for ABAP/BOR classes)

  SWOR (Runtime System)

  SDTX (Desktop Access)

- To introspect and query SAP ABAP queries and InfoSets, the following function groups are required, in addition the authorizations above:

  AQCF (SAP Query: Catalog functions)

  AQRC (SAP Query: Remote query call)

- To introspect and query SAP tables, in addition the authorizations above, the following authorization object is required:

  Class: BC_A (Basis: Administration), Object: S_TABU_DIS (Table Maintenance)

Troubleshooting Security-Related Errors

Some of the common security-related errors and their remedies are described below.

- RFC Authorization, page 26
- QUERY_TABLE Authorization, page 26

RFC Authorization

ERROR: User TESTUSER1 has no RFC authorization for function group SYST.

TDV is logging into SAP with a SAP user that lacks the authorization object S_RFC. Set authorization S_RFC_ALL to grant access to all RFCs.

S_RFC grants the user permission to introspect and invoke functions. But functions may still fail unless the appropriate authorizations for its parent application are granted. Depending on the function, application-level authorization errors may be reported in a return code. For example, invoke BAPI_COMPANYCODE_GETLIST without the proper application-level authorization and the field RETURN MESSAGE contains:

> You do not have authorization to display company codes.

QUERY_TABLE Authorization

ERROR: com.sap.mw.jco.JCO$AbapException: (126) NOT_AUTHORIZED: User not authorized to access QUERY_TABLE.

TDV is logging into SAP with a SAP user that lacks the authorization object S_TABU_DIS. Set authorization S_TABU_ALL to allow display of all table classes, or limit access to specific groups of tables using the appropriate authorizations.

Custom Security on SAP Tables

The granularity of access control provided by S_TABU_DIS may be too coarse for some security requirements. For example, SAP administrators may prefer to allow access table by table to TDV users. In this case the authorization check on S_TABU_DIS in RFC_READ_TABLE is insufficient.

TDV supports the ability to call a custom implementation of RFC_READ_TABLE. This is accomplished by editing the data source to set the Table read function and Table row length to match the function name and field size of the custom function. The column names and data types in the custom function must be identical to those in RFC_READ_TABLE.

Pooled and Cluster Tables

This section describes TDV SQL support for SAP pooled and cluster tables.

- Understanding Pooled and Cluster Tables, page 27
- HR Clusters, page 29

Understanding Pooled and Cluster Tables

SAP uses three approaches to table storage:

- Transparent tables are stored as regular tables in the back-end database.
- Pooled tables are several logical tables stored as a single table, or "table pool," in the database.
- Cluster tables are logical tables stored as a single table, or "table cluster," in the database.

The difference between table pools and table clusters is that the component tables of a table cluster share common keys, while component tables of a table pool do not.

The TDV can access all three types of SAP tables and all views based on them.

Table Type	Table Class	Supported?	Examples
Transparent	TRANSP	Yes	KNA1, AFKO
Cluster	CLUSTER	Yes	BSEG, BSET
Pooled	POOL	Yes	A001, FINK
View	VIEW	Yes	V_T001, CUSPROJ

Some tables in SAP have misleading names. For example, “HR Cluster 1” (PCL1) is actually a transparent table. To ascertain the storage type of a table, go to the ABAP Dictionary (transaction code SE11), provide the name of the table, and display its technical settings.

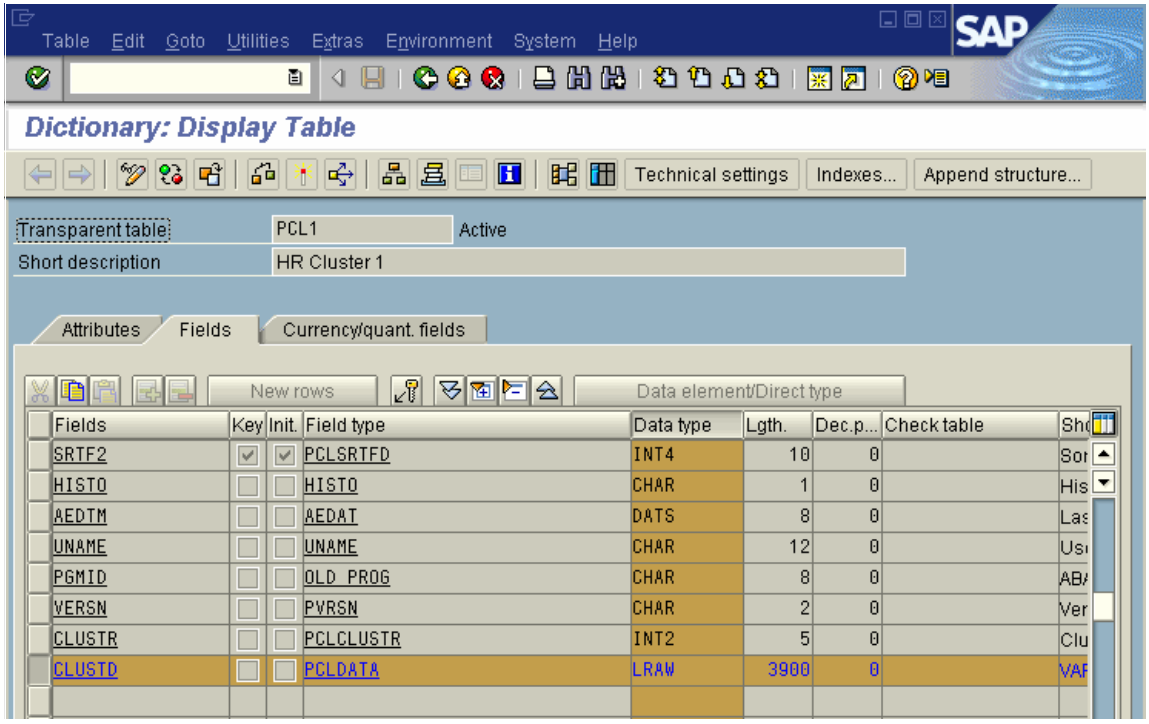
Working with the definitions provided above, TDV is able to access all three types of tables: transparent, pooled, and cluster. They all look to TDV like any other table resource. However, pool and cluster tables are more likely to contain columns that TDV cannot read:

Types LRAW and RAW—Due to limitations of the SAP function RFC_READ_TABLE, which TDV uses to access SAP tables, the column types RAW and LRAW are not properly handled.

Columns greater than 512 bytes—RFC_READ_TABLE does not support any type of columns that are greater than 512 bytes. Such columns are ignored during introspection, and a WARNING message is added to the log.

HR Clusters

HR functional areas contain several transparent tables with columns of type RAW and LRAW. Many have “cluster” in their names even though they are not cluster tables. “HR Cluster 1” (table PCL1) is one example. The inaccessible CLUSTD column is highlighted below.



Because SAP is unable to provide TDV access to RAW columns or columns of this length, administrators must write custom BAPIs to expose their data. TDV can then access the custom functions like any other resource.

SAP Global Properties

Global properties apply to all instances of an SAP data source. You can edit the properties to control how TDV works with SAP. Global properties are stored in the file:

<TDV_install_dir>\apps\dlm\app_ds_sap\conf\product.properties

Edit the properties using a text editor such as Notepad on Windows. TDV must be restarted for property changes to take effect.

The global properties are described below in alphabetical order.

sap.annotations.enabled

Default Value: true

Description:

If set to false, introspection of functions (BAPIs and RFCs) will not include their annotations.

sap.aq.introspectDbAccesses

Default Value: 1

Description:

ABAP Query database access limit to be used during introspection.

During introspection of ABAP queries, each introspected query is executed with no parameters to obtain a list of its output fields. (If this is not possible, other metadata query methods are used). Because the goal is to gather metadata only, the number of database accesses should be set to the smallest value that still results in at least one row.

If the database access limit is set too low, no data is selected from the ABAP query and the fallback method of metadata access is used, which may not include some calculated fields.

If problems are encountered with ABAP query metadata, test the ABAP query using transaction SQ01 until it returns rows, then increase this property value to match.

sap.debug

Default Value: false

Description:

If set to true, all data returned from SAP requests is logged to CSV files in the SAP driver folder. Each request becomes a separate CSV file. This can be helpful in debugging joins and performing data validation.

sap.feature.key

Default Value: BRT

Description:

Do not change this. TDV uses it internally for SAP certification.

sap.handle.erroneous.date

Default Value: USE_DEFAULT

Description:

Commented out by default. If enabled, specifies how to handle erroneous dates returned from the SAP server. SE_DEFAULT tells the connector not to throw an exception, but instead to use the default date specified by sap.handle.erroneous.date.default, page 31 (the SAP default property for handling erroneous dates).

sap.handle.erroneous.date.default

Default Value: 99991231

Description:

Commented out by default. If enabled, specifies the default date to use if an erroneous date is returned by the SAP server.

sap.introspect.tableFolders

Default Value: T5,T7,V_,/B,/BI,/BIC,/BI0,/BIC/,/BI0/,/BIC/B,/BIC/D,/BI0/D

Description:

Comma-separated list of table folder name patterns to contain subfolders.

During introspection, SAP tables are split by default into two levels of folders by name. For example, folder A contains AA, AB, AC, and so forth.

Some table name patterns may contain so many tables that SAP throws an exception when opening them. This property allows such name patterns to be expanded into an additional level of subfolders. For example, expanding T5 results in further folders T5A, T5B, etc.

sap.jco.traceLevel

Default Value: 2

Description:

Turns on the SAP JCo trace. Allowed levels are 0 through 10. The commonly used ones are:

0 - nothing

1 - errors

2 - errors and warnings

3 - info messages, errors and warnings

4 - execution path, info messages, errors and warnings

5 - verbose execution path, info messages, errors and warnings

6 - verbose execution path, limited data dumps, info messages, errors and warnings

7 - full execution path, data dumps with metadata, verbose info messages, errors and warnings

8 - full execution path, full data dumps with metadata, verbose info messages, errors and warnings

sap.table.decimalFix

Default Value: false

Description:

If true, SAP Note 758278 or equivalent has been applied to SAP so that the table read function encodes decimal values properly.

sap.table.functionName

Default Value: RFC_READ_TABLE

Description:

An SAP function to be used for accessing table resources.

sap.table.maxOptions

Default Value: 500

Description:

This property should never be changed.

sap.table.memoryAvailable

Default Value: 50

Description:

Memory available for SAP table queries, in megabytes. This is a critical property for performance tuning of SAP table access.

This property determines how many table rows can be requested from SAP in one round-trip, and thus how many round-trips to SAP are required to satisfy a user query. Each round-trip can only fetch the permitted number of rows, based on this memory constraint.

To calculate the number of rows per request, take available memory after applying sap.table.requestFactor and divide by two times the table row length. To examine requests and the number of rows in each, enable SAP driver. The ROWCOUNT parameter is the number of rows TDV is requesting from SAP.

If this property is set too high, the JVM hosting TDV can terminate abruptly when SAP's client connectivity library (JCo) attempts to allocate too much memory from its native code. The main purpose of this property is to guard TDV against these low memory conditions.

If this property is set too low, queries are split into many small requests to SAP, resulting in poor performance. With many concurrent requests, the amount of memory available may fall so low that the rows requested fall below the property sap.table.minimumSize. In that case, queries will block until requests are finished and memory is freed. If memory is still not available after a period of time (property sap.table.memoryMaxWait), the query is terminated with an exception.

The maximum heap size of the JVM (-Xmx setting) must be considered when changing this property value. If this value is increased without a corresponding reduction in the Java heap size, JVM crashes could result.

For example, a machine with 2 GB of RAM, a Java heap size of 600 MB permits a sap.table.memoryAvailable setting of up to 200 MB. Reducing the Java heap size to 500 MB enables an increase to 300 MB.

sap.table.memoryMaxWait

Default Value: 60

Description:

Amount of time to wait (in seconds) for memory to become available for an SAP request. If there is insufficient memory available for greater than this time period, an exception is thrown and the query is terminated.

The exception text is:

Could not reserve memory for request {0}: lowerBound = {1}, upper bound = {2}, reserved = {3}, total {4}.

The five variables in the message (in order from 0 to 4) are: number of concurrent requests, minimum size of query (bytes), maximum query size (bytes), amount of memory reserved (bytes) globally for SAP table access, and amount of memory available (bytes) globally for SAP table access.

If queries are being terminated with this error message, decrease Java heap size and increase sap.table.memoryAvailable, or optimize SAP queries to avoid table scans.

sap.table.minimumSize

Default Value: 500

Description:

Minimum number of rows allowed for SAP table requests. When the system is in a low memory state, requests become fragmented, resulting in more round-trips to SAP to fetch data. Splitting a query into too many small requests typically degrades performance more than waiting for enough available memory to make fewer, larger requests.

This property forces queries to block until other queries release memory. If a query blocks for longer than sap.table.memoryMaxWait (seconds), an exception is thrown.

sap.table.profiler.enabled

Default Value: false

Description:

TDV does not use this.

sap.table.requestFactor

Default Value: 0.8, 0.25, 0.15, 0.1

Description:

Comma-separated list of values used to determine how much memory is available for reading SAP tables based on the number of concurrent requests. The number of concurrent requests is used as an index into this list. If this number is greater than the number of values, the last value is used.

For example (using the default value): The first request receives 90% of the available memory. If a second request is made before the first completes, it receives 50% of the remaining memory.

sap.table.rowLength

Default Value: 512

Description:

Length of data field returned by table read function, in bytes.

sap.table.safetyFactor

Default Value: 0.9

Description:

TDV does not use this.

sap.use.logon.lang.introspection

Default Value: false

Description:

If false, the SAP connection uses the language configured in the data source. If true, the SAP connection uses the language set in the SAP connection attributes, which originate from SAP's configuration of the SAP user.

TIBCO Product Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join the TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the TIBCO Data Virtualization page.

- Users
  - TDV Getting Started Guide
  - TDV User Guide
  - TDV Web UI User Guide
  - TDV Client Interfaces Guide
  - TDV Tutorial Guide
  - TDV Northbay Example
- Administration
  - TDV Installation and Upgrade Guide
  - TDV Administration Guide
  - TDV Active Cluster Guide
  - TDV Security Features Guide
- Data Sources
  - TDV Adapter Guides
  - TDV Data Source Toolkit Guide (Formerly Extensibility Guide)
- References
  - TDV Reference Guide
  - TDV Application Programming Interface Guide

- **Other**

  TDV Business Directory Guide

  TDV Discovery Guide

- *TIBCO TDV and Business Directory Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Release Version Support

TDV 8.5 is designated as a Long Term Support (LTS) version. Some release versions of TIBCO Data Virtualization products are selected to be long-term support (LTS) versions. Defect corrections will typically be delivered in a new release version and as hotfixes orservice packs to one or more LTS versions. See also

https://docs.tibco.com/pub/tdv/general/LTS/tdv_LTS_releases.htm.

How to Contact TIBCO Support

Get an overview of TIBCO Support. You can contact TIBCO Support in the following ways:

- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support website.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to TIBCO Support website. If you do not have a user name, you can request one by clicking **Register** on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the TIBCO Ideas Portal. For a free registration, visit TIBCO Community.

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, TIBCO logo, Two-Second Advantage, TIBCO Spotfire, TIBCO ActiveSpaces, TIBCO Spotfire Developer, TIBCO EMS, TIBCO Spotfire Automation Services, TIBCO Enterprise Runtime for R, TIBCO Spotfire Server, TIBCO Spotfire Web Player, TIBCO Spotfire Statistics Services, S-PLUS, and TIBCO Spotfire S+ are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (https://www.tibco.com/patents) for details.

Copyright © 2002-2021. TIBCO Software Inc. All Rights Reserved.