 **tEXASCDELT**

**Function http://www.cimt-ag.de**

tEXASCDELT reflects and tracks changes in a dedicated EXASOL database.

**Purpose**

tEXASCDELT creates and carries out SCD transformations from a source to a target table in an EXASOL database. This is an ELT component which means that all operations are executions of statements inside the database.

**Talend-Integration**

This component can be found in the palette under Databases -> EXASolution and

Business Intelligence -> DB SCD.

**Parameters**

**Connection configuration:**

|  |  |
| --- | --- |
| **Property** | **Content** |
| Property Type | Choose database connection from metadata or use build-in mode to setup individual configurations.  Built-in: No property data stored centrally. Enter properties manually.  Repository: Select the repository file where Properties are stored. The fields that come after are pre-filled in using the fetched data. |
| Use an existing connection | True: choose an existing connection component in your job  False: configure components own connection. |
| Host | Host (IP address or hostname) of your EXASOL server. |
| Port | Port where the instance is listening. Default is 8563. |
| Additional JDBC Parameters | Semicolon separated list of key=value pairs with JDBC parameters.  The default pair in this component is: retrieveMessagesFromServerOnGetMessage=true which cause in case of errors a readable error message instead of getting only the SQLCODE. |
| Database | The database you want to work with |
| Source DB Schema | The database schema containing the source table. |
| Username | User name  User authentication data for a dedicated Exasol database. |
| Password | Password of the user  To enter the password, click the [...] button next to the password field, and then in the pop-up dialog box enter the password between double quotes and click OK to save the settings. |

**Source configuration:**

|  |  |
| --- | --- |
| **Property** | **Content** |
| Overwrite source schema from existing connection | Change schema when using an existing connection. Visible only if “Use an existing connection” is enabled. |
| Source table | Name (without schema) of the input / source table (sometimes also called “staging table”). |
| Where condition | The condition to select the source datasets. Use this if you don’t want the complete source to be processed. The keyword *where* is not needed here (it will be added in the SQL code generation in case this attribute is not empty). |
| Schema from input table | Talend Schema chooser for the input table.  The schema is a row description. It defines the number and names of fields (columns) to be processed and passed on the SCD table. |
| Enable ignoring source columns | Allows to mark columns as ignored, i.e. they will not be in target schema. |
| Enable SCD type 1 | Allows setting SCD type 1 for individual columns. |
| Enable SCD type 2 | Allows setting SCD type 2 for individual columns. If selected, the target schema contains columns marking the valid time period of rows (time slices). |
| Enable SCD type 3 | Allows setting SCD type 3 for individual columns. If SCD 3 is selected but SCD 2 is not selected, target schema contains one column marking the time of the last update of an SCD 3 value. |
| Use self-defined source key | The component can take the keys from the input schema. In case there are no keys defined in this schema (e.g. the source table is probably a view) it is possible to define the keys which are part of the source key. It is necessary to have at least one input field declared as source key. Source key field will not be updated and will be used to find a single unique dataset. |
| ‘Valid from’ timestamp in source column | If this option is checked, a source column can be selected that contains the timestamp representing the beginning of the ‘valid time’ period. |
| Supplementary Source Column Config | Per source column you can specify:  *Ignore:* If true the column will be ignored. This is helpful if you prefer using a schema from the repository and it contains not needed columns.  *Is source key*: Check all columns which identify a unique source dataset. Visible only if option “Use self-defined source key” is enabled.  *Timestamp:* If true the column will be used to choose the ‘valid from’ time. Visible only if option “‘Valid from’ timestamp in source column” is enabled. Must not be selected for more than one column.  *SCD type 1(only current)*: Check all columns that shall have this SCD type. Visible only if option “Enable SCD type 1 columns” is enabled.  *SCD type 1(incl. historic)*: Check all columns that shall have this SCD type. Visible only if option “Enable SCD type 1 columns” and “Enable SCD type 2 columns” are enabled. The difference to the first SCD type 1 is, that this also updates historic SCD 2 time slices.  *SCD type 2*: Check all columns that shall have this SCD type. Visible only if option “Enable SCD type 2 columns” is enabled.  *SCD type 3*: Check all columns that shall have this SCD type. Visible only if option “Enable SCD type 3 columns” is enabled. SCD 3 columns must have an additional column defined. |
| Use this value for valid-timestamp | Set a default time value for rows that don’t have a timestamp defined. Must be in format of “yyyy-MM-dd HH-mm-ss.SSS” with time part being optional (it is padded with zeros if missing). If no value is set system time will be used. |

**Target configuration:**

|  |  |
| --- | --- |
| **Property** | **Content** |
| Target DB Schema | Database schema for the target table. |
| Target table | Name (without schema) of the target table. |
| Create table if not exists | If enabled and the table does not exist, it will be created. |
| Create primary key | If enabled and the table is created, all key columns (and ‘valid from’ column if SCD type 2 is enabled) are used as primary key. |
| Create surrogate key | If enabled and the table is created, a surrogate key column (identity) is added. |
| Surrogate Key column | Column for the surrogate key |
| Use current flag | If enabled, target table contains a column of type boolean with value ‘true’ for time slices that are valid. Visible only if option “Enable SCD type 2 columns” is enabled. |
| Column for current flag | Column for current flag. Visible only if options “Enable SCD type 2 columns” and “Use current flag” are enabled. |
| Use version numbers | If enabled, target table contains a column of type decimal(10,0) which contains version numbers for time slices. Visible only if option “Enable SCD type 2 columns” is enabled. |
| Column for version | Column for version numbers. Visible only if options “Enable SCD type 2 columns” and “Use version numbers” are enabled. |
| Start with version number | The number for initial versions (usually 1 or 0). Visible only if options “Enable SCD type 2 columns” and “Use version numbers” are enabled. |
| Valid time-period start column | Name of the column holding the timestamp for start of valid time-period. Visible only if option “Enable SCD type 2 columns” is enabled. |
| Valid time-period end column | Name of the column holding the timestamp for end of valid time-period. Visible only if option “Enable SCD type 2 columns” is enabled. |
| Value for end column of current data | The value that is used in “valid time-period end column” for currently valid time slices. Visible only if option “Enable SCD type 2 columns” is enabled. |
| Time of last SCD 3 change column | Column that contains timestamp for last update of an SCD type 3 column. Visible only if option “Enable SCD type 2 columns” is disabled and option “Enable SCD type 3 columns” is enabled. |
| Die on error | If selected, causes the job to die on errors happening during the execution of this component. |

**Advanced settings**

|  |  |
| --- | --- |
| **Property** | **Content** |
| Set different schema for temp table | Specify the schema where the temporary data for the scd transformation is stored. Default value is target schema. |
| Execute nothing mode | Statements are generated but not executed. Use for tests. |
| Debug mode | Will cause statements and other information to be written to the console. |
| Log statements | Write all statements to a special log file. |
| Statement log file | File for special log. |

**Return values / Global Variables**

To fill up a field or expression with a variable, press Ctrl + Space to access the variable list and choose the variable to use from it.

|  |  |
| --- | --- |
| **Return value / Variable** | **Content** |
| ERROR\_MESSAGE | Last error message |
| COUNT\_SCD2\_INSERT\_NEW\_RECORDS | Number of inserted records with new business key. Only used if columns of SCD type 2 are defined. |
| COUNT\_SCD1\_CHANGED\_RECORDS | Number of updates for SCD type 1 changes. If no columns of SCD type 2 are defined this will also contain inserts of new records (for optimal performance this happens in one statement and can’t have separate counters). |
| COUNT\_SCD2\_CHANGED\_RECORDS | Number of changed records for SCD type 2. |
| COUNT\_SCD3\_CHANGED\_RECORDS | Number of changed records for SCD type 3 |
| COUNT\_OUTDATED\_RECORDS | Number of detected outdated records in source. (Outdated records can’t always be detected with SCD method and may then cause inconsistent data.) |

**Additional information**

If a column has more than one selection, the following sequence determines precedence: Ignore, key, timestamp, SCD.

It is possible to give one column multiple SCD types. Some combinations make sense (e.g. types 2 and 3) while others don’t (but they don’t cause errors in execution). If no type is selected, column has SCD type 0 (i.e. it will only be used for new records or new time slices).

Multiple updates with the same key are only permitted if the source contains a timestamp. Updates will then automatically execute in the right order. If no timestamp column is specified, the SCD transformation is cancelled to prevent inconsistent data. (Multiple updates with the same key negatively impact performance because they must be processed separately. Runtime can multiply by the highest number of duplicates for a single key in the source.)

Outdated updates (i.e. records for which a newer update has been processed) should be prevented by the user as they may cause inconsistent data. The SCD method doesn’t keep a full history of all seen records so it is impossible to always detect outdated updates automatically. A possible way for the user to avoid this problem is a “pre-staging area” from which data is moved to the “staging area” when it is certain that no later updates will arrive.

Performance depends far more on the size of the target table than on the size of the source. Few large updates usually take less total time than many small updates. This behaviour may change for a very large source depending on the used database system.

Errors may occur for several reasons. Please check this list first:

* At least one field must be selected as key.
* At most one field must be selected as timestamp.
* Columns must not have names of words reserved by the database system.
* The target schema must exist.
* The target table must exist if auto-create is disabled.
* The target table can’t be created automatically if length or precision is required for a column but undefined in the schema.
* If the target table already exists it must be compatible with all settings in the component.
* The user (that is selected in connection configuration) needs privileges to read from source, write to target and create a table for temporary data.
* Multiple records in the source with the same key are only allowed if the source contains a timestamp column.
* SCD type 3 columns must have an additional column defined (in supplementary source column config).
* Default values in the component can be substituted with other valid values, but invalid values (e.g. different data type or null) may cause errors.

**Changelog**

1.2:

* Timestamps in source can be provided even without SCD 2 or 3. This is necessary when input contains key duplicates.
* New option (Basic settings): Overwrite source schema from existing connection
* New option (Advanced settings): Set different schema for temp table (default is same as target schema)
* Output of SQLExceptions on console