

Document Version: 1.0 – 2018-06-01

Saving a History of Source System Changes in the Target System

SAP Landscape Transformation Replication Server SP11 and Higher



Typographic Conventions

Type Style	Description
<i>Example</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Textual cross-references to other documents.
Example	Emphasized words or expressions.
EXAMPLE	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
Example	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
Example	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example>	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE	Keys on the keyboard, for example, F2 or ENTER.

Document History

Version	Date	Change
1.0	2018-06-01	Initial Version

Table of Contents

1	Introduction	5
2	Using History Column Tables in SAP HANA	6
2.1	Overview	6
2.2	Manual Actions	6
2.3	Example	6
3	Using Event-Based Rules.....	8
3.1	Overview	8
3.1.1	Adjusting the Target Table Structure.....	8
3.1.2	Creating the Include Program	9
3.1.3	Creating the Event-Based Rule.....	10
3.2	Example	11
3.2.1	Adjusting the Target Table Structure.....	11
3.2.2	Creating the Include Program	11
3.2.3	Creating the Event-Based Rule.....	13
4	Using Templates to Handle Multiple Tables	15
4.1	Overview	15
4.1.1	Creating the Template	15
4.1.2	Configuring the Template	15
4.1.3	Assigning the Template to the Configuration.....	16
4.1.4	Result.....	17
4.2	Example	17
4.2.1	Creating the Template.....	17
4.2.2	Configuring the Template.....	17
4.2.3	Assigning the Template to the Configuration.....	19

1 Introduction

SAP Landscape Transformation Replication Server facilitates the real-time replication of data from a source system to a target system. After an initial load of data, only delta data (which is captured by database triggers and logging tables) is replicated to the target system. The default behavior is that any insert, update, or delete operation in the source system is also transferred as an insert, update, or delete operation in the target system.

However, there might be use cases where you want to keep a history of the data changes in the target system as you want to be able to access a history of changes for a specific time period. In this case, you need to make sure, that any insert, update, or delete operation in the source system is transferred to the target system as an insert operation ("insert only").

This guide outlines two solutions to do this. The first one is covered in [chapter 2](#) and makes use of **SAP HANA History Column Tables**. This option is only relevant for SAP HANA target systems. By making use of the time travel feature in your queries, you will be able to access the data of a table with reference to a certain point in time.

The second way is more generic and uses an **Event-Based Rule** ([chapter 3](#)). You will have to adopt the target structure of your table by adding additional fields, that store the timestamp and type of an operation. The event-based rule will ensure that those fields are populated correctly.

Both solutions can also be realized by using **templates**, which is described in [chapter 4](#). This approach keeps manual effort to a minimum, which is especially helpful if there are multiple tables in scope.

2 Using History Column Tables in SAP HANA

2.1 Overview

If the target system is based on SAP HANA, you can use History Column Tables to change the default writing behavior to "insert only". The History Column Option creates a column that enables the use of the time travel feature. For more information on SAP HANA History Column Tables and how they are used in SQL statements, see [HISTORY COLUMN Option \(Time Travel\)](#) on the SAP Help Portal.

2.2 Manual Actions

1. In the SAP Landscape Transformation Replication Server System, create a configuration and specify all settings as usual that are required for your project.
2. Open transaction `LTRS` and choose your configuration.
3. Right-click [Table Settings](#) and choose [Add Table](#). Enter the name of the database table for which you would like to apply the "insert only" behavior.
4. Under [Table Properties](#), change the value of the [Table Store](#) field from [Column \(default\)](#) to [History Column store](#).
5. Choose [Save](#).

Note

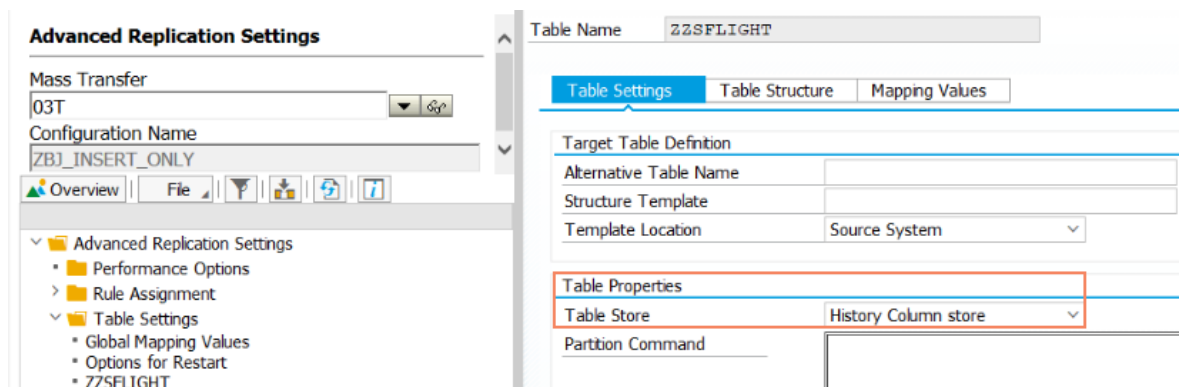
Ideally, the steps above are done before replication starts. If the configuration is however already in replication, it need to be stopped and restarted after the steps above are complete.

After your table has been replicated to the target system, you will be able to read data out of your table specifying a time frame in your query. Therefore, your select statement needs to include the `<time_travel>` syntax element. For example, it could look like the following:

```
SELECT * FROM <tablename> AS OF UTCTIMESTAMP <timestamp>
```

2.3 Example

The following graphic shows a configuration for table ZZSFLIGHT. The value of the field [Table Store](#) has been changed to [History Column store](#).



Change of Table Store to History Column store

3 Using Event-Based Rules

3.1 Overview

If the target system is not an SAP HANA system, or if there are additional requirements for the "insert only" behavior that cannot be fulfilled with History Column Tables, event-based rules can be used. This solution allows more flexibility, as inside the rule there will be an include program called where any custom logic to transform or filter data can be applied.

In order to identify the latest inserted dataset for a key field in a source table, there needs to be an additional key field added that tracks the timestamp of the insert operation in the target system. Optionally, an additional field can be added, for instance a non-key field that captures the original operation from the source system (insert, update or delete operation). In a second step, the include program needs to be created. The include program changes any writing operation to an insert operation, and populates the additional fields accordingly. Finally, the include program is assigned to the rule.

3.1.1 Adjusting the Target Table Structure

1. In the SAP Landscape Transformation Replication Server system, create your configuration and specify settings as usual that are required for your project.
2. Open transaction `LTRS` and choose your configuration.
3. Right-click *Table Settings* and choose *Add Table*. Enter the name of the database table for which you would like to apply the "insert only" behavior.
4. Under *Table Structure*, on the right-hand side, choose *View Modified Table Structure* and add an additional key field to your target structure by clicking *Add Field* (on the left-hand side). As an example, you could insert the following values:

Field	Value
<i>Field Name</i>	SLT_UPDATE
<i>Position</i>	5
<i>Data Type</i>	DEC
<i>Length</i>	15

5. Choose *Confirm*.
6. Optionally, add an additional non-key field to the target structure. For example, you could insert the following values:

Field	Value
<i>Field Name</i>	SLT_OPERAT
<i>Position</i>	16
<i>Data Type</i>	CHAR
<i>Length</i>	1

This field will later be populated with "I" for insert, "D" for delete or "U" for update. Choose [Confirm](#).

7. Save your changes.
8. The system displays the modified target table structure on the right-hand side.

Note

The field values proposed in steps 4 and 5 are only examples based on the `SFLIGHT` table structure. Note that any key field needs to be positioned immediately after (or before) a key field so that the new field also becomes a key field. Non-key fields need to be inserted after a non-key field.

3.1.2 Creating the Include Program

1. In transaction `SE38`, enter an appropriate name for the include program, for example **Z_INSERT_ONLY**, and choose [Create](#).
2. Ensure, that the value of the field *Type* is **Include program**, that the value of the field *Status* is **Test Program**.
3. Save and choose a [Package](#).
4. The system displays the include program in the ABAP Editor.
5. Implement your custom logic. You can use the code example below, and adjust if required.

```
*&-----*
*&  Include          ZSLT_INSERT_ONLY
*&-----*

DATA: lv_name(30) TYPE c,
      lv_operat_src TYPE char1.

FIELD-SYMBOLS: <ls_record>      TYPE any,
                <lv_operat_trg> TYPE any,
                <lv_operat_src> TYPE any,
                <lv_update>     TYPE any.

* get target structure
```

```

CONCATENATE ' _WA_R_ ' _COBJ_ALIAS INTO lv_name.
ASSIGN (lv_name) TO <ls_record>.
ASSIGN COMPONENT 'IUUC_OPERAT_FLAG' OF STRUCTURE <ls_record> TO <lv_operat_trg>.

IF sy-subrc = 0.
    lv_operat_src = <lv_operat_trg>.

*** change operation to insert
    <lv_operat_trg> = 'I'.

*** set additional target fields
    ASSIGN COMPONENT 'SLT_OPERAT' OF STRUCTURE <ls_record> TO <lv_operat_src>.
    If sy-subrc = 0.
        <lv_operat_src> = lv_operat_src.
    ENDIF.

*** set updated timestamp
    ASSIGN COMPONENT 'SLT_UPDATE' OF STRUCTURE <ls_record> TO <lv_update>.
    IF sy-subrc = 0.
        GET TIME STAMP FIELD <lv_update>.
    ENDIF.
ENDIF.

```

6. Save and activate.

3.1.3 Creating the Event-Based Rule

1. In transaction `LTRS`, right-click *Rule Assignment* and choose *Add Table*. Enter the name of the database table for which you would like to apply the "insert only" behavior.
2. Choose *New* and select *Event-Based Rule*.
3. As *Target Event* choose *BOR Begin of Record* and confirm.
4. Choose *Save*.
5. Double-click the entry in the *Rule Overview*. Enter the name of the include program for field *Include Name*.
6. Change the Status to *Released*.
7. Save.

Note

Ideally those steps are maintained before replication starts. If the configuration is however already in replication, it need to be stopped and restarted after the steps above are completed.

Result

After your table has been replicated to the target system, you will be able to access the history of changes by specifying queries that use the two added fields.

3.2 Example

3.2.1 Adjusting the Target Table Structure

The following screen shows the final target table structure with the two added fields `SLT_UPDATE` and `SLT_OPERAT`.

Table Name: ZZSFLIGHT

Table Settings | **Table Structure** | Mapping Values

Mass Change

Original Table Structure						Modified Table Structure					
Field Name	Position	Key	Data Type	Length	Dec.	Field Name	Position	Key	Data Type	Length	Dec.
MANDT	1	<input checked="" type="checkbox"/>	CLNT	3	0	MANDT	1	<input checked="" type="checkbox"/>	CLNT	3	0
CARRID	2	<input checked="" type="checkbox"/>	CHAR	3	0	CARRID	2	<input checked="" type="checkbox"/>	CHAR	3	0
CONNID	3	<input checked="" type="checkbox"/>	NUMC	4	0	CONNID	3	<input checked="" type="checkbox"/>	NUMC	4	0
FLDATE	4	<input checked="" type="checkbox"/>	DATS	8	0	FLDATE	4	<input checked="" type="checkbox"/>	DATS	8	0
PRICE	5	<input type="checkbox"/>	CURR	15	2	SLT_UPDATE	5	<input checked="" type="checkbox"/>	DEC	15	0
CURRENCY	6	<input type="checkbox"/>	CUKY	5	0	PRICE	6	<input type="checkbox"/>	CURR	15	2
PLANETYPE	7	<input type="checkbox"/>	CHAR	10	0	CURRENCY	7	<input type="checkbox"/>	CUKY	5	0
SEATSMAX	8	<input type="checkbox"/>	INT4	10	0	PLANETYPE	8	<input type="checkbox"/>	CHAR	10	0
SEATSOCC	9	<input type="checkbox"/>	INT4	10	0	SEATSMAX	9	<input type="checkbox"/>	INT4	10	0
PAYMENTSUM	10	<input type="checkbox"/>	CURR	17	2	SEATSOCC	10	<input type="checkbox"/>	INT4	10	0
SEATSMAX_B	11	<input type="checkbox"/>	INT4	10	0	PAYMENTSUM	11	<input type="checkbox"/>	CURR	17	2
SEATSOCC_B	12	<input type="checkbox"/>	INT4	10	0	SEATSMAX_B	12	<input type="checkbox"/>	INT4	10	0
SEATSMAX_F	13	<input type="checkbox"/>	INT4	10	0	SEATSOCC_B	13	<input type="checkbox"/>	INT4	10	0
SEATSOCC_F	14	<input type="checkbox"/>	INT4	10	0	SEATSMAX_F	14	<input type="checkbox"/>	INT4	10	0
						SEATSOCC_F	15	<input type="checkbox"/>	INT4	10	0
						SLT_OPERAT	16	<input type="checkbox"/>	CHAR	1	0

View Modification Overview

Comparison of ZZSFLIGHT source table structure (original) and target table structure

3.2.2 Creating the Include Program

In transaction `SE38`, enter the name of the new include program, for example, `Z_INSERT_ONLY`, and choose [Create](#). The system displays the following screen:

ABAP: Program Attributes ZSLT_INSERT_ONLY Change

Title: Z_INSERT_ONLY

Original language: EN English

Created: 04.05.2018 JOCHUMBR

Last changed by:

Status: New(Revised)

Attributes:

Type: INCLUDE program

Status: Test Program

Application:

Authorization Group:

☐ Editor lock

Save Cancel Help Print Close

Creating the Include Program for the Event-Based Rule

The code example below demonstrates how any operation (insert, update, delete) can be changed to insert only. Variable `_COBJ_ALIAS` is a constant for the current table name to keep the code generic. The information about the original operation on the source system is stored in the newly added non-key field `SLT_OPERAT`. The key field `SLT_UPDATE` is populated with the current timestamp.

ZSLT_INSERT_ONLY	Active
<pre> *-----* * Include ZSLT_INSERT_ONLY *-----* DATA: lv_name(30) TYPE c, lv_operat_src TYPE char1. FIELD-SYMBOLS: <ls_record> TYPE any, <lv_operat_trg> TYPE any, <lv_operat_src> TYPE any, <lv_update> TYPE any. * get target structure CONCATENATE 'WA_R' _COBJ_ALIAS INTO lv_name. ASSIGN (lv_name) TO <ls_record>. ASSIGN COMPONENT 'IUUC_OPERAT_FLAG' OF STRUCTURE <ls_record> TO <lv_operat_trg>. IF sy-subrc = 0. lv_operat_src = <lv_operat_trg>. *** change operation to insert <lv_operat_trg> = 'I'. *** set additional target fields ASSIGN COMPONENT 'SLT_OPERAT' OF STRUCTURE <ls_record> TO <lv_operat_src>. IF sy-subrc = 0. <lv_operat_src> = lv_operat_src. endif. *** set updated timestamp ASSIGN COMPONENT 'SLT_UPDATE' OF STRUCTURE <ls_record> TO <lv_update>. IF sy-subrc = 0. GET TIME STAMP FIELD <lv_update>. endif. ENDIF. </pre>	

Code Example for include Z_INSERT_ONLY

3.2.3 Creating the Event-Based Rule

In the last step, the include program needs to be assigned to the rule. The status of the rule has been changed to *Released*. The include program will only be called if the rule has the status *Released*.

Table Name

Rule Overview

Status	Rule Type	Assignment Target	Implementation
Released	Event-Related	Begin of Record	Include : ZSLT_INSERT_ONLY_EVENT

Details of Rule for Begin of Record

Rule Type	Event-Related
Status	Released
Import Parameter 1	
Import Parameter 2	
Import Parameter 3	
Include Name	ZSLT_INSERT_ONLY_EVENT
Line of Code	

Releasing Event-Related Rule

4 Using Templates to Handle Multiple Tables

4.1 Overview

If you want to apply the "insert only" behavior to multiple tables, you can use templates for both the options History Column Table and event-based rules. This reduces the number of manual steps.

To do this, you create a template that contains all required settings for the both options. The template is then assigned to the configuration. During the replication process, the system then applies these settings to the replicated data depending on the selection criteria defined.

4.1.1 Creating the Template

1. In the SAP Landscape Transformation Replication Server system, create your configuration and specify the settings as usual that are required for your project.
2. Open transaction `LTRS` and choose your configuration.
3. In the menu, choose [Goto](#) and then [Template Maintenance](#).
4. Choose [Create](#) and enter a value for the [Template Name](#) (for instance `INSERT_ONLY`) and [Template Component Name](#) (for instance `INSERT_ONLY_COMP`).

4.1.2 Configuring the Template

If you are using the option History Column Table, proceed as follows:

1. Under [General Table Settings](#), in the field [Table Store](#), choose [History Column store](#).
2. Choose [Save](#).

If you are using the option Event-Based Rule, proceed as follows:

1. Under [Table Structure Deviations](#) choose [Create](#) and add an additional key field to your target structure. Enter the following values:

Field	Value
<i>Deviation Type</i>	Add
<i>Field Name</i>	SLT_UPDATE
<i>Position</i>	1
<i>Data Type</i>	DEC
<i>Length</i>	15

- Choose *Confirm*.
Note that position 1 is chosen to ensure that the added field will be a key field.
- Optionally, add another non-key field to the target structure. For example, you could enter the following values:

Field	Value
<i>Deviation Type</i>	Add
<i>Field Name</i>	SLT_OPERAT
<i>Position</i>	999
<i>Data Type</i>	CHAR
<i>Length</i>	1

- Note that position 999 is chosen to ensure that the added field will be a non-key field.
- Choose *Save*.
 - Under *Rule Assignment* create a new Event-Based Rule as described in 3.1.3.
 - Inside your template component, double-click *Patterns* and choose *Create*.
 - Select *Table Pattern* and enter a rule to select the relevant tables. The rule component will automatically be assigned to any table of the configuration, that fits to this selection criteria. For example, if you enter the value instance **z***, the template will be used for any tables starting with Z.
 - Choose *Save*.

4.1.3 Assigning the Template to the Configuration

- Go back to the initial screen of transaction **LTRS** (or press F3).

2. Choose the icon [Manage Templates](#) (on the left-hand side), and assign the template you have created before (for instance `INSERT_ONLY`) by selecting it. Choose [Confirm](#).

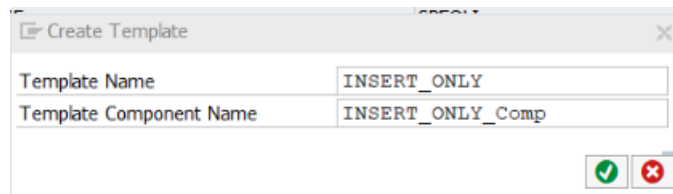
4.1.4 Result

Any table that is in scope of the configuration and matches the selection criteria specified in the pattern will now make use of the settings of the template.

4.2 Example

4.2.1 Creating the Template

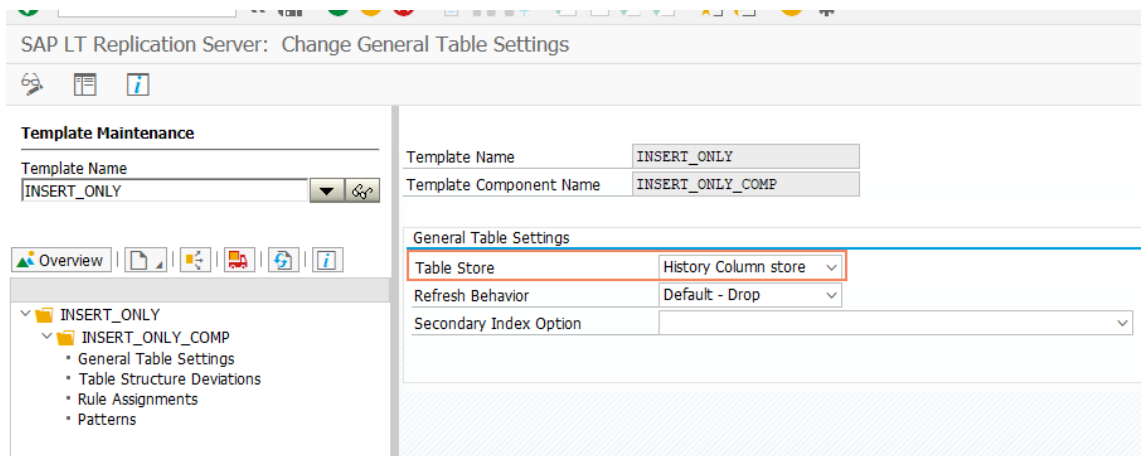
You create a template and a template component.



Creating a Template

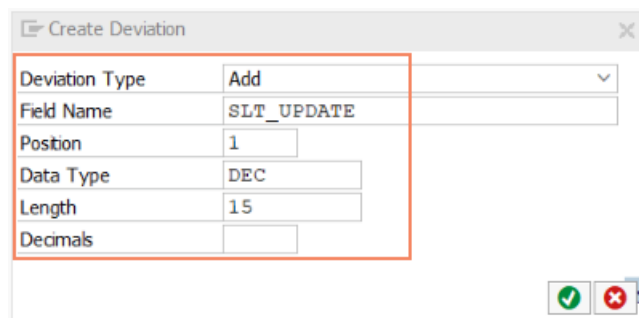
4.2.2 Configuring the Template

For History Column Tables, the value of the [Table Store](#) field in the template is changed to **History Column store**.



Template Configuration for History Column Table

For an event-based rule, you create a *Table Structure Deviation*. The key field `SLT_UPDATE` is added at position 1 to ensure that it will be added as a key field.



Adding the key field `SLT_UPDATE`

Note

When you add key fields to a table structure, the added key fields will automatically become part of the primary key. As usual, SAP LT Replication Server will create the target table and the primary key in the target system. If you are adding key fields to a table that already exists in the target system, the settings you specify using SAP LT Replication Server will only be considered if you have manually adjusted the table structure. This means also adjusting the primary key. However, exercise caution, as this might negatively affect performance if you have already applications in use that use this table.

The field `SLT_OPERAT` is added in the same way, however it is put at position 999 to ensure that it will be added as a non-key field.

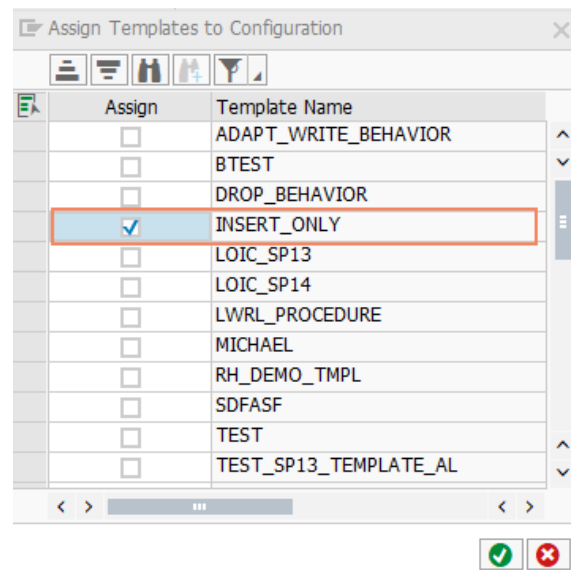
Adding the non-key field SLT_OPERAT

The following screenshot shows the pattern specified for the template component INSERT_ONLY_COMP. All tables starting with Z* will make use of the template.

Table pattern for template component

4.2.3 Assigning the Template to the Configuration

In the Advanced Replication Settings (transaction LTRS), the template INSERT_ONLY is now assigned to the configuration.



Assigning the template to the configuration



www.sap.com/contactsap

© 2018 SAP. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

National product specifications may vary.

These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries. Please see www.sap.com/corporate-en/legal/copyright/index.epx#trademark for additional trademark information and notices.

Material Number: NA