Optional Features of ServiceNow Table Maps

Once you are familiar with the basics of ServiceNow table maps, including the difference between inbound and outbound table maps, how to create a table map, and how to add a table map to dynamic/bulk shares, there are optional features you can configure based on your needs.

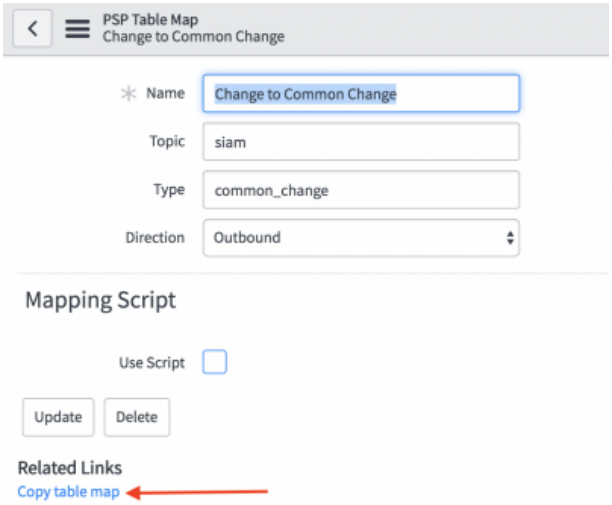
What's on this page?

Copy a ServiceNow table map

Existing table maps, as well as transform maps and all other related mappings, can be duplicated or copied. If you plan to modify any of the Common Table Maps, e.g. Incident to Common Incident, we recommend that you modify a new copied table map.

Here's how:

1. In your ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**
2. Select the table map that you want to copy
3. Under the **Related Links** section at the bottom, click **Copy table map**



Go to top of page

Customize your target database schema using table maps

By default, data replicated from existing tables will automatically create the same schema at the replicated database. The targeted database schema of replication needs to be flexible and customized in certain situations, such as:

- An existing schema that the replicated data need to fit in
- A need to rename the field names or transform the field values using script at the source

- The resulting field types need to be modified to a specific type or field length

Customizing your target database schema is done by using table maps in your dynamic share or bulk share.

Here's how:

Prerequisites:

- Install and configure DataSync for ServiceNow

Instructions:

1. In your sharing ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**
2. Select the table map you want to use (for the example, we'll use the table **Incident to Common Incident)**, review the information in the fields, and check the **Generate schema** box.

   ⚠️ **WARNING**: If there are multiple table maps with the same **Target table name**, the agent will return the first record when querying against ServiceNow for the schema.

   NOTES:
   - Checking the **Generate schema** box is required in the initial sharing of your records.
   - Specifying **Field Type** and **Field Length** is optional except for designating at least one field as **Sys ID (GUID)**. This is required for the target database table to be created and queried for updates and deletes.
   - The value specified in the **Target table name** field will be used as the name of the table in the database. When the DataSync Agent queries the instance for the table's schema, if it cannot find a table map with a matching **Target table name**, it will then look for a table that matches the value in the **Type** field instead.
3. To share the table map output, you need to crate a dynamic share or bulk share and select the table map from step 1 in the **table map** field. The generated payload can be subscribed by an agent and the scheme created dynamically.

   In this example, the **common incident** table map will result in the following table schema, if consumed by a database agent for MySQL.

```
mysql> describe common_incident;
+---------------------+--------------+------+-----+---------+-------+
| Field               | Type         | Null | Key | Default | Extra |
+---------------------+--------------+------+-----+---------+-------+
| state               | mediumtext   | YES  |     | NULL    |       |
| short_description   | varchar(160) | YES  |     | NULL    |       |
| description         | mediumtext   | YES  |     | NULL    |       |
| priority            | mediumtext   | YES  |     | NULL    |       |
| attachments         | mediumtext   | YES  |     | NULL    |       |
| category            | mediumtext   | YES  |     | NULL    |       |
| correlation_id      | varchar(100) | YES  |     | NULL    |       |
| number              | varchar(40)  | YES  |     | NULL    |       |
| correlation_display | varchar(100) | YES  |     | NULL    |       |
| caller_id           | mediumtext   | YES  |     | NULL    |       |
| caller_email        | varchar(100) | YES  |     | NULL    |       |
| caller_full_name    | mediumtext   | YES  |     | NULL    |       |
| comments            | mediumtext   | YES  |     | NULL    |       |
| sys_id              | varchar(224) | NO   | PRI |         |       |
| work_notes          | mediumtext   | YES  |     | NULL    |       |
+---------------------+--------------+------+-----+---------+-------+
15 rows in set (0.00 sec)
```

ⓘ Even though we specified the **number** field as the required **Sys ID (GUID)** field in the **table map**, the **number** field does not become the **G UID**. Instead, the **sys_id** field is created as **GUID (PRI key)** and contains **number** values to be used for lookup during updates/deletes. This is consistent with replicating database views.

Go to top of page

Use scripts to serialize records

Table maps can be configured to use scripts to serialize the record yourself. Within the script, you can set the variable **answer** to be the value you want the record to be serialized and returned as.

This is an advance configuration. Contact Perspectium Support for more information.

The following variables are available to use:

Variable	Description
current	This represents the record that is being shared
gr_tablemap	This represents the outbound table map itself. For example if you want to access the table map's name, you can use gr_tablemap.u_name

Instructions:

1. In your ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**.
2. Select the table map that you want to add a script to or create a new table map.
3. Under the **Mapping Script** section, check the **Use Script** box.
4. Fill the **Script** field with the appropriate script. See below for an example script.
5. Click **Update**.

Example of a table map script

Click to reveal

```
/*
 * Custom Table Map
 */

//Serialize 'current' record into an XMLDocument
var recordSerializer = (typeof GlideRecordXMLSerializer != 'undefined') ? new GlideRecordXMLSerializer() :
new Packages.com.glide.script.GlideRecordXMLSerializer();
var xmlstr = recordSerializer.serialize(current);
var xmlDoc = new XMLDocument(xmlstr);

//Process Display Values As Necessary
var pspUtil = new PerspectiumUtil();
var addDisplayValues = pspUtil.getPspPropertyValue("com.perspectium.replicator.add_display_values", "true");
var currentFieldsOnly = pspUtil.getPspPropertyValue("com.perspectium.replicator.share_current_fields",
"false");
if (currentFieldsOnly == "true" || addDisplayValues == "true") {
    addDVFields();
}

/*
 * Any extra mapping, to add
 */

// Send the XMLDoc string to our answer
answer = xmlDoc.toString();

// Helper Functions

//Standard DV Field Processing
function addDVFields(){
    var fl = (typeof GlideFieldList != 'undefined') ? new GlideFieldList() : new Packages.com.glide.
processors.FieldList();
    var tableName = current.getTableName();
    var fieldNames = fl.get(current.getTableName(),"");
    var arrFields = current.getFields();

    for (var i = 0; i < arrFields.size(); i++) {
        var glideElement = arrFields.get(i);
        var ed = glideElement.getED();
        var elName = glideElement.getName();

        if (!fieldNames.contains(elName) || (currentFieldsOnly == "true" && tableName != ed.getTableName()))
{
            removeElement(elName);
        }

        // Create dv fields for reference, choice, or lists
        if (ed.isReference() || ed.isChoiceTable() || ed.getInternalType() == "glide_list") {
            addElement("dv_" + elName, glideElement.getDisplayValue());
        }

        if (!glideElement.hasValue()) {
            continue;
        }
    }
}

//Remove an element from xmlDoc
function removeElement(elName){
    var nn = xmlDoc.getElementByTagName(elName);
    if(nn && nn.parentNode) {
        nn.parentNode.removeChild(nn);
    }
}
//Add an element from xmlDoc
function addElement(elName, elValue){
    xmlDoc.createElement(elName, elValue);
}
```

Go to top of page

Always share field

ServiceNow table maps abide by the share only updated fields property that is set in the dynamic share record. However, if you want to override this setting, you can do so in the table field map record. This can be useful if you have a field that is designated as the record's unique ID field (such as the sys_id field) that you always want to share out, regardless of whether it's updated or not.

Prerequisites:

- Commit a Perspectium update set containing table maps to your ServiceNow instance.
- Create a ServiceNow dynamic share or bulk share.
- Add a table map to your dynamic share.

Instructions:

1. In your sharing ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**.
2. Find and click the name of the table map that you want to modify.
3. In the **PSP Table Field Maps** section, select the field that you want to always hare out regardless of the field being updated or not.
4. Check the **Always Share Field** box.
5. Click **Update**.

Go to top of page

Hide empty fields

ServiceNow table maps will often generate common documents with empty elements, depending on the script that is used or the source field being mapped. This may not be the desired result and hiding these empty elements may be more favorable. This can be done using the **hide empty fields** f eature that is built into table maps.

Prerequisites:

- Commit a Perspectium update set containing table maps to your ServiceNow instance.
- Create a ServiceNow dynamic share or bulk share.
- Add a table map to your dynamic share.

Instructions:

1. In your ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**.
2. Find and click the name of the table map that you want to modify.
3. Check the **Hide Empty Fields** box.
4. Click **Update**.

Go to top of page

Field mappings in table maps

This feature allows you to map fields in the outbound record based on values from the ServiceNow record to be shared.

Prerequisites:

- Create a ServiceNow table map

Instructions:

1. In your ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**.
2. In the **PSP Table Field Maps** section, click **New**.

   Alternatively, you can click **Add all source table fields.** This allows you to quickly add all fields of the specified source table as field maps. This can be useful for cases where you want all fields to be in the outbound table map and only want to modify a few fields to have different values.
3. Set the following required fields:

   - Set the **Source Field** with the name of the field in the source table
   - Set the **Target Field** with the name of the field in the target table
4. Click **Submit**.

Go to top of page

Use script in field mappings

Using scripts, you an enhance or create new columns of data in your table map's field mapping. Here's how:

Prerequisites:

- Create a new field mapping

Instructions:

1. In your ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**.
2. Find and click into the desired table map.
3. In the **PSP Table Field Maps** section, click the desired field map.
4. Check the **Use Script** box, which will reveal a **Source Script** field.
5. Enter your script int o the **Source Script** field (find a few examples below)
6. Click **Update**.

Examples of using scripting in field mappings

There are various ways you can use scripting in your field mappings. Here are some examples (click to reveal):

Set mapping field value

To set an overall value for your field mapping, use the following:

```
answer = "";
```

Conversion to a new format

To script the transformation of an entire record to a new format (ex: convert to a custom JSON format), use the `current` variable to reference values in the source record and set the `answer` variable to be the new value returned.

The following example builds a JSON string from the value of correlation_id and short_description in the current record:

```
var attributes = {"type":"Case"};
var o = {};
o.attributes = attributes;
o.Id = current.correlation_id;
o.Subject = current.short_description.toString();

var j = new JSON();
j = j.encode(o);

answer = j.toString();
```

Ignore mapping fields

To ignore mapping fields for cases such as specified conditions not being met, use the script:

```
ignore = true;
```

In the following example, the mapping field will be ignored if the the **work_notes** field is empty:

```
if (current.work_notes == "")
   ignore = true;
else
   answer = current.work_notes;
```

Go to top of page

Create filter conditions for a table map

This feature allows you to share data only when certain conditions are met. You can also set up filter conditions for dynamic shares and bulk shares. T able map filter conditions can be created per source field either with the ServiceNow condition builder or programatically. For more information about creating filter conditions in ServiceNow, see condition builder.

Here's how to create filter conditions for a ServiceNow table map:

1. In your sharing ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**.
2. Find and click into the table map that you want to configure.
3. Scroll down to the **PSP Table Field Maps** tab. Then, locate and click into the name of the source field that you want to create filter conditions for. You can create conditions for your source field either by typing a script in the scripting window or using the ServiceNow condition builder below the scripting window. For more information about creating filter conditions in ServiceNow, see condition builder.
4. Click **Update**.

Go to top of page

Include embedded records in a table map

Embedded records are fields in the table map that cannot be mapped by singular values. Instead, they are entirely different records included (i.e. **emb edded**) inside the table map to be created on the subscribing system.

To include embedded records for a ServiceNow table map, do the following:

1. In your sharing ServiceNow instance, go to **Perspectium** > **Control and Configuration** > **Table Maps**.
2. Find and click into the table map that you want to configure.
3. Scroll down to the **PSP Table Field Maps** tab. Then, locate and click into the name of the source field that will contain the embedded record.
4. Update the **Source Field** to the following available:

Target Field	Source Field	Common Document
approvers	${TM:psp_approver;sysapproval=$[GR:sys_id]} ${TM:psp_approver;document_id=$[GR:sys_id]}	Common Change Common Request Common Request Item Common Knowledge
attachments	${TM:psp_attachment;table_sys_id=$[GR:sys_id]} ${TM:psp_attachment;table_sys_id=$[GR:sys_id];msp_client_application_sent;skip_insert} ${TM:psp_attachment;table_sys_id=$[GR:sys_id];msp_client_application_sent;psp_action=create}	Common Incident Common Change Common Problem Common Request Common Request Item Common Task
affected_cis	${TM:psp_affected_ci;task=$[GR:sys_id]}	Common Incident Common Change
affected_products	${TM:psp_affected_product;kb_knowledge=$[GR:sys_id]}	Common Knowledge
catalog_tasks	${TM:psp_catalog_task;request_item=$[GR:sys_id]}	Common Request Item
change_tasks	${TM:psp_change_task;change_request=$[GR:sys_id]}	Common Change
delegates	${TM:psp_delegate;user=$[GR:sys_id]}	Common User
groups	${TM:psp_group;user=$[GR:sys_id]}	Common User
group_approvals	${TM:psp_group_approval;parent=$[GR:sys_id]}	Common Request Common Request Item
impacted_services	${TM:psp_impacted_service;task=$[GR:sys_id]}	Common Change
incidents	${TM:psp_incident;problem_id=$[GR:sys_id]}	Common Problem
incidents_caused	${TM:psp_incident;caused_by=$[GR:sys_id]}	Common Change
incidents_fixed	${TM:psp_incident;rfc=$[GR:sys_id]}	Common Change
knowledge_feedbacks	${TM:psp_knowledge_feedback;article=$[GR:sys_id]}	Common Knowledge
manage_subscriptions	${TM:psp_manage_subscription;user=$[GR:sys_id]}	Common User
problems	${TM:psp_problem;rfc=$[GR:sys_id]}	Common Change
problem_tasks	${TM:psp_problem_task;problem=$[GR:sys_id]}	Common Problem
recurring_prices	${TM:psp_recurring_price;request=$[GR:sys_id]}	Common Request
requested_items	${TM:psp_requested_item;request=$[GR:sys_id]}	Common Request
roles_list	${TM:psp_role;user=$[GR:sys_id]}	Common User

variables	${TM:psp_requested_item_variables;request_item=$[GR:sys_id]}	Common Request Item

NOTE: You must have the referenced table map in your instance. For example, for **attachments**, you need to have the **psp_attachment** table map installed in your instance. See **Common Documents** for more information.

5. Click **Update**.

Go to top of page

For other ways to utilize scripting in your field mappings, contact Perspectium Support.