Master Data Management System (MDMS)

User Guide
Contents

Chapter 1  Introduction............................................................................................................................................. 1-3

Chapter 2  Glossary...................................................................................................................................................... 2-3

Chapter 3  Master Company Processing ..................................................................................................................... 3-4

   3.1. DATA SET...................................................................................................................................................... 3-4
   3.2. REPLICATION................................................................................................................................................. 3-7
   3.3. RUNNING REPLICATION......................................................................................................................... 3-9
   3.4. Synchronizing Records......................................................................................................................... 3-12

Chapter 4  ADVANCED FEATURES ............................................................................................................................ 4-14

   4.1. RECORD-LEVEL REPLICATION.............................................................................................................. 4-14
   4.2. DELETING RECORDS............................................................................................................................... 4-17
   4.3. MDMS EVENTS.............................................................................................................................................. 4-18
Chapter 1  Introduction

Master Data Management System (abbrev. MDMS) is a module that ensures data consistency when replicating data between multiple Microsoft Dynamics 365 Business Central databases. For this purpose, MDMS offers functions that enable the user to set up any set of data within Dynamics 365 Business Central database and replicate it to another Dynamics 365 Business Central database.

With its features for replication, master data management and data consistency, the MDMS application supports companies that use Business Central in multiple subsidiaries.

In MDMS, target companies with databases, to which data will be replicated are set up as receivers. It is possible to set up multiple receivers.

The structure of data replicated is set up as data sets.

Users can use any number of created data sets and set them up to be replicated to a particular receiver, by using the replication card. Replication can be set up as either Full or Incremental. While processing incremental replication only data changed or added within the last replication will be exported.

The module is particularly useful when a company wants to create and maintain standardized Dynamics 365 Business Central cards (e.g. items, BOMs, vendor cards) in one main company/headquarters – called “master” company, and afterwards synchronize the cards and data with its subsidiaries (“receiver” companies).

The MDMS application uses SOAP Web Services as a data transport layer.

Chapter 2  Glossary

MDMS: Master Data Management System.
Master Company: A company where all data considered global is stored
Receiver Company: A company to which data is sent.
Chapter 3  Master Company Processing

3.1. **Data Set**

Data sets contain information about the structure of data to be replicated. Data sets can be displayed in the **MDMS / Data Sets** window:

![Data Set Window]

The following fields are available in the Data Set header:

1. **No.**
   - Specifies a data set unique identifier.

2. **Description**
   - Specifies a description of the data set

3. **Last Date Modified**
   - This is a read only field. It contains a date of the last modification

4. **Status**
   - This is a read-only field that specifies the data set status. The status can be either Open or Released and can be adjusted using the Release and Reopen actions.

The following fields are available in data set lines:

1. **Code**
   - Specifies a unique identifier of the data set line.

2. **Name**
   - Specifies a description of the data set line.

3. **Table No.**
   - Specifies a number of the table to be replicated.
4. **Table Name**  
   Specifies a name of the table to be replicated; the field is filled in automatically when the Table No. field is filled in.

5. **On Record Exists**  
   Specifies the action which will be taken when it is found that a replicated record already exists in the target database. The following action options are available: Update and Skip.

6. **Record Level Replication**  
   Specifies if all the table data should be replicated or the user (with the MDMS Receiver role assigned) should manually mark the records to be replicated to the user’s local database (see more under the Record-Level Replication chapter).

7. **Disable Local Insert**  
   If this field is selected, users will not be able to insert new records in their local database.

8. **Disable Local Delete**  
   If this field is selected, users will not be able to delete records in their local database.

9. **Block Exclude Filter**  
   Specifies a filter (using C/AL SETVIEW syntax) to exclude certain records from the blocking rules. 
   Sample filters:  
   - Cross-Reference Type=Filter(<>Bar Code)  
   - Code=FILTER(FA CIP)  
   - Dimension Code=FILTER(FA CIP)  
   - No.”=FILTER(11111.11198|11211..11297)

10. **Run Trigger on Insert**  
    If a new record is inserted into a local database, INSERT(TRUE) will be used during the operation. 
    By default, it is INSERT(FALSE).

11. **Run Trigger on Modify**  
    If a record is updated in a local database, MODIFY(TRUE) will be used during the operation. 
    By default, it is MODIFY(FALSE).

12. **Target Table No.**  
    Specifies a number of the table into which records should be inserted in a receiver company. 
    The default value is 0 which means records will be transferred to the same table they are exported from.

   The user can set up which fields should be included in the data set using the Line / Show Details function.
The following fields are available on the Data Set Line Details page:

1. **Code**
   Specifies a unique identifier of a data set field.

2. **Field No.**
   Specifies a field number.

3. **Field Name**
   Specifies a name of the field.

4. **Target Field No**
   Specifies a number of the target field. Therefore, it is possible to set up a data set to import the data to another field from the same table.

5. **Validate Field**
   Specifies whether the field should be validated upon importing on the receiver side.

6. **Disable Local Modify**
   Specifies whether it should be allowed to edit fields at the local level, on the receiver’s side.

7. **Field Type**
   Specifies a type of the field.

8. **Processing Order**
   Specifies an order in which record fields will be processed. This is particularly important for the data validation flow.

9. **Table No.**
   Specifies a number of the table

10. **Keep Local Value**
    Specifies if a selected field value should be transferred from the global/master company to a local company, however the value is not overwritten afterwards. It will be possible to adjust the field’s value locally and this value will be kept.

11. **Skip Export**
    If this field is selected, the selected field will be excluded from the export.

**Note:** The Add Fields function:
The Copy function enables the user to copy data sets.

### ITEMS • Items Data Set

<table>
<thead>
<tr>
<th>New</th>
<th>Process</th>
<th>Actions</th>
<th>Fewer options</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Functions</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>![Check Release](Check Release)</td>
<td>Release</td>
</tr>
<tr>
<td></td>
<td></td>
<td>![Check Reopen](Check Reopen)</td>
<td>Reopen</td>
</tr>
<tr>
<td></td>
<td></td>
<td>![Check Copy](Check Copy)</td>
<td>Copy...</td>
</tr>
</tbody>
</table>

#### 3.1.1. Releasing / Reopening

After all the setup has been defined, it is necessary to mark the data set card as completed by using the **Functions / Release** action which will change its status to **Released**. The data set has to be released before it is included in any replication process. If the user wants to make any further adjustments to a previously released data set, it is necessary to reopen it first by using the **Functions / Reopen** action.

#### 3.1.2. Data Set filters

Using the **Line / Show Filters** action, it is possible to specify additional filters that will be applied to data set records when the replication process is run.

**NOTE:** If filters have been defined both for a data set and replication for the same table, filters on the data set level will be overwritten.

#### 3.1.3. Data Set Line Relation

The **Line / Show Relation** action is used to specify the relation between indented record and its parent.

#### 3.2. Replication

Replication links data sets to a particular receiver and specifies the type of data to be replicated. It is also possible to set up additional filters which will be applied when replication is processed. Replication entries can be displayed under MDMS / Replications:
The following fields are available in the replication header:

1. **No.**
   Specifies a unique identifier of the replication.

2. **Description**
   Specifies a description of the replication.

3. **Receiver No.**
   Specifies a number of the receiver.

4. **Last Run Date and Time**
   Specifies a date and time of the last successful replication.

5. **Status**
   This a read-only field and it specifies the replication status. The status can be either Open or Released and can be adjusted by using the **Functions / Release** and the **Functions/Reopen** actions.

The following fields are available in replication lines:

1. **Data Set No.**
   Specifies a number of the replicated data sets.

2. **Description**
   Specifies a description of the replicated data set.

### 3.2.1. Replication filters

The Line / Show Filters action is used to specify additional filters that will be applied to data set records while processing the replication:
The following fields are available:

1. **Data Set Line No.**
   Specifies a line number of the data set to be replicated.

2. **Data Set Line Code**
   Specifies a line code of the data set to be replicated. It is filled in automatically as soon as a number in the Data Set Line No. field is entered.

3. **Field No.**
   Specifies a number of the field for which the filter will be applied.

4. **Field Caption**
   The field is field in automatically as soon as a field number is entered.

5. **Value**
   Specifies a value that is a required filter value.

**NOTE:** Please remember that **Value** is a text-based field. If you want to set up a filter for an option-type field, you should ensure that the filter value is correct.

### 3.2.2. Releasing/Reopening

After required setup has been completed, it is necessary to mark replication cards as completed by using the **Functions / Release** action which will change its status to **Released**. Each replication has to be released before it is run. If the user wants to make any further adjustments to a previously released replication, it is necessary to reopen it first using the **Functions / Reopen** action.

### 3.3. Running Replication

While running a replication process, the system first checks whether all data sets included in the current replication are released and displays an error message if otherwise. Moreover, each replication has to be released as well.

After the replication process is completed successfully, the system creates record synchronization entries and updates the **Last Run Date and Time** field on the **Replication** card page.

### 3.3.1. Manual

The user can run the replication process manually, from a relevant replication card or replication list, using the **Run (Full)** or **Run (Incremental)** functions:
NOTE: It is possible to select multiple replications on the list and run them altogether.

### 3.3.2. Automatic (using Job Queue)

It is possible to set up replication to be run automatically using the **Job Queue** module (in Dynamics 365 Business Central).

**Codeunit • 20020699 • Run Replication (MDMS)**

**General**

<table>
<thead>
<tr>
<th>Object Type to Run</th>
<th>Codeunit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object ID to Run</td>
<td>20020699</td>
</tr>
<tr>
<td>Object Caption to Run</td>
<td>Run Replication (MDMS)</td>
</tr>
<tr>
<td>Description</td>
<td>Run Replication (MDMS)</td>
</tr>
<tr>
<td>Last Ready State</td>
<td>02.07.2019 15:05</td>
</tr>
<tr>
<td>Earliest Start Date/Time</td>
<td>...</td>
</tr>
</tbody>
</table>

**Parameter String**

It is necessary to set up a codeunit no. 20020699 by selecting the type and number in the **Object Type to Run** and **Object ID to Run** fields and then, selecting **Run Replication (MDMS)** in the **Object ID to Run** field.

In the **Parameter String** field, the following structure has to be applied:

\[
<Replication No.>,<Replication Type>
\]

Where:

- Replication No. – Specifies a number of the replication to be run;
- Replication Type – Specifies a replication type with any of two possible values: “FULL” and “INCREMENTAL”.

**Example Parameter String values are:**

- R0001,INCREMENTAL
- R0025,FULL
If the value in the **Parameter String** field is invalid, the job queue fails, and an appropriate error message is displayed. To enable the user to read incoming confirmations, it is necessary to set up Codeunit 20020701 with **Read Confirmations**.

### 3.3.3. Full vs Incremental

Each replication can be run in one of the two modes: Full or Incremental. Full replication loops through all the records from a data set, including filters set up on the replication line (if any). If no filters have been set up, all the records will be exported each time the replication process is run.

Incremental replication includes only data that has recently been modified (since the **Last Run Date and Time** field as specified on the Replication card). For each record that would normally be included in full replication, it performs additional check (using the internal change log functionality) to verify if the record should be exported.

**NOTE:** Only top-level changes are recognized.

Each successful replication process updates the **Last Run Date and Time** field on Replication card page.

### 3.3.4. Record Synchronization Entries

Each time the replication process is run, it creates record synchronization entries for each processed record. The entries can be viewed from the replication card by choosing **Navigate / Replication/Record Synchronization Entries:**
The following fields are available:

1. **Entry No.**
   Specifies a sequential number.

2. **Date and Time**
   Specifies a date and time of the replication to be run.

3. **Status**
   Specifies if the entry has been processed successfully.

4. **Processed Date & Time and Processed By**
   Specifies if the entry has been processed and the user who ran the process.

5. **Error Text**
   Specifies a text of the message displayed after a processing error occurs.

6. **Old Record ID**
   Specifies an old record ID to be used in case of record renaming. It points to the record that was renamed to be included in the receiver company.

7. **Receiver No.**
   Specifies a number of the receiver which is important when the group receiver feature is used.

### 3.4 Synchronizing records

After the replication process is run, no data is sent to the receiver company. Under Process Record Synchronization Entries, there are 2 action options available. The Synchronize on Replication Card action processes all entries pending for replication.
The Process on Record Synchronization Entries action processes all selected entries.
Chapter 4  ADVANCED FEATURES

4.1. **Record-Level Replication**

4.1.1. General information

By default, replication processes all records from given table, and while processing incremental replication, only recently modified records are exported. It may be however necessary to specify a certain subset of records that should be replicated to certain receivers.

Using Record-level Replication, every receiver has to manually mark which records they want to receive.

4.1.2. Setup

To enable Record-level Replication feature it is necessary to put a checkmark in Record-level Replication field in Data Set line:

![Record-Level Replication Example]

4.1.3. Processing

When Record-level Replication option is enabled for a data set, all replications that use this data set will consider only records that receiver has manually marked to replication.

To see a list of records, following page needs to be opened:
Note: Please remember that data set will appear on the list only when it is included in released replication. List is rebuilt while releasing a replication.

To see a list of database records user needs to run Records function. Following window will appear:

**Data Records** window contains following columns:

1. **Record ID**
   - Contains unique record ID of each record,

2. **Replicated**
   - Specifies whether record is being replicated,

3. **Status**
   - Record replication status,

4. **<data set columns>**
   - List of first 10 data columns from particular data set.

**Status** field may have one of two values:

- Under Revision,
- Released.

Only Released records are replicated. Under Revision status has been introduced to allow user to mark some of replicated records to further investigation. They will not be replicated until their status is switched back to Released.

Following functions are available in **Data Records** window allowing users to easily change status for many records at once:

- Set Replicated,
- Unset Replicated,
- Replicate All,
- Replicate None,
- Set Released,
- Set Under Revision.

**NOTE:** Above functionality may cover any table from the database, therefore it is built in very general way. It may be however necessary to implement it more user-friendly way on certain
application areas. For instance, if it is required to replicate customers data, customer card can be adjusted to allow users to set certain records as replicated directly on customer cards.

4.1.4. **Group Receivers**

When using the Group Receivers feature, it is possible to process one replication simultaneously for multiple receivers.

On the Receiver card, select **Group Receiver** to display the **Group Receiver Lines** subpage. It is possible to set up a list of linked receivers.

When the replication process is run for a group receiver, record synchronization entries are created for each of the receiver lines.
### 4.2. Deleting Records

Records can be deleted in the master company. Records that have never been replicated yet, can be deleted when the **Allow Deleting Before Replicated** checkbox is selected on the **MDMS Setup** page.

For records that have been replicated, the system will automatically issue a webservice call for all receiving companies that have ever received this particular record and will try to delete the record from the companies’ databases. If the process is completed successfully, records will be deleted from the master company database as well.

**NOTE:** The Block Record Delete setup parameters should be disabled on the **MDMS Setup** page first.

The following dialog window will be displayed:

![System will now try to delete current record (Item: 1000) in all companies it was sent to. Do you want to continue?](image)

When the message is confirmed, the system will attempt to delete the record from all receiver companies.

If there are any errors (e.g. when the item being deleted is on stock), they will be listed in this summary:
Only after deleting is completed successfully in all the companies, the record will be deleted in the master company.

4.3. MDMS Events

In order to efficiently incorporate advanced business logic, MDMS now publishes the following Events:

1. Codeunit Replication Mgt.
   1.1 OnBeforeExportRecord
      Allows the user to specify if given record should be exported (e.g. when exporting a list of currencies – to avoid publishing a local currency. This can also be set up by using filters on the Replication card).
      Parameters:
      - ReplicationNo [Code20]
      - ReceiverNo [Code20]
      - ExportedRecord [RecordID] = Specifies a record to be exported.
      - IncludeRecord [Boolean] = Specifies if the record will be exported.
      - VAR OverrideIncludeRecord [Boolean] = The parameter should be set to TRUE in order to override a default value.
      - VAR NewIncludeRecord [Boolean] = The parameter should be set to TRUE/FALSE in order to include or skip the record. This setup only works when combined with OverrideIncludeRecord set to TRUE.

Example: (Currency CAD to be skipped from export):

```plaintext
LOCAL [EventSubscriber] BeforeRecordExported(ReplicationNo : C)
IF ExportedRecord.NUMBER = DATABASE::Currency THEN BEGIN
  ExportedRecord.SETTABLE(MyCurrency);
  IF MyCurrency.Code = 'CAD' THEN BEGIN
    OverrideIncludeRecord := TRUE;
    NewIncludeRecord := FALSE;
  END;
END;
```

1.2 OnAfterExportRecord
   Allows you to perform actions on exported records.
   Parameters:
   - ReplicationNo [Code20]
   - ReceiverNo [Code20]
   - TableNo [Integer]
   - RecordID [RecordID] = Specifies the record to be exported (RecordID)
• ExportedRecord [RecordRef] = Specifies the record to be exported (RecordRef)

**Example usage** (Fill in “Replication Comment” = new field on Customer card):

```plaintext
LOCAL [EventSubscriber] AfterRecordExported(ReplicationNo : Code[20]; ReceiverNo
    IF ExportedRecord.NUMBER = DATABASE::Customer THEN BEGIN
        ExportedRecord.SETTABLE(MyCustomer);
        MyCustomer."Replication Comment" := 'Published ' + FORMAT(CURRENTDATETIME);
        MyCustomer.MODIFY;
    END;
END;
```

2. **Codeunit Record Synchronization Mgt.**

2.1 **OnAfterImportedRecordModify**

Allows the user to perform actions on imported records.

- **Parameters:**
  - **VAR ModifiedRecord [RecordRef]** = Specifies the record to be updated.
  - **IsNewRecord [Boolean]** = Specifies if the record was new or it existed before.

**Example:** (Updating the Currency Code field on the Customer card, this time on the receiving company side):

```plaintext
LOCAL [EventSubscriber] AfterRecordImported()
    IF ModifiedRecord.NUMBER = 18 THEN BEGIN
        ModifiedRecord.SETTABLE(MyCustomer);
        IF MyCustomer."Currency Code" = 'CAD' THEN
            MyCustomer."Currency Code" := '';
            MyCustomer.MODIFY;
        END;
    END;
END;
```

3. **OnBeforeExportField:**

Allows you to override a default field value with your own function.

- **Parameters:**
  - **ReplicationNo [Code20]**
  - **ReceiverNo [Code20]**
  - **TableNo [Integer]**
  - **FieldNo [Integer]**
  - **RecordID [RecordID]** = Specifies the record to be exported.
  - **FieldValue [Text]** = Specifies the currently exported field value
  - **VAR OverrideFieldValue [Boolean]** = This parameter should be set to TRUE in order to override default value
  - **VAR NewFieldValue [Text]** = This parameter should be to a new field value.

**Example:** (Currency Code CAD on Customer Card to be changed to <Blank> when exporting to Canada Receiver):

```plaintext
LOCAL [EventSubscriber] AfterFieldExported(Replication
    IF ReceiverNo <> 'CA' THEN
        EXIT;
    IF TableNo = DATABASE::Customer THEN BEGIN
        IF FieldNo = 22 THEN BEGIN
            IF FieldValue = 'CAD' THEN BEGIN
                NewFieldValue := '';
                OverrideFieldValue := TRUE;
            END;
        END;
    END;
END;
```