Data Express 4.0

Data Generation Guide

Micro Focus
The Lawn
22-30 Old Bath Road
Newbury, Berkshire RG14 1QN
UK
http://www.microfocus.com

2024-03-15

Contents

Data Generation Guide

This guide outlines the concepts and procedures used by the Data Express Data Generation module, which uses user-specified selection criteria, rules, and data to create a new test environment from scratch.

You begin with an existing but empty test environment. Then, use the Data Generation module to:

- Create rules that derive a test environment from the client side
- Populate job submissions derived from the client side
- Load tables via the Distributed Loader for distributed data stores, or via the external interface for z/OS data stores
- Define and associate selection classes

Use this process to generate a full test environment, or a partial test environment. A partial test environment typically includes the applications not present in production. In this case, the remainder of the test environment comes from production through Data Subset Extraction.

Who Should Read This Guide

This guide is for Micro Focus users who are interested in using a subset of data to populate empty tables independent of the content of a production environment. It explains the Data Generation configuration procedure, and describes the steps required to populate the files in your application.

Before using this product, we recommend that you carefully review the other Data Express documentation, including the *Front End Guide* and the *Data Subset Extraction Guide* in particular.

Before Starting with Data Generation

Before using the Data Generation module for Data Express bear in mind the following points:

- The Data Generation module is available for both Data Express for Distributed Systems and Data Express for z/OS. This requires that you have the correct license. See the *Installation Guide* for details.
- To use the Data Generation module, it is not necessary for you to analyze application or program source code.
- The Data Generation module manages all types of z/OS data stores processed by the Data Builder module except DL/I. , in addition to all types of ODBC distributed data stores processed by the Data Builder module.

Getting Started

Provides an overview of the Data Generation module. We recommend that you start by reading this section before using the module.

Mode

Data Express provides two modes:

Standard Mode	Supports direct access to the data resident on RDBMS from your PC.

Client/Server Mode	Supports access to the data resident on RDBMS through the Data Express kbde-Server, which is based on a three-tier architecture with the second tier handling data transfer optimization. This enables you to access data over the Internet with a minimal amount of administration and using efficient network capacity. In addition, Client/Server mode can use the same functions as Standard mode, but does not require any additional software such as the DB2 client or the Borland Database Engine (BDE).

Data Express for z/OS can be used in either Standard or Client/Server mode, whereas Data Express for Distributed Systems can be used in Standard mode only.

For Data Subset Extraction configuration information and instructions, see the *Appendix A. Standard Mode* and *Appendix B. Client/Server Mode* sections in the *Front End Guide*.

Data Generation UI Guidelines

The following guidelines apply to windows and screen components of the Data Generation module:

- The main window, **Data Generation - [Master Form]**, cannot be closed except when exiting the module.
- Multiple MDI windows may be open at the same time.
- To view a list of commands that apply to a specific item or area, right-click on the item or area.

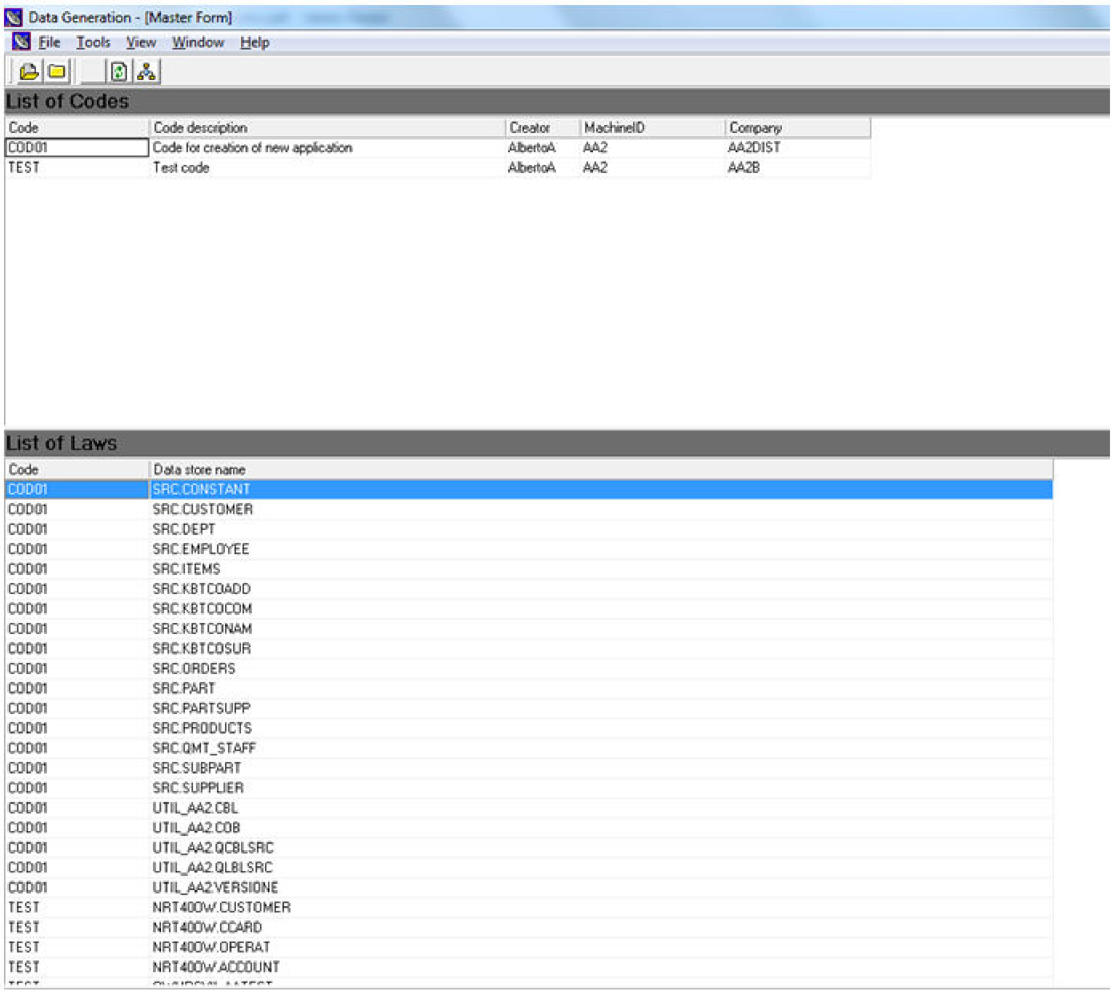
Starting the Data Generation Module

To start the Data Generation module:

1. Click **Start > All Programs > Micro Focus Data Express 4.0 > Data Generation**.

   Note: If you are using Data Express on Windows Vista and User Access Control is enabled, you must run Data Generation as an Administrator.
2. Select the required database with which to launch the connection. By default, the highlighted item in the list of available databases is the last database you connected to using the Data Generation module.

   Note: To connect to the database for first time, you need your user ID and password.
3. Select the required schema. The **Data Generation - [Master Form]** window appears:



List of Codes

Code	Code description	Creator	MachineID	Company
COD01	Code for creation of new application	AlbertoA	AA2	AA2DIST
TEST	Test code	AlbertoA	AA2	AA2B

List of Laws

Code	Data store name
COD01	SRC.CONSTANT
COD01	SRC.CUSTOMER
COD01	SRC.DEPT
COD01	SRC.EMPLOYEE
COD01	SRC.ITEMS
COD01	SRC.KBTCOADD
COD01	SRC.KBTCOCOM
COD01	SRC.KBTCONAM
COD01	SRC.KBTCOSUR
COD01	SRC.ORDERS
COD01	SRC.PART
COD01	SRC.PARTSUPP
COD01	SRC.PRODUCTS
COD01	SRC.QMT_STAFF
COD01	SRC.SUBPART
COD01	SRC.SUPPLIER
COD01	UTIL_AA2.CBL
COD01	UTIL_AA2.COB
COD01	UTIL_AA2.QCBLSRC
COD01	UTIL_AA2.QLBLSRC
COD01	UTIL_AA2.VERSIONE
TEST	NRT400W.CUSTOMER
TEST	NRT400W.CCARD
TEST	NRT400W.OPERAT
TEST	NRT400W.ACCOUNT

Figure 1: Data Generation - [Master Form] Window

Tip: Once you are connected to a database, you can open and close a database from the **Data Generation - [Master Form]** window by clicking **File > Open** or **File > Close**.

Directive Types

The following directives, applied at column level, enable the generation of synthetic data according to their defined rules. If no directive is specified for a given column, the column is populated with the default value for column type; for example, zero for numeric columns:

Generation by dictionary	Column values are obtained from a dictionary. Data Express provides some pre-loaded dictionaries (for masking purposes), and allows you to add dictionaries as well. To generate using a dictionary, you specify the dictionary name in the directive properties.
Generation by domain value list	Column values are obtained from a user-defined list. For example,if your column is a CHAR(1) and you want to assign it the values M and F (as male and female) with probability distribution values to represent 60% M and 40% F, you can specify these values in the directive properties.
Generation by random value list	Columns are randomly assigned values in the column's domain of definition.

Generation by referential integrity (other file dependency)	Column values are obtained from a list of values present in a column from another table. To ensure this, the parent and child column must be of the same class. **Note:** This feature is similar to the "filter by filtered list" feature in Data Subset Extraction.

Distributed Exporter Utility

The Distributed Exporter utility that creates and provides necessary information to the Extension Technology that enables the generation of a distributed data store.

This utility generates the following files after you have successfully exported from the Data Generation module:

`method.rc`	Coded content of source and target databases information.
`method.txt`	Information for the environment and the method in it which are about to be masked.
`filter.txt`	All created filters in the test environment.
`elab.txt`	Information about tables involved in Data Generation process.
`Dir.txt`	Information about the directives to be used
`cbfld.txt`	Information about combined fields used in the method.
`CREATETABLE.sql`	A list of the tables that need to be masked; a generic CREATE TABLE statement is provided for each table.
`CREATEINDEX.sql`	A list of all the indexes for the tables that need to be masked; a generic CREATE INDEX statement is provided for each index.
`ALTERTABLE_RI.sql`	A list of all the primary and foreign keys for the tables that need to be masked; a generic ALTER TABLE statement is provided for appropriate primary and foreign keys.

There are two areas of interest when it comes to using the Distributed Exporter with distributed data stores.

- If you use ODBC-enabled data stores, you can subset across all your data stores with one invocation of the ODBC Extension. This feature is not available with the Oracle Extension.
- You also have the ability to subset within one distributed data store, as long as a distinct target schema name is provided.

If you use the Distributed Exporter utility on one machine, and do your generation on another, make sure that the target ODBC DSN matches the ODBC DSN used for the actual data generation.

Executing Data Generation

Data Generation execution for distributed data stores is done from the command line using the `dxestart` command, which reads the configuration files exported by Distributed Exporter, and populates the target environment according to the laws and directives specified.

The `dxestart` executable resides in the `C:\Program Files (x86)\Micro Focus\Data Express 4.0\Synthetic\ODBC` directory by default.

For complete information about using the `dxestart` command, see *The dxestart Command* topic in your *Getting Started with Distributed Data Stores* documentation.

Catalog Distributed Data

You catalog the target database in Data Express using the Distributed Loader feature of the Data Builder module. A detailed description of this feature is in the *Using Distributed Loader* section of your *Getting Started with Distributed Data Stores* documentation. Consider the following:

- When using Distributed Loader, the target database should be accessible via ODBC using the `dxegenloadfile` utility either from the machine where the Data Builder module is installed or from the machine where the distributed engine is installed. The `dxegenloadfile` utility is detailed in your *Getting Started with Distributed Data Stores* documentation.
- Distributed Loader allows you to specify an ODBC instance and schema, and presents a list of tables from which you select tables to load into the Data Express knowledge base.
- After confirming which tables to load, Distributed Loader launches an interactive process that writes the catalog and structure information for the selected tables to the knowledge base.

Catalog z/OS Data

For details on the process of cataloging z/OS data in Data Express, see the *Getting Started with z/OS Data Stores* topics.

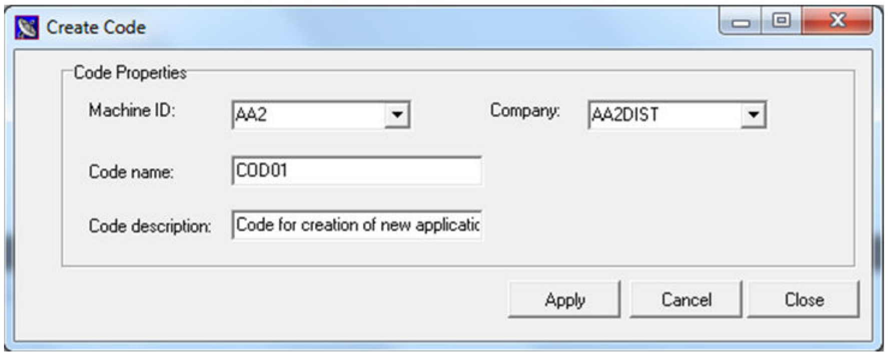
Classify Data

The Data Generation module requires that you assign a class to each column that uses referential integrity directives. The process used to classify distributed and z/OS data stores is essentially the same. For specific instructions on assigning classes for distributed data stores, see the *Importing classes* section in your *Getting Started with Distributed Data Stores* documentation. For instructions on assigning classes for z/OS Data, see *Getting Started with z/OS Data Stores*.

Note: While you can assign classes to columns that are not used in Data Generation, it is not required.

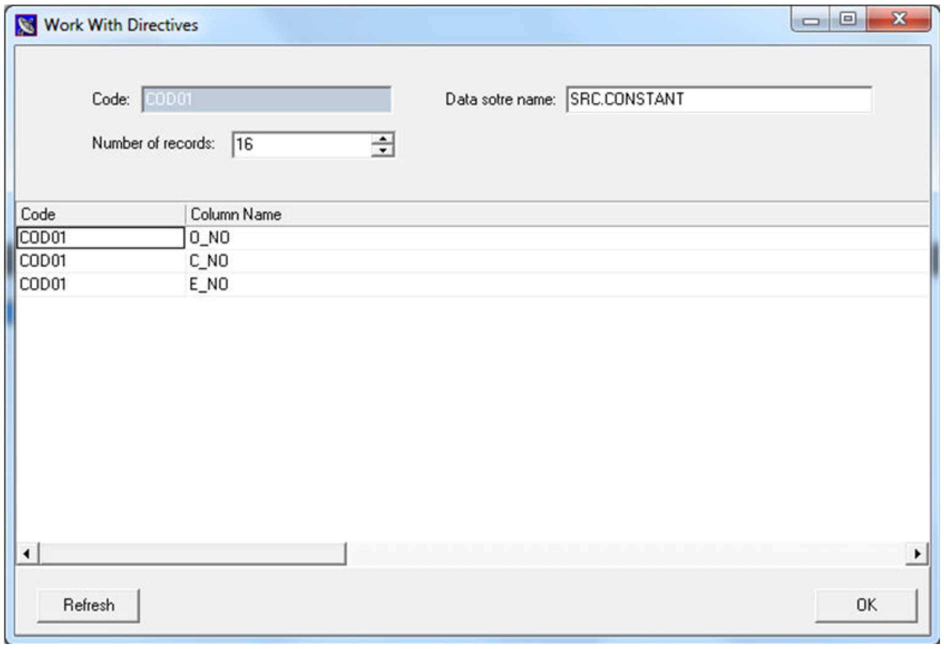
Define Rules

Use the Create Code wizard to define Data Generation rules for both distributed and z/OS data stores. The Create Code page of the wizard enables you to create, name, and describe a code set for a given machine and company:



Machine ID: AA2
Company: AA2DIST
Code name: COD01
Code description: Code for creation of new applicatic

After applying the parameters specified on the Create Code page, you then progress to the Work With Directives page. On this page, you associate the code set with tables by adding a law set rule for each table. Each law rule definition includes a parameter that specifies the number of records used to populate the associated table.



Code: COD01
Data sotre name: SRC.CONSTANT
Number of records: 16

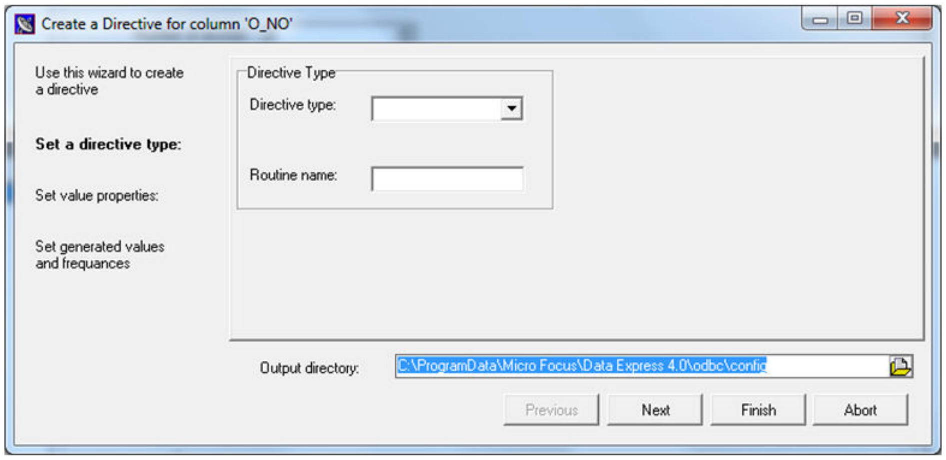
Code	Column Name
COD01	O_NO
COD01	C_NO
COD01	E_NO

From this page, you can open each law rule, enabling you to specify directives when necessary. To do this, right-click a **Column Name**; then select **Set Directive**. This opens the Create a Directive for column page.

You can select the directive type, and assign a routine name to the directive type. Each directive type obtains column values differently as follows:

Other file dependency These directives obtain all relevant information from the class assignment. The values for the new column are obtained from the column in the parent table that is associated with the same class.

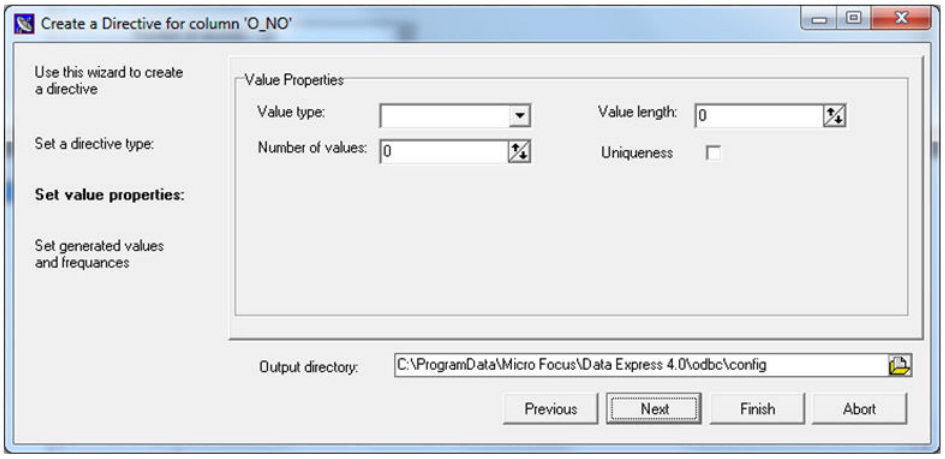
Create a Directive for column, Other file dependency


Create a Directive for column 'O_NO'
Use this wizard to create a directive
Set a directive type:
Set value properties:
Set generated values and frequances
Directive Type
Previous
Next
Finish
Abort

Directive type:
Routine name:
Output directory: C:\ProgramData\Micro Focus\Data Express 4.0\odbc\config

Dictionary These directives require that you specify the dictionary name. For further information on Data Express dictionaries, see the *Masking Routines* section in your *Getting Started with Distributed Data Stores* documentation.

Create a Directive for column, Dictionary

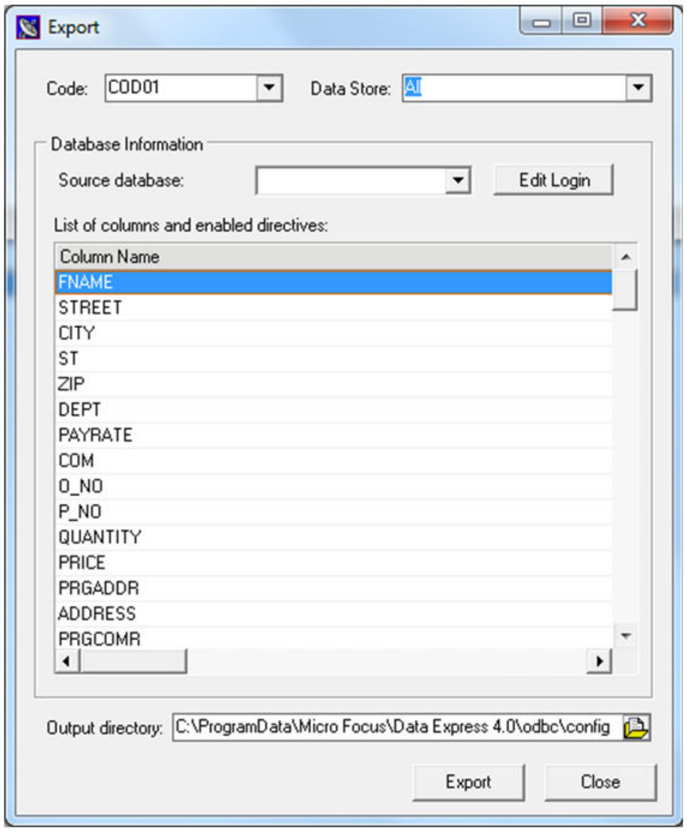

Create a Directive for column 'O_NO'
Use this wizard to create a directive
Set a directive type:
Set value properties:
Set generated values and frequances
Value Properties
Value type:
Value length: 0
Number of values: 0
Uniqueness
Output directory: C:\ProgramData\Micro Focus\Data Express 4.0\odbc\config
Previous
Next
Finish
Abort

Domain value list These directives require that you specify the different values the column can have, and a frequency for each value.

Important: Columns with no directives associated receive the default value for the associated column type.

Export Rules

For distributed data stores only, you can use the Export wizard to export rules at the code set level. The wizard enables you to choose the machine, company, and code, and to specify the output directory. The output directory should be the folder that contains the configuration files.



Note: The **List of columns and enabled directives** is on the Export page as a summary, and is not editable.

Execute Rules

For distributed data stores only, you can execute Data Generation rules. However, to do this you must have a run-time knowledge base. A run-time knowledge base is tables in the same database instance that contains the target tables to be populated, or in a separate instance if you use a centralized run-time knowledge base. Details about run-time knowledge base creation, including a centralized run-time knowledge base, are described in the DBA Tasks for Extension Technology section of your *Getting Started with Distributed Platforms* documentation.

Once the run-time knowledge base is in place, the execution of rules is done by executing the `dxestart` command. See *The dxestart Command* topic in your *Getting Started with Distributed Platforms* documentation.

Data Generation Jobs

For Data Express for z/OS only, the Work with Jobs area contains information about the function related to the execution of jobs for test environment generation and is part of the Data Builder module. Jobs are submitted using the Work with Jobs area of Data Builder.

For more information about the Work with Jobs area, see *Work with Jobs* in the *Front End Guide*.

Introduction to Data Generation Jobs

To access the Work with Jobs area:

1. Start Data Builder by clicking **Start > All Programs > Micro Focus Data Express 4.0 > Data Builder**.
2. Click ▶ (Work with Jobs) in the toolbar of the **Data Builder** main window.

Job Creation

This section describes the procedures for creating jobs relevant to data generation.

Data Generation Jobs

After all files have been allocated for extraction, launching the Data Generation job is the final step in creating the test environment.

To create a Data Generation job:

1. Create a new Data Generation job from the Work with Jobs area in the Data Builder module. See *Job Creation* in the *Front End Guide* for basic instructions. The **Secondary Options** window for the Data Generation job is displayed.
2. Specify the name of the `Code` from the corresponding drop-down list. If desired, specify the range of steps for which you want to launch the elaboration.
3. Click **OK** to submit the job.
4. Wait for the data generation phase to be completed.

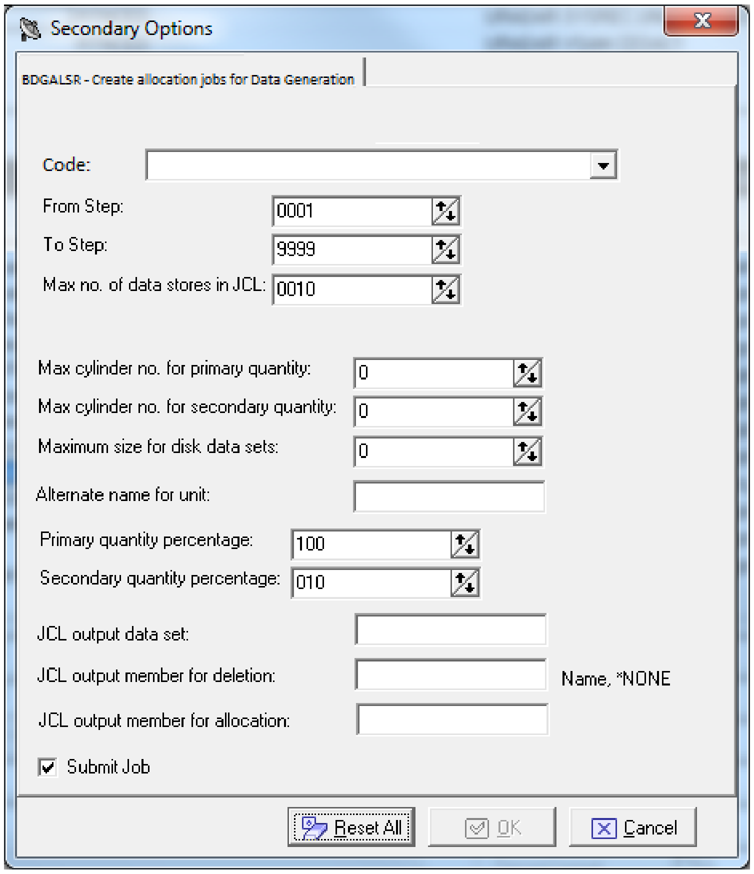
Create Allocation Jobs

The Create Allocation job executes a process to create a JCL that will delete and allocate all the output for the data set associated with the data Generation according to a given code.

Job Creation

To create a Create Allocation job for Data Generation:

1. From the **Work with Jobs** window, click **New**.
2. In the List of Jobs section, select **Create Allocation Jobs for Data Generation**.
3. In the General section, select the appropriate `Machine Id` and `Company name`.
4. Click **Apply** to submit the job. The **Secondary Options** window appears:



Code:	
From Step:	0001
To Step:	9999
Max no. of data stores in JCL:	0010
Max cylinder no. for primary quantity:	0
Max cylinder no. for secondary quantity:	0
Maximum size for disk data sets:	0
Alternate name for unit:	
Primary quantity percentage:	100
Secondary quantity percentage:	010
JCL output data set:	
JCL output member for deletion:	Name, *NONE
JCL output member for allocation:	
Submit Job	✓

5. In the **Secondary Options** window, select the required `Code` value, and specify values for the following fields:

From Step / To Step	Range of values for a subset of steps for the selected code.
Max no. of files in JCL	Maximum number of files you want process in a single JCL. If your code exceeds this limit, a new JCL is created for the other files. The default value of the parameter is 2000.
Max cylinder no. for primary quantity	Maximum number of cylinders that can be assigned to a primary quantity. The value of 0 indicates to use the default value of 4300 cylinders.
Maximum cylinder No. for secondary quantity	Maximum number of cylinders that can be assigned to a secondary quantity. The value of 0 indicates to use the default value of 4300 cylinders.
Maximum size for disk data sets	Maximum number of cylinders (not for the data set) to be allocated on an alternative unit. If the primary quantity exceeds this dimension, the data set will be allocated on an alternative unit.
Alternative name for unit	Name of the alternative unit where you want to allocate the data set that exceeded the **Maximum size for disk data sets** value.
Primary quantity percentage	Percentage of cylinder you want to assign at the primary quantity in regard to the total dimension of output data set. The default value is 10%.

Secondary quantity percentage	Percentage of cylinder you want to assign at the secondary quantity in regard to the total dimension of primary quantity. The default value is 15%.
JCL output data set	Library where the generated JCLs will be stored.
JCL output member for deletion	First part of the name of the generated JCLs (the second part will be a 3-digit progressive number starting from 001).
JCL output member for allocation	First part of the name of the generated JCLs (the second part will be a 3-digit progressive number starting from 001).
Submit Job	Indicator of whether to submit the newly created JCLs.

JCL Information

The JCLs are built starting from the CBDGALS1 and CBDGALS2 skeletons, which can be customized.

The structure of the skeletons comprises a starting part and an ending part that are repeated once in the created JCLs, and a central part that is delimited by the following keywords:

```
//* SINGLE FILE ELABORATION and
//* SINGLE FILE END ELABORATION, that is repeated for each file.
```

In the skeleton, some variables are used and are then changed into the JCL values using the values stored in the Knowledge Base or included in the launch parameters.

The following table describes the used variables:

Variable	Description
`&TYPE`	Name of the sequential data set (output of extraction process)
`&PCYL`	Primary quantity
`&LRECL`	Record length
`&UNIT`	SYSDA or the value put in the launch screen, depending on the above-described rules
`&CYL`	`TRK` if the space is less than a cylinder, or `CYL` otherwise
`&SCYL`	Secondary quantity
`&JOBN`	It is actualized with the JCL name

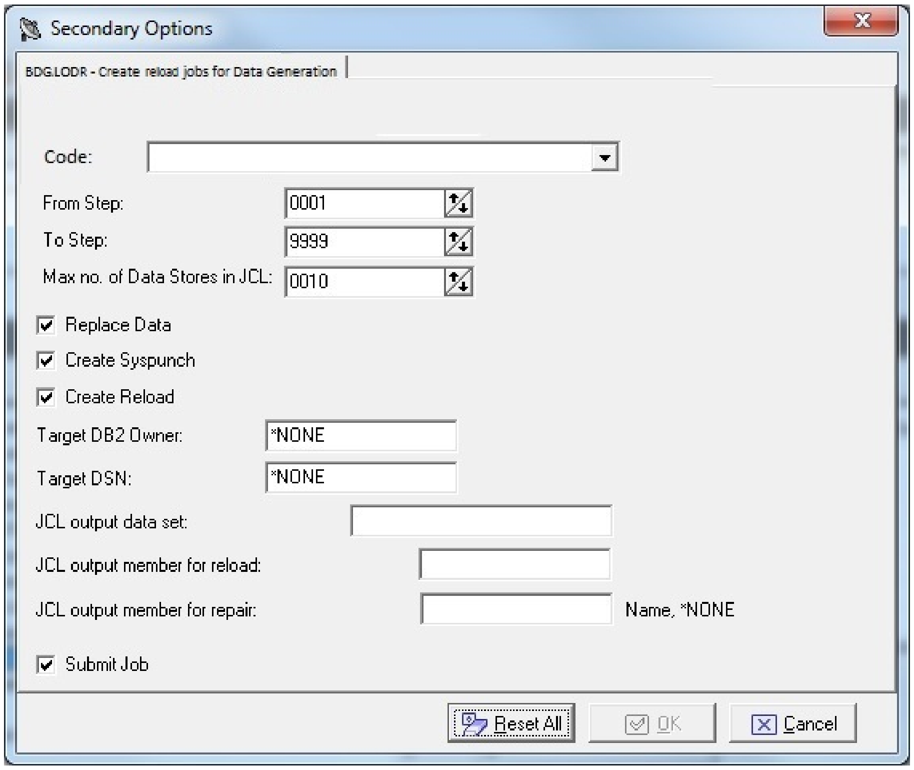
Create Reload Jobs

This function allows you to create a JCL to reload all DB2 tables with direct access included in a code.

Job Creation

To create a `Create Reload` job:

1. From the **Work with Jobs** window, click **New**.
2. In the List of Jobs section, select **Create Reload Jobs for Data Generation**.
3. In the General section, select the appropriate `Machine ID` and `Company name`.
4. Click **Apply** to submit the job. The `Secondary Options` window appears.


Secondary Options

BDG.LODR - Create reload jobs for Data Generation

Code:

From Step: 0001

To Step: 9999

Max no. of Data Stores in JCL: 0010

☑ Replace Data

☑ Create Syspunch

☑ Create Reload

Target DB2 Owner: *NONE

Target DSN: *NONE

JCL output data set:

JCL output member for reload:

JCL output member for repair: Name, *NONE

☑ Submit Job

5. In the **Secondary Options** window, select the required `Code` value, and specify values for the following fields:

From Step / To Step	Range of values for a subset of steps for the selected code.
Max no. Data Stores in JCL	Maximum number of data stores you want process in a single JCL. If your method exceeds this limit, the function creates a new JCL for the other files. The default value is 2000.
Replace Data	Indicator of whether data replacement in the target database is enabled. If not selected, data items are added without deletion.
Create Syspunch	Indicator of whether to create the part of the JCL performing the SYSPUNCH.
Create Reload	Indicator of whether to create the part of the JCL performing the reload.
Target DB2 Owner	Name of the target DB2 owner. If this parameter is not specified, this value is the same as cataloged in Data Express.
Target DSN	Name of the target database subsystem. If this parameter is not specified, this value is the same as cataloged in Data Express.
JCL output data set	Library where the generated JCLs will be stored.
JCL output member for reload	Name of the generated JCLs used for reload.
JCL output member for repair	Name of the generated JCLs used for repair.
Submit Job	Indicator of whether to submit the newly created JCLs.

JCL Information

The JCLs are built starting from CBDGLOD1 and CBDGLOD2 skeletons, which can be customized. The CBDUNLD2 skeleton is an alternative skeleton where the UNLOAD utility is used instead of the DSNTIAUL utility in order to create SYSPUNCHs.

The structure of the skeleton comprises a starting and ending part that are repeated once in the created JCLs, and a central part that is delimited by the following keywords:

```
//* SINGLE FILE ELABORATION and
//* SINGLE FILE END ELABORATION, that is repeated for each file.
```

In the skeleton some variables are used, and they are changed into the JCL values using the values stored in the Knowledge Base or included in the launch parameters.

The following table describes the used variables:

Variable	Description
`&NUM`	Numeric progressive
`&UNLOUTNAM`	Name of the sequential data set (output of extraction process)
`&OWNAME`	Owner of the original table
`&FILNAME`	Name of the original table
`&DBNAME`	Database name of the original table
`&TSNAME`	Tablespace name of the original table
`&UNLINPNAM`	Name of the unload as specified in the Work with files area
`&STEP`	Step name for the SYSPUNCH generation (automatically generated)
`&STP1`	Step name for the MISYSPUN program call (automatically generated)
`&STP2`	Name of the reload step (automatically generated)
`&SYSPARM`	It can be ADD,YES, *number of sortkeys* if no replace parameter, REP,YES, *number of sortkeys* if replace parameter
`&DSNNAME`	Name of the input DSN
`&METHOD`	Name of the code
`&JOBN`	Name of the job actualized with the JCL name

Index