

CODD Metadata Processor

Version 1.0

Database Systems Lab

Supercomputer Education & Research Centre

and

Department of Computer Science & Automation

Indian Institute of Science, Bangalore, India

February 2015

©Indian Institute of Science, Bangalore, India

[CODD 1.0 Documentation Index](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

DOCUMENTATION INDEX

(version 1.0 - February 2015)

1. [Introduction](#)
2. [Installation Instructions](#)
3. [Usage Guide](#)
 - o [Database Connection Setup](#)
 - o [Choosing Relations for Dataless Operations](#)
 - o [Metadata Construct Mode](#)
 - [DB2](#)
 - [Oracle](#)
 - [SQL/MX](#)
 - [PostgreSQL](#)
 - [Graph Histogram](#)
 - o [Metadata Retain Mode](#)
 - o [Metadata Transfer Mode](#)
 - o [Metadata Scaling Mode](#)
 - [Size based Scaling](#)
 - [Time based Scaling](#)
 - o [CODD through Command Line](#)
4. [License Information](#)
5. [Development Team](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

INTRODUCTION

Welcome to the [CODD Metadata Processor software](#) developed at the [Database Systems Lab, Indian Institute of Science](#), Bangalore, India, by [this team](#).

The COConstructing Dataless Databases (CODD) tool is an easy-to-use graphical tool for the automated creation, verification, retention, scaling and porting of database metadata configurations. It is written entirely in Java and operational on a suite of industrial-strength database engines (currently DB2, Oracle, SQL Server, SQL/MX and PostgreSQL are supported).

Effective design and testing of database engines and applications is predicated on the ability to easily construct alternative scenarios with regard to the database contents. A limiting factor, however, is that the time and/or space overheads incurred in creating and maintaining these databases may render it infeasible to model the desired scenarios. CODD is a graphical tool that attempts to alleviate these difficulties through the construction of "dataless databases". Specifically, CODD implements a unified visual interface through which databases with the desired metadata characteristics can be efficiently simulated without persistently generating and/or storing the data.

The following are the different modes of operations available in the CODD tool:

1. **Metadata Construct Mode:** It is the most potent mode, where the user is allowed to directly create (from scratch) or edit an existing metadata without the existence of any prior database instance. CODD validates the user input before updating it to the database catalogs for legality and consistency checks. CODD also provides a graphical interface to alter the data distribution of the columns by reshaping the histogram graph via changing the bucket boundaries using the mouse.
2. **Metadata Retain Mode (Data Drop Mode):** This mode applies to pre-existing databases, where the storage occupied by the database can be fully reclaimed without affecting the database metadata.
3. **Metadata Transfer Mode:** This mode allows the user to port the statistical metadata across different engines, thereby facilitating the comparative study of different database systems.
4. **Metadata Scaling Mode:** CODD supports two kinds of scaling methods. First, size based scaling where the relations size (in bytes) is scaled linearly. Second, time based scaling where the relations size is scaled so that the time (cost) of executing a query workload on the database is scaled linearly.
 - a. Size based scaling - relations' size (in bytes) are scaled linearly. This is in line with the TPC-H based scaling method.
 - b. Time based scaling - relations' size are scaled such that the time (cost) of executing a query workload on the database is scaled linearly.

The name of the tool comes from the English word 'cod', whose archaic meanings include "empty shell" as well as "fake", both of which are appropriate to our dataless context. It also coincidentally happens to be the name of [Edgar Codd](#), the father of relational databases.

[Documentation Home](#)

CODD Metadata Processor

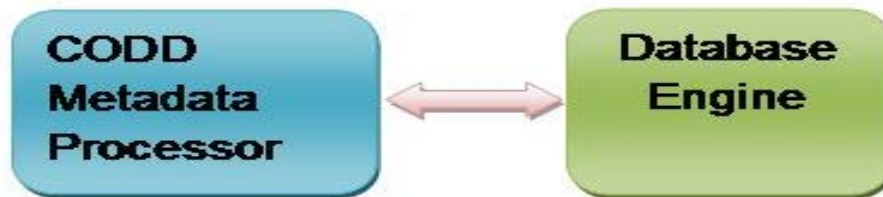
©Indian Institute of Science, Bangalore, India

INSTALLATION INSTRUCTIONS

Overview

There are two processes involved in CODD, as shown in the figure:

- (a) the **CODD Metadata Processor**, through which the user creates the desired metadata characteristics using the dataless modes;
- (b) the **Database Engine**, which stores the metadata of the relational tables.



These two processes can all execute on the same machine or on different machines. The machine in which the CODD Metadata Processor is run should support Java compilation and execution. A few third-party libraries for database connection and Time Scaling are required for CODD to function – the details are given in [License Information](#) (for convenience, this support software (other than the library for Time Scaling) is included with the CODD code-base in the **full** version).

CODD is completely written in Java and should, in principle, operate in a platform-independent manner. It has been successfully tested on the following system and database environments:

System platforms:

- (a) **Windows 64-bit:** Windows Vista Business 64-bit, Sun Ultra 24 Intel® Core™2 QuadCore 3GHz, 8 GB Ram,
- (b) **Windows 64-bit:** Windows 7 Professional, Intel® Core™i3 2.10GHz, 4 GB RAM
- (c) **Windows 64-bit:** Windows 8/8.1, Intel® Core™i5 2.60GHz, 6 GB RAM

Database engines: DB2 9/10, Oracle 11g, SQL Server 2008, SQL/MX 3.1, PostgreSQL 9.3

Installation Steps

I Setup the Database Engine

1. Install a database engine. Database Engines can be obtained from the vendor websites and installed by following their installation steps.
2. Populate the database with data. This step may be needed only for certain modes of CODD (like Metadata Retain mode). Details can be found when we explain the different modes.
Note: CODD can be used with generic relational database schemas and SQL query templates. The illustrative examples in the CODD documentation are with respect to the [TPC-H benchmark](#).

II Download the CODD Software

1. From [CODD Download](#), download the CODD code (version 1.0) as a zip file. Extract its contents. A directory called **CODD1.0** in which the entire code-base is contained will be created. All paths mentioned in this document and the supporting documentation are with reference to this directory.
2. Contents of the extracted zip file:
The extracted CODD1.0 folder contains the source code, CODDDoc folder, folder named Libraries, which contains the associated library files and another folder named PostgreSQLModifications, which contains files required for PostgreSQL to work with CODD. (see [PostgreSQL](#) for details) All the necessary library files are included in the Libraries folder except [SuanShu](#), which is needed for Time Scaling. Obtain the [SuanShu](#) library, license file and put it in the lib folder. License related information about the tool and libraries can be found in [License Information](#)
3. Read through the documentation given in the [CODDDoc](#) directory.

III Getting Started with the tool

Prerequisite: Java 1.7 or above. It must be present in the CLASSPATH Environment variable. i.e. Command "java -version" in the command prompt should give the version number more than 1.7.

1. Compiling and running the Main.java present in the folder `iisc/dsl/codd/` starts the tool. We suggest the user to import CODD folder as a project in [Netbeans/Eclipse](#) IDE and execute the project to start the tool.
2. Running Main.java takes the user to a screen as in Figure 1, indicating that the tool has started successfully.

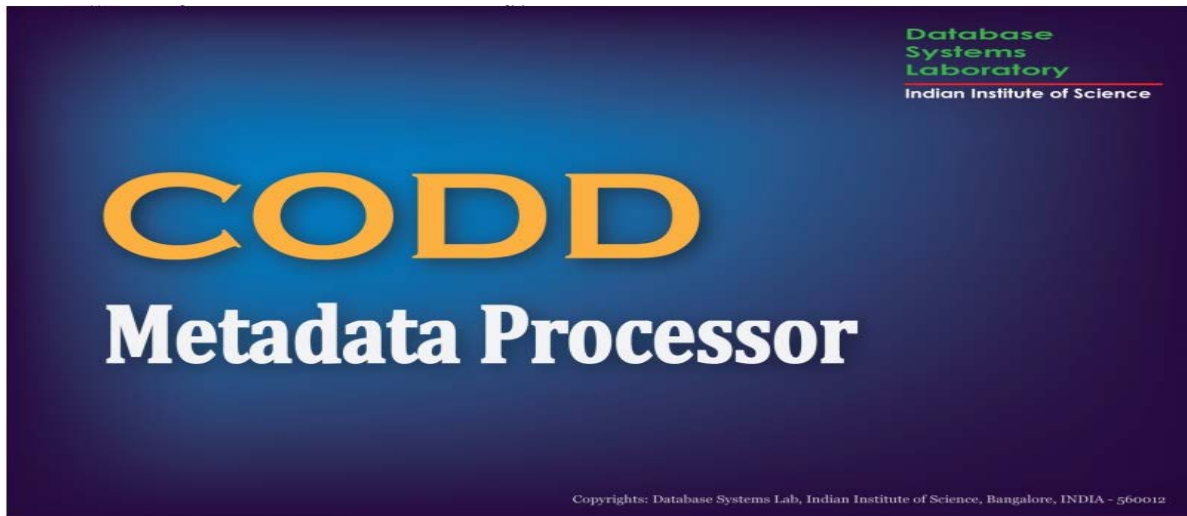


Figure 1 - CODD Initial Screen

To learn how to use CODD, proceed to the [usage guide](#).

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

USAGE GUIDE

The motivation and conceptual framework underlying the CODD Metadata Processor software are presented in this [DBTest 2012 paper](#). This section describes various modes of operations in CODD explained with the GUI controls.

The subsections are as follows:.

- 1) [Database Connection Setup](#)
- 2) [Choosing Relations for Operation](#)
- 3) [Metadata Construct Mode](#)
- 4) [Metadata Retain Mode](#)
- 5) [Metadata Transfer Mode](#)
- 6) [Metadata Scaling Mode](#)

The first two sections describe the common setup in using the different modes of CODD. The other sections describe the dataless modes.

Execution of the tool information is logged in a log file (a log file will be created every time the tool is started), which can be found in the 'Logs' directory. 'Logs' directory is created in the current working directory from which the tool has started the execution. This log file information would be helpful in debugging / fixing bugs.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

DATABASE CONNECTION SETUP

On starting the tool CODD, the user will be shown with CODD initial screen (Figure 2).

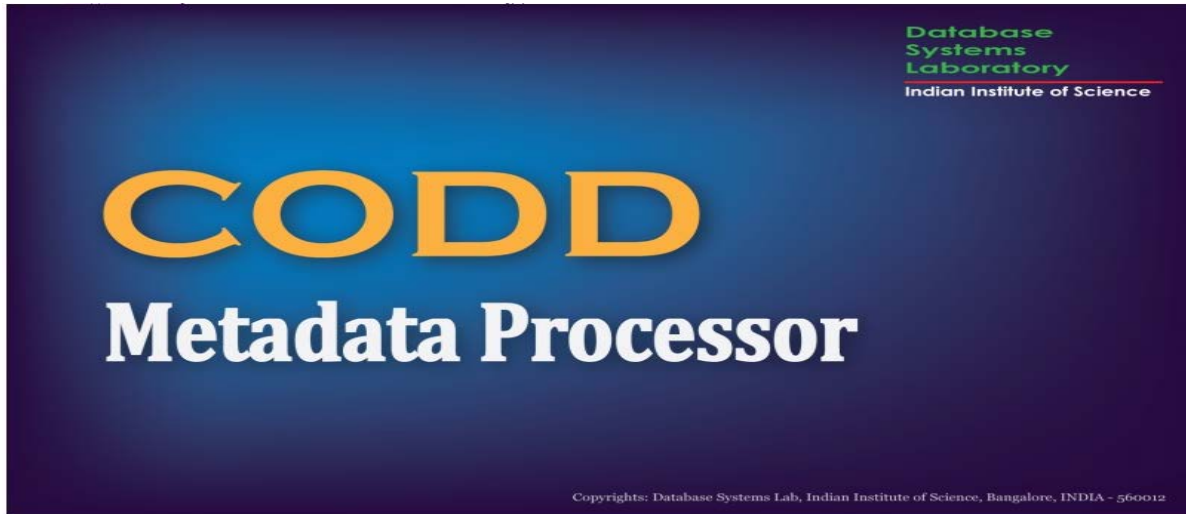


Figure 2 - CODD Initial Screen

User should automatically be redirected to the Connection Information window (Figure 3), where the user has to enter the database connection parameters. The frame validates the form entries and uses it to make a connection with the database engine. This database connection is used to read/write the metadata from/to the database engine catalogs, which is the core operation of CODD dataless modes.

The image shows a window titled 'Database Connection Properties'. It has a light blue border and standard window controls (minimize, maximize, close). At the top, there is a dropdown menu labeled 'Choose From Pre-Saved Connections:' with 'oracleTpch' selected. Below this, there are two main sections: 'Database Server Details' and 'Schema Details'. 'Database Server Details' includes fields for 'Engine:' (set to 'ORACLE'), 'Machine Name / IP:' (set to 'localhost'), 'Port:' (set to '1521'), and 'Server Instance:' (a greyed-out field). 'Schema Details' includes fields for 'Database:' (set to 'XE'), 'Catalog:' (a greyed-out field), 'Schema:' (set to 'SYSTEM'), and 'Version:' (a greyed-out field). Below these sections is a 'Credentials' section with 'User Name:' (set to 'SYSTEM') and 'Password:' (a masked field with seven dots). At the bottom, there are four buttons: 'Save Config', 'Refresh', 'Delete Config', and 'Connect'. In the bottom right corner, there is a logo for 'CODD Metadata Processor'.

Figure 3 - Database Connection Information

CODD also provides features to save and load database connection configurations.

1. Clicking on **Save Config** button, shows a "**Save**" window, where the user has to give a file name (which is the configuration name [Provide an unique name]) and click **Save**. This saves the current connection information provided in the form fields to the file after validating the form contents (No empty fields).
2. Saved database configurations are listed (by its unique configuration name) on the **Connection Properties combo box**. The user can select any one of the saved configuration to use it. Selecting any of the configuration, fills the form fields with the saved connection information contents.
3. **Refresh** button resets the connection form fields.
4. **Delete Config** button removes the selected configuration from saved configurations.
5. Clicking on **Connect** button, validates the form fields, makes connection to the database and returns the [Choosing Relations for Operation](#) window to the user to select the set of relations to operate on.

CODD errors / exceptions given at this stage and troubleshooting information are as follows:

S.No	Error / Warning/ Exception	Error / Warning / Exception Information	Action / Troubleshooting Information
1	Error: Entries in the fields are not correct. Please enter the correct values.	Some of the form fields are left empty.	Fill the appropriate connection parameters.
2	Error: Could not get database object. Error: Could not able to connect to database.	CODD is not able to connect to the database with the provided connection parameters.	Connection parameters may be wrong. Check the connection parameters.
3	Warning: Not able to stop auto maintenance of statistics.	CODD is not able to stop the database engine automatic statistics collecting utility.	User can continue with the CODD dataless modes operation. But the updates made in the metadata may be changed by the database engine at any time.
4	Exception: Exception in Connecting to the database /	CODD query given to Database engine has	Look for the log file contents.

Getting the Relation Frame visible / setting AUTO MAINT OFF.	resulted in error.	
--	--------------------	--

Table 1 - Connection Information Error Information

Note: Clicking on **Connect** button also stops/disables the automatic statistics collecting utility on each of the supported database engines after making connection to them. But for PostgreSQL, the user has to manually stop the automatic statistics collecting utility, which can be done by following the procedure given below:

- Open the **postgresql.conf** file
- Changes in **RUNTIME STATISTICS** as follows:
 - track_activities = off
 - track_count = off
- Changes in **AUTOVACUUM PARAMETERS** as follows:
 - autovacuum = off
- Save and Close the file.
- Restart PostgreSQL server.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

CHOOSING RELATIONS FOR OPERATION

After establishing the connection with the database, the relations of the provided 'database user' is shown to the user as in Figure 4. The user can select the relations on which he / she would like to perform different CODD operations.

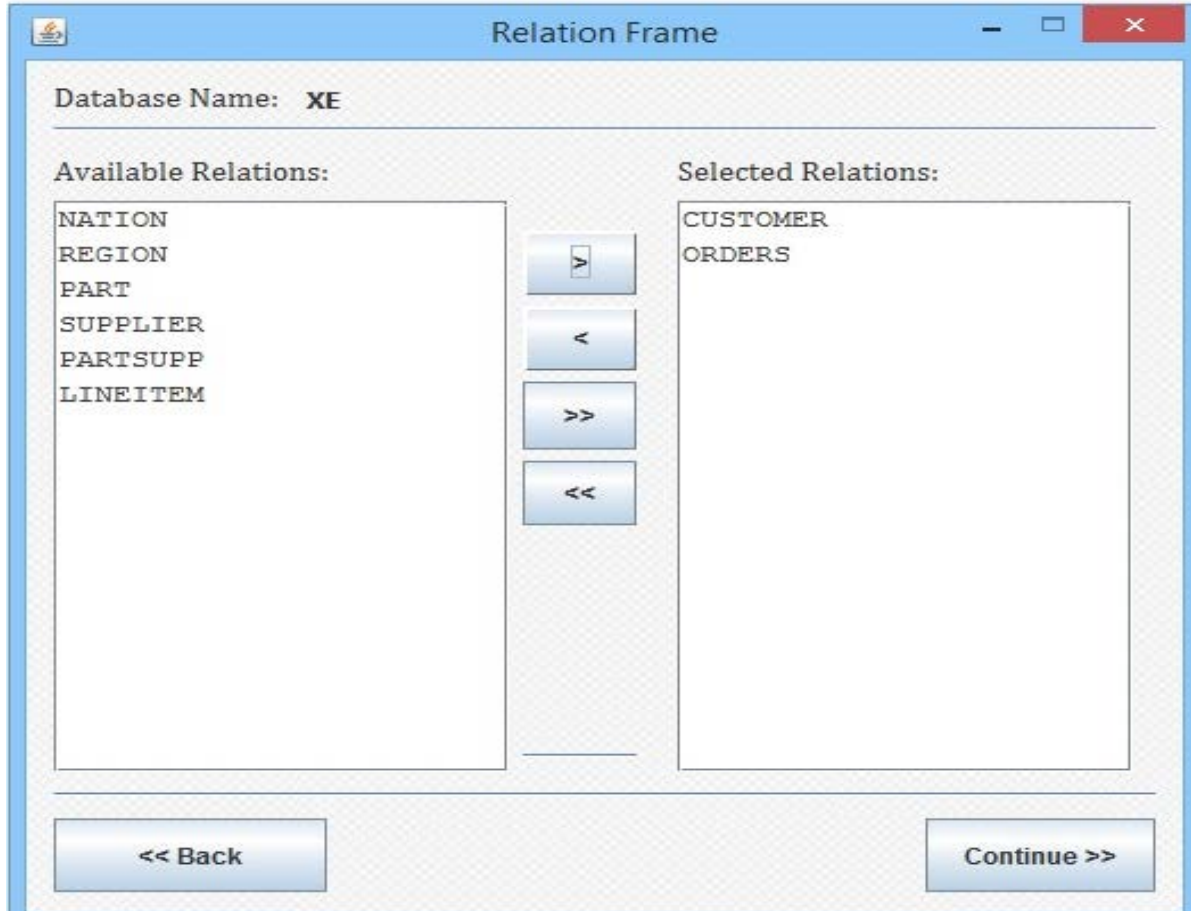


Figure 4 - Choosing Relations for Operations

The left-column **Available Relations** lists 'database user' relations and the user can select relations with the buttons {>, <, >>, <<} to move to the right-column **Selected Relations**.

Button ">" moves the selected left-column entries to the right-column.

Button "<" moves the selected right-column entries to the left-column.

Button ">>" moves the all the left-column entries to the right-column.

Button "<<" moves the all the right-column entries to the left-column.

Clicking on **Continue** button, takes the user to **Mode Selection Window** (Figure 5). **Back** button takes

the user to the Database Connection Information Window.

Note: The relations in the right-column **Selected Relations** will be used in dataless modes. In the Figure 4 relations CUSTOMER, ORDERS are chosen for the CODD operations.

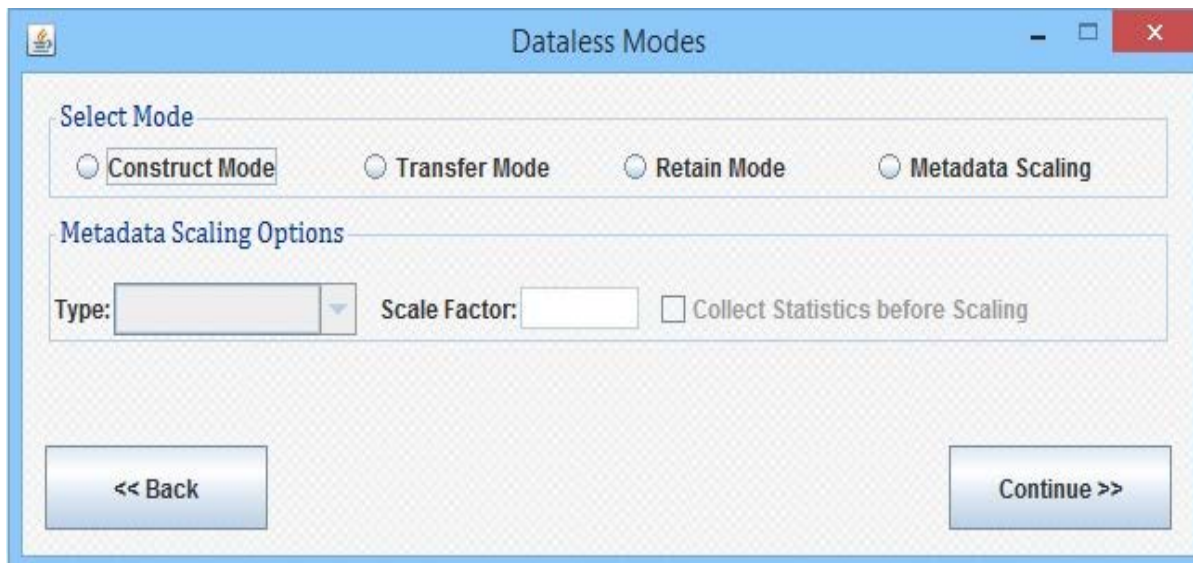


Figure 5 - Mode Selection Window

The mode selection window presents the various modes supported by CODD. The user has to choose any one of them and click **Continue** button to perform the selected operation mode. The operating modes and how to perform the dataless operation are described in the following sections.

[Metadata Construct Mode](#)

[Metadata Retain Mode](#)

[Metadata Transfer Mode](#)

[Metadata Scaling Mode](#)

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA CONSTRUCT MODE

Metadata Construct mode allows the user to create or edit the statistical metadata without requiring presence of any prior data instance. Upon choosing **Construct Mode** option from the Mode Selection Window, the user will be shown the **Construct Mode Window** (Figure 6), where the user has to choose the type of construct mode.

Option **Construct from Scratch** lets the user to construct the metadata statistics from scratch.

Option **Construct from baseline configuration** lets the user to construct the metadata based on the previously transferred metadata(See [Transfer Mode](#) to know about the transfer of metadata to a file).

Click on the **Browse** button to specify the path to the metadata transferred file. The user has to choose the directory in which the transferred files are present. The directory must have a file name named 'dbType' and metadata file for each chosen relation in the **Choosing Relations for Operations** window. The metadata relation file names must be same as the relation names.

Checking the **Modify and Update** check box, uses the transferred metadata file as initial values for the construct mode. If it is unchecked, CODD performs metadata transfer (see [Transfer Mode](#) for details), where the source is the saved metadata files and the target is the connected database.

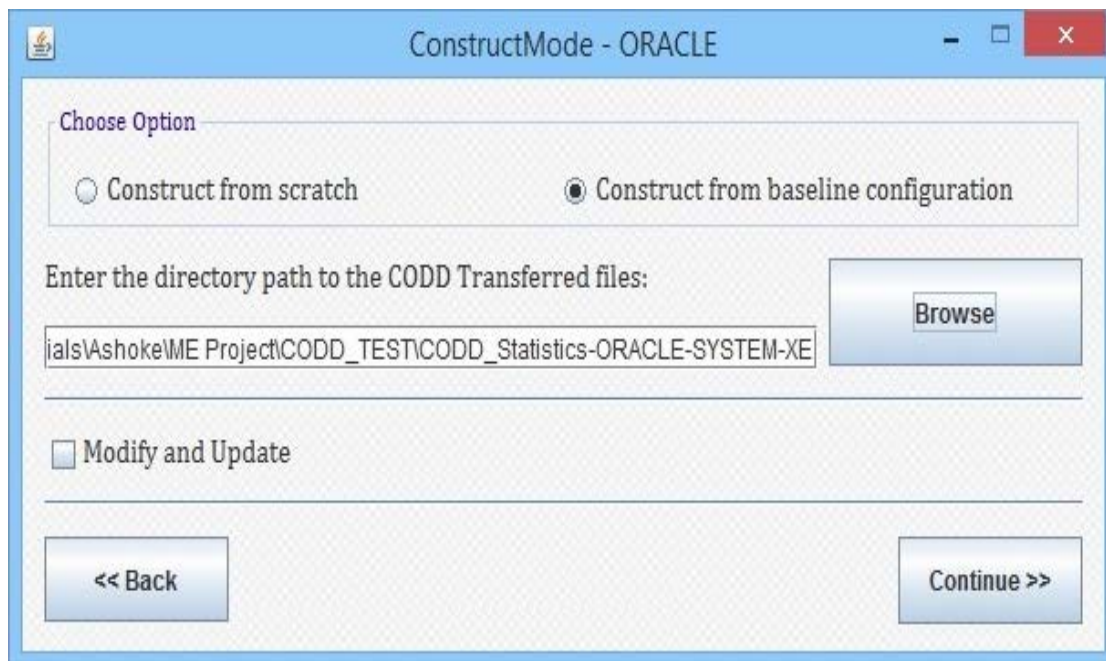


Figure 6 - Metadata Construct Mode Window

Clicking on the **Continue** button takes the user to construct mode window, which is specific to the Database Engine. If Construct from baseline is chosen with **Modify and Update** unchecked, metadata is transferred from the files to the database and a success or failure message is shown to the user.

CODD errors / exceptions which could happen for Construct Mode and troubleshooting information are as follows:

S.No	Error / Warning/ Exception	Error / Warning / Exception Information	Action / Troubleshooting Information
1	Exception: Could not read metadata object dbType file.	Exception in reading 'dbType' file from the path specified.	Make sure the 'dbType' file is present in the specified folder.
2	Exception: Could not read metadata object from the file 'RelationName'.	Exception in reading the metadata file for 'RelationName'.	Make sure the 'RelationName' file is present in the specified folder.

Table 2 - Construct Mode Error Information

Engine Specific Construct Mode

The Construct Mode Window for DB2, Oracle, SQL/MX and PostgreSQL are shown in Figures 7, 8, 9 and 10 respectively. CODD does not support construct mode for SQL Server as the column statistics format is currently proprietary.

The relations selected in [Choosing Relations for Operations](#) window are the relations which can be constructed here. All the metadata (Relational level, Column level [includes Index level]) of these relations are initialized with the default values (if **Modify and Update** is checked, the transferred metadata values are the default values). The construct mode form fields only updates the default values with the provided values and finally it is updated to the database catalogs.

Though the relations / columns metadata can be updated in any order, CODD lets the user to follow an ordering in which the relation / column level metadata must be updated (It is to ensure the consistency of metadata).

Relation level metadata of the selected relations can be updated in any order. Relation level metadata must be updated before updating the column level metadata. The column level metadata fields would be enabled only after updating the relation level metadata.

In case of Foreign Key column, if the Primary Key column is also chosen to construct, then the FK column can be constructed only after the PK column.

Index metadata can be updated along with column level metadata. The index must have been created already (**Update index statistics** is checked automatically if index exists on the selected column), only then the metadata will be updated.

In case of composite index (Index on multiple columns), the first key column of the index column level metadata must be used to update the index level metadata.

The relations selected in [Choosing Relations for Operations](#) window is listed in the drop down box present just after the **Relation Name** label. The user can select any of the relation, which enables the relation

level metadata fields. The user can fill in the relation level metadata values. Clicking on **Update** button (present after Overflow [DB2] / Avg Row Len [Oracle] / Cardinality [SQL/MX] field / Number of Pages [PostgreSQL]) updates the default relation level metadata values with the provided values and loads the list of columns of the selected relation in to the drop down box present just after the **Attribute Name** label. The user can choose columns and update the column level metadata. After filling in the values, clicking on the **Update Column** button to update the default column [index] level metadata values. The user can repeat this procedure to update other relations and columns metadata.

The combo box for the relations (present just after the Relation Name label) and columns (present just after the Attribute Name label) lets the user to select and update (construct) only the required columns / relations metadata to construct. For other columns the default values are used. The button **Construct** present at the bottom is enabled always, so that as soon as the user updated the required column and relation level metadata, he/she can click on the **Construct** button. This button does the actual construction of metadata by updating the database catalogs with the provided metadata values. Currently CODD validates metadata input only for the attributes and relations for which the **Update** button action is performed. If not all the relations or attributes are updated and **Construct** button is clicked, then there is a possibility of metadata validation error with the non updated attributes and relations. A warning message is shown to the user whenever **Construct** button is used. We suggest the user to go through all the attributes one by one and validate the input values by performing **Update** action to avoid construct mode failure.

Actual updates to database catalogs is done either after clicking on **Construct** button or after updating the metadata for all chosen relations and their columns. Closing the form exits the construct mode, without updating anything. The database catalog remains in the same state as it was before. **Reset Values** button resets the column level metadata fields with the default values.

The screenshot shows the 'DB2 Construct Mode' window. At the top, the 'Relation Name' is set to 'CUSTOMER'. Below this, there are input fields for 'Please Input Values' including 'Cardinality' (150000), 'NPAGES' (3330), 'Overflow' (0), and 'FPAGES' (3330), with an 'Update' button. The 'Attribute Name' is set to 'C_ACCTBAL'. Below this, there are input fields for 'Please Input Values' including 'Cardinality (Distinct)' (131072), 'Null Count' (0), 'High2Key' (0000009999.96), 'Low2Key' (0000009999.96), and 'Avg. Col. Len.' (8). There are tabs for 'Frequency Value' and 'Quantile Value'. The 'Set Value' section has a 'Set the number of Buckets' field (50) and a 'Create' button. Below it is an 'Upload From File' button and a 'Write to File' checkbox. A table displays 'COLVALUE' and 'VALCOUNT' data. The 'Index Statistics' section shows a message: '[There is a system generated index on this attribute.]' and a 'Show Quantile Value Graph' button. At the bottom, there are input fields for 'Index Cardinality', 'NLeaf', 'NLevels', 'Density', 'NumRID', and 'NumEmptyLeaves'. The bottom right corner has buttons for '<< Back', 'Reset Values', 'Update Column', and 'Construct'.

COLVALUE	VALCOUNT
+0000000003449.33	150
+0000000004464.79	150
-0000000000981.63	75
-0000000000975.18	75
-0000000000969.78	75
-0000000000964.54	75
-0000000000964.13	75
-0000000000963.26	75
-0000000000956.07	75
-0000000000948.94	75
-0000000000942.53	75
-0000000000940.67	75
-0000000000937.70	75
-0000000000936.66	75

Figure 7 - Metadata Construct Mode DB2 Window

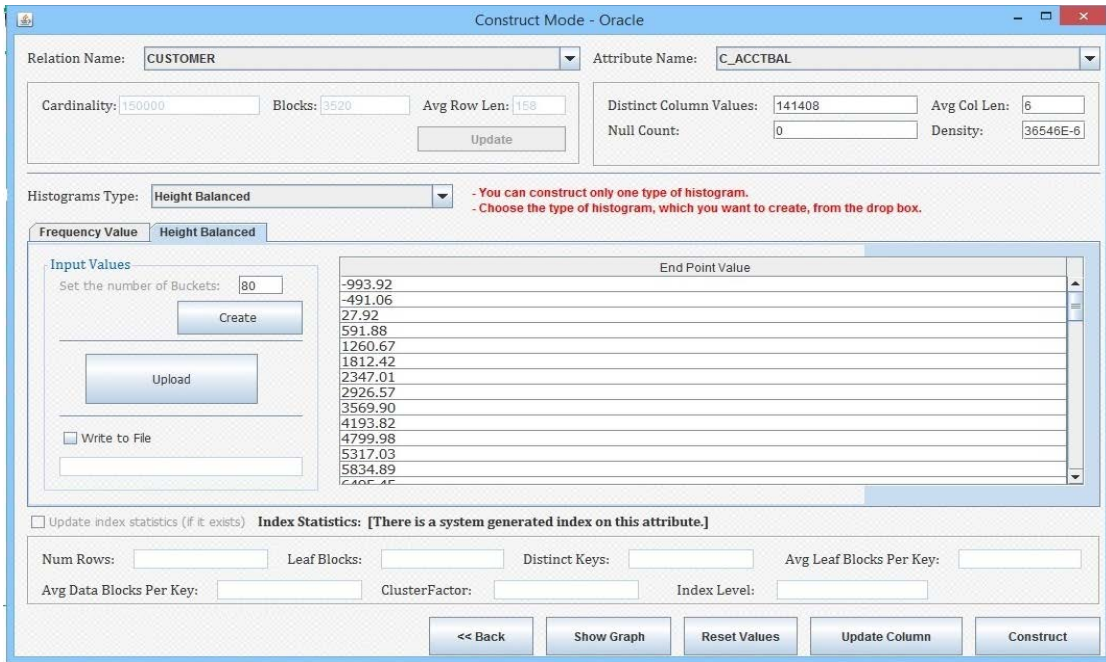


Figure 8 - Metadata Construct Mode Oracle Window

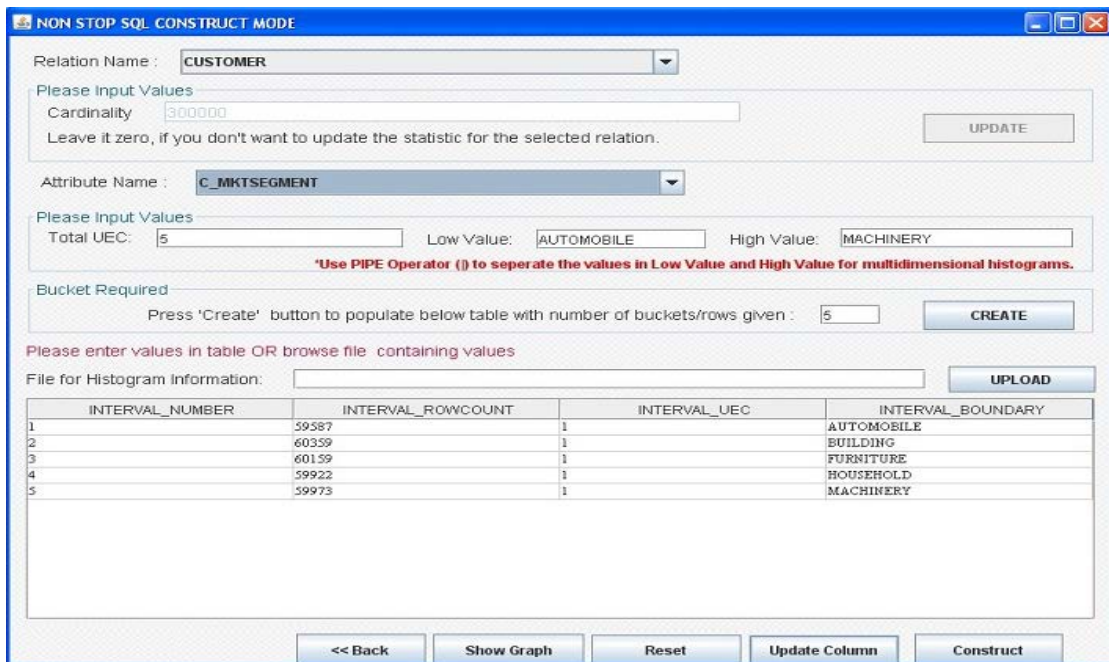


Figure 9 - Metadata Construct Mode SQL/MX Window

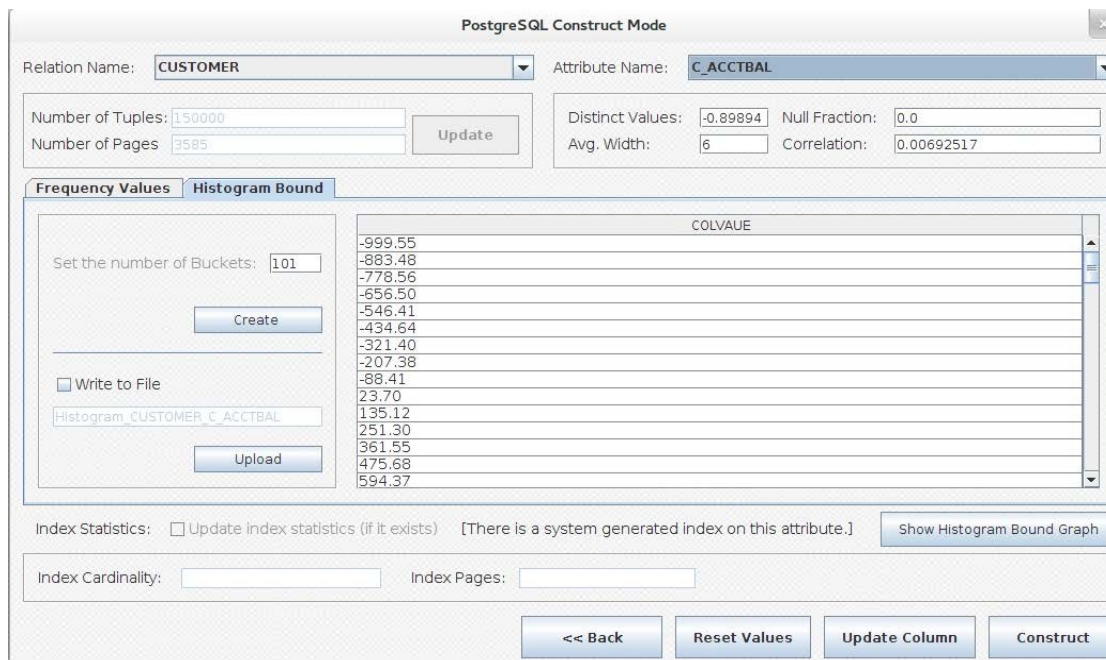


Figure 10 - Metadata Construct Mode PostgreSQL Window

- Check box **Write to File** allows the user to write the histogram in to a file. The file name has to be specified in the textbox below the check box.
- **Upload** button allows the user to load the pre saved histogram file in to the table. **Write to File** and **Upload** options are given for both Frequency histogram and Quantile / Height Balanced histogram.
- **Show Graph** allows the user to modify the data distribution (Quantile / Height Balanced Histogram) through the graphical interface, which is described in section [Graph Histogram](#). Initial values must be present in the Quantile value / Height Balanced histogram table to start the graph histogram.
- After each update (relation / column level), the CODD tool checks for the consistency of input metadata and reports to the user if there is an inconsistency in the metadata values with an appropriate error message. The constraint checks and the other limitations on the construct mode are described in [DB2](#), [Oracle](#), [SQL/MX](#) and [PostgreSQL](#) for the respective database engines.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA CONSTRUCT MODE - DB2

CODD supports the following data types in the Construct Mode for DB2.

1. **BOOLEAN**
2. **VARCHAR**
3. **NUMERIC**
4. **INTEGER**
5. **REAL**
6. **DECIMAL**
7. **DOUBLE**
8. **FLOAT**
9. **DATE**
10. **TIME, TIMESTAMP**
11. **CHARACTER**

CODD Construct mode can be used only for the relations which has only the above list of data types in their columns.

The Relation, Column and Index level metadata of the construct mode form are described in the following tables with the constraint associated with each field and the default values. Each field also specifies the data type of it. The input value in the form field must be in between the smallest and largest value of the data type and adhere to the constraints associated with it. Violating any of the constraints reports an error to the user.

We suggest the user to follow the constraints when they fill the values in the form field.

Metadata Field	Default Value	Consistency Constraints
Cardinality	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or $0 \leq \text{Cardinality} \leq 9223372036854775807$ <u>Consistency Constraints:</u> NIL
NPages	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or $\text{NPages} \geq 0$ <u>Consistency Constraints:</u> 1) $\text{NPages} \leq \text{Cardinality}$
FPages	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or $\text{FPages} \geq 0$ <u>Consistency Constraints:</u> 1) $\text{FPages} \geq \text{NPages}$
Overflow	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or $\text{Overflow} \geq 0$ <u>Consistency Constraints:</u> NIL

Table 3 - DB2 Relation Level Metadata Fields

Metadata Field	Default Value	Consistency Constraints
Column Cardinality	-1	<p><u>Structural Constraint:</u> BIGINT</p> <p>1) -1 or Column Cardinality ≥ 0</p> <p><u>Consistency Constraints:</u></p> <p>1) Column Cardinality \leq Cardinality</p> <p>2) Column Cardinality = Cardinality, if Unique column</p> <p>3) Column Cardinality \leq Primary Key Column Cardinality, if Foreign Key Column</p>
Null Count	-1	<p><u>Structural Constraint:</u> BIGINT</p> <p>1) -1 or Null Count ≥ 0</p> <p><u>Consistency Constraints:</u></p> <p>1) Null Count \leq Cardinality</p> <p>2) Null Count = 0, if NotNULL column</p> <p>3) Null Count + Column Cardinality \leq Cardinality</p>
High2Key	Empty String	<p><u>Structural Constraint:</u> VARCHAR(254)</p> <p>1) Empty String or Value whose type is same as column type</p> <p><u>Consistency Constraints:</u></p> <p>1) High2Key \leq PK.High2Key, if Foreign Key Column</p>
Low2Key	-1	<p><u>Structural Constraint:</u> VARCHAR(254)</p> <p>1) Empty String or Value whose type is same as column type</p> <p><u>Consistency Constraints:</u></p> <p>1) Low2Key \geq PK.Low2Key, if Foreign Key Column</p>
High2Key Low2Key	-	<p><u>Consistency Constraint:</u></p> <p>1) High2Key $>$ Low2Key, if Column Cardinality > 3</p>
Avg Col Len	-1	<p><u>Structural Constraint:</u> INTEGER</p> <p>1) -1 or Avg Col Len ≥ 0</p> <p><u>Consistency Constraints:</u></p> <p>NILL</p>
Frequency Histogram	Null	<p><u>Structural Constraint:</u></p> <p>1) COLVALUE : VARCHAR(254) : Value whose type is same as column type</p> <p>2) VALCOUNT : BIGINT : -1 or VALCOUNT ≥ 0</p> <p><u>Consistency Constraints:</u></p> <p>1) VALCOUNT must be increasing.</p> <p>2) Sum (VALCOUNT's) \leq Cardinality</p> <p>3) Number of buckets \leq Column Cardinality</p> <p>4) There can be one entry greater than high2key and one entry lesser than low2key.</p>
Quantile Histogram	Null	<p><u>Structural Constraint:</u></p> <p>1) COLVALUE : VARCHAR(254) : Value whose type is same as column type</p> <p>2) VALCOUNT : BIGINT : -1 or VALCOUNT ≥ 0</p> <p>3) DISTCOUNT : BIGINT : Empty or DISTCOUNT ≥ 0</p> <p><u>Consistency Constraints:</u></p> <p>1) COLVALUE, VALCOUNT, DISTCOUNT must be increasing.</p> <p>2) DISTCOUNT_i \leq VALCOUNT_i , where i indicates the ith bucket of the</p>

histogram.

3) The largest COLVALUE's VALCOUNT must be equal to Cardinality.

4) The largest COLVALUE's DISTCOUNT must be equal to Column Cardinality.

Optional Consistency Constraints:

These constraints are informed to the user, but not enforced strictly.

1) There can be one unique entry greater than High2key and one unique entry lesser than Low2key.

2) The VALCOUNT of a Quantile Histogram bin b must be greater than the sum of VALCOUNTs in the Frequency Histogram whose COLVALUE is less than the b's COLVALUE.

Table 4 - DB2 Column Level Metadata Fields

Metadata Field	Default Value	Consistency Constraints
Index Cardinality	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or Index Cardinality >= 0 <u>Consistency Constraint:</u> 1) Index Cardinality = Cardinality
NLeafs	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or NLeafs >= 0 <u>Consistency Constraint:</u> NILL
NLevels	-1	<u>Structural Constraint:</u> SMALLINT 1) -1 or NLevels >= 0 <u>Consistency Constraint:</u> 1) NLevels <= NLeafs
Density	-1	<u>Structural Constraint:</u> INTEGER 1) -1 or 0 <= Density <= 100 <u>Consistency Constraint:</u> NILL
Num RIDs	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or Num RIDs >= 0 <u>Consistency Constraint:</u> 1) Num RIDs >= Index Cardinality
Cluster Factor	-1	<u>Structural Constraint:</u> DOUBLE 1) -1 or 0 <= Cluster Factor <= 1 <u>Consistency Constraint:</u> NILL
Num Empty Leafs	-1	<u>Structural Constraint:</u> BIGINT 1) -1 or Num Empty Leafs >= 0 <u>Consistency Constraint:</u> 1) Num Empty Leafs <= NLeafs

Table 5 - DB2 Index Level Metadata Fields

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA CONSTRUCT MODE - ORACLE

CODD supports the following data types in the Construct Mode for Oracle.

1. **BOOLEAN**
2. **VARCHAR2**
3. **NUMBER**
4. **INTEGER**
5. **REAL**
6. **DECIMAL**
7. **DOUBLE**
8. **FLOAT**
9. **DATE**
10. **TIMESTAMP**
11. **CHAR**

CODD Construct mode can be used only for the relations which has only the above list of data types in their columns.

The Relation, Column and Index level metadata of the construct mode form are described in the following tables with the constraint associated with each field and the default values. Each field also specifies the data type of it. The input value in the form field must be in between the smallest and largest value of the data type and adhere to the constraints associated with it. Violating any of the constraints reports an error to the user.

We suggest the user to follow the constraints when they fill the values in the form field.

Oracle Number data type is implemented with BigInteger in Java. The Number data type value must be lesser than or equal to 9223372036854775807.

Metadata Field	Default Value	Consistency Constraints
Cardinality	0	<u>Structural Constraint:</u> NUMBER 1) Cardinality ≥ 0 <u>Consistency Constraints:</u> NIL
Blocks	0	<u>Structural Constraint:</u> NUMBER 1) Blocks ≥ 0 <u>Consistency Constraints:</u> 1) Blocks \leq Cardinality
Avg Row Len	0	<u>Structural Constraint:</u> NUMBER 1) Avg Row Len ≥ 0 <u>Consistency Constraints:</u> NIL

Table 6 - Oracle Relation Level Metadata Fields

Metadata Field	Default Value	Consistency Constraints
Distinct Column Values	0	<u>Structural Constraint:</u> NUMBER 1) Column Cardinality ≥ 0 <u>Consistency Constraints:</u> 1) Column Cardinality \leq Cardinality 2) Column Cardinality = Cardinality, if Unique column 3) Column Cardinality \leq Primary Key Column Cardinality, if Foreign Key Column
Null Count	0	<u>Structural Constraint:</u> NUMBER 1) Null Count $\Rightarrow 0$ <u>Consistency Constraints:</u> 1) Null Count \leq Cardinality 2) Null Count = 0, if NotNULL column 3) Null Count + Column Cardinality \leq Cardinality
Avg Col Len	0	<u>Structural Constraint:</u> NUMBER 1) Avg Col Len ≥ 0 <u>Consistency Constraints:</u> NILL
Density	0	<u>Structural Constraint:</u> NUMBER 1) Density ≥ 0 <u>Consistency Constraints:</u> NILL
Frequency Histogram	NONE	<u>Structural Constraint:</u> 1) COLVALUE : Value whose type is same as column type 2) FREQUENCY : INTEGER : VALCOUNT ≥ 0 <u>Consistency Constraints:</u> 1) Sum (VALCOUNT's) \leq Cardinality 2) Number of buckets \leq Column Cardinality
Quantile Histogram	NONE	<u>Structural Constraint:</u> 1) COLVALUE : Value whose type is same as column type <u>Consistency Constraints:</u> 1) COLVALUE must be increasing.

Table 7 - Oracle Column Level Metadata Fields

Metadata Field	Default Value	Consistency Constraints
Index Num Rows	-1	<u>Structural Constraint:</u> NUMBER 1) Index Num Rows ≥ 0 <u>Consistency Constraint:</u> 1) Index Num Rows = Cardinality
Leaf Blocks	-1	<u>Structural Constraint:</u> Number 1) Leaf Blocks ≥ 0 <u>Consistency Constraint:</u> NILL
Distinct Keys	-1	<u>Structural Constraint:</u> NUMBER 1) Distinct Keys ≥ 0

		<u>Consistency Constraint:</u> NILL
Avg Leaf Blocks Per Key	-1	<u>Structural Constraint:</u> NUMBER 1) Avg Leaf Blocks Per Key >= 0 <u>Consistency Constraint:</u> NILL
Avg Data Blocks Per Key	-1	<u>Structural Constraint:</u> NUMBER 1) Avg Data Blocks Per Key >= 0 <u>Consistency Constraint:</u> NILL
Clustering Factor	-1	<u>Structural Constraint:</u> NUMBER 1) 0 <= Clustering Factor <= 1 <u>Consistency Constraint:</u> NILL
Index Levels	-1	<u>Structural Constraint:</u> NUMBER 1) Index Levels >= 0 <u>Consistency Constraint:</u> NILL

Table 8 - Oracle Index Level Metadata Fields

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA CONSTRUCT MODE - SQL/MX

CODD supports the following data types in the Construct Mode for SQL/MX.

1. CHAR
2. VARCHAR
3. DATE, TIME, TIMESTAMP
4. FLOAT
5. SIGNED/UNSIGNED NUMERIC
6. SIGNED/UNSIGNED DECIMAL
7. SMALLINT,LARGEINT, INTEGER
8. INTERVAL
9. REAL

CODD Construct mode can be used only for the relations which has only the above list of data types in their columns.

The Relation, Column and Index level metadata of the construct mode form are described in the following tables with the constraint associated with each field and the default values. Each field also specifies the data type of it. The input value in the form field must be in between the smallest and largest value of the data type and adhere to the constraints associated with it. Violating any of the constraints reports an error to the user.

We suggest the user to follow the constraints when they fill the values in the form field.

Metadata Field	Default Value	Consistency Constraints
Cardinality	0	<u>Structural Constraint:</u> NUMBER 1) Cardinality ≥ 0 <u>Consistency Constraints:</u> NIL

Table 9 - SQL/MX Relation Level Metadata Fields

Metadata Field	Default Value	Consistency Constraints
Total UEC	-1	<u>Structural Constraint:</u> NUMBER 1) Total UEC ≥ 1 <u>Consistency Constraints:</u> 1) Total UEC \leq Cardinality, if Total UEC > 0 2) Total UEC = Cardinality, if Unique column in case of single column histograms
Low Value	Null	<u>Structural Constraint:</u> 1)Value whose type is same as column type. 2)Value cannot be null.

		<p>3) In case of Multi Column Histograms (MCH), Low Value should have same number of values pipe separated as number of multiple columns.</p> <p><u>Consistency Constraints:</u></p> <p>1) Low Value <= High Value.</p> <p>2) Low Value of column in MCH should be same as low value of its individual column.</p>
High Value	Null	<p><u>Structural Constraint:</u></p> <p>1) Value whose type is same as column type.</p> <p>2) Value cannot be null.</p> <p>3) In case of Multi Column Histograms (MCH), High Value should have same number of values pipe separated as number of multiple columns.</p> <p><u>Consistency Constraints:</u></p> <p>1) Low Value <= High Value.</p> <p>2) High Value of column in MCH should be same as high value of its individual column.</p>
Interval RowCount	Null	<p><u>Structural Constraint:</u> NUMBER</p> <p>2) COLVALUE : Value >= 0</p> <p>3) COLVALUE : Value >= Interval UEC</p> <p><u>Consistency Constraints:</u></p> <p>1) COLVALUE must be increasing.</p> <p>2) SUM(Interval RowCount) <= Cardinality</p>
Interval UEC	Null	<p><u>Structural Constraint:</u> NUMBER</p> <p>1) COLVALUE : Value >= 0</p> <p>2) COLVALUE : Value <= Interval RowCount</p> <p><u>Consistency Constraints:</u></p> <p>1) COLVALUE must be increasing.</p> <p>2) SUM(Interval UEC) should be equal to Total UEC</p>
Interval Boundary	Null	<p><u>Structural Constraint:</u></p> <p>1) COLVALUE : Value whose type is same as column type</p> <p>2) Low Value <= Interval Boundary <= High Value</p> <p><u>Consistency Constraints:</u></p> <p>1) COLVALUE must be increasing.</p> <p>1) Last bucket value same as High Value.</p>

Table 10 - SQL/MX Column Level Metadata Fields

Note: Index level statistics are not taken into account, as they are maintained within system metadata tables that are not user updatable.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA CONSTRUCT MODE - POSTGRESQL

CODD supports the following data types in the Construct Mode for PostgreSQL.

1. **BOOLEAN**
2. **VARCHAR2**
3. **NUMERIC**
4. **INT2, INT4, INT8**
5. **REAL**
6. **DECIMAL**
7. **DOUBLE**
8. **FLOAT4, FLOAT8**
9. **DATE**
10. **BPCHAR**
11. **CHAR**

CODD Construct mode can be used only for the relations which has only the above list of data types in their columns.

PostgreSQL doesn't allow modifications to its column level metadata details from outside. Hence, in case of PostgreSQL, we have added functionality inside PostgreSQL database engine code. Hence, in order to use CODD with PostgreSQL, users have to install PostgreSQL from Source code. After installation, replace the files of PostgreSQL with the files given as part of the CODD zip file. Care should be taken to replace as there are certain files which have same names across different folders in PostgreSQL.

The list of files to be replaced with their location is given in Table 11.

File Name	File Location
analyze.c	src/backend/commands/analyze.c
gram.y	src/backend/parser/gram.y
kwlist.h	src/include/parser/kwlist.h
parsenodes.h	src/include/nodes/parsenodes.h
copyfuncs.c	src/backend/nodes/copyfuncs.c
equalfuncs.c	src/backend/nodes/equalfuncs.c
plancat.c	src/backend/optimizer/util/plancat.c
guc.c	src/backend/utils/misc/guc.c
pgstat.h	src/include/pgstat.h

Table 11 - Files to be replaced in PostgreSQL.

You also need to define the following new parameter under RUNTIME STATISTICS in postgresql.conf, the configuration file for PostgreSQL:

```
skip_pagetest = ON
```

The user is also given an option to **turn off** the CODD mode. In postgres.h file, the user can comment **#define CODD_ENABLED** and uncomment **#undef CODD_ENABLED** to turn off CODD and run PostgreSQL normally.

Note: The version of PostgreSQL that was tested is 9.3.

The Relation, Column and Index level metadata of the construct mode form are described in the following tables with the constraint associated with each field and the default values. Each field also specifies the data type of it. The input value in the form field must be in between the smallest and largest value of the data type and adhere to the constraints associated with it. Violating any of the constraints reports an error to the user.

We suggest the user to follow the constraints when they fill the values in the form field.

Metadata Field	Default Value	Consistency Constraints
RelTuples	0	<u>Structural Constraint:</u> NUMBER 1) RelTuples >= 0 <u>Consistency Constraints:</u> NIL
RelPages	0	<u>Structural Constraint:</u> NUMBER 1) RelPages >= 0 <u>Consistency Constraints:</u> 1) RelPages <= RelTuples

Table 12 - PostgreSQL Relation Level Metadata Fields

Metadata Field	Default Value	Consistency Constraints
Distinct Values	0	<u>Structural Constraint:</u> NUMBER 1) Distinct Values >= -1 <u>Consistency Constraints:</u> 1) Distinct Values <= RelTuples, if Distinct Values > 0 2) Distinct Values = RelTuples or Distinct Values = -1, if Unique column 3) Distinct Values <= Primary Key Column Distinct Values, if Foreign Key Column
Null Fraction	0	<u>Structural Constraint:</u> NUMBER 1) 0 <= Null Fraction <= 1 <u>Consistency Constraints:</u> 1) Null Fraction = 0, if NotNULL column 2) Null Fraction + Distinct Values <= RelTuples

		(after converting to positive ratio format)
Avg Width	0	<u>Structural Constraint:</u> NUMBER 1) Avg Col Len ≥ 0 <u>Consistency Constraints:</u> NIL
Correlation	0	<u>Structural Constraint:</u> NUMBER 1) $-1 \leq \text{Density} \leq 1$ <u>Consistency Constraints:</u> NIL
Frequency Histogram	Null	<u>Structural Constraint:</u> 1) COLVALUE : Value whose type is same as column type 2) FREQUENCY : Double : $0 < \text{FREQ} \leq 1$ <u>Consistency Constraints:</u> 1) Sum (FREQ's) ≤ 1 2) Number of buckets \leq Distinct values
Histogram Bounds	Null	<u>Structural Constraint:</u> 1) COLVALUE : Value whose type is same as column type <u>Consistency Constraints:</u> 1) COLVALUE must be increasing.

Table 13 - PostgreSQL Column Level Metadata Fields

Metadata Field	Default Value	Consistency Constraints
Index RelTuples	0	<u>Structural Constraint:</u> NUMBER 1) Index RelTuples ≥ 0 <u>Consistency Constraint:</u> 1) Index RelTuples = RelTuples
Index RelPages	0	<u>Structural Constraint:</u> Number 1) Index RelPages ≥ 0 <u>Consistency Constraint:</u> Index RelPages = RelPages

Table 14 - PostgreSQL Index Level Metadata Fields

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

CONSTRUCT MODE - GRAPH HISTOGRAM

CODD provides a feature to modify the data distribution of a column through graphical interface, when the user wants to edit the histogram of a column in the [Metadata Construct Mode](#). Figure 11 shows the graphical interface of histogram. CODD uses [JFreeChart](#) for the graphical histogram. Features of [JFreeChart](#) like zoom in/out on the graph are inherently present in the graph histogram.



Figure 11 - Graph Histogram Window

Components of the Graph Histogram GUI

1. **Vertical Slider:** The vertical slider on the left side of the graph histogram window represents the percentage of tuple (0 -100). Upon selecting a bucket, the slider value is set to the percentage of tuple present in the bucket.
2. **Vertical Slider Textbox:** The vertical slider textbox represents the percentage of tuple (0 - 100). Upon selecting a bucket, the textbox value is set to the percentage of tuple present in the bucket.

3. **Data Distribution Graph:** The central part of the window is the data distribution graph, which is constructed using [JFreeChart](#). The top label above the graph represents the histogram value (Frequency Count or Distinct Count) being shown. At the bottom it shows the column name, for which the histogram is shown (Figure 11 graph represents the data distribution of C_ACCTBAL of CUSTOMER relation). A bucket can be selected by clicking on the bucket. The selection of bucket is shown by a GREEN color border around the bucket. Selection of a bucket also changes the Vertical Slider, Horizontal Slider, Vertical Slider Textbox, Horizontal Slider Textbox and Bucket Information label.
4. **Zoom-in operation:** Select the area of the histogram, which you want to zoom in by dragging the mouse. Selecting a bucket a in the zoom-in mode, brings the graph back to original representation.
5. **Zoom-out operation:** Right click on the zoomed-in graph and choose AutoRange -> Both Axes.
6. **Add/Delete Bucket at[Front/Back] Buttons:** These buttons lets the user add / remove bucket at the start or end of the bucket.
7. **Bucket Information label:** The label present in between the **Add/Delete Bucket at[Front/Back]** buttons represent the bucket information. Upon selecting a bucket, this label changes and shows the bucket range and frequency / distinct count present in the bucket.
8. **Horizontal BiSlider:** The horizontal slider is a [BiSlider](#), which helps in changing the bucket range. Upon selecting a bucket, the BiSlider sets the left and right knob at the bucket boundaries.
9. **Horizontal BiSlider TextBox:** The horizontal slider textbox helps in changing the bucket range. Upon selecting a bucket, the two textboxes value is set to the bucket boundaries.
10. **Distribute mode buttons: Distribute Buckets** button is enabled if the excess / less percentage of tuples is not zero. Initially, the graph represents all the 100% of tuples. As the graph is changed the total percentage of tuple represented by the graph may go less or excess. This button lets the user distribute the excess / less percentage of tuples into a selected list of buckets. **Finish** button will be enabled only if there is no excess / less percentage of tuples. In Figure 11, the graph is excess of 8.25% of tuples.
11. **Merge mode buttons:** Selecting **Start Merging Buckets** button starts the merging of buckets. The user can select multiple adjacent buckets and click on **Merge Buckets** button to merge the selected buckets.
12. **Split a Bucket in Two Buckets Button:** Clicking on this button lets the user split the selected bucket into two.
13. **Redo, Undo buttons:** Graph Histogram supports to store the last 10 history of operations. Operations can be Undone / Redone in case if the user wants to revert back an operation with the help of **Undo**, **Redo** buttons.
14. **Finish button: Finish** button exits the graph and updates the histogram in the Construct mode with the graph values. If the button was clicked when the graph was representing the frequency count of buckets and distinct counts are present for the buckets, then it shows the distinct count graph, where the user can modify the distribution of distinct count on the buckets. Closing the graph doesn't affect the histogram in the Construct mode window.

Features of Graph Histogram

The graph histogram supports the following features:

1. Adjust the height of a histogram bucket:

- Upon selecting a bucket, the vertical slider value is set to the percentage of tuples present in the bucket. The user can move the knob to increase / decrease the percentage of tuples present in the bucket.
- Each bucket has minimum (1% of total row / distinct counts) height beyond which the bucket can't be decreased. If the column is of integer type and if there is a unique constraint, then the height can't be increased beyond the max height possible (which is the number of unique integer values in the bucket range).
- And also for distinct count graph the bucket can't be increased beyond the bucket's row count. The increase / decrease in the height is reflected in the Excess / Less percentage of tuple of the histogram label. Alternatively, the user can also enter the tuple percentage using Vertical Slider textbox.

2. Adjust the width of a histogram bucket:

- CODD Graph Histogram supports changing width of the bucket only for columns with numeric data types (Integer, Double).
- Upon selecting a bucket, the horizontal bislider represents the bucket boundary. The user can move the knobs to change the width of the bucket. Each bucket has a minimum width (1 i.e the minimum bucket range (right boundary - left boundary) is 1), beyond which it can't be decreased.
- Alternatively, the user can change the bucket boundaries using the Horizontal Slider textboxes by directly entering a valid value for the bucket boundaries.

3. Split a bucket into two:

- Upon selecting a bucket and clicking on the **Split a Bucket in Two Buckets** button, the user is asked for the split column value (which should be in between the left and right boundary of the bucket) and height split percentage for the bucket split.
- The bucket is split into two buckets based on the inputs.

4. Merge one or more nearer buckets into one bucket:

- If the user chooses the **Start Merging Buckets** button, the graph goes to merge mode.
- The user can select multiple adjacent buckets and click on **Merge Buckets** button. This merges the selected bucket into one bucket.

5. Add or Remove bucket from the start / end of the histogram:

- Clicking on the **Add/Delete Bucket at[Front/Back]** buttons asks the user for the input (add / remove) and if it is add the user has to give the minimum / maximum value of the new bucket.
- Based on the input the end of the histogram is modified and **Excess / Less percentage of tuple of the histogram** updated accordingly.

6. Distribute the excess / less percentage of tuples in to a selected list of buckets:

- If the excess / shortage percentage of row / distinct count is not zero and the user has clicked on the **Distribute Excess/Less Values** button, then the graph goes to Distribute mode.
- This enables the **Selected, Not Selected** buttons. The user can select multiple buckets by clicking on the buckets. A selected bucket can be unselected by clicking on the bucket again. And also clicking outside the graph, un-selects all the selected buckets.
- For excess percentage, the user has to select buckets whose total percentage of row / distinct count is more than excess percentage. After selecting the buckets, the user can click on the

Selected / Not Selected buttons to distribute (weighted distribution) the excess / less percentage on the **selected / un selected** buckets. After this operation, the **Excess / Less percentage of tuple of the histogram** changes.

7. **Undo and Redo the operations:**

- Clicking on **Undo** button undoes the previous operation. **Redo** button is enabled only if there was an earlier undo operation; clicking on it redoes the undone operation.

After each operation, the graph is refreshed and the new graph is shown to the user.

Operating data modes of Graph Histogram

The graph histogram is used in two modes of operations as given below.

1. **Frequency Mode:** The graph operates on the frequency values of the buckets.
2. **Distinct Count Mode:** The graph operates on the distinct count values of the buckets. Distinct Count Mode is carried out only after completing the Frequency Mode. The number of buckets and their positions are fixed (based on the changes in the Frequency Mode) in this mode i.e., the user will be able to adjust only the height of the histogram and distribute the excess / less percentage of tuples. Changing bucket width, split, merge, add / remove is not supported in this mode.

The graph starts with the Frequency mode. Once the user is done with the changes on frequency histogram, he / she can click on the **Finish** button, which will take the user to distinct count mode graph. In Figure 11, the graph is in Frequency Mode.

Graph Histogram Features support for database engines

Table 15 shows the supported histogram features for DB2, Oracle, SQL/MX and PostgreSQL.

Feature	DB2 and SQL/MX	Oracle and PostgreSQL
Frequency Mode	Supports	Supports
Distinct Count Mode	Supports	Does not Support
Adjust Height	Supports	Does not Support
Adjust Width	Supports (Numeric Datatype)	Supports (Numeric Datatype)
Split Bucket	Supports	Supports
Merge Bucket	Supports	Supports
Add / Remove Bucket	Supports	Supports
Undo / Redo	Supports	Supports

Table 15 - Graph Histogram support for DB Engines

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA RETAIN MODE

In environments where the metadata is required to be explicitly created from a database instance, CODD supports the subsequent dropping of some or all of the raw data, permitting reclamation of the storage space occupied by the data.

The Metadata Retain Mode window is shown in Figure 12. The relations selected in [Choosing Relations for Operations](#) window is listed in the upper text box. Dropping parent relation data also requires to child relation data to be dropped. All the dependent relations are listed in the lower text box. If the user does not want to drop data of any of the relations, he / she can remove that relations by selecting the relation and then clicking on **Remove** button. Clicking on **Confirm** button will perform the Metadata Retain mode operation, where all the data of the chosen relations are dropped and only the Metadata is kept in the database. Once the operation is completed, CODD shows the success message or the error information to the user.

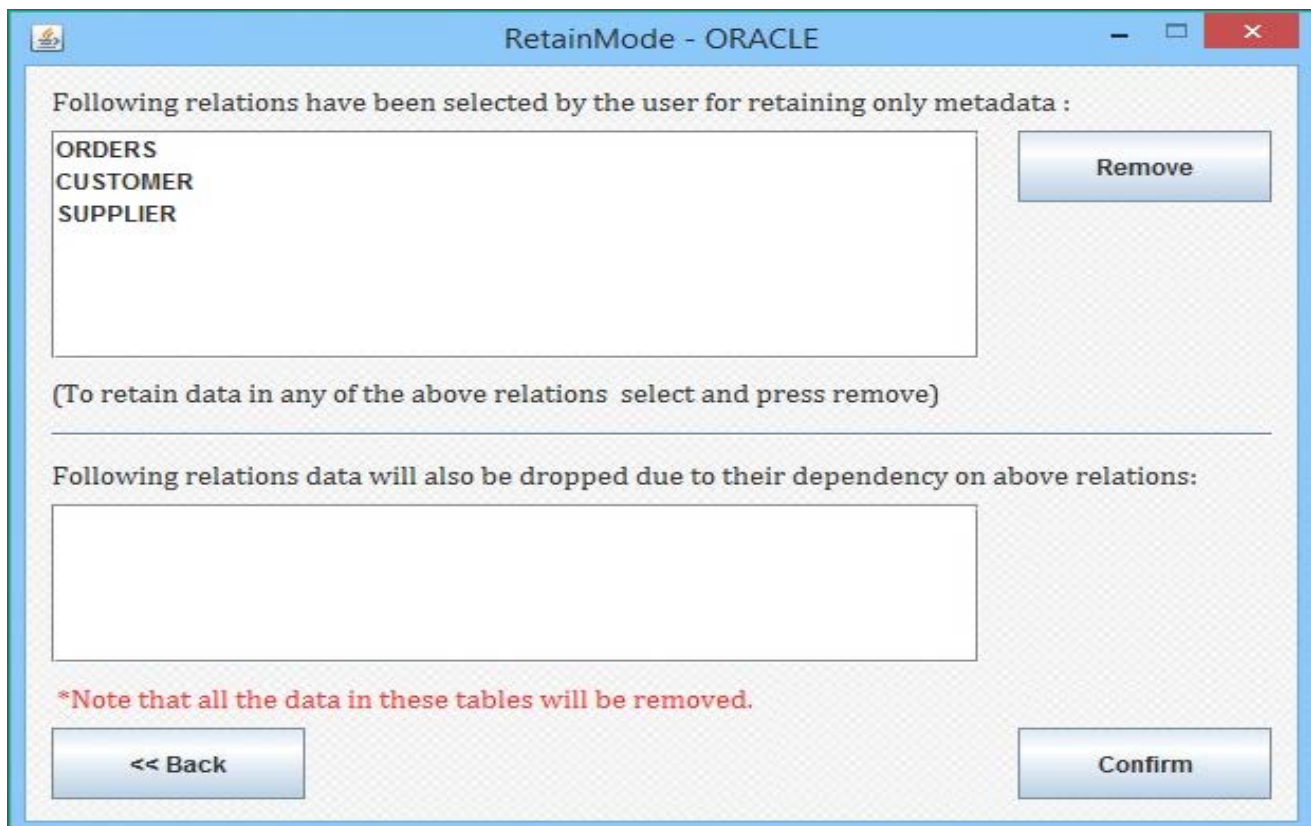


Figure 12 - Metadata Retain Mode Window

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA TRANSFER MODE

CODD supports automatic porting to the extent possible, of the statistical metadata across database engines, thereby facilitating comparative studies of systems as well as early assessments of the impact of data migration.

CODD supports two kinds of Metadata Transfer.

Native Engine Transfer

Metadata of relations of a DB Engine (say DB2) can be transferred **completely** to another database relations of same DB Engine type (DB2).

Inter Engine Transfer

Metadata of relations of a DB Engine (say DB2) can be transferred to another database relations of different DB Engine type (Oracle). CODD transfers **most of the metadata statistics** in Inter Engine Transfer.

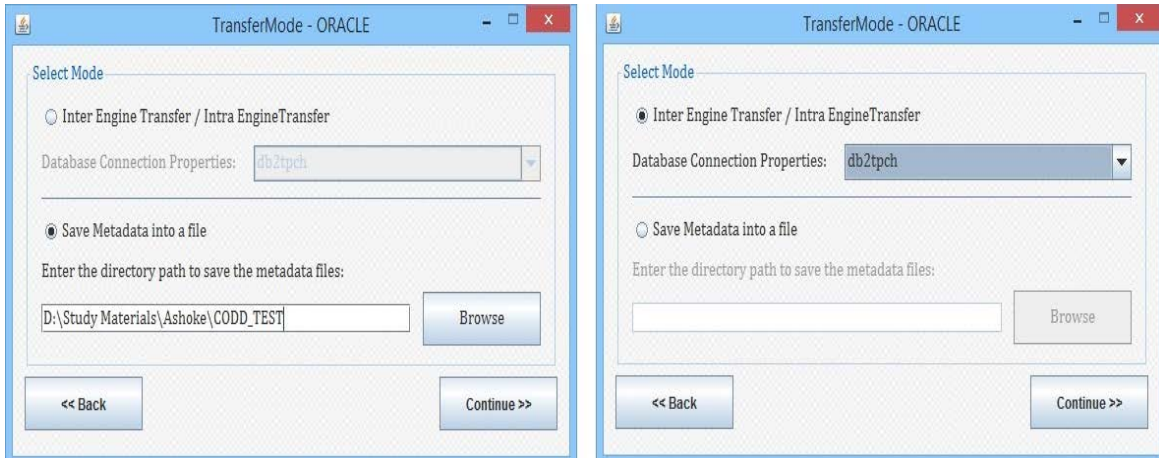


Figure 13 - Metadata Transfer Mode Window

Figure 13 shows the Metadata Transfer Mode Window. The user has to choose any one of the two following two options.

Inter Engine Transfer / Native Engine Transfer:

If the destination database (native / other DB Engine) is up and running in the same machine or in the network, then the user can select this option and input the destination database connection configurations to continue with metadata transfer. Based on the destination database type, most of the metadata is transferred.

Save Metadata into a file:

If the user wants to store the metadata into a file and load it later to the destination database, then he/ she can select this option and input the path to store the metadata file. This creates a file called 'dbType' and metadata files for each chosen relations. These files can be loaded back through **Construct from CDD transferred metadata** option of [Metadata Construct Mode](#) window to the database with / without modification (**Modify and Update** option) .

After Choosing one of the options and clicking on **Continue** button, the metadata will be transferred to the destination database / to a file. Once the operation is completed, CDD shows the success message or the error information to the user.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

METADATA SCALING MODE

A common activity in database engine testing exercises is to assess the behavior of the system on scaled versions of the original database, and this is the reason that benchmarks such as TPC-H and TPC-DS are available in a variety of scale factors.

CODD supports [size-based scaling](#) model based on the TPC-H and TPC-DS scaling model. In addition, it also provides a novel [time-based scaling](#) model. Here, the aim is to scale the baseline metadata such that the optimizer's estimated cost of executing a given query workload Q on the scaled version is a multiple, α (scaling factor), of the cost of executing it on the original database.

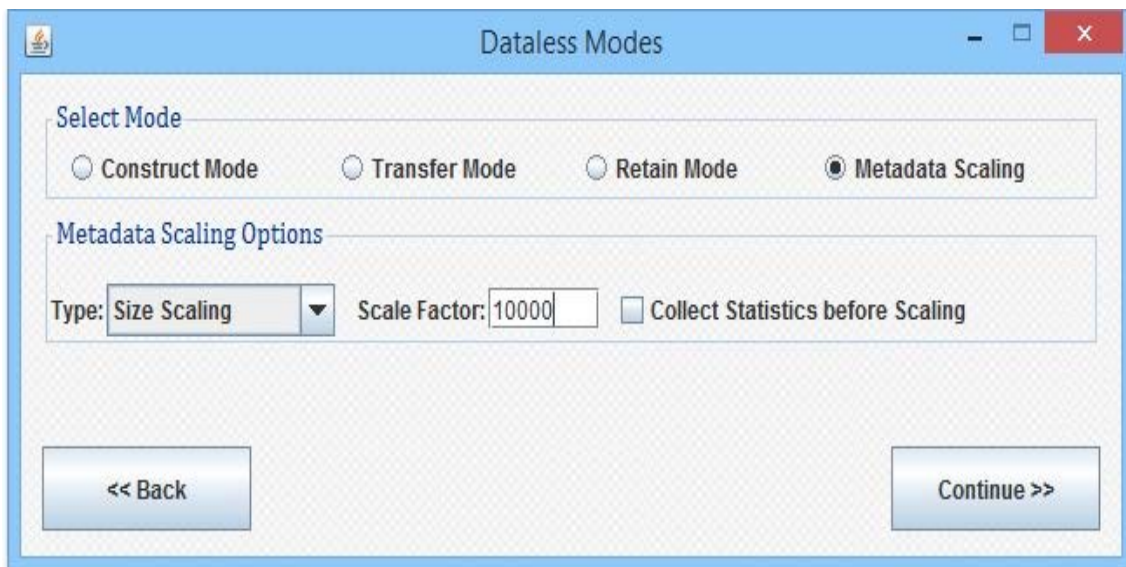
[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

SIZE-BASED METADATA SCALING

Choosing the Metadata Scaling option in the Mode Selection window, enables the input fields of metadata scaling. Figure 14 shows that the TPC-H style scaling is chosen with scaling factor 10000. Checking **Run Statistics** check box collects the statistics on the relations before the scaling process starts. The user should note that, this should be checked only if the data is present in the database. If the data has been removed through CODD Retain Mode, then selecting this option would not scale the relations. After entering the input values and clicking on continue takes the user to the Size-Based Metadata Scaling window, shown in Figure 15.



The screenshot shows a window titled "Dataless Modes" with a light blue border. Inside, there are two main sections. The first section, "Select Mode", contains four radio buttons: "Construct Mode", "Transfer Mode", "Retain Mode", and "Metadata Scaling". The "Metadata Scaling" radio button is selected. The second section, "Metadata Scaling Options", contains a "Type" dropdown menu set to "Size Scaling", a "Scale Factor" text box containing "10000", and a "Collect Statistics before Scaling" checkbox which is unchecked. At the bottom of the window, there are two buttons: "<< Back" on the left and "Continue >>" on the right.

Figure 14 - Size Based Scaling Window-1

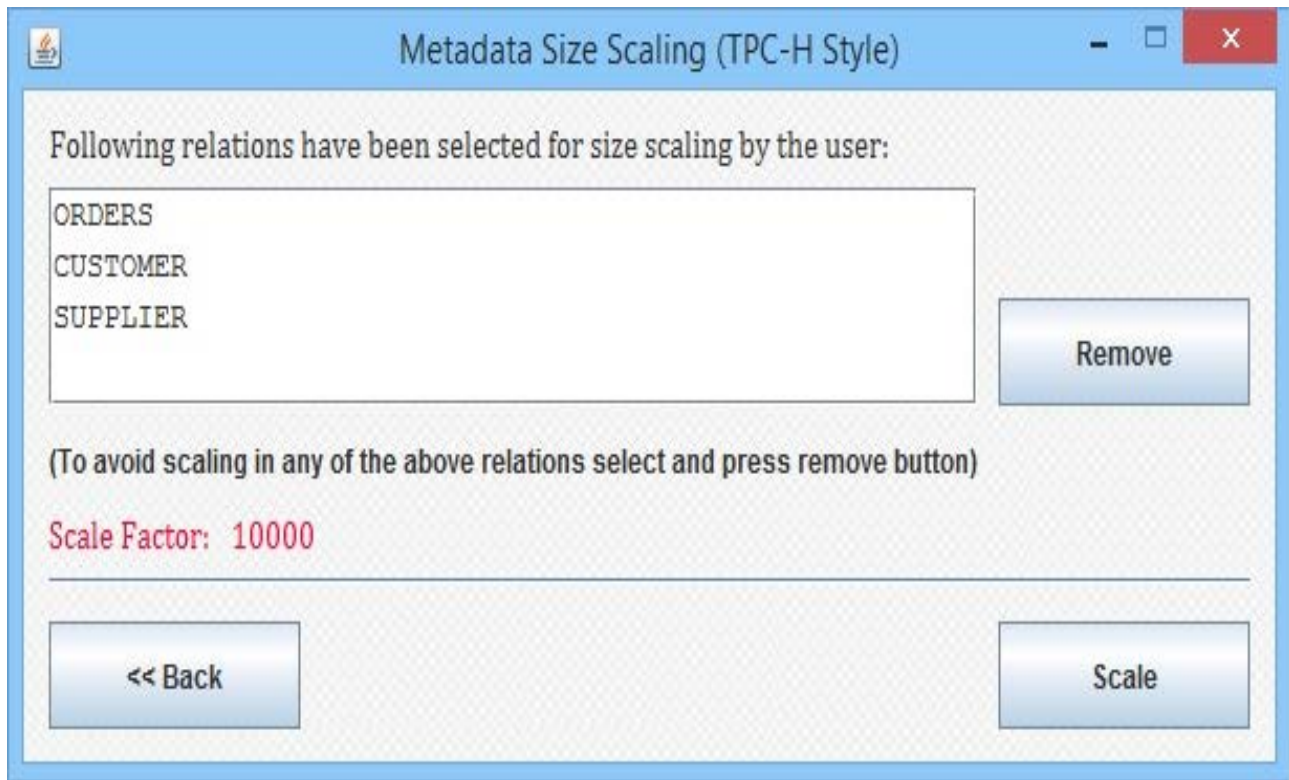


Figure 15 - Size Based Scaling Window-2

Size Based scaling window lists the chosen relations that will be scaled and describes the scaling mechanism. If the user want to remove any of the relations for scaling, he can select the relation and click on the **Remove** button. Clicking on **Scale** button, CODD performs the size scaling (linear scaling of relation cardinalities by the scale factor assuming that *tuple* width is same for original and scaled relations) on the chosen relations. Once the operation is completed, CODD shows the successful message or the error information to the user.

[Documentation Home](#)

CODD Metadata Processor

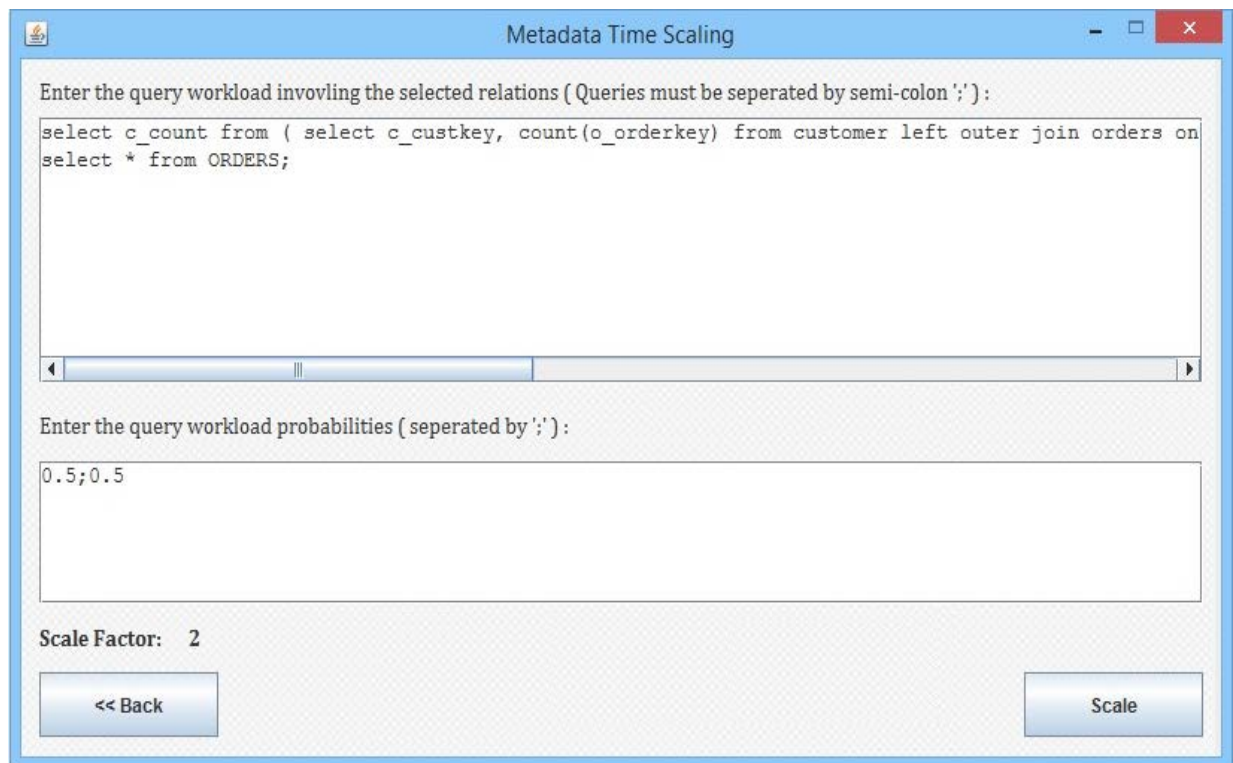
©Indian Institute of Science, Bangalore, India

TIME-BASED METADATA SCALING

Choosing the time scaling option in Mode Selection window, takes the user to the Time scaling window shown in Figure 16. To know more about the Time based scaling, the user can refer to the [technical report](#). The user has to give the query workloads and the probabilities (separated by semicolon) in the corresponding fields. The probability field can be left empty, in which case the uniform probability distribution is used. Choosing **Scale** button, validates the user input. The following checks are done on the inputs:

1. Number of queries is same as the number of input probabilities (if input probability is given)
2. Sum of probabilities is 1

If there are any validation error, it is reported to the user. Once the input is validated CODD performs the time scaling on the selected relations with respect to the query workloads (QW).



Metadata Time Scaling

Enter the query workload involving the selected relations (Queries must be seperated by semi-colon ';') :

```
select c_count from ( select c_custkey, count(o_orderkey) from customer left outer join orders on  
select * from ORDERS;
```

Enter the query workload probabilities (seperated by ';') :

0.5;0.5

Scale Factor: 2

<< Back

Scale

Figure 16 - Cost Based Scaling Window

The time scaling steps are as follows:

Cost of executing query on the original database is obtained from the optimizer's execution plan [OriginalCost].

Cost of executing query on the scaled database is modeled as a function of input scaling factors [ScaledCost].

The following optimization problem is solved to obtain the time scaling factors for the chosen relations.

$$\text{Minimize } \sum_{\text{query in QW}} [(\text{ScaledCost}_{\text{query}} / \text{OriginalCost}_{\text{query}}) - \text{ScaleFactor}]^2$$

$$\text{such that } \sum_{\text{query in QW}} \text{ScaledCost}_{\text{query}} = \text{ScaleFactor} * (\sum_{\text{query in QW}} \text{OriginalCost}_{\text{query}})$$

$$\text{Individual relation scaling factors} > 0$$

CODD uses [SuanShu](#) Optimization library to solve the above optimization problem. It is a commercial library. License information is provided in [License Information](#).

Relations are linearly scaled (metadata values are scaled) with the obtained scaling factors.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

CODD THROUGH COMMAND LINE

There is a command line (batch mode) option added for the scaling operation of CODD. This is to automate invoking of CODD and scale any given table histogram without any user intervention. The usage of the command is as follows:

Usage: `coddCmd - connection_config_string -scale_factor -run_stats <0/1> (optional) -table (optional) \n";`
[The arguments are expected in the above order (except the optional argument which can be swapped among them)]

Here "connection_name" is the name of the connection created using the CODD GUI. "SF no" is any positive integer value that you want the histogram to be scaled to. "run_stats" is an optional argument that specifies if we should run the database specific command to generate statistics before we start the scaling. This is very useful if there is existing data in the tables and we want to start scaling from the statistics of the given data. Also this option would be helpful if we want to start from a clean and well known set of statistics. "table_name" would be optional so that we can scale a particular table in a database and NOT all the tables in the given database catalog & schema.

This command line option makes the following assumptions to make the processing easier and simple.

1. Assumes an initial connection configuration to be available and created using the CODD GUI. This is a reasonable assumption to make and it simplifies the processing a lot.
2. The arguments are expected in the above order and only very simple processing of the arguments are done.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

LICENSE INFORMATION

1. CODD Software

The CODD Metadata Processor software is copyrighted by the [Indian Institute of Science](#), Bangalore, India, as per this [Copyright Certificate](#) issued by Govt. of India, on September 10, 2013. The entire software package, including the source code, is provided **completely free of charge** on an *as-is-where-is* basis. Downloading the CODD software automatically implies that you accept and acknowledge copyright ownership by the [Indian Institute of Science](#), and that your usage of the CODD software is governed by the terms of this [licensing agreement](#).

2. Third-Party Software

Third-party software required for CODD to function is comprised of libraries for

- (a) graphics layout and visualization
- (b) establishment of connections with the various database engines
- (c) optimization library to solve time scaling minimization problem.

All rights on these support libraries rest entirely with the respective vendors.

To assist in setup and testing of your initial CODD installation, all essential libraries (other than the SuanShu optimization library) are included with the full version of the CODD code-base in the **Libraries** folder – however, we *strongly recommend* that users should visit the vendor URLs given below for complete details about the licensing of the support libraries, and make sure that their usage complies with the vendor's requirements. Users are fully responsible for their usage of the support software and Indian Institute of Science is not liable for any improper usage. As a general policy, it is advisable for CODD users to directly download the support libraries from the vendor web-sites, or from alternative third-party sources, and update the **Libraries** folder with these downloaded files.

2.1 Graphics

BiSlider: This is a Slider library to have two knobs, so that a range of values can be represented in the slider. This library is redistributable as per [this document](#).

Libraries used: *BiSlider.jar*

JFreeChart: This is a Chart Library used to visualize and modify the data distribution (histogram), which comes under LGPL.

Libraries used: *gnujaxp.jar iText-2.1.5.jar jcommon-1.0.17.jar jfreechart-1.0.14.jar jfreechart-1.0.14-experimental.jar jfreechart-1.0.14-swt.jar junit.jar servlet.jar swtgraphics2d.jar*

2.2 Databases

[DB2 JDBC Driver](#): This is a JDBC driver for connecting to DB2 databases, redistributable as per [this document](#).

Libraries used: *db2jcc.jar db2jcc_license_cu.jar*

[Oracle JDBC Driver](#): This is a JDBC driver for connecting to Oracle databases, redistributable as per this [licensing agreement](#).

Libraries used: *ojdbc14.jar*

[MSSQL JDBC Driver 2.0](#): This is a JDBC driver for connecting to Microsoft SQL Server, redistributable as per this [licensing agreement](#).

Libraries used: *sqljdbc4.jar*

[PostgreSQL JDBC Driver](#): This is a JDBC driver for connecting to PostgreSQL, distributed under a BSD license, redistributable as per [this document](#).

Libraries used: *postgresql-8.0-311.jdbc3.jar*.

Driver Sites:

Comprehensive lists of database drivers, including both vendor drivers and third-party drivers, are available at the following sites:

- Sun: <http://developers.sun.com/product/jdbc/drivers>
- Minq Software: <http://www.minq.se/products/dbvis/drivers.html>

2.3 Optimization Solver

[SuanShu](#): An Optimization library to solve non linear optimization problem, used in time scaling. It is available for purchase under academic as well as commercial licenses. A limited trial version is also offered. [**Note**: This library is not included in the CODD distribution.]

Libraries used: *suanshu-3.1.2.jar*.

[Documentation Home](#)

CODD Metadata Processor

©Indian Institute of Science, Bangalore, India

DEVELOPMENT TEAM

- [Jayant Haritsa](#) (Project Lead)

(primary student contributors in chronological order of participation)

- [Rakshit Trivedi](#) (PA)
- [I. Nilavalagan](#) (ME, CSA, IISc)
- [Deepali Nemade](#) (ME, CSA, IISc)
- [Ankur Gupta](#) (ME, CSA, IISc)
- [Ashoke S](#) (ME, CSA, IISc)

[Documentation Home](#)

End CODD Documentation

©Indian Institute of Science, Bangalore, India