Converting Netezza DDL to SQL Data Warehouse

# Overview

This blog is intended to provide guidance for converting a Netezza platform to Azure Data Warehouse(SQLDW). This also provides a PowerShell script that will convert an extracted Netezza DDL to SQLDW.

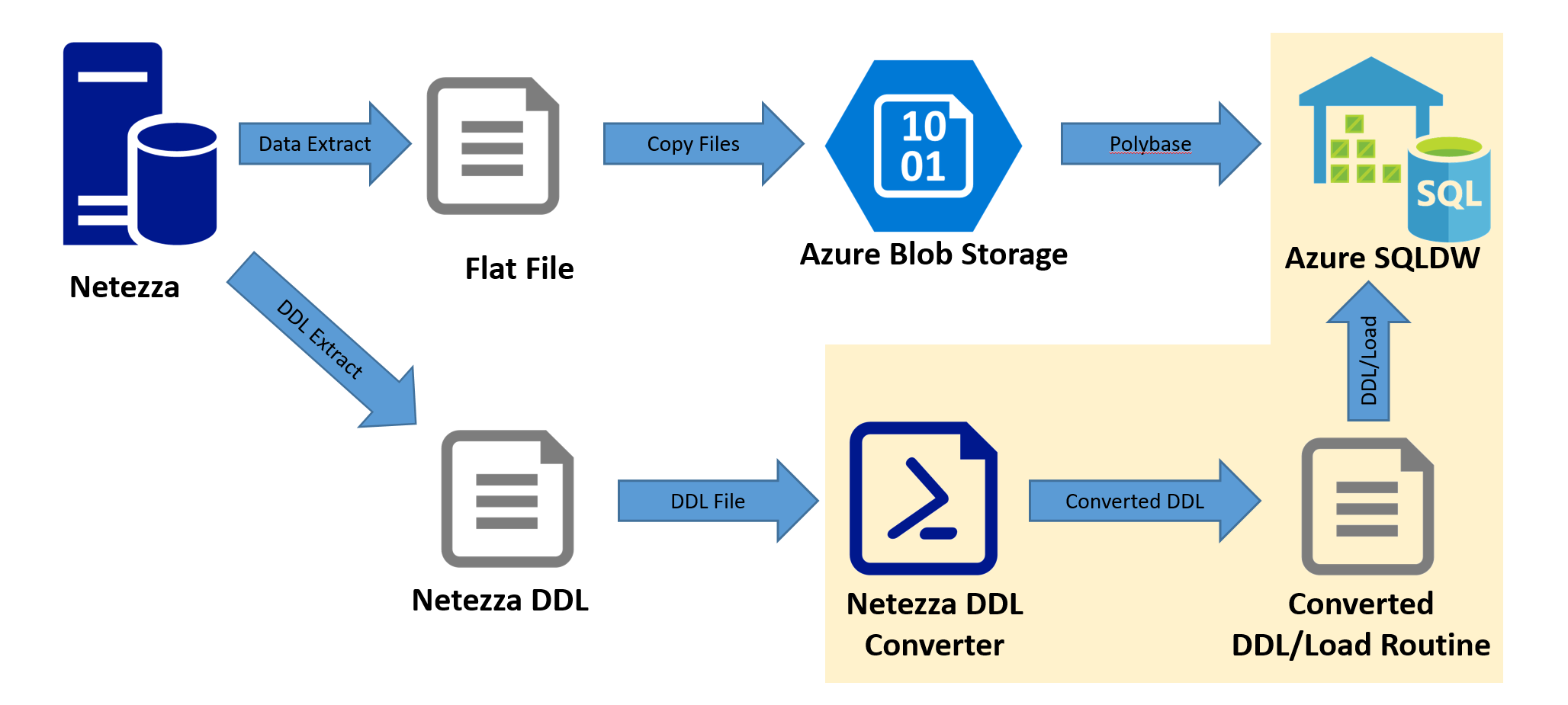
# Problem

Migrating Netezza to SQLDW requires several steps.

1. Data needs to be extracted from Netezza
2. The Netezza DDL needs to be converted to SQLDW format.
3. The data needs to be loaded into SQLDW.

This article will assume step 1 (Data Extraction) has been completed, the data is in UTF8 format, uploaded to Azure Blob Storage and ready to load into SQLDW.

This article will concentrate on how to convert the DDL and load the data into SQLDW.



# Solution

The diagram above shows the flow of data and DDL from Netezza to SQLDW. The DDL is first extracted from Netezza and then read into the DDL converter. Currently, the tool only converts Tables and does not support all Netezza objects.

The DDL converter completes the following tasks:

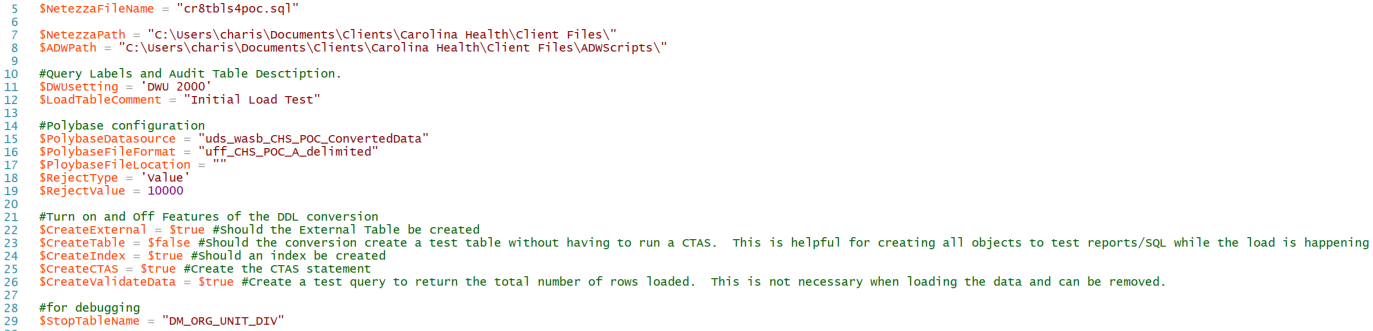
1. Creates the External tables necessary to load the extracted Data into SQLDW. Appendix A: Netezza DDL Sample
2. Loads the data from the external table into the destination table
3. Creates a Clustered Column Store Index on the on the Destination Table.

Prerequisites:

1. Netezza DDL is extracted using the Netezza Tools. DDL is extracted into a single file. This should just be the Create Tables. Views, Stored Procedures and other Netezza objects are considered out of scope for hit tool at this time.
2. SQLDW is configured for Polybase
3. Data Source for Polybase has been created
4. File Format for Polybase has been created
5. Create the Log table on SQLDW for tracking the Load Process. See Appendix B.

To get stared Open the ConvertNetezza\_to\_SQLDW.ps1 File in a Powershell Editor.

In the Script there are several line that need to be edited for your environment.



Edit the Following lines:

* Line 5: Enter the name of the Extracted Netezza DDL.
* Line 7: Location where the Extracted Netezza DDL is stored.
* Line 8: Location where the converted .SQL file will be stored.
* Line 11: This is the text that is entered into the log table when running the converted DDL and loading the data to indicate the DWU the SQLDW was running.
* Line 12: This is the text for the LoadComment Column in the Load\_Times Table.
* Line 15: Name of the Polybase Data Source created in SQLDW.
* Line 16: Name of the Polybase File Format created in SQLDW.
* Line 17: Location in WASB where the FlatFiles are stored. This is the Directory structure. It is assumed that each table will have its own location based on the table name.
* Line 18: RejectType to use when loading the data using Polybase.
* Line 19: RejectValue to use when loading the data using Polybase.

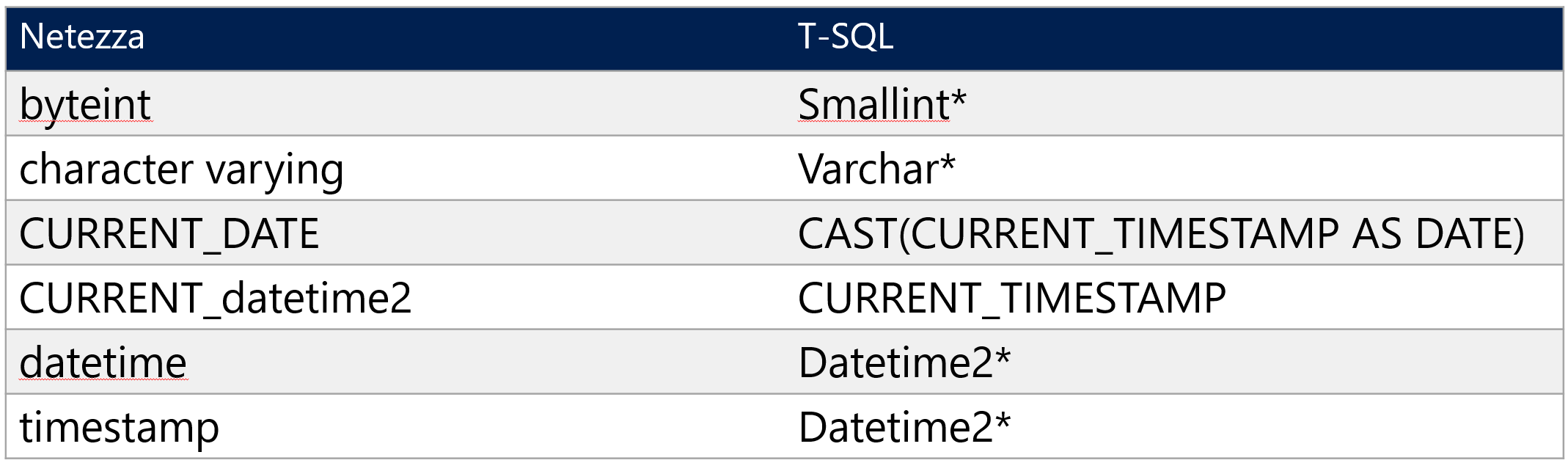
This section controls what is generated by the DDL migration process.

* Line 22: Should the External Table be created?
* Line 23: Should the conversion create a test table without having to run a CTAS. This is helpful for creating all objects to test reports/SQL while the load is happening.
* Line 24: Should an index be created
* Line 25: Create the CTAS statement
* Line 26: Create a test query to return the total number of rows loaded. This is not necessary when loading the data and can be removed.

# What Gets Converted

Currently the converter can only convert table structures and has limited functionality to perform the following:

**Data Types conversions:**



**Current limitations:**

Default values are removed from all columns as external tables do not support defaults.

# Output from Converter

After running the converter, each table in the Netezza DDL will generate a separate script for creating the External Table and loading the data into the final table. Each of these scripts can then be run individually through SSMS or through the Load Automation PowerShell script.

The Following DDL would be converted into the following script:

**Netezza Script**

CREATE TABLE CD\_DMN

(

DMN\_SK integer not null,

DMN\_NM character varying(30) not null,

DMN\_DESC character varying(250) not null,

DMN\_COL\_NM character varying(1024),

LOAD\_INFO\_SK bigint not null,

MAP\_REQD\_IND numeric(1,0) not null,

TENANT\_CMN\_CD\_OVRD\_IND numeric(1,0),

SRC\_ACT\_IND integer not null default 1,

SRC\_CRE\_TS timestamp,

SRC\_UPDT\_TS timestamp

,Constraint CD\_DMN\_UK UNIQUE (DMN\_NM)

)

DISTRIBUTE ON (DMN\_SK)

;

\echo

\echo \*\*\*\*\* Adding Primary Key Constraint: CD\_DMN

ALTER TABLE CD\_DMN ADD PRIMARY KEY (DMN\_SK);

**Converter Output:**

A file is generated for each table in the DDL. This File is name after the Table: CD\_DMN.sql

--- CREATE EXTERNAL DW TABLE ---

if Exists(select 1 from sys.tables where name = 'Ext\_CD\_DMN') Drop External Table [dbo].[Ext\_CD\_DMN]

CREATE EXTERNAL TABLE Ext\_CD\_DMN

(

DMN\_SK integer not null,

DMN\_NM varchar(30) not null,

DMN\_DESC varchar(250) not null,

DMN\_COL\_NM varchar(1024),

LOAD\_INFO\_SK bigint not null,

MAP\_REQD\_IND numeric(1,0) not null,

TENANT\_CMN\_CD\_OVRD\_IND numeric(1,0),

SRC\_ACT\_IND integer not null ,

SRC\_CRE\_TS datetime2,

SRC\_UPDT\_TS datetime2

)

WITH (DATA\_SOURCE = uds\_wasb\_CHS\_POC\_ConvertedData,

LOCATION = N'CD\_DMN',

FILE\_FORMAT = uff\_CHS\_POC\_A\_delimited,

REJECT\_TYPE = Value,

REJECT\_VALUE = 10000

)

--- VERIFY EXTERNAL DATA ---

SELECT TOP 10 \* FROM [dbo].[EXT\_CD\_DMN];

--- LOAD DATA VIA CTAS ---

if Exists(select 1 from sys.tables where name = 'CD\_DMN') Drop Table [dbo].[CD\_DMN]

Declare @start datetime

Set @start = getdate()

Create table CD\_DMN With (Distribution = Hash(DMN\_SK), Heap) As

Select \* from [dbo].[Ext\_CD\_DMN] option (label = 'Data Load for Table: CD\_DMN At: DWU 2000')

Insert into Load\_Times(Tablename, Duration, DWU, Load\_Comment, CreateDate) Select 'CD\_DMN', DateDiff(ss, @start, getdate()), 'DWU 2000', 'Initial Load Test', getdate()

--- CREATE THE INDEX ---

Create Clustered ColumnStore Index IdxCCI\_CD\_DMN on CD\_DMN

--- VALIDATE THE LOADED DATA ---

select count(\*) as Row\_Count, 'CD\_DMN' as TableName from [dbo].[CD\_DMN]

# Appendix A – Netezza DDL Sample

\echo

\echo \*\*\*\*\* Creating table: "CAL\_DM"

CREATE TABLE CAL\_DM

(

CAL\_DK bigint not null,

CAL\_DT timestamp,

CAL\_DT\_FULL\_NM character varying(30) not null,

CAL\_DT\_ABBREV character varying(150) not null,

YR\_NBR integer not null,

MTH\_OF\_YR\_NBR smallint not null,

MTH\_OF\_YR\_ABBREV character(3) not null,

MTH\_OF\_YR\_NM character varying(30) not null,

MTH\_AND\_YR\_NBR integer not null,

MTH\_AND\_YR\_ABBREV character varying(20) not null,

MTH\_AND\_YR\_NM character varying(30) not null,

QTR\_OF\_YR\_NBR smallint not null,

QTR\_OF\_YR\_ABBREV character(2) not null,

QTR\_AND\_YR\_NBR integer not null,

QTR\_AND\_YR\_ABBREV character varying(20) not null,

DY\_OF\_YR\_NBR smallint not null,

DY\_OF\_MTH\_NBR smallint not null,

DY\_OF\_WK\_NBR smallint not null,

DY\_OF\_WK\_ABBREV character(3) not null,

DY\_OF\_WK\_NM character varying(30) not null,

WK\_OF\_YR\_NBR smallint not null,

PRV\_YR\_NBR integer not null,

PRV\_MTH\_OF\_YR\_NBR smallint not null,

PRV\_MTH\_OF\_YR\_ABBREV character(3) not null,

PRV\_MTH\_OF\_YR\_NM character varying(30) not null,

PRV\_MTH\_AND\_YR\_NBR integer not null,

PRV\_MTH\_AND\_YR\_ABBREV character varying(20) not null,

PRV\_MTH\_AND\_YR\_NM character varying(30) not null,

PRV\_QTR\_OF\_YR\_NBR smallint not null,

PRV\_QTR\_OF\_YR\_ABBREV character(2) not null,

PRV\_QTR\_AND\_YR\_NBR integer not null,

PRV\_QTR\_AND\_YR\_ABBREV character varying(20) not null,

FY\_NBR integer,

FY\_BEGIN\_DT timestamp,

FY\_END\_DT timestamp,

FISC\_QTR\_NBR smallint,

FISC\_QTR\_ABBREV character(2),

FISC\_QTR\_BEGIN\_DT timestamp,

FISC\_QTR\_END\_DT timestamp,

FISC\_MTH\_NBR smallint,

FISC\_MTH\_ABBREV character(3),

FISC\_MTH\_NM character varying(30),

FISC\_MTH\_BEGIN\_DT timestamp,

FISC\_MTH\_END\_DT timestamp,

PRV\_FY\_NBR integer,

PRV\_FY\_BEGIN\_DT timestamp,

PRV\_FY\_END\_DT timestamp,

PRV\_FISC\_QTR\_NBR smallint,

PRV\_FISC\_QTR\_ABBREV character(2),

PRV\_FISC\_QTR\_BEGIN\_DT timestamp,

PRV\_FISC\_QTR\_END\_DT timestamp,

PRV\_FISC\_MTH\_NBR smallint,

PRV\_FISC\_MTH\_ABBREV character(3),

PRV\_FISC\_MTH\_NM character varying(30),

PRV\_FISC\_MTH\_BEGIN\_DT timestamp,

PRV\_FISC\_MTH\_END\_DT timestamp,

BUS\_DY\_IND numeric(1,0) not null,

HOL\_IND numeric(1,0) not null,

DT\_ERR\_IND numeric(1,0) not null,

DT\_ERR\_DESC character varying(150),

ADJ\_PD\_IND numeric(1,0),

CHS\_ALL\_YR\_CURR\_MTH\_IND numeric(1,0) not null,

CHS\_ALL\_YR\_CURR\_QTR\_IND numeric(1,0) not null,

CHS\_ALL\_YR\_YTD\_IND numeric(1,0) not null,

CHS\_CURR\_YR\_CURR\_MTH\_IND numeric(1,0) not null,

CHS\_CURR\_YR\_CURR\_QTR\_IND numeric(1,0) not null,

CHS\_CURR\_YR\_YTD\_IND numeric(1,0) not null,

CHS\_CURR\_YR\_IND numeric(1,0) not null,

CHS\_FIRST\_DY\_MTH\_DT\_NBR integer not null,

CHS\_FY\_NM character varying(30) not null,

CHS\_IDX\_BILL\_LOAD\_OFFSET\_MTH\_NBR integer,

CHS\_IDX\_BILL\_LOAD\_OFFSET\_YR\_NBR integer,

CHS\_LAST\_DY\_MTH\_IND numeric(1,0) not null,

CHS\_LAST\_YR\_CURR\_MTH\_IND integer not null,

CHS\_LAST\_YR\_CURR\_QTR\_IND integer not null,

CHS\_LAST\_YR\_YTD\_IND integer not null,

CHS\_LAST\_YR\_IND integer not null,

CHS\_LAST\_DY\_MTH\_NBR integer not null,

CHS\_OFFSET\_DY\_NBR integer,

CHS\_OFFSET\_MTH\_NBR integer,

CHS\_OFFSET\_QTR\_NBR integer,

CHS\_OFFSET\_WK\_NBR integer,

CHS\_OFFSET\_YR\_NBR integer,

CHS\_OVRALL\_DY\_NBR integer not null,

CHS\_OVRALL\_MTH\_NBR integer not null,

CHS\_OVRALL\_QTR\_NBR integer not null,

CHS\_OVRALL\_WK\_NBR integer not null,

CHS\_OVRALL\_YR\_NBR integer not null,

CHS\_WK\_BEGIN\_DT timestamp,

CHS\_WK\_NBR\_BY\_DY\_NBR integer not null,

CHS\_WKDY\_WKEND\_DESC character varying(30) not null,

DWM\_LOAD\_INFO\_SK bigint not null

)

DISTRIBUTE ON (CAL\_DK)

;

\echo

\echo \*\*\*\*\* Adding Primary Key Constraint: CAL\_DM

ALTER TABLE CAL\_DM ADD PRIMARY KEY (CAL\_DK);

/\*

Number of columns 92

(Variable) Data Size 363 - 1043

Row Overhead 41

====================== =============

Total Row Size (bytes) 404 - 1084

\*/

\echo

\echo \*\*\*\*\* Creating table: "CD\_DMN"

CREATE TABLE CD\_DMN

(

DMN\_SK integer not null,

DMN\_NM character varying(30) not null,

DMN\_DESC character varying(250) not null,

DMN\_COL\_NM character varying(1024),

LOAD\_INFO\_SK bigint not null,

MAP\_REQD\_IND numeric(1,0) not null,

TENANT\_CMN\_CD\_OVRD\_IND numeric(1,0),

SRC\_ACT\_IND integer not null default 1,

SRC\_CRE\_TS timestamp,

SRC\_UPDT\_TS timestamp

,Constraint CD\_DMN\_UK UNIQUE (DMN\_NM)

)

DISTRIBUTE ON (DMN\_SK)

;

\echo

\echo \*\*\*\*\* Adding Primary Key Constraint: CD\_DMN

ALTER TABLE CD\_DMN ADD PRIMARY KEY (DMN\_SK);

/\*

Number of columns 10

(Variable) Data Size 46 - 1350

Row Overhead 30

====================== =============

Total Row Size (bytes) 76 - 1380

\*/

# Appendix B – Load\_Times Table

CREATE TABLE [dbo].[Load\_Times] (

[TableName] varchar(255) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL,

[Duration] bigint NULL,

[DWU] varchar(25) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL,

[LoadComment] varchar(200) COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS NULL,

[CreateDate] datetime

)

WITH (HEAP, DISTRIBUTION = ROUND\_ROBIN);

/\*\*\*This Artifact belongs to the Data SQL Ninja Engineering Team\*\*\*/