



Master Data Management




Installation Guide

Release 4.9
B035-0000-9701
March 2024

Copyrights or Trademarks

All copyrights and trademarks used in Teradata documentation are the property of their respective owners. For more information, see [Section : “Teradata Trademark and Trademark Attributions.”](#)

Product Safety

Safety Type	Description
	Indicates a situation which, if not avoided, could result in damage to property, such as to equipment or data, but not related to personal injury.
	Indicates a hazardous situation which, if not avoided, could result in minor or moderate personal injury.
	Indicates a hazardous situation which, if not avoided, could result in death or serious personal injury.

Warranty Disclaimer

Except as may be provided in a separate written agreement with Teradata or required by applicable law, the information available from the Teradata Documentation website or contained in Teradata information products is provided on an "as-is" basis, without warranty of any kind, either express or implied, including the implied warranties of merchantability, fitness for a particular purpose, or noninfringement.

The information available from the Teradata Documentation website or contained in Teradata information products may contain references or cross-references to features, functions, products, or services that are not announced or available in your country. Such references do not imply that Teradata Corporation intends to announce such features, functions, products, or services in your country. Please consult your local Teradata Corporation representative for those features, functions, products, or services available in your country.

The information available from the Teradata Documentation website or contained in Teradata information products may be changed or updated by Teradata at any time without notice. Teradata may also make changes in the products or services described in this information at any time without notice.

Feedback

To maintain the quality of our products and services, e-mail your comments on the accuracy, clarity, organization, and value of this document to: docs@teradata.com.

Any comments or materials (collectively referred to as "Feedback") sent to Teradata Corporation will be deemed nonconfidential. Without any payment or other obligation of any kind and without any restriction of any kind, Teradata and its affiliates are hereby free to (1) reproduce, distribute, provide access to, publish, transmit, publicly display, publicly perform, and create derivative works of, the Feedback, (2) use any ideas, concepts, know-how, and techniques contained in such Feedback for any purpose whatsoever, including developing, manufacturing, and marketing products and services incorporating the Feedback, and (3) authorize others to do any or all of the above.

Teradata Trademark and Trademark Attributions

Teradata, BYNET, Claraview, Covalent, DecisionCast, IntelliBase, IntelliCloud, IntelliFlex, IntelliSphere, nPath, QueryGrid, SQL-MapReduce, Stacki, "Teradata" logo, Teradata Analytics Platform, Teradata Decision Experts, "Teradata Labs" logo, Teradata ServiceConnect, and Teradata Vantage are trademarks or registered trademarks of Teradata Corporation or its affiliates in the United States and other countries.

Adaptec and SCSISelect are trademarks or registered trademarks of Adaptec, Inc.

Amazon Web Services, AWS, Amazon Elastic Compute Cloud, Amazon EC2, Amazon Simple Storage Service, Amazon S3, AWS CloudFormation, and AWS Marketplace are trademarks of Amazon.com, Inc. or its affiliates in the United States and/or other countries.

AMD Opteron and Opteron are trademarks of Advanced Micro Devices, Inc.

Apache, Apache Avro, Apache Hadoop, Apache Hive, Hadoop, and the yellow elephant logo are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries.

Apple, Mac, and OS X all are registered trademarks of Apple Inc.

Axeda is a registered trademark of Axeda Corporation. Axeda Agents, Axeda Applications, Axeda Policy Manager, Axeda Enterprise, Axeda Access, Axeda Software Management, Axeda Service, Axeda ServiceLink, and Firewall-Friendly are trademarks and Maximum Results and Maximum Support are servicemarks of Axeda Corporation.

CENTOS is a trademark of Red Hat, Inc., registered in the U.S. and other countries.

Cloudera and CDH are trademarks or registered trademarks of Cloudera Inc. in the United States, and in jurisdictions throughout the world.

Data Domain, EMC, PowerPath, SRDF, and Symmetrix are either registered trademarks or trademarks of EMC Corporation in the United States and/or other countries.

GoldenGate is a trademark of Oracle.

Hewlett-Packard and HP are registered trademarks of Hewlett-Packard Company.

Hortonworks, the Hortonworks logo and other Hortonworks trademarks are trademarks of Hortonworks Inc. in the United States and other countries.

Intel, Pentium, and XEON are registered trademarks of Intel Corporation.

IBM, CICS, RACF, Tivoli, IBM Spectrum Protect, and z/OS are trademarks or registered trademarks of International Business Machines Corporation.

Linux is a registered trademark of Linus Torvalds.

LSI is a registered trademark of LSI Corporation.

Microsoft, Azure, Active Directory, Windows, Windows NT, and Windows Server are registered trademarks of Microsoft Corporation in the United States and other countries.

NetVault is a trademark of Quest Software, Inc.

Novell and SUSE are registered trademarks of Novell, Inc., in the United States and other countries.

Oracle, OpenJDK, Java, and Solaris are trademarks or registered trademarks of Oracle and/or its affiliates.

QLogic and SANbox are trademarks or registered trademarks of QLogic Corporation.

Quantum and the Quantum logo are trademarks of Quantum Corporation, registered in the U.S.A. and other countries.

Red Hat is a trademark of Red Hat, Inc., registered in the U.S. and other countries. Used under license.

SAP is the trademark or registered trademark of SAP AG in Germany and in several other countries.

SAS and SAS/C are trademarks or registered trademarks of SAS Institute Inc.

Sentinel® is a registered trademark of SafeNet, Inc.

Simba, the Simba logo, SimbaEngine, SimbaEngine C/S, SimbaExpress and SimbaLib are registered trademarks of Simba Technologies Inc.

SPARC is a registered trademark of SPARC International, Inc.

Unicode and the Unicode logo are registered trademarks of Unicode, Inc. in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Veritas, the Veritas Logo and NetBackup are trademarks or registered trademarks of Veritas Technologies LLC or its affiliates in the U.S. and other countries.

Other product and company names mentioned herein may be the trademarks of their respective owners.

Purpose

Welcome to Teradata Master Data Management® (MDM). Master Data Management provides a way to integrate and consolidate master data without having to replace existing infrastructure investments.

You can stage, consolidate, validate, cleanse, store, augment, cross-reference, and publish data to systems in your IT infrastructure. By ensuring cross-system data consistency, MDM can enable flawless execution of business processes – while leveraging existing IT investments and reducing the total cost of ownership to manage business critical data.

Topics:

- [About Teradata Master Data Management](#)
- [About This Book](#)
- [Related Documentation](#)
- [Customer Support](#)
- [Documentation Feedback](#)

About Teradata Master Data Management

Master Data Management helps you synchronize key data elements across disparate systems and geographies. Master Data Management enables you with:

- Data Staging for loading, cleansing, validating, aggregating, and publishing data.
- Data Model Maintenance for model extension and model repository.
- Master Data Lifecycle Maintenance for data maintenance workflows and versioning, editing and archiving data.

MDM simplifies deployment, integration of heterogeneous systems, is easier to manage, provides high quality data, and decouples data from business application software. Consider using Master Data Management at these levels in your organization:

- Enterprise level
- Division level
- Process level (for example, Supply Chain, Customer Management)
- Sub-Process level (for example, Replenishment, and Master Planning)

About This Book

This document describes how to install Teradata MDM.

Target Audience

This guide is intended for the users who are installing Teradata MDM.

What You Should Know

Some sections of this document assume knowledge of third party software.

Document Structure

This book has the following chapters:

- [Chapter 1: “MDM System Requirements.”](#) Describes MDM system requirements and supported platforms.
- [Chapter 2: “MDM Installation.”](#) Provides a step by step procedure for installing MDM and CRDM.
- [Chapter 3: “MDM Database Preparation.”](#) Describes about database preparation.
- [Chapter 4: “MDM WebClient Deployment.”](#) Describes about WebClient deployment.
- [Chapter 5: “Launch MDM Server and Application Server.”](#) Describes about starting the MDM server and client.
- [Chapter 6: “MDM Deployment Manager.”](#) Describes MDM Deployment manager.
- [Chapter 7: “MDM Upgrade.”](#) Describes MDM Upgrade process.
- [Appendix A: “WebClient War Deployment on Weblogic”](#) Provides the WebClient war deployment on Weblogic.

Changes to This Book

The following changes were made to this book in support of the current release. For a complete list of changes to the product, refer *MDM Platform Release Definition* associated with this release.

Date and Release	Description
November 2013, 3.3	Replaced all the Installer Panels in Chapter 2: “MDM Installation” and Chapter 10: “Uninstalling MDM” . Updated Section : “RDM Sandbox Sizing Guidelines” MDM Upgrade chapter updated.

Date and Release	Description
April 2014, 3.3.1	Replaced all the Installer Panels in Chapter 2: “MDM Installation” and Chapter 10: “Uninstalling MDM” . Updated Section : “WebClient Deployment on Tomcat” for Web Profiler support. Section : “WebClient Deployment in WebSphere Liberty Server” and Section : “WebClient Deployment in Oracle WebLogic” for Web Profiler support. Updated Chapter 7: “MDM Upgrade” .
July 2014, 3.3.1	In Chapter 7: “MDM Upgrade” , updated Section: “Upgrade from MDM 3.03.01.00 to MDM 3.04.00.00. Included Section: “Upgrade from MDM 3.02.01.00 to MDM 3.03.01.00, Section: “Upgrade from MDM 3.02.00.01 to MDM 3.03.01.00, Section: “Upgrade from MDM 3.01.00.01 to MDM 3.03.01.00.
November 2014, 3.4	Updated Chapter 7: “MDM Upgrade” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” and Chapter 10: “Uninstalling MDM” .
December 2014, 3.4.0.1	Updated Chapter 7: “MDM Upgrade” and few notes added.
January 2015, 3.4.0.1	Updated Chapter 7: “MDM Upgrade” and included WebProfile configuration steps in Chapter 4: “MDM WebClient Deployment” .
February 2015, 3.4.1	Updated Chapter 7: “MDM Upgrade” and Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” .
November 2015, 3.5	Updated Chapter 7: “MDM Upgrade” and Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” .
March 2016, 3.5.1	Updated Chapter 7: “MDM Upgrade” and Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” .
June 2016, 3.5.2	Updated Chapter 7: “MDM Upgrade” and Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” .
November 2016, 3.5.3	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” .
June 2017, 4.0.0	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” . Upgrade chapter updated.
September 2017, 4.0.1	Updated system requirements in Chapter 1: “MDM System Requirements” . Upgrade chapter updated.

Date and Release	Description
December 2017, 4.1.0	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” .
June 2018, 4.2.0	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” . Included MDM Patch Installation chapter.
April 2019, 4.3.0	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” Chapter 6: “MDM Deployment Manager” and Chapter 10: “Uninstalling MDM” . Removed all instances of CRDM.
November 2019, 4.4.0	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” and Chapter 10: “Uninstalling MDM” .
June 2020, 4.5.0	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” and Chapter 10: “Uninstalling MDM” . Updated upgrade chapter with changes in Chapter 7: “MDM Upgrade”
March 2021, 4.6.0	Updated system requirements in Chapter 1: “MDM System Requirements” . Replaced all the Installer Panels in Chapter 2: “MDM Installation” and Chapter 10: “Uninstalling MDM” .
February 2022, 4.7.0	Updated system requirements in Chapter 1: “MDM System Requirements” Updated Script based Installation in Chapter 2: “MDM Installation” Updated Script based Installation in Chapter 6: “MDM Deployment Manager” Minor updates in Chapter 7: “MDM Upgrade” Removed chapter MDM Silent Installation, MDM Patch Installation, Uninstalling MDM
June 2022, 4.7.1	Minor updates related to release version
August 2022, 4.8	Release Version Updated Updated SAML Configuration in Appendix: Spring Security Added New appendix for SSL Configuration Minor update related to system requirement
March 2024, 4.9	Release Version Updated Vault installation and upgrade steps updated Unicode support provided for Master OOTB Tables System requirement section updated for JDBC, Database, Ngnix , JDK, browser

Conventions

Table i lists examples of the typographic conventions used to display different types of information in this document.

Table i: Typographic conventions used in this document

Item	Example	Explanation
Code	Call NotifyPending;	File names, executable code, commands, and configuration statements are shown in monospaced font.
Class Names	Make the Class Configurations pointer in the Module Configuration class a primary key.	Class names appear in bold.
Interface element	Click Organization Management in the toolbar.	Button names, field names, window names are shown in a bolded sans-serif font.
Pathname	C:\Teradata\webdriver or /Teradata/webdriver	Windows pathnames are shown in monospaced font, with backslash path separators.
Meta-variable	<i>Teradata_Home</i> \webdriver or <i>Teradata_Home</i> /webdriver	Portions of code that you replace with specific values are shown in italic monospaced font.
Documentation or book names	<i>Master Data Management (MDM) - Installation Guide</i>	Document or book names referenced in this book are shown in italics.

Any of the following types of notes may appear in this book:

Note: This kind of note contains information that is useful or interesting but not essential to an understanding of the main text.

Caution: This kind of note contains instructions that are especially important to follow for proper functioning of the product.

Warning: This kind of note contains instructions that must be followed to avoid potential crashes or loss of data.

Related Documentation

For more information on MDM, refer the following documents:

- *Master Data Management Release Definition*
(Master Data Management 4.9.0 Release Definition.pdf)
- *Master Data Management Developer Guide*

- (Master Data Management 4.9.0 Developer Guide.pdf)
- *Master Data Management Reference Guide*
(Master Data Management 4.9.0 Reference Guide.pdf)
- *Master Data Management Studio User Guide*
(Master Data Management 4.9.0 Studio User Guide.pdf)
- *Master Data Management Server Guide*
(Master Data Management 4.9.0 Server Guide.pdf)

The above Teradata documents are available at: <https://docs.teradata.com>

Documentation Accessibility

To read the pdf files, you must have Adobe Acrobat Reader, version 4.0 or higher. If you do not have Acrobat Reader on your machine, you can download it from Adobe's Web site at <http://www.adobe.com>.

Customer Support

Customer support is available at the Teradata customer support Web site (<https://access.teradata.com>), where you can:

- Request shipment of software.
- Download software documentation.
- Submit new issues or cases.
- Track the status of current issues or cases.

Documentation Feedback

Please share your thoughts and ideas:

- Send feedback to docs@teradata.com.
- Navigate to <https://teradata-documentation.ideas.aha.io/ideas/new> and provide your ideas.

Table of Contents

Prefacevii

Purpose	vii
About Teradata Master Data Management.	vii
About This Book.	viii
Target Audience.	viii
What You Should Know	viii
Document Structure	viii
Changes to This Book	ix
Conventions	xiv
Related Documentation	xiv
Documentation Accessibility	xv
Customer Support.	xv
Documentation Feedback	xv

Chapter 1: MDM System Requirements 1

System Requirements	1
Database User Requirements	4
Create Database User	6
Grant Database User Rights	6
Optional User Rights	8
Topology Database	9
Create Publication and Sandbox Databases	11
Grant Database User Rights on Publication and Sandbox Databases	11
Create Staging Databases for MDM Topology	12
Grant Database User Rights on Staging Databases	12
Configuring User for Default Database Setup	13

Chapter 2: MDM Installation 15

Packaging Overview	15
Prerequisites for MDM Installation	15
Installation and Configuration of Nginx	16

Configure HashiCorp Vault with MDM Installer	20
Vault: Frequently Asked Questions	25
Installing MDM	28
Script Based Installation	29
Creating Script Based Installation Property File for MDM	29
LDAP Configuration	36
Starting the Script Based Installation Process	36
Installing MDM on Default Database Setup	40
Vault Setup After MDM Installation	40
Configure Vault After Installation	40
Additional Details	41
DBC Object Details	41
Fallback on MDM Database Tables	43
Enable Output Staging in MDM	43
 Chapter 3: MDM Database Preparation	 46
Preparing Database for MDM	46
 Chapter 4: MDM WebClient Deployment	 48
WebClient Deployment in WebSphere Liberty Server	48
Installing the Application	48
WebClient Deployment in Oracle WebLogic	49
WebClient Deployment on Tomcat	50
Web Services	52
Troubleshooting	52
 Chapter 5: Launch MDM Server and Application Server	 56
MDM Server & Application Server Startup Process	56
Start Services	58
Log Files	60

Chapter 6: MDM Deployment Manager 62

Deploying Custom Application using the properties file.....	62
Script Based DM Installation Process	63
Creating Property File for Deployment Manager	63
Starting the Script Based Installation Process for DM.....	67
Debugging Deployment Process.....	69

Chapter 7: MDM Upgrade..... 71

Introduction.....	71
Upgrade Process	72
Upgrade Steps	75
Upgrade: Restart and Logs.....	79
Troubleshooting	80
Sync Up Custom Application.....	84
Backdown Procedure for MDM	87
Troubleshooting.....	88

Appendix A: WebClient War Deployment on Weblogic 91

WebClient War Deployment on Weblogic.....	91
---	----

Appendix B: SSL Configuration with MDM 94

Configure SSL with MDM.....	94
Setup SSL Certificate in MDM and Shared Services.....	97

Appendix C: Table Name Changes During Upgrade 101

Introduction.....	101
List of Tables Upgrading During Upgrade.....	102

Appendix D: Spring Security 105

Overview	105
Namespace Configuration	106
Security Configuration	106
MDM Spring Security Login and Logout	107
MDM Spring Security Architecture	107
Spring Security Integration with MDM	108
Configure Authentication Provider	108
User Service Authentication Provider	109
LDAP Authentication Provider	110
ADS Authentication Provider	114
Kerberos Authentication Provider	115
SiteMinder Authentication Provider	117
SAML Authentication Provider	118
ADFS Authentication Provider	122
HTTP Strict Transport Security (HSTS)	124
OAuth2 Authentication	125
Troubleshooting	128

Index 132

List of Figures

Figure 1: Database User Requirements Flowchart	5
Figure 2: Database Topology	10
Figure 3: MDM Database Preparation	47
Figure 4: Upgrade Process Flow Diagram	73
Figure 5: Weblogic Console	91
Figure 6: Weblogic Console Home Page	92
Figure 7: Summary of Deployments.	92
Figure 8: Install Application Assistant	92
Figure 9: Install Application Assistant—Choose Targeting Style	92
Figure 10: Install Application Assistant—Choose Targeting Style	92
Figure 11: MDM Spring Security Architecture	108
Figure 12: Simple SAML Flow	119
Figure 13: High Level Authentication of SAML	120
Figure 14: ADFS High Level Concept	123

List of Tables

Table i: Typographic conventions used in this document xiv

Table 1: Hardware Specifications for Typical Workstation (Development Environment) . . . 1

Table 2: Hardware Specifications for Server (Production Environment) 2

Table 3: System Requirements for MDM Server and MDM Studio 3

Table 4: Specifications for MDM Server (Cloud) 4

Table 5: MDM Script Based Installation Property Values. 30

Table 6: Collapsed Mode and Co-located Mode—MDM Server Start Up Process. 57

Table 7: DM Script Based Installation Property Values 63

Table 8: Error Messages in SYS_UPGRADE_TASK_LOG Table. 82

Table 9: OAuth Configuration Parameters. 127

CHAPTER 1 MDM System Requirements

What's In This Chapter

This chapter provides information about MDM system requirements and supported platforms.

Topics include:

- [System Requirements](#)
- [Database User Requirements](#)

System Requirements

Before installing MDM, ensure to have the following hardware and software configurations on the machine where you would be installing MDM. [Table 1](#) and [Table 2](#) lists hardware specifications for typical workstation (development environment) and server (production environment) respectively:

Table 1: Hardware Specifications for Typical Workstation (Development Environment)

Hardware	Recommended
Processor	1 CPU
Memory	8.0 GB
Disk Space	500 GB

Table 2: Hardware Specifications for Server (Production Environment)

Hardware	Recommended
Processor	2 CPU
Memory	16.0 GB
Disk Space	500 GB



Minimum free disk space required to install MDM is 2GB.

MDM can be installed on any of the following supported operating systems:

- Windows 10, Windows Server 2019, 2022
- SUSE Linux (SLES 12.3/15.2)
- Red Hat Linux (RHEL) 8.x

[Table 3](#) lists other system requirements for MDM Server and MDM Studio on any of the above operating systems.

Table 3: System Requirements for MDM Server and MDM Studio

Components	Software Version	Required For
Development Environment	Eclipse IDE for Java EE (IDE v2018-09) and Teradata Plug-in for Eclipse 15.10 works with JDK 8 only.	MDM Studio
Application Server	Apache Tomcat 9.0.31 or above and below 10 Or IBM Liberty WebSphere Application 22.0.0.8	MDM Server
JDK	Oracle JDK 11.x, 15.x, 18.x & Zulu JDK 11.x and 15.x only	Application Server and MDM Server
Database Server	TD 17.0 TD 17.10 TD 17.20 TD 20.x Vantage 2.0	Database Host
Database Client	Teradata JDBC 20.x TTU 17.0, TTU 17.10, TTU 17.20, 20.x BTEQ	MDM Server
Browser	Microsoft Edge 120.x or Firefox 120.x or Chrome 120.x	MDM Client
Web Server	Nginx version 1.20.1, 1.23.x Apache httpd server version 2.4.48 Hashicorp Vault 1.11.3 and above and below 1.15.1	MDM Shared Services

Table 3: System Requirements for MDM Server and MDM Studio

Components	Software Version	Required For
Git Client	Git Bash 2.16.2(Only for Windows) Linux Bash: Greater than 4.x+	MDM Server

MDM Server Installation Requirements for AWS

Below is the system requirements for the MDM Server Installation in AWS:

Table 4: Specifications for MDM Server (Cloud)

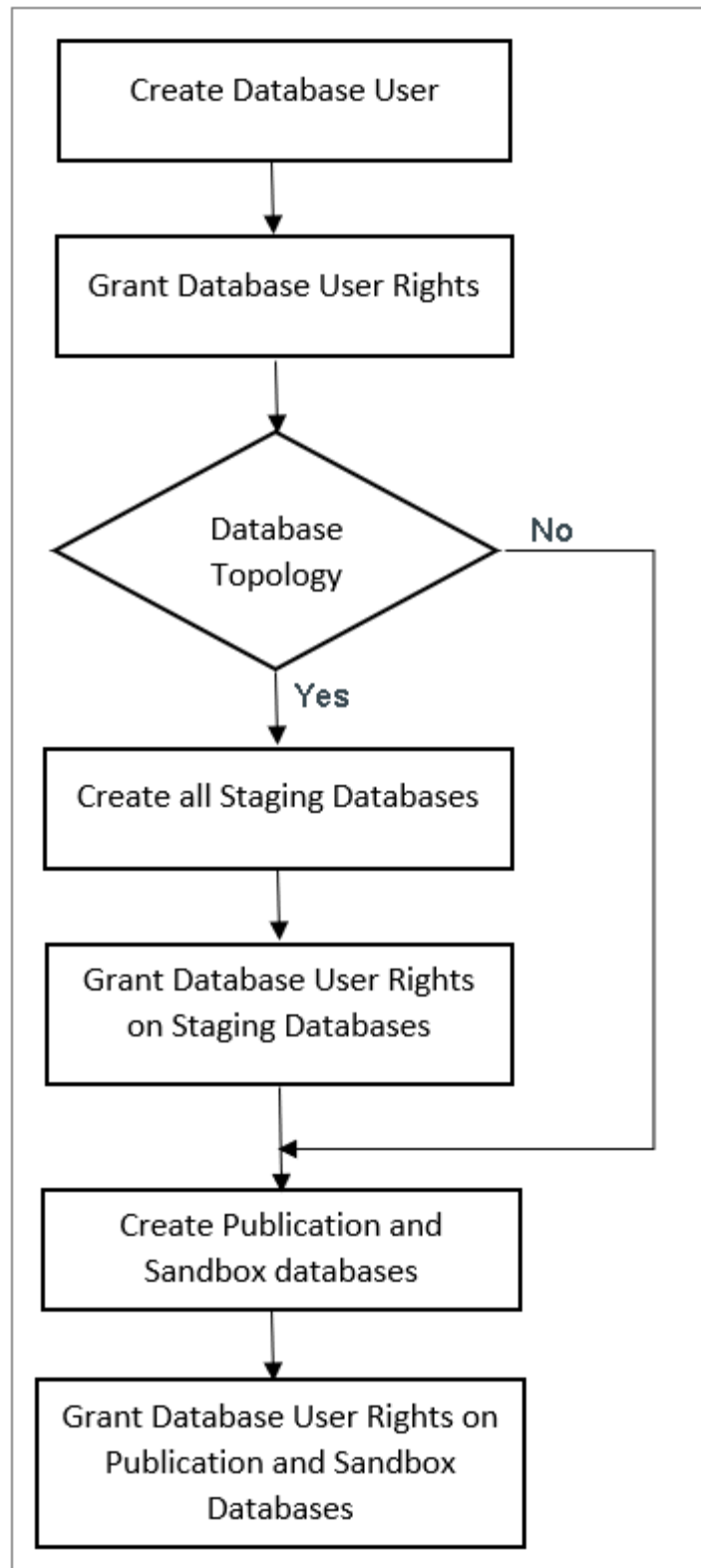
Components	Recommended
Operating System	Microsoft Windows Server 2016 Datacenter AWS Cloud instance
Storage	Standard hard drive, 100 GB space
Processor	Intel(R) Xeon(R) CPU E5- 2676 v3 @ 2.40 GHz 2.40 GHz
RAM	16 GB
Instance Type	M5.xlarge

Database User Requirements

Prerequisites:

- MDM can be installed successfully only on a Teradata database that supports the Extended Object Name (EON) feature.
- Before installing MDM, configure the Teradata database for MDM installation, follow the flowchart ([Figure 1](#)) and the steps described in the below section.

Figure 1: Database User Requirements Flowchart



Create Database User

Create a user database with a minimum of 200MB permanent space, 50MB spool space and 50MB temp space.

```
CREATE USER <MDM_USER> AS PASSWORD=temppwd, PERM=209715200,  
SPOOL=52428800, TEMPORARY=52428800;
```



In production environment based on the number of tables required, the recommended configurations would be minimum of 12GB perm space, 5GB spool and temp space for the MDM user and 5GB perm space for all the MDM staging databases used in the MDM installation. For Sandbox staging database, minimum perm space should be 12GB. Default character set in MDM is LATIN.

Grant Database User Rights

The following grant privileges are required for the database user to create and manage all database objects in MDM.

- 1 To execute UDFs that are required for C&S and Matching

```
GRANT CREATE FUNCTION, EXECUTE FUNCTION, DROP FUNCTION ON SYSLIB TO  
<MDM_USER>;  
GRANT CREATE FUNCTION , EXECUTE FUNCTION ,DROP FUNCTION,CREATE  
PROCEDURE ,EXECUTE PROCEDURE ON SQLJ TO <MDM_USER>  
GRANT UDTUSAGE ON SYSUDTLIB TO < MDM_USER >;  
GRANT UDTTYPE ON SYSUDTLIB TO MDM;
```

- 2 Grant user rights to create and manage all database objects in MDM.

```
GRANT CREATE MACRO, CREATE TABLE, CREATE VIEW, DELETE, DROP MACRO,  
DROP TABLE, DROP VIEW, INSERT, RETRIEVE, SELECT, UPDATE ON <MDM_USER>  
TO <MDM_USER>;  
  
GRANT EXECUTE PROCEDURE, ALTER PROCEDURE, CREATE PROCEDURE, DROP  
PROCEDURE, EXECUTE ON <MDM_USER> TO <MDM_USER>;
```



The below listed access rights are required for schema generation or upgrade on MDM. After MDM schema generation, the below rights can be revoked by the user.

```
REVOKE CREATE MACRO, DROP MACRO, CREATE PROCEDURE, DROP PROCEDURE ON  
<MDM_USER> TO <MDM_USER>;
```

- 3 Grant user rights on selected DBC Tables

MDM refers to few DBC metadata for its operation as most of the processing in MDM is dynamic and requires DBC metadata for constructing run time queries and processing logic. In a large Teradata database system, performance issues would arise when MDM

tries to access the DBC's metadata. A solution to the above problem is implemented by maintaining locally in MDM all the required DBC metadata (specific to a particular MDM installation).

For MDM user to access the tables on DBC, you can provide grant access on ALL the table of DBC using the following query:

```
GRANT SELECT ON dbc TO <MDM_USER>;
```

If you do not want to provide access to all DBC tables, use the following query to provide access to selected DBC tables which are must for functioning of MDM features.

```
GRANT SELECT on DBC.TABLES to <MDM_USER>;
GRANT SELECT on DBC.ALL_RI_PARENTSV to <MDM_USER>;
GRANT SELECT on DBC.ALLRIGHTS to <MDM_USER>;
GRANT SELECT on DBC.DATABASESV to <MDM_USER>;
GRANT SELECT on DBC.ALL_RI_CHILDRENV to <MDM_USER>;
GRANT SELECT on DBC.ALL_RI_PARENTS to <MDM_USER>;
GRANT SELECT on DBC.ALLRIGHTSV to <MDM_USER>;
GRANT SELECT on DBC.ALLROLERIGHTS to <MDM_USER>;
GRANT SELECT on DBC.ALLROLERIGHTSV to <MDM_USER>;
GRANT SELECT on DBC.ALLSPACE to <MDM_USER>;
GRANT SELECT on DBC.ALLSPACEV to <MDM_USER>;
GRANT SELECT on DBC.COLUMNS to <MDM_USER>;
GRANT SELECT on DBC.COLUMNSV to <MDM_USER>;
GRANT SELECT on DBC.DATABASES to <MDM_USER>;
GRANT SELECT on DBC.DBCINFO to <MDM_USER>;
GRANT SELECT on DBC.DBCINFOV to <MDM_USER>;
GRANT SELECT on DBC.ERRORMSGs to <MDM_USER>;
GRANT SELECT on DBC.INDICES to <MDM_USER>;
GRANT SELECT on DBC.RI_DISTINCT_CHILDRENV to <MDM_USER>;
GRANT SELECT on DBC.ROLEMEMBERSV to <MDM_USER>;
GRANT SELECT on DBC.SHOWTBLCHECKS to <MDM_USER>;
GRANT SELECT on DBC.TABLESIZE to <MDM_USER>;
GRANT SELECT on DBC.TABLESV to <MDM_USER>;
GRANT SELECT on DBC.USERS to <MDM_USER>;
GRANT SELECT on DBC.DISKSPACE to <MDM_USER>;
GRANT SELECT on DBC.DISKSPACEV to <MDM_USER>;
GRANT SELECT on DBC.INDICESV to <MDM_USER>;
GRANT SELECT on DBC.ALL_RI_CHILDREN to <MDM_USER>;
GRANT SELECT on DBC.RI_DISTINCT_CHILDREN to <MDM_USER>;
GRANT SELECT on DBC.ROLEMEMBERS to <MDM_USER>;
GRANT SELECT on DBC.SHOWCOLCHECKS to <MDM_USER>;
GRANT SELECT on DBC.SHOWCOLCHECKSV to <MDM_USER>;
GRANT SELECT on DBC.SHOWTBLCHECKSV to <MDM_USER>;
GRANT SELECT on DBC.TABLESIZEV to <MDM_USER>;
GRANT SELECT on DBC.USERSV to <MDM_USER>;
```

Optional User Rights

Following are optional grants which are required by MDM for some specific modules.

- 1 For MDM user to access Geospatial attributes, the Geospatial access right must be provided.

Grant UDTUsage on SYSUDTLIB to <MDM_USER>
- 2 For MDM User to access non temporal attributes, grant the below rights. User can use the NONTEMPORAL prefix to perform nontemporal operations on transaction-time and bitemporal tables in the database and required for schema generation process of Temporal tables.

```
GRANT NONTEMPORAL ON <MDM_USER> TO <MDM_USER>;
```

3 For Non MDM, grant the below rights.

- No access rights required for registering an external source. Select or DML privilege has to be given to MDM user depending on usage.
- Cleansing and Standardization

```
GRANT SELECT, UPDATE ON <NON MDM> TO <MDM_USER>;
```



For any NON MDM view with table referencing to a different database (external database), the Non MDM database should have SELECT and UPDATE WITH GRANT OPTION access on external database.

```
GRANT SELECT ON <NON MDM> TO <MDM_USER> WITH GRANT OPTION;
```

- Matching

```
GRANT SELECT ON <NON MDM> TO <MDM_USER>;
```

- Survivorship

```
GRANT SELECT ON <NON MDM> TO <MDM_USER>;
```

- Configurable UI

```
GRANT SELECT, UPDATE, INSERT, DELETE ON <NON MDM> TO <MDM_USER>;
```



- UPDATE, INSERT, DELETE operations work only on Simple Views
 - For any NON MDM view with table referencing to a different database (external database), the Non MDM database should have SELECT, UPDATE, INSERT, DELETE WITH GRANT OPTION access on external database.
-

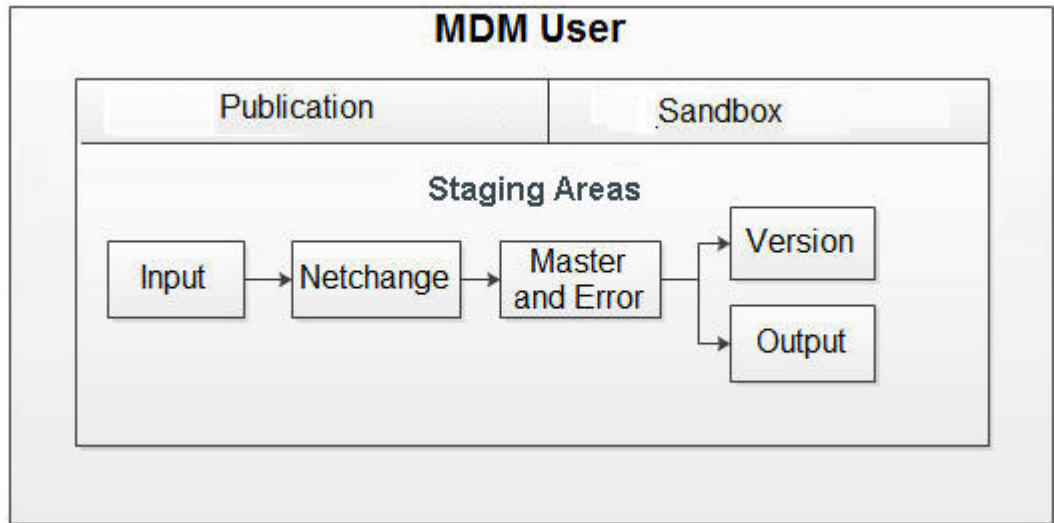
Topology Database

MDM database supports both topology based and non topology based database configurations.

In MDM topology based database configuration, you can optionally create separate and distinct databases for different staging areas as in [Figure 2](#).

In non topology database configuration, you cannot create separate databases for different staging areas and by default, all staging areas will be created under the user database.

Figure 2: Database Topology



The different staging databases includes the following:

- **Input Staging:** optionally create a separate database for holding all the input staging data. The input staging area is the data acquisition area where the data is loaded from source systems.
- **Netchange Staging:** optionally create a separate database for holding all the netchange staging data. The Net change area holds the difference between the existing data and the data present in the input staging area.
- **Master and Error Staging:** optionally create a separate database for master and error staging data. The Master tables in master staging area receive data from the input staging tables and this data is then synchronized with the data in the output staging tables.
- **Version Staging:** optionally create a separate database for version tables.
- **Output Staging:** optionally create a separate database for output tables. The output tables contain the synchronized and cleansed data.
- **Publication Staging:** it is mandatory to have a separate database for the publication in both the database configurations (topology and non topology). The publication tables contain the data to be published to downstream applications.
- **Sandbox Staging:** it is mandatory to have a separate database sandbox in both the database configurations (topology and non topology). Sandbox staging tables enables you to create a copy of the existing source tables on which Cleansing and Standardization Rules can be applied and used by other Reference Data Management (RDM) processes.

Create Publication and Sandbox Databases

Create Publication and Sandbox databases using the below SQLs.

- **CREATE DATABASE <MDM_PUB> as PERM=100000000;**
<MDM_PUB> is publication staging database.
- **CREATE DATABASE <MDM_CS> as PERM=100000000;**
<MDM_CS> is sandbox staging database.

Grant Database User Rights on Publication and Sandbox Databases

Provide grant access on Publication database and Sandbox databases as below:

- GRANT CREATE TABLE, CREATE VIEW, DELETE, DROP TABLE, DROP VIEW, INSERT, SELECT, UPDATE ON <MDM_PUB> TO <MDM_USER>;
- GRANT CREATE TABLE, CREATE VIEW, DELETE, DROP TABLE, DROP VIEW, INSERT, SELECT, UPDATE ON <MDM_CS> to <MDM_USER>;



For any view with table referencing to a different database (X_DB), the SANDBOX should have SELECT, UPDATE, INSERT, DELETE WITH GRANT OPTION access on X_DB.

Create Staging Databases for MDM Topology

Create staging databases using the below SQLs

- CREATE DATABASE <MDM_MST> as PERM=100000000;
<MDM_MST> is master staging database.
- CREATE DATABASE <MDM_IN> as PERM=100000000;
<MDM_IN> is input staging database.
- CREATE DATABASE <MDM_OUT> as PERM=100000000;
<MDM_OUT> is output staging database.
- CREATE DATABASE <MDM_VER> as PERM=100000000;
<MDM_VER> is version staging database.
- CREATE DATABASE <MDM_NC> as PERM=100000000;
<MDM_NC> is netchange staging database.

Grant Database User Rights on Staging Databases

Provide grant access on staging databases as below:

- GRANT CREATE MACRO, CREATE TABLE, CREATE VIEW, DELETE, DROP MACRO, DROP TABLE, DROP VIEW, INSERT, RETRIEVE, SELECT, UPDATE, EXECUTE PROCEDURE, ALTER PROCEDURE, CREATE PROCEDURE, DROP PROCEDURE, EXECUTE ON <MDM_MST> TO <MDM_User>;
- GRANT CREATE MACRO, CREATE TABLE, CREATE VIEW, DELETE, DROP MACRO, DROP TABLE, DROP VIEW, INSERT, RETRIEVE, SELECT, UPDATE, EXECUTE PROCEDURE, ALTER PROCEDURE, CREATE PROCEDURE, DROP PROCEDURE, EXECUTE ON <MDM_IN> TO <MDM_User>;
- GRANT CREATE MACRO, CREATE TABLE, CREATE VIEW, DELETE, DROP MACRO, DROP TABLE, DROP VIEW, INSERT, RETRIEVE, SELECT, UPDATE, EXECUTE PROCEDURE, ALTER PROCEDURE, CREATE PROCEDURE, DROP PROCEDURE, EXECUTE ON <MDM_OUT> TO <MDM_User>;
- GRANT CREATE MACRO, CREATE TABLE, CREATE VIEW, DELETE, DROP MACRO, DROP TABLE, DROP VIEW, INSERT, RETRIEVE, SELECT, UPDATE, EXECUTE PROCEDURE, ALTER PROCEDURE, CREATE PROCEDURE, DROP PROCEDURE, EXECUTE ON <MDM_NC> TO <MDM_User>;

- ```
PROCEDURE, ALTER PROCEDURE, CREATE PROCEDURE, DROP PROCEDURE,
EXECUTE ON <MDM_VER> TO <MDM_User>;
```
- **GRANT CREATE MACRO, CREATE TABLE, CREATE VIEW, DELETE, DROP MACRO, DROP TABLE, DROP VIEW, INSERT, RETRIEVE, SELECT, UPDATE, EXECUTE PROCEDURE, ALTER PROCEDURE, CREATE PROCEDURE, DROP PROCEDURE, EXECUTE ON <MDM\_NC> TO <MDM\_User>;**



You can use below access rights for schema generation or upgrade on MDM. After MDM schema generation, you can revoke the rights.

```
REVOKE CREATE MACRO, DROP MACRO, CREATE PROCEDURE, DROP PROCEDURE ON
<Topology databases> TO <MDM_User>;
```

## Configuring User for Default Database Setup

Default Database set up is provided for Customers who want to install MDM on Teradata user with 0 perm space.

- Create Teradata user (no perm space) with a default database on which user want to install MDM.

Example:

```
CREATE USER <MDM_USER> AS PASSWORD=temppwd, PERM=209715200,
SPOOL=52428800, TEMPORARY=52428800, DEFAULT DATABASE =< MDM _DB>;
```

Access Rights Required for Default Database

- Created user should have all the access rights as in the [Section : “Grant Database User Rights.”](#)
- In addition to above access rights, the created user should have the below access rights:

If MDM\_USER is User and MDM\_DB is Default DB

```
GRANT CREATE TABLE, CREATE VIEW, DELETE, DROP TABLE, DROP VIEW,
INSERT, SELECT, UPDATE, CREATE EXTERNAL PROCEDURE, EXECUTE
PROCEDURE ON <MDM_DB> TO <MDM_USER>;
```

```
GRANT CREATE OWNER PROCEDURE ON <MDM_DB > TO <MDM_USER> ;
```

If Topology setup is used, all access privilege listed in [Section : “Topology Database.”](#) should be provided to MDM\_USER as well as to MDM\_DB.

When access provided to MDM\_DB, mention WITH GRANT OPTION;

Example: If MDM\_MST is the master database.

- **GRANT CREATE TABLE, CREATE VIEW, DELETE, DROP TABLE, DROP VIEW, INSERT, SELECT, UPDATE ON <MDM\_MST> TO <MDM\_USER>;**
- **GRANT CREATE TABLE, CREATE VIEW, DELETE, DROP TABLE, DROP VIEW, INSERT, SELECT, UPDATE ON < MDM\_MST > TO < MDM\_DB > WITH GRANT OPTION;Installing Out of the Box Match Attribute**

## Installing Out of the Box Match Attribute Functions

Out of the box match attribute functions can be installed via script provided with MDM 4.3 installer.

**Note:** For Matching to work on 4.3, all the OOTB MDM UDFs must be reinstalled using the new package available at <MDM\_Install\_Directory>\batch\TMDM\_udfs\TMDM\_udfs.zip

The old UDFs can still be retained and can be used in custom code. Optionally you can drop all Pre 4.3 UDFs, for details see section Section : “Uninstall UDFs.”

To execute UDFs that are required for Matching, the following grant access must be provided:

```
GRANT CREATE FUNCTION, EXECUTE FUNCTION, DROP FUNCTION ON SYSLIB
TO <MDM_USER>;
```

Install UDFs on Windows

You must provide below grants to the mdm user:

Grant execute procedure on SQLJ.install\_JAR to <MDM\_User>;

Grant execute procedure on SQLJ.ALTER\_JAVA\_PATH to <MDM\_User>;

Grant UDTTYPE on SYSUDTLIB to <MDM\_User>;

Perform the following functions to install user defined matching attribute functions on Windows server:

Copy the zip file (TMDM\_udfs.zip) from <MDM\_Install\_Directort>/batch/TMDM\_udfs to C:/ folder.

**Note:** Below steps are defined on assumption that Zip folder is extracted to C:/ .If zip is not extracted to C:/, you may have to update the new file location in the below files before proceeding to next step:

mdmcreate\_judf.btq

mdmcomp\_scriptudf.btq

The above files are available at: C:\TMDM\_udfs\bteq

Replace the text "C:\TMDM\_udfs" with new file location

Example: If new file location of the extracted file is D:\TDUDF\SQL\TMDM\_udfs, then replace "C:\TMDM\_udfs" with "D:\TDUDF\SQL\TMDM\_udfs" without quotes in the above files.

Open LOGIN\_INFO.sql and enter the Teradata login details as below:

If Database IP is 153.65.183.79

User name is SYSDBA

PASSWORD is SYSDBA1234, then enter as below

```
.logon 153.65.183.79/SYSDBA,SYSDBA1234;
```

**Note:** You should have CREATE, DROP and Execute function rights on SYSLIB and SYSUDTLIB database . Execute Install.bat from C:\TMDM\_udfs\setup

Install UDFs on Linux

Perform the following changes in BTEQ scripts while installing Teradata UDFs in Linux server.

1. Unzip TMDM\_udfs.zip file under "/root" directory and run the "./install.sh" file from "setup" directory only.

For example: <LINUX SERVER>TMDM\_udfs/setup #./install.sh

2. Open mdmcreate\_sqludf.btq script

Replace ".run file C:\TMDM\_udfs\LOGIN\_INFO.sql" with ".run file ../LOGIN\_INFO.sql;"

3. Open mdmcreate\_udt.btq script

Replace ".run file C:\TMDM\_udfs\LOGIN\_INFO.sql" with ".run file ../LOGIN\_INFO.sql;"

4. Open mdmcomp\_scriptudf.btq script,

a. Replace ".run file C:\TMDM\_udfs\LOGIN\_INFO.sql" with ".run file ../LOGIN\_INFO.sql"

b. In function UF\_GETNAME, replace EXTERNAL NAME path,  
EXTERNAL NAME 'CS!UF\_GETNAME!../lib/getname.c';

c. In function UF\_EDITDISTANCE\_U, replace EXTERNAL NAME path,  
EXTERNAL NAME 'CS!strUF\_EDITDISTANCE\_U!../lib/  
editdistance\_u.c!F!UF\_EDITDISTANCE\_U';

d. In function UF\_EDITDISTANCE\_U, replace EXTERNAL NAME path,  
EXTERNAL NAME 'CS!strUF\_EDITDISTANCE\_U\_OPT!../lib/  
editdistance\_u\_opt.c!F!UF\_EDITDISTANCE\_U\_OPT';

e. In function UF\_JAROC, replace EXTERNAL NAME path,  
EXTERNAL NAME 'CS!UF\_jaroc!../lib/jaro.c!F!UF\_JAROC';

5. Open mdmcreate\_judf.btq script,

Replace ".run file C:\TMDM\_udfs\LOGIN\_INFO.sql" with ".run file ../LOGIN\_INFO.sql"

Note: perform the following changes in CALL SQLJ.INSTALL\_JAR while creating



below UDFs:

- CALL SQLJ.INSTALL\_JAR('cj!../lib/commons-codec-1.10.jar', 'commons-codec-1.10', 0);
- CALL SQLJ.INSTALL\_JAR('cj!../lib/MetaphoneMDMJarFile.jar', 'MetaphoneMDMJarId', 0);
- CALL SQLJ.INSTALL\_JAR('cj!../lib/DoubleMetaphoneHandlerJarFile.jar', 'DoubleMetaphoneHandlerJarId', 0);

6. Open mdmudf\_Dropall.btg scripts. Replace ".run file C:\TMDM\_udfs\LOGIN\_INFO.sql" with ".run file ../

LOGIN\_INFO.sql;"

7. Open mdmudf\_DropPre42\_UDFs.btg script,

Replace ".run file C:\TMDM\_udfs\LOGIN\_INFO.sql" with ".run file ../

LOGIN\_INFO.sql;"

Testing

1. Once Installation is complete, make sure all the functions (UF\_SNDX\_B, UF\_SDX, UF\_EXDA, UF\_FRSTCHAR, UF\_FRST3CHAR, UF\_FRST2CHAR, UF\_EXMO, UF\_EXYR, UF\_GETNAME, UF\_DTEDIF1, UF\_DTEDIF2, UF\_EDITDISTANCE\_U, UF\_NGRAMM\_U, UF\_STRINGCOMP, UF\_STRSIM4, UF\_STRSIM3, UF\_STRSIM2, UF\_STRSIM) gets created in SYSLIB database.

2. Testing: Run Below SQLs from BTEQ or SQLA to make sure all functions are created properly. The below function execution should not fail.

```
SelectUF_SNDX_B('ASDF1');
SelectUF_SDX('ASDF12','ASKSD');
SelectUF_EXDA(1999-01-01);
SelectUF_FRSTCHAR('ASDF1');
SelectUF_FRST3CHAR('ASDF1');
SelectUF_FRST2CHAR('ASDF1');
SelectUF_EXMO(1999-01-01);
SelectUF_EXYR(1999-01-01);
SelectUF_GETNAME('ASDF1');
SelectUF_DTEDIF1(1999-01-01,1999-01-02);
SelectUF_DTEDIF2(1999-01-01,1999-01-02);
SelectUF_EDITDISTANCE_U('ASDF12','ASKSD');
SelectUF_NGRAMM_U('ASDF12','ASKSD',1);
```

```
SelectUF_STRINGCOMP('ASDF12','ASKSD');
SelectUF_STRSIM4('ASDF12','ASKSD');
SelectUF_STRSIM3('ASDF12','ASKSD');
SelectUF_STRSIM2('ASDF12','ASKSD');
SelectUF_STRSIM('ASDF12','ASKSD');
```

Note: The Seed data for all the above said Functions will get populated into the SYS\_FM\_FUNCTION table as part of Installation.

#### Uninstall UDFs

Perform the below steps to uninstall UDFs:

1. Copy the zip file (TMDM\_udfs.zip) from <MDM\_Install\_Directort>/batch/TMDM\_udfs to C:/ folder.

Note: Below steps are defined on assumption that Zip folder is extracted to C:/ .If zip is not extracted to C:/, you may have to update the new file location in the below files before proceeding to next step:

mdmcreate\_judf.btq

mdmcomp\_scriptudf.btq

The above files are available at: C:\TMDM\_udfs\bteq Replace the text "C:\TMDM\_udfs" with new file location

Example: If new file location of the extracted file is D:\TDUDF\SQL\TMDM\_udfs, then replace "C:\TMDM\_udfs" with "D:\TDUDF\SQL\TMDM\_udfs" without quotes in the above files.

2. Open LOGIN\_INFO.sql and enter the Teradata login details as below:

If Database IP is 153.65.183.79

User name is SYSDBA

PASSWORD is SYSDBA1234, then enter as below

.logon 153.65.183.79/SYSDBA,SYSDBA1234;

**Note:** You should have CREATE, DROP and Execute function rights on SYSLIB and SYSUDTLIB database.

3. To remove, execute dropall.bat from C:\TMDM\_udfs\setup. Refer log in the console for any failure.

4. To remove all pre 4.2 UDFs, execute dropPre42UDFs.bat from C:\TMDM\_udfs\setup. Refer log in the console for any failure.

## CHAPTER 2 MDM Installation

---

### What's In This Chapter

This chapter provides guidelines for installing MDM.

Topics include:

- [Packaging Overview](#)
- [Prerequisites for MDM Installation](#)
- [Installing MDM](#)
- [Starting the Script Based Installation Process](#)
- [Additional Details](#)

## Packaging Overview

The Teradata MDM Platform 4.9 or above package contains a CD with MDM installer.

## Prerequisites for MDM Installation

[“Installation and Configuration of Nginx”](#)

[“Installation of Nginx in Linux”](#)

[“Configure HashiCorp Vault with MDM Installer”](#)(optional)

[“Installing MDM”](#)

### Installation and Configuration of Nginx

Before installing MDM follow the below steps to install and configure Nginx :

- 1 Go to [nginx:download](#).
- 2 Download and unzip nginx.zip to C drive.
- 3 Update the config file at <NGINX\_HOME>/conf/nginx.conf. Refer to the sample config below to populate your configuration for NGINX set-up.

```
#user nobody;
worker_processes 1;
#error_log logs/error.log;
#error_log logs/error.log notice;
#error_log logs/error.log info;
```

```
#pid logs/nginx.pid;
events {
 worker_connections 1024;
}
http {
 include mime.types;
 default_type application/octet-stream;

 #log_format main '$remote_addr - $remote_user [$time_local]
"$request" '
 # '$status $body_bytes_sent "$http_referer" '
 # '"$http_user_agent" "$http_x_forwarded_for"';

 #access_log logs/access.log main;
 sendfile on;
 #tcp_nopush on;
 #keepalive_timeout 0;
 keepalive_timeout 65;
 #gzip on;
 server {
 listen 80;
 server_name <Host Name>;
 proxy_buffer_size 256k;
 proxy_buffers 4 256k;
 proxy_busy_buffers_size 256k;
 proxy_cache off;
 proxy_redirect off;
 client_header_buffer_size 64k;
 large_client_header_buffers 4 64k;
 proxy_set_header Host $host;
 proxy_set_header X-Real-IP $remote_addr;
 proxy_set_header X-Forwarded-For $remote_addr;
 proxy_set_header X-Forwarded-Host $host;
 proxy_set_header X-Forwarded-Server $host;
 proxy_cookie_path ~*^/.*/;
 proxy_read_timeout 600s;

 #charset koi8-r;
 #access_log logs/host.access.log main;
 location / {
 root html;
 index login.html index.htm;
 }

 location /mdm {
 proxy_pass http://hostname:8080/mdm;
 }

 location /analyst {
 proxy_pass http://hostname:8080/mdm;
 }
 location /dashboard {
 proxy_pass http://hostname:8080/mdm;
 }

 location /login {
 proxy_pass http://hostname:8080/mdm/login;
```

```

 }

 location /authentication {
 proxy_pass http://hostname:8080/mdm;
 }

 location /customer-360 {
 proxy_pass http://hostname:8115/customer-360;
 }

 location /connected-identity {
 proxy_pass http://hostname:8116/connected-identity;
 }

 location ~ "^/connected-identity/assets/help/([A-Za-z0-9-]*)/
 (.\.\. (? :md)) $" {
 proxy_pass http://hostname:8116/connected-identity/
 assets/help/$1/$2;
 add_header content-type "text/markdown";
 }

 location /metadata {
 proxy_pass http://hostname:8103/metadata;
 }

 location ~ "^/mdm/covalent/([A-Za-z0-9-]*)/help/ (.\.\. (? :md)) $"
 { proxy_pass http://localhost:8080/mdm/covalent/$1/
 help/$2;
 add_header content-type "text/markdown"; }
 #error_page 404 /404.html;
}

```



The default port of Tomcat server is 8080 and the default port for shared services are 8103, 8116, and 8115 respectively. For more information on changing the port, see [Appendix D: “Configure Nginx Port”](#).

#### 4 Start nginx using below command.

```
start nginx
```



- Command to stop Nginx: `nginx -s quit`
  - Make sure port 80 is free and not used (command to check this: `netstat -ano`).
- Note:** To run Nginx on different port, change the port number at listen 80 in nginx config file.

- Sometime 80 is used by ISS service to uninstall iis, follow below steps.
  - Go to *Control Panel > Programs and Features*.
  - Click *Turn Windows features on or off*.
  - Scroll down to Internet Information Services.
  - Click on the square next to *Internet Information Services* so it becomes empty.
  - Click OK and reboot if required.



To install nginx as a Windows service, check and see [Appendix E: “Install Nginx as Windows Service.”](#)

## Installation of Nginx in Linux

You must follow below steps to install Nginx in Linux:

- 1 To install prerequisites, follow below command:  
`zypper install curl ca-certificates gpg2`
- 2 To set up the zypper repository for nginx packages, run the following command:  
`zypper addrepo --gpgcheck --type yum --refresh --check 'http://nginx.org/packages/sles/$releasever_major' nginx-stable`
- 3 To import an official nginx signing key so that zypper/rpm can verify the packages authenticity, use following command:  
`curl -o /tmp/nginx_signing.key https://nginx.org/keys/nginx_signing.key`
- 4 To verify that the downloaded file contains the key, follow below command:  
`gpg --with-fingerprint /tmp/nginx_signing.key`
- 5 To import the key to the rpm database, follow below command:  
`rpmkeys --import /tmp/nginx_signing.key`
- 6 To install nginx, run the following command:  
`zypper install nginx`

## Configure Nginx Port

Follow below steps to make changes to the Nginx port and listen at port 9090:

- 1 Navigate to the nginx installation folder and open nginx.conf under conf folder to update the port.
- 2 Append **:9090** to the following lines:  
`proxy_set_header Host $host:9090;`  
`proxy_set_header X-Real-IP $remote_addr:9090;`  
`proxy_set_header X-Forwarded-For $remote_addr:9090;`

- ```
proxy_set_header X-Forwarded-Host $host:9090;
proxy_set_header X-Forwarded-Server $host:9090;
```
- 3 Restart nginx servers
 - 4 Add the port to the url in oauth2.properties file.
 - 5 In application.properties file, add the port to server.hostname for shared service as mentioned below:
`server.hostname=${tdaa.proxy.hostname:localhost}:9090`
 - 6 Change the port for all the script.js files in covalent folder from 8080 to 9090.
 - 7 Add the port to oauth.token.endpoint in td-workflow-connected-identity.properties file under Installer\cfg folder.
For Collapsed setup, add the port to same file in Installer\web\mdmclient\WEB-INF\bcm\cfg also.
 - 8 Run mkcoloc jar.
 - 9 Restart the servers.

Configure HashiCorp Vault with MDM Installer

HashiCorp Vault is used for management of Database secrets for MDM. MDM 4.9 release uses Vault (which is optional) for management of secrets of the services. The Installer script allows you to configure the vault using either Token based or Certificate based Vault integration with MDM.

Before configuring the HashiCorp vault, ensure that you:

- Download Hashicorp Vault from <https://www.vaultproject.io/downloads> and follow the steps mentioned in the site to install the vault for your Environment.
- Set environment path to execute (test by `vault -h` in windows gitbash / Linux Terminal)
- You must have access and folder permission before proceeding. The script requires admin/super user access to generate Certificates, ensure that you login as Administrator/ root user when executing the script. Choose Run as Administrator while starting Git Bash.
- Update <MDM_Install_Dir>/MDM4.9.0.0/install/mdm.properties file with `SECRET_MANAGEMENT_MODE=VAULT`.
- Keep a note of the Unseal Key and the root token when you install the vault for the first time. You need this information for the subsequent startup of vault in future if it goes down for some reason.

Token Based Vault Integration

Token based configuration of HashiCorp vault uses token to authenticate the MDM.



For token based installation, you must create **Data** folder at location `VAULT_HOME`.

Before configuring Token based vault, ensure that you:

- Set VAULT_ADDR - Required for Token based. You must set VAULT_ADDR='http://<IP_ADDR>:8200' for Token Based.
- Create data, certificates, conf, tls, policies folders manually for Token based.
- Delete content of data, certificates, conf, tls, policies folders from Vault before any Fresh Installation (If not using Existing configuration).

Follow below steps to configure Token based vault:

Note: All the “steps to configure Token based vault” has to updated before running ./installMDM.sh

- 1 Update /MDM4.9.0.0/shared_services/configuration/templates/VaultToken.hcl with system IP/HOSTNAME.
- 2 Open gitbash/linux terminal and execute the command: *cd <MDM INSTALL PATH>/MDM4.9.0.0/shared_services/configuration*
- 3 `vault server -config=./templates/VaultToken.hcl`
- 4 Provide IP address to the vault: `export VAULT_ADDR='http://<IP_ADDR>:8200'`
- 5 `vault operator init -key-shares=1 -key-threshold=1`
- 6 `vault operator unseal <with Unseal Key 1>`
- 7 Update the following properties in vault.properties file: VAULT_AUTH=TOKEN & VAULT_ADDR & ROOT_TOKEN in ./vault.properties file.
- 8 Run ./installMDM.sh from /MDM4.9.0.0/install folder which will internally call vaultSetup.sh.

If you restart the system or vault, you must unseal the vault to continue using MDM with Vault configuration.

For Certificate Based Vault Setup, vaultSetup.sh script handles the Vault configuration, however, you may need to run them in case the server restarts for some reason.

- 1 Run `vault server -config=<VAULT_PATH>/conf/SharedServices.hcl`
- 2 `export VAULT_ADDR='http://<IP_ADDR>:8200'`



Check Vault Server is up and running by accessing `http://<IP_ADDR>:8200/ui/vault`

- 3 Run `vault operator init -key-shares=1 -key-threshold=1`
- 4 Run `vault operator unseal <with Unseal Key 1>`

If you restart the system or vault, you must unseal the vault to continue using MDM with Vault configuration.

For Certificate Based Vault Setup, vaultSetup.sh script handles the Vault configuration, however, you may need to run them in case the server restarts for some reason.

- 1 Run vault server -config=<VAULT_PATH>/conf/SharedServices.hcl
- 2 export VAULT_ADDR='http://<IP_ADDR>:8200'



Check Vault Server is up and running by accessing http://<IP_ADDR>:8200/ui/vault

- 3 Run vault operator init -key-shares=1 -key-threshold=1
- 4 Run vault operator unseal <with Unseal Key 1>

Certificate Based Vault Integration

Certificate Based configuration of HashiCorp Vault is one of the most trusted implementation. MDM recommends the use of Certificate based Vault Integration for Production.

- Ensure that Vault is properly installed and you can run vault command from any location in Git Bash/Linux Terminal to start the vault.



Ensure that Vault is not running already. The script vaultSetup.sh assumes Vault is already stopped.

- Set Environment path for vault_home and vault_addr. For certificate based configuration, set VAULT_ADDR='https://<IP_ADDR>:8200'
- When you run vaultSetup.sh script deletes and recreates data, certificates, conf, tls, policies folders. Take backup, if required.

Follow below steps to configure Certificate based vault:

- 1 Update <MDM INSTALL PATH>/MDM4.9.0.0/shared_services/configuration/vault.properties with below properties (In windows provide path in Unix format).

Rename intermediate certificate and key as ca-int.crt.pem & ca-int.key.pem and copy to CERT_PATH. Script deletes vault/data folders, and creates. Take backup if required.

```
CA_ROOT=GENERATE
CERT_PATH="/D/vault/certificates"
KEYSTORE_PASSWORD="changeit"
VAULT_PATH=/D/etc/vault
```



If you already have CA root & intermediate certificates then update
CA_ROOT=USE_EXISTING in vault.properties for using the same to configure Vault.
CERT_PATH should be pointing to the location of ca-root.cnf, ca-int.cnf files.

```
CA_ROOT=USE_EXISTING
CERT_PATH=<PATH>
KEYSTORE_PASSWORD=<Password>
VAULT_PATH=/D/etc/vault
```

- 2 Update below properties in ca-root.cnf, ca-int.cnf for new self signed and connected-identity.cnf, metadata.cnf, customer-360.cnf & vault.cnf for shared services SSL certificate generation in <MDM INSTALL PATH>/MDM4.9.0.0/shared_services/configuration/templates.

Update countryName, stateOrProvinceName, localityName, organizationName, organizationalUnitName, emailAddress, commonName, DNS.1 and IP.1 as per requirement in the templates files.



Take backup of <JAVA_HOME>/lib/security folder before proceeding the Installation to ensure the files do not get corrupted. If files are corrupted, you can face anchor error during Vault Setup. Vault may show handsake error. You may need to reinstall JDK in this case.

- 3 Open gitbash / linux terminal & execute below command:
cd <MDM INSTALL PATH>/MDM4.9.0.0/shared_services/configuration
- 4 Run ./installMDM.sh from /MDM4.9.0.0/install folder which will internally handle vaultSetup.sh
vaultSetup.sh:
 - Creates Self Signed Certificates for ca-root and ca-int using information given in ca-root.cnf and ca-int.cnf files.
 - Generates SSL Certificates for Services using connected-identity.cnf, metadata.cnf, customer-360.cnf & vault.cnf files.
 - Generates bundle certificate with all service certificates and Imports certificates to JAVA lib/security using keytool.



Once installation completes, take backup and remove vault_config_XXXXXXX.log file, as the file contains sensitive information.

If you restart your system or vault, you must unseal Vault to continue using MDM with Vault Configuration. The script vaultSetup.sh handles configuration, however, you can run the steps in case the server restart for some reason.

- 1 Run `vault server -config=<VAULT_PATH>/conf/SharedServices.hcl`
- 2 `export VAULT_ADDR='https://<IP_ADDR>:8200'`



Ensure that Vault Server is up and running by accessing `https://<IP_ADDR>:8200/ui/vault`.

- 3 Run `vault operator init -key-shares=1 -key-threshold=1`
- 4 Run `vault operator unseal <with Unseal Key 1>`

To Customize the Security ciphers for Vault, you can update "tls_cipher_suites" parameter in SharedServices.hcl file generated under the conf folder (creates when executing vaultSetup.sh script) and regenerate the certificates accordingly.

Vault: Frequently Asked Questions

- 1 What is the sequence in which we need to start the servers?

Ans: You can start the servers in following sequence:

- a Vault
- b MDM Server and App Server (if Colocated Mode, else start App Server)
- c Shared Services
- d nginx/httpd



You can start nginx/httpd servers before MDM as well.

- 2 How to stop vault to proceed with new MDM installation?

Ans: You can terminate the existing Vault process to stop Vault. Execute below command to find and terminate the vault process:

```
ps -elf | grep vault  
kill -9 <pid>
```

- 3 Are there any changes required to the Vault Setup if reinstalling MDM?

Ans: You can do the following changes when reinstalling MDM:

- Use `CA_ROOT=USE_EXISTING` in the `vault.properties` file to use the existing CA root & intermediate certificates.
- All Shared Services related certificates will regenerate.
- Stop the Vault before executing `vaultSetup.sh`

4 Do we need to regenerate certificates or reuse the existing certificates?

Ans: Regenerate certificates or reuse existing certificates is optional. You can choose `USE_EXISTING` or `GENERATE` based on the requirement.

5 What to do if vault is down for some reason?

Ans: You can run below commands for bringing up the vault again:

- `vault server -config=<PATH TO server-config.hcl>`
`server-config.hcl` will be available under `/etc/vault`
- `vault operator unseal <UNSEAL_KEY>`
`UNSEAL_KEY` has to be saved during initial configuration.

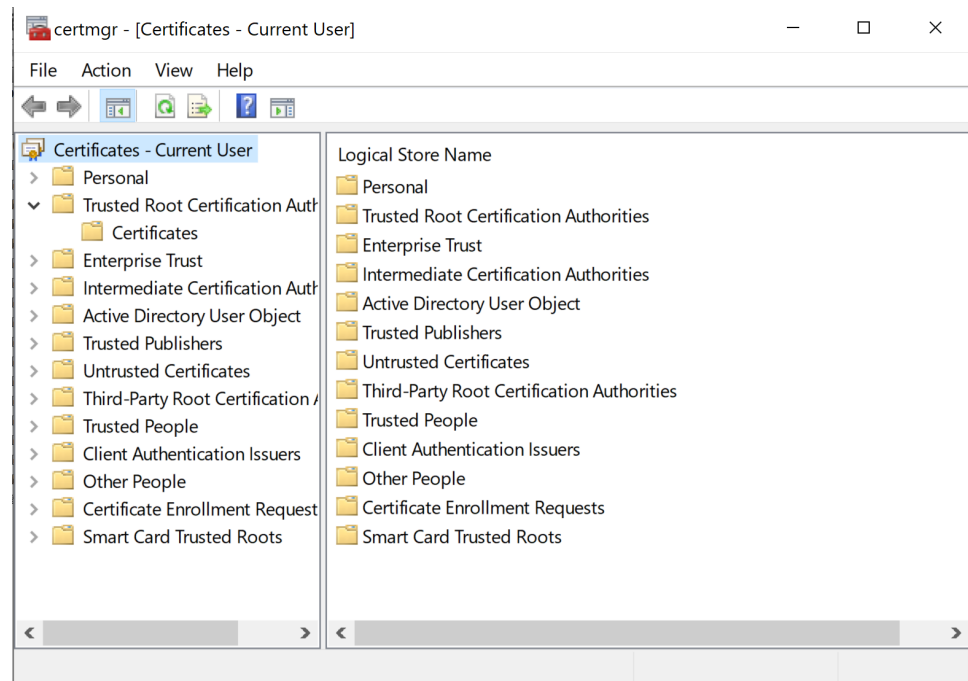
6 What to do if getting the below error:

Error unsealing: Put "https://localhost:8200/v1/sys/unseal": x509: certificate signed by unknown authority. Vault Login Status: Error authenticating: a token must be passed to auth

Ans: Add the ca-root-certificate in the trust certificates:

To add the root certificate, follow below procedure:

- 1 On Run prompt, type `certmgr.msc`.
The `certmgr` window opens.
- 2 Expand **Trusted Root Certification Authorities** and right click **Certificates > All Tasks > Import**.



- 3 Import your root certificate using **Import Certificate Wizard**.
- 4 Choose certificate store as “Trusted Root Certification Authorities”
- 5 In vault.properties file, update CA_ROOT property as USE_EXISTING .
- 6 Execute vaultSetup.sh script.

Installing MDM

This section gives you step-by-step instructions for installing MDM from the Teradata Master Data Management (MDM) 4.9 CD. Prior to installing MDM on your computer, make sure that you comply with the minimum system requirements as detailed in [Chapter 1: “MDM System Requirements”](#).



- 1 **Set the “Java” VM to be used by the installer:** The installer is a Java based installer, hence requires a JDK to run the installation. The installer looks for JDK versions in JAVA_HOME variables and PATH variables. Refer to system requirements in [Chapter 1: “MDM System Requirements”](#) for the exact version of JDK to be used.
- 2 For successful execution of MDM installation on windows environment GitBash should be installed in the machine Refer to system requirements in [Chapter 1: “MDM System Requirements”](#) for the exact version of Gitbash to be used.
- 3 For successful execution of MDM installation scripts in Linux environment, bash (Bourne Again Shell) should exist at “!/bin/sh”. If does not exists, create slink with the same path.
Note: Ensure that the Bash version is higher than 4.



Ensure that you select JDK path not the JRE path. JDK is required to execute jar command during installation.

Script Based Installation

The Script Based installation enables an installer to run without any user interaction using a properties file (usually available at <MDM_Install_Directory>\install\mdm.properties).

Script Based installation is fully supported on Windows and in all UNIX platforms.

Note: Ensure that you install Git Bash before you proceed with the installation on Windows environment.

Script Based Installation Process

Script Based installation process has two main steps:

- 1 Create a properties file with settings for properties as in section [“Creating Script Based Installation Property File for MDM”](#)
- 2 Start the script based installation process and use the values specified in the property file. For the detailed procedure, see [Starting the Script Based Installation Process](#).

Creating Script Based Installation Property File for MDM

You can now install the MDM using script. To install MDM using script based installer, the installation program uses a property file to determine the installation option. You must create a property file for installation before running the installation program.

Follow [Table 5](#) to modify the values for parameters in mdm.properties file available at the location <MDM_Install_Directory>/install/.

Table 5: MDM Script Based Installation Property Values

Variable	Description	Example
* Indicates required field.		
* JAVA_HOME	Provide Java home path	JAVA_HOME=/C/Program Files/Java/jdk_version
*PORT	Provide MDM Server Port details (free/open in firewall & usable)	PORT=14444
*HOST_NAME	Provide Canonical Host Name	Hostname.domain.com
* MODE_OF_DEPLOYMENT	Set the Value to COLLAPSED/COLOCATED	MODE_OF_DEPLOYMENT=COLLAPSED
LDAP_CUSTOMIZED	Provide Database Authorization details for LDAP. Set the Value to Yes/No	LDAP_CUSTOMIZED=YES
DB_CUSTOM_PARAMETER	Set the database custom parameter value to Yes/No if LDAP Authentication is set as Yes	DB_CUSTOM_PARAMETER=LOGMECH=LDAP
* DB_HOST	Set database hostname/IP address details	Teradatadbhost
* DB_USER	Set database user name	User
* DB_PASSWORD	Set database password details	****
MASTER_STG_DB	Provide name of the Master staging database	MASTER_STG_DB=Master Staging Database name
INPUT_STG_DB	Provide name of the Input staging database	INPUT_STG_DB=Input Staging Database name
OUTPUT_STG_DB	Provide name of the Output staging database	OUTPUT_STG_DB=Output Staging Database name
VERSION_STG_DB	Provide name of the Version staging database	VERSION_STG_DB=Version Staging Database name
NETCHANGE_STG_DB	Provide name of the Net Changing staging database	NETCHANGE_STG_DB=Net Change Staging Database name
PUBLISHING_SERVICE_STG_DB=dbuser_pub	Provide name of the Publishing Service Staging Database	PUBLISHING_SERVICE_STG_DB=dbuser_pub

Table 5: MDM Script Based Installation Property Values

Variable	Description	Example
PUBLISHING_AUDIT_STG_DB	Provide name of the Publishing Audit Staging Database	PUBLISHING_AUDIT_STG_DB=dbuser_pub
SANDBOX_STG_DB	Provide name of the Sandbox Staging Database	SANDBOX_STG_DB=dbuser_cs
GENERATE_SCHEMA	Provide the value as Yes/No	GENERATE_SCHEMA=YES
CHARACTER_SET	Set the character set as Latin or Unicode	CHARACTER_SET=UNICODE
APP_SERVER	Provide details for Application Server. Set the value to WEBSPHERE/ TOMCAT	APP_SERVER=TOMCAT
TOMCAT_PATH	Provide path for the Tomcat server if you set APP_SERVER as TOMCAT	TOMCAT_PATH="/C/Apache Tomcat 9.0.35"
TOMCAT_CONF_HOST	Provide configuration host name for the Tomcat server if you set APP_SERVER as TOMCAT	TOMCAT_CONF_HOST="/C/Apache Tomcat 9.0.35/conf/Catalina/localhost"
SMTP_HOST (optional)	Provide Email configurations	SMTP_HOST=smtptserver.com
EMAIL_ID (optional)	Provide Email configurations	EMAIL_ID= admin@teradata.com
SECRET_MANAGEMENT_MODE	Set the value as VAULT to handle Passwords in secure & configure Vault after this installation. Otherwise set as DEV_MODE.	SECRET_MANAGEMENT_MODE=DEV_MODE
ABORT_INSTALLATION_AGAINST_WARNINGS	Set the value as Yes/No for Installation Warning Configuration details	ABORT_INSTALLATION_AGAINST_WARNINGS=NO



Setting the Character set as Unicode updates all the Character set of string data type columns to Unicode in the OOTB master and related staging tables.

Sample mdm script based installer properties file is provided below for reference:


```
# If there are any space in the Installation Paths of JDK or App Servers
then update those paths within quotes.
# All Path information should be mentioned in linux format

# Java Home Location details (Java version specified should be 11 or
above)
# Eg. JAVA_HOME=/C/Program Files/Java/jdk_version
JAVA_HOME=/C/platform_dep/zulu11.52.13

# MDM Server Port details (free/open in firewall & usable)
# Eg. PORT=14444
PORT=14444

# Host Name (Canonical Host Name Eg.Hostname.domain.com)
HOST_NAME=<Host Name>

#Set Proxy Host Name if it exist
PROXY_HOST_NAME=<Proxy Host Name>

# Deployment Mode details
# Set the Value to COLLAPSED/COLOCATED. Eg. For Collapsed Mode
deployment, SET MODE_OF_DEPLOYMENT=COLLAPSED
MODE_OF_DEPLOYMENT=COLOCATED

# Database Authorization details
# Set the Value to YES/NO. Eg. if Authentication is required,
LDAP_CUSTOMIZED=YES.
LDAP_CUSTOMIZED=NO
# Set DB_CUSTOM_PARAMETER value, if LDAP_CUSTOMIZED value is set to YES.
Eg. For LDAP Authentication set DB_CUSTOM_PARAMETER=LOGMECH=LDAP
DB_CUSTOM_PARAMETER=

# Teradata Database details
# DB_HOST=Database Hostname/IP Address
# DB_USER=Database Username
# DB_PASSWORD=Database Password
DB_HOST=<Hostname>
DB_USER=<DB Username>
DB_PASSWORD=<DB Password>

# Topology Staging Database details
# If not using Topology, Database Username should be mentioned for
MASTER_STG_DB, INPUT_STG_DB, NETCHANGE_STG_DB, OUTPUT_STG_DB and
VERSION_STG_DB
# MASTER_STG_DB=Master Staging Database name
# INPUT_STG_DB=Input Staging Database name
# OUTPUT_STG_DB=Output Staging Database name
# VERSION_STG_DB=Version Staging Database name
# NETCHANGE_STG_DB=Net Change Staging Database name
# PUBLISHING_SERVICE_STG_DB=Publishing Service Staging Database name
```

```
# PUBLISHING_AUDIT_STG_DB=Publishing Audit Staging Database name
# SANDBOX_STG_DB=Sandbox Staging Database name
MASTER_STG_DB=<DB_USR>_<Table_Name>
INPUT_STG_DB=
OUTPUT_STG_DB=
VERSION_STG_DB=
NETCHANGE_STG_DB=
PUBLISHING_SERVICE_STG_DB=dbuser_pub
PUBLISHING_AUDIT_STG_DB=dbuser_pub
SANDBOX_STG_DB=dbuser_cs

# Generate Database Schema details
# Set the value to YES/NO. Eg. For Generating the Database Schema during
the installation set GENERATE_SCHEMA=YES.
GENERATE_SCHEMA=YES

#Set the value as Unicode if you need UNICODE character set, otherwise
set as LATIN.
CHARACTER_SET=UNICODE

# Application Server details
# Set value to WEBSPPHERE/ TOMCAT. Eg. To Use Weblogic Server, Set
APP_SERVER=TOMCAT

# Tomcat Configuration details
# Set the below properties if APP_SERVER=TOMCAT
# Eg. TOMCAT_PATH="/C/Apache Tomcat 9.0", TOMCAT_CONF_HOST="/C/Apache
Tomcat 9.0/conf/Catalina/localhost")
TOMCAT_PATH=/C/apache-tomcat-9.0.52
TOMCAT_CONF_HOST=/C/apache-tomcat-9.0.52/conf/Catalina/localhost
TOMCAT_PORT=<Tomcat Port Number>

# Email configuration details
# SMTP_HOST=<SMTP Hostname/IP Address>
# EMAIL_ID=<Administrator Email Id>
SMTP_HOST=relay1.td.teradata.com
EMAIL_ID=admin@teradata.com

# Protect Mode Configuration details
# Set the value as VAULT to handle Passwords in secure & configure Vault
after this installation. Otherwise set as DEV_MODE.
SECRET_MANAGEMENT_MODE=VAULT

# Installation Warning configuration details
# Set the below property to "YES", if Installation should fail for
Warnings during Installation.
ABORT_INSTALLATION_AGAINST_WARNINGS=NO
```



Ensure that the path mentioned in mdm.properties file is in Linux format.



After successful installation of MDM, you must remove all the password entries.

LDAP Configuration

After installing MDM, you must configure the setting for LDAP. Below are the settings for LDAP setup post the installation:

- Modify the below mentioned properties in <MDM_Installation>/web/mdmclient/WEB-INF/spring/config/ldap.properties:
 ldap.URL=ldap://serve.domain.com:10xxx/ou=system
 ldap.manager-dn=uid=admin,ou=system
 ldap.manager-password=<pwd>
- Update the web.xml authentication to ldap at <mdm_installation>/web/mdmclient/WEB-INF/web.xml

```
<context-param>
<param-name>contextConfigLocation</param-name>
<param-value>/WEB-INF/spring/ldap-authentication-provider.xml</param-value>
</context-param>
```

Starting the Script Based Installation Process

To start the Script Based installation process on Windows/Linux sy

- 1 .Extract the content of the mdm_indep_<ver.no>.tar.gz file to the preferred installation location.

Windows:

From Windows Git Bash/Linux terminal, run

```
tar -zxvf mdm_indep_<ver_no.>.tar.gz
```

- 2 .Update the file (Vault.properties) in <MDM_Install_Directory>/shared_services/configuration/templatesRun ./install.sh from <MDM_Install_Directory>/install with SECRET_MANAGEMENT_MODE=VAULT

- 3.Update the installation properties in the mdm.properties file available at <MDM_Install_Directory>\install

- 4.Update credentials.properties and dbadd.txt at <MDM_Install_Directory>/cfg and run UpgradeVault.bat/sh

- 5.Run the installation using installMDM.sh file available at <MDM_Install_Directory>\install

Windows:

From Git Bash Command/Linux terminal, run

```
sh installMDM.sh
```



- For MDM installation using Weblogic application servers, the installer edits the config.xml file present in the specified directory. If the specification is incorrect, then the edit will not occur and the client will not start. Ensure that you enter the following just above the </ Domain> tag in the config.xml file.

```
<app-deployment>
<name>mdmclient</name>
<target>AdminServer</target>
<module-type>war</module-type>
<source-path>C:\Teradata\MDM\web\mdmclient</source-path>
<security-dd-model>DDOnly</security-dd-model>
</app-deployment>
<admin-server-name>AdminServer</admin-server-name>
```

- Warning on Geospatial (SYSUDTLIB) usage gets displayed during MDM Installation, if the MDM database user does not have UDT usage access right on SYSUDTLIB database.



For Topology Installation, the physical databases must already exist in your Teradata system and have the appropriate permissions granted. For grant access, refer to [Chapter 1: “MDM System Requirements.”](#)

The predefined staging areas are used as repositories as below:

- Master—repository for storing MDM Master staging tables.
- Input—repository for storing MDM Input staging tables.
- Output—repository for storing MDM Output staging tables.
- Version—repository for storing MDM Version staging tables.
- Netchange—repository for storing MDM Netchange staging tables.
- Publishing Service—repository for storing publishing target tables.
- Publishing Audit—repository for storing publishing audit history tables.
- Sandbox—repository for storing Source tables and Match process results tables.



For Publication and Sandbox, MDM installation will progress only if you provide a valid database name for Sandbox. MDM staging databases cannot be used as Sandbox.

The MDM user should have DDL and DML rights on Sandbox.



Preparing database for use by MDM is an optional installation operation that is performed by the installer.

You must prepare the database for MDM Server to start. If you do not select prepare the database during the installation process and run ISG after installation, add user Id, password and JDBC URL to dbadd.txt available at: <MDM_Install_Directory>\bin and run the script call_GenDB.bat/call_GenDB.sh. The logs will be available at gendb.log in the <MDM_Install_Directory>/logs folder.



During the Service Setup process, you may get warnings as below under Validating Model set section (output of Model Validator process). These warnings can be ignored and schema generation process can continue to perform with these warnings.

- QueryExecutionLog's result is a RESERVED word
- QueryExecutionLog's result is a RESERVED word
- QueryParam's parameter is a RESERVED word
- QueryParam's sequence is a RESERVED word
- QueryParam's parameter is a RESERVED word
- QueryParam's sequence is a RESERVED word
- RelationalObjProperties's Name is a RESERVED word
- RelationalObjProperties's Nullable is a RESERVED word
 - RelationalObjProperties's Name is a RESERVED word
 - RelationalObjProperties's Nullable is a RESERVED word
- SystemProperties's type is a RESERVED word
- SystemProperties's type is a RESERVED word

Installing MDM on Default Database Setup

To install MDM on a database with 0 perm space, perform the following steps:

- 1 For default database setup installation with user with 0 perm space, update the mdm.properties file and staging database with the default user name and default database details.

For details on the database configuration for default user and database, see [Section : “Configuring User for Default Database Setup.”](#)

- Enter the Database User name and database name in mdm.properties file.
- 2 All other remaining steps remains same as normal MDM installation.

Install and Configure Shared Services

To install and configure shared services, follow below steps:

- For Windows, run the below command:

```
<MDM_INSTALL_FOLDER>\shared_services\<Shared_Service_Name>\bin\install_service.bat
```

- For Linux, use below template file to configure as systemd service with the help of your system administrator.

```
<MDM_INSTALL_FOLDER>/shared_services/<Shared_Service>/bin/  
<Shared_Service>.service
```

Vault Setup After MDM Installation

You can setup Hashicorp vault for shared services after installing the MDM if you have not setup during installation.

Prerequisite:

- Download Hashicorp Vault from <https://www.vaultproject.io/downloads> and follow the steps mentioned in the site to install the vault for your Environment.

Configure Vault After Installation

Follow below procedure to configure vault:

- 1 Go to the location <MDM-HOME>\shared_services\configuration and edit the vault.properties file.
- 2 Go to the location <MDM_HOME>\shared_services\configuration\templates and update the properties DNS1 and IP1 in the following files:
 - ca-int
 - ca-root
 - connected-identity
 - customer-360
 - metadata
 - vault
- 3 Execute the vaultSetup.sh file from <MDM_HOME>\shared_services\configuration
- 4 During execution, vault unseal key and token generates in the generated vaultInit file at location <MDM_HOME>\shared_services\configuration



Additional Details

The following section provides additional details on the following:

- [DBC Object Details](#)
- [Fallback on MDM Database Tables](#)
- [Enable Output Staging in MDM](#)

DBC Object Details

The following list provides the MDM equivalent tables of the DBC objects accessed by MDM.

- SYS_DBC_COLUMNS
- SYS_DBC_TABLES
- SYS_DBC_INDICES
- SYS_DBC_ALL_RI_CHILDREN
- SYS_DBC_ERRORMSGs
- SYS_DBC_SHOWCOLCHECKS
- SYS_DBC_DATABASES
- SYS_DBC_ALL_RI_PARENTS

The following list provides the MDM views pointing to DBC objects directly.

Certain MDM features like Schema Generation (SG), Incremental Schema Generation (ISG) and import from relational database directly refer to the DBC object due to its feature requirements.

By default, the SYS_DBC copies are refreshed (delete/insert) during ISG and SG, but can also be refreshed outside of SG/ISG because of the availability of the run time object creation feature in MDM.



Local copy of the following tables are not created, as these tables are accessed occasionally and the data in these tables are very transient.

- SYS_DBC_DBCINFO
- SYS_DBC_TABLESIZE
- SYS_DBC_ALLRIGHTS

- SYS_DBC_TRIGGERS
- SYS_DBC_DISKSPACE
- SYS_DBC_USERS

Perform one of the following steps to refresh the MDM DBC copies:

- Call refreshSysDBC x-rule from
`<MDM_Install_Directory>\cfg\xservice\toolkit\data\refreshSysViews.xml` to
execute REFRESH_SYS_DBC stored procedure.

Fallback on MDM Database Tables

Fallback is one of the unique feature of Teradata Database, which manages copies of data on alternate storage subsystems within a single database. This secondary copy of the data is referred to as the fallback copy. Thus, the Teradata Database can minimize the impact of major failure scenarios. The FALLBACK tables use twice as much disk space as NON-FALLBACK rows. It is recommended to use fallback only on critical tables since it will double your disk usage.

You can enable fallback on MDM database tables using any of the following procedure:

- On the MDM Teradata Administrator, in **Tools** menu, point to **Create** and select **Database**. In the **Create Database/User** window, select the option Fallback. By default, the Fallback option would be selected.
- When MDM schema generation (SG) / Incremental Schema Generation (ISG) is executed with No-Run option, it generates the SQLs for Fallback, but does not execute them. The generated SQLs can then be edited as required and SG/ISG can be executed to enable FallBack.

Enable Output Staging in MDM

By default, output staging services is disabled in MDM. If required, you can enable output staging service by performing the following steps:

- 1 Set SchemaGen = "true" in `<MDM_Install_Directory>/cfg/properties/toolkit-output.xml` file. In case of custom project, the file path is `<MDM_Install_Directory>/custom/<custom_project>/cfg/properties/<custom_project>-output.xml` file.

```
<schemaGenerate Value="true">
  <file Value="../../db/output.sql"/>
  <generateTables Value="true"/>
  <viewSqlDir Value="xservice/toolkit/staging/output/viewsql"/>
  <dbname Value=""/>
  <maxTableNameLength Value="30"/>
  <maxColumnNameLength Value="30"/>
</schemaGenerate>
```


- 2 Add <Service File="toolkit-output.xml"/> in <MDM_Install_Directory>/cfg/startup_config.xml. In case of custom project, add <Service File="<custom_project>-output.xml"/> in <MDM_Install_Directory>/ custom/<custom_project>/cfg/startup_config.xml

```
<PlatformServerConfig>
  <XserverFile File="xserver.xml"/>
  <Services>
    <Service File="platform.xml"/>
    <Service File="mdmservices.xml"/>
    <Service File="toolkit.xml"/>
    <Service File="toolkit-input.xml"/>
    <Service File="toolkit-netchange.xml"/>
    <!--<Service File="toolkit-output.xml"/>-->
    <Service File="toolkit-version.xml"/>
  </Services>
</PlatformServerConfig>
```

- 3 Set Skip="false" in <Service Value="toolkit-output.xml" Skip="true"/> in <MDM_Install_Directory>/cfg/ SchemaGen_Incr_MetaInfo.xml and SchemaGen_MetaInfo.xml files and for custom Project in <MDM_Install_Directory>/ custom/<custom_project>/bin/custom_SchemaGen_Incr_MetaInfo.xml and custom_SchemaGen_MetaInfo.xml files

```
<Services>
  <Service Value="platform.xml" Skip="false"/>
  <Service Value="mdmservices.xml" Skip="false"/>
  <Service Value="toolkit.xml" Skip="false"/>
  <Service Value="toolkit-input.xml" Skip="false"/>
  <Service Value="toolkit-netchange.xml" Skip="false"/>
  <Service Value="toolkit-output.xml" Skip="true"/>
  <Service Value="toolkit-version.xml" Skip="false"/>
</Services>
```

- 4 Set Skip="false" in <File FileName="xservice/toolkit/staticdata/outputStagingServiceStaticData.xml" ServiceName="BCM_MASTER" Skip="true"/> in <MDM_Install_Directory>/cfg/ SchemaGen_Incr_MetaInfo.xml and SchemaGen_MetaInfo.xml files and for custom Project in <MDM_Install_Directory>/ custom/<custom_project>/cfg/custom_SchemaGen_Incr_MetaInfo.xml and custom_SchemaGen_MetaInfo.xml files

```
<ServicesStaticFileData IsIncremental="true" Skip="false">
  <FacetDir Value="..\cfg/xservice/toolkit/mdm_setup/facets"/>
  <PreLoad>
    <File FileName="xservice/mdmservices/modules/common/staticdata/isg/metadataCleanup.xml" ServiceName="MDMServices" />
  </PreLoad>
  <StaticFiles>
    <File FileName="xservice/mdmservices/modules/common/staticdata/isg/tk_incr_userActivities.xml" ServiceName="PLATFORM"/>
    <File FileName="xservice/toolkit/staticdata/outputStagingServiceStaticData.xml" ServiceName="BCM MASTER" Skip="true"/>
    <File FileName="xservice/toolkit/staticdata/PopulateSysTableColMap.xml" ServiceName="BCM_MASTER"/>
    <File FileName="mdm_setup/entity_templates.xml" ServiceName="MDMServices"/>
  </StaticFiles>
</ServicesStaticFileData>
```

- 5 Add toolkit-output.xml and <CustomApp>-output.xml in
<MDM_Install_Directory>\web\WEB-INF\classes\x2_collapsed.properties (Only for
Collapsed Setup)

```
xcore.xservices=
xserver.xml,ui_workflow.xml,bpemeta.xml,toolkit.xml,toolkit-output.xml,toolkit-version.xml,toolkit-input.xml,userSecurity
.xml,messaging.xml,validation.xml,toolkit-netchange.xml,E2E.xml,dataProfiling.xml,dataupload.xml,email.xml,bpeui.xml,time
rsink.xml,admin_ui.xml,hierarchy.xml,approval.xml,scheduler.xml,tas_service.xml
```

- 6 To generate coloc jar, go to MDM_Install_Dir\web\mdmclient\bin and execute:
mkcolocar.bat/sh
To generate customcoloc jar for Custom project, go to
MDM_Install_Dir\web\mdmclient\bin\custom and execute: *mkcustomcolocar.bat/sh*
- 7 Run ISG.



If custom project is available on MDM installation, the above changes must be done at
<MDM_Install_Directory>custom for the respective files.

CHAPTER 3 MDM Database Preparation

What's In This Chapter

This chapter provides a diagrammatical representation of the various options for preparing the database for MDM.

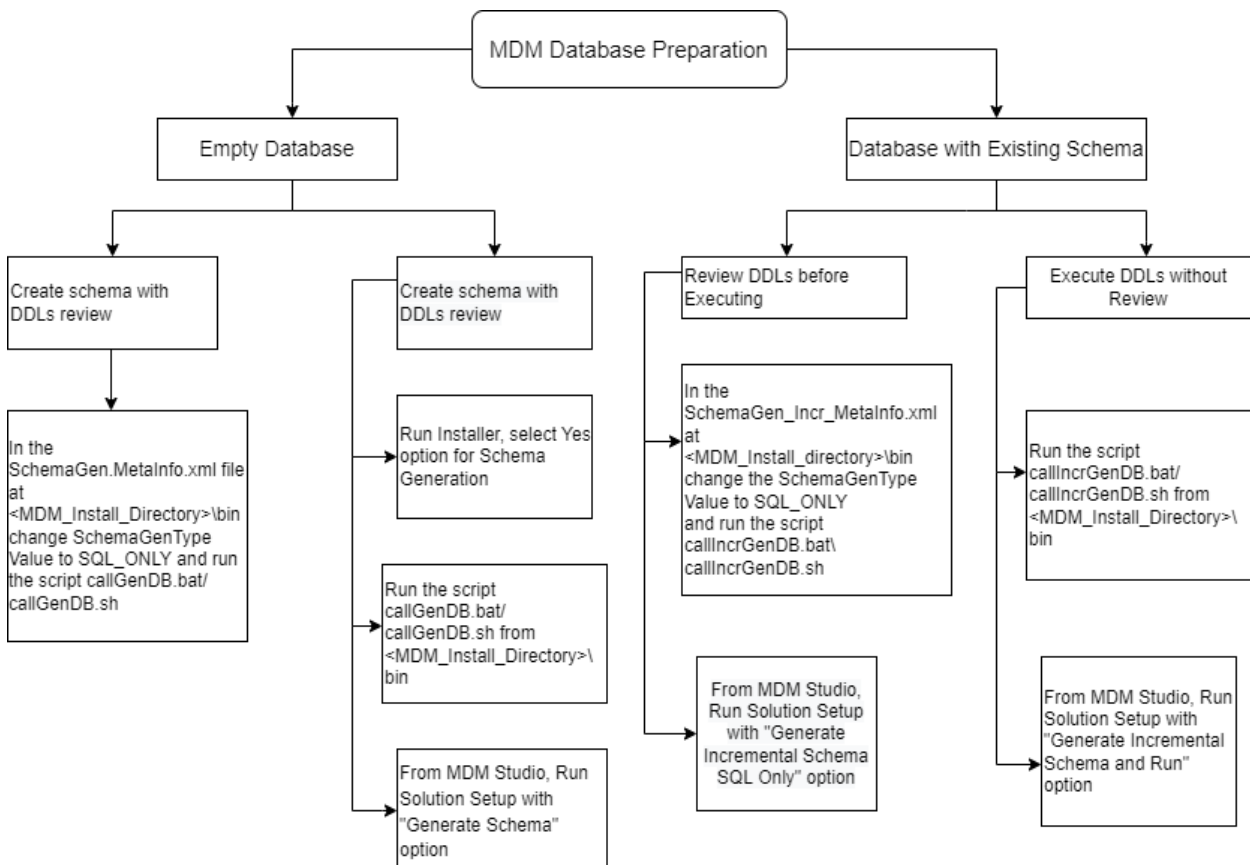
Topics include:

- [Preparing Database for MDM](#)

Preparing Database for MDM

The [Figure 3](#) displays the flow diagram of the various options for preparing the database for MDM.

Figure 3: MDM Database Preparation



For details on solution setup, refer to *Solution Setup section in Master Data Management Studio User Guide*. For details on schema generation scripts, refer to *MDM Schema Generation Process Appendix in Master Data Management Server Guide*.

CHAPTER 4 MDM WebClient Deployment

What's In This Chapter

This chapter provides information about MDM WebClient deployment.

Topics include:

- [WebClient Deployment in WebSphere Liberty Server](#)
- [WebClient Deployment in Oracle WebLogic](#)
- [WebClient Deployment on Tomcat](#)
- [Troubleshooting](#)

WebClient Deployment in WebSphere Liberty Server

The MDM WebClient is deployed on Windows/Linux using IBM WebSphere.

Installing the Application

- 1 [Create a server](#) named MDM Server by using the command: *server create MDMserver*.
- 2 Copy the MDM.war application from <MDM_INSTALL_LOC>/web/mdmclient into the /usr/servers/MDMserver/apps directory.
- 3 For Changing the HTTP port number for Websphere server, In the server.xml file that was created by the server create command, change the default HTTP port of the server MDMserver to <PORT_NO> by replacing the attribute value httpPort="9080"

```
<server description="MDM Application Server">
  <featureManager>
    <feature>webProfile-8.0</feature>
  </featureManager>
  <httpEndpoint id="defaultHttpEndpoint" httpPort="9080" host="*" httpsPort="9443" />
</server>
```

- 4 Configure the MDM application by updating the server.xml. Define the application by using an application element.

```
<server description="MDM Application Server">
  <featureManager>
    <feature>webProfile-8.0</feature>
  </featureManager>
  <httpEndpoint id="defaultHttpEndpoint" httpPort="9080" host="*" httpsPort="9443" />
  <application context-root="mdm" type="war" id="mdm"
    location="mdm.war" name="mdm"/>
</server>
```

- 5 Start the server in foreground by using the command run MDMserver.
- 6 Start the Websphere Application Server and launch UI. For details, refer [Chapter 5: “Launch MDM Server and Application Server.”](#)

WebClient Deployment in Oracle WebLogic

The MDM WebClient is deployed using WebLogic. The installer performs the necessary configuration to WebLogic’s config.xml properties file, such that the MDM Web Client starts (or stops) whenever WebLogic is started (or stopped).

- 1 Before starting Weblogic, add *xercesImpl.jar* and *xml-apls.jar* from
`<MDM_Install_Directory>/lib/3plib to`
`<WEBLOGIC>\user_projects\domains\<DOMAIN_NAME>\lib.`
- 2 If the following entry is not available, enter the details as below in the config.xml file at:
`<WEBLOGIC>\user_projects\domains\<DOMAIN_NAME>\config`

```
<app-deployment>
  <name>mdmclient</name>
  <target>AdminServer</target>
  <module-type>war</module-type>
  <source-path><MDM_Install_Directory>\web\mdmclient</source-path>
  <security-dd-model>DDOnly</security-dd-model>
</app-deployment>
<admin-server-name>AdminServer</admin-server-name>
```
- 3 Start the Weblogic Application Server and launch UI. For details, refer to [Chapter 5: “Launch MDM Server and Application Server.”](#)



For deploying web services, add below entry in config.xml before closing tag for MDM Web services in Weblogic

```
<security-configuration>
...
<enforce-valid-basic-auth-credentials>false</enforce-valid-basic-
auth-credentials>
```

</security-configuration>

WebClient Deployment on Tomcat

The MDM WebClient is deployed using Tomcat.

1 Changes to Tomcat for MDM web client deployment

Below listed configuration changes required for Tomcat application server files for successful MDM web client deployment.

A. Add the following to the xml document available at:

<Install_Tomcat_Location>\conf\Catalina\localhost

```
<Resources cachingAllowed="true" cacheMaxSize="100000"/>
```

For example

```
<Context path="/mdm" debug="0" privileged="true"
docBase="<MDM_Install_Location>/web/mdmclient" crossContext="true">

<Resources cachingAllowed="true" cacheMaxSize="100000"/>

</Context>
```

• Changes to web.xml of (<TOMCAT_INSTALL>\conf)

Update web.xml with below snippet.

```
<servlet>
    <servlet-name>jsp</servlet-name>
    <servlet-class>org.apache.jasper.servlet.JspServlet</
servlet-class>
    <init-param>
        <param-name>fork</param-name>
        <param-value>>false</param-value>
    </init-param>
    <init-param>
        <param-name>compilerSourceVM</param-name>
        <param-value>1.8</param-value>
    </init-param>
    <init-param>
        <param-name>xpoweredBy</param-name>
        <param-value>>false</param-value>
    </init-param>
    <init-param>
        <param-name>tagpoolMaxSize</param-name>
        <param-value>0</param-value>
    </init-param>
    <load-on-startup>3</load-on-startup>
</servlet>
```



In higher versions of Tomcat, servlet with name jsp may already be available. In such case, replace the old servlet with new servlet.

- Copy xercesImpl.jar from <MDM_Install_Directory>\lib\3plib location to <Tomcat_Install>\lib folder.
- 2 Start the Tomcat Application Server and launch UI. For details, refer to [Chapter 5: “Launch MDM Server and Application Server.”](#).

Web Services

Note: The WSDL-based web service has been deprecated and removed from the MDM 4.9.0.0 release due to the following reasons.

AXIS2 based web services are currently experiencing lower popularity compared to Rest-based web services, resulting in reduced development activity for AXIS2.

The AXIS2 web service dependent jars have been identified as vulnerable by our internal security scanning tool, and unfortunately, there are no fixes available at the present time. MDM customers are encouraged to make use of REST-based web services, which are light-weight, efficient, and support a wide range of authentication mechanisms. REST is straightforward and easy to grasp and learn. It operates on the basis of the HTTP(s) protocol and also supports JSON in addition to XML. Furthermore, it is stateless.

Troubleshooting

This section helps you to identify and resolve the problems that you may experience while deploying MDM WebClient.

Q. In Linux, the result is displayed in incorrect format after setting support for UTF-8 and UTF-16.

A. Perform the following steps in Linux:

- In Linux console, enter edit ~/.bashrc
- Check whether the command “export LC_CTYPE=en_US” is written to the file, else enter the command and save it by pressing Esc+:+wq.
- If the command already exists, close the file without saving by pressing Esc+q+!
- Run extracti18n_new.sh from location
<MDM_Install_Directory>\web\mdmclient\bin
- Restart the MDM server and Application server to see the required changes.

Q. To avoid ‘Out of Memory’ errors in WebLogic and Tomcat.

A. For WebLogic, modify JVM startup parameters in startWeblogic.cmd to setup the heap size and other attributes.

For Tomcat, update below details in catalina.bat\sh file:

For Linux: export CATALINA_OPTS="-Xms1024M -Xmx4096M"

For Windows: set CATALINA_OPTS=-Xms1024M -Xmx4096M

Q. To generate MDM war file for tomcat server.

A. Perform the below steps to create the war file:

- Open command prompt and navigate to project's web folder:
`<MDM_Install_Directory>/web/mdmclient`
- Enter the command to create the jar :`jar cvf mdmclient.war *`
 The war file is generated at the specified location.

Q. HeadersTooLargeException is thrown on trying to start tomcat with Shared Services.

- Add below entry in server.xml file of Tomcat to resolve the issue:
`<Connector port="8080" maxHttpHeaderSize="65536" protocol="HTTP/1.1" connectionTimeout="20000"redirectPort="8443" />`

Q. During MDM installation, to change the context rule of the MDM Deployment.

- If you change the base URL to something like `{$hostname}/mdm1`, then you must manually change the context path to the below mentioned location:
 - a Catalina inside Tomcat, replace the mdm file with mdm1 where context path is changed to /mdm1
 - b Nginx config file, replace mdm with mdm1 everywhere
 - c **oauth2.property** file inside `<INSTALL_LOC>\web\mdmclient\WEB-INF\spring\config -->`
 Change value of `oauth2.issuer` property
 - d **td-workflow-connected-identity.property** file inside `<INSTALL_LOC>\web\mdmclient\WEB-INF\bcm\cfg ->`
 Change value of `oauth.token.endpoint` property
 - e Change issuer and `userinfoEndpoint` inside **script.js** file under each folder in `<INSTALL_LOC>\web\mdmclient\covalent`
 - f **application.property** file inside `<INSTALL_LOC>\shared_services\connected-identity\conf -->`
 Change the value of `services.mdm.context.prefix` property
 - g Change issuer and `userinfoEndpoint` inside **script.js** file in front-end folder of each shared service jar

Ensure to run `mkcolocJar` once changes are done

Q. Facing issue during install MDM 4.6 on AWS after genDB process executes and MDM file system installs successfully.

Perform below steps for successful installation:

- 1 SSH to ec2 instance
- 2 Enter cmd "hostname"
- 3 `sudo cat /etc/hosts`
- 4 Add the dns name returned by hostname cmd like the below example
 Before adding: `127.0.0.1 localhost localhost.localdomain`
 After adding: Example: `127.0.0.1 localhost localhost.localdomain ip-10-1-0-112`

Q. The error “Request Entity Too Large (413)” generating while importing large number of records (more than 10k)

- Set client_max_body_size in the http block of Nginx.conf file (by default it takes 1MB), after client_header_buffer_size

```
http {  
    client_max_body_size 10MB;  
}
```

This directive defines the maximum amount of data Nginx accept in an HTTP request.

Q. When databases are restored using BAR utility, reference access issue generates even after providing the access in DB level.

To address this issue ,we may require to provide access for to every table explicitly. Below SQL can be used to generate Grant SQL for all the tables associated to MDM

```
sel 'Grant references on '||tablename||' to ' ||<MDM User>||';'
```

FROM DBC.TABLESV

WHERE DATABASENAME IN (select PHYSICAL_DB from <MDM User>.SYS_DB_MAP) and Tablekind='T';

CHAPTER 5 Launch MDM Server and Application Server

What's In This Chapter

This chapter provides detailed information about starting the MDM server and Web application server.

Topics include:

- [MDM Server & Application Server Startup Process](#)

MDM Server & Application Server Startup Process

You can install MDM either in collapsed mode or co-located mode.



To install the MDM with vault, install the vault first and then begin the MDM installation. For more details on vault configurations, see [Section : “Configure HashiCorp Vault with MDM Installer.”](#).

Follow [Table 6](#) for MDM server and application server startup process in case of collapsed and co-located modes.

Table 6: Collapsed Mode and Co-located Mode—MDM Server Start Up Process

Collapsed Mode	Co-located Modes
The MDM server runs inside the Web application server. MDM server gets started or stopped along with the web application server.	The MDM server and the application server sit separately. Both MDM and web application server needs to be started separately in the order MDM server followed by application server.

Table 6: Collapsed Mode and Co-located Mode—MDM Server Start Up Process

Collapsed Mode	Co-located Modes
<p>Start Application server</p> <p># For WebSphere, execute the below command</p> <pre><WebSphere_Home>/profiles/AppSrv01/bin/ startServer.sh server1</pre> <p>depending on the WebSphere profile used (AppSrv01, AppSrv02 etc).</p> <p># For Tomcat, start server using startup.bat/.sh from <Tomcat_Install_Directory>\bin</p> <p># For Weblogic Server, run startweblogic.cmd/.sh from <Weblogic>\user_projects\domains\<domain_name>\bin</p> <p>Stop Application server</p> <p># For WebSphere, execute the below command</p> <pre><WebSphere_Home>/profiles/AppSrv01/bin/ stopServer.sh server1</pre> <p>depending on the WebSphere profile used (AppSrv01, AppSrv02 etc).</p> <p># For Tomcat, stop server using shutdown.bat/.sh from Tomcat install location.</p> <p># For Weblogic Server, run stopweblogic.cmd/.sh from <Weblogic>\user_projects\domains\<domain_name>\bin Stop App</p>	<p>Start MDM server from <MDM_Install_Directory>\bin\startServer.bat on Windows or <MDM_Install_Directory>\bin\startServer.sh on AIX/Unix</p> <p>Stop MDM server from <MDM_Install_Directory>\bin\stopServer.bat on Windows or <MDM_Install_Directory>\bin\stopServer.sh on AIX/Unix</p> <p>Note: To stop MDM server, the application server should be up.</p>
	Start Application server - same as in collapsed mode.

Start Services

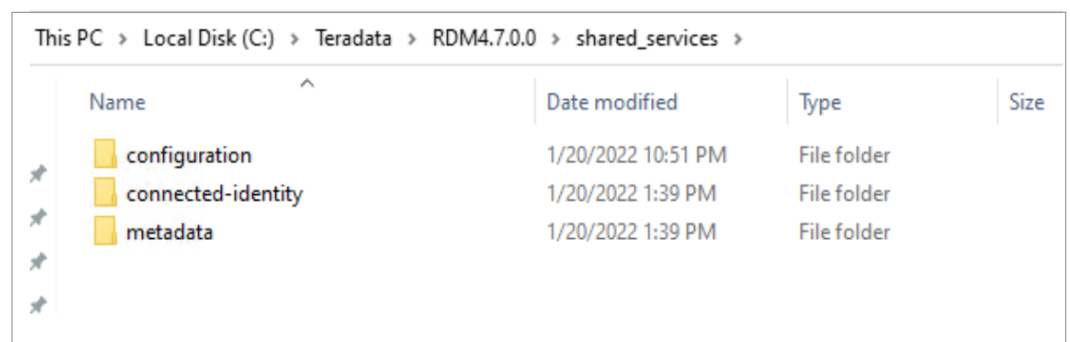
Follow below steps to start Services:

Note: Among shared services, start Metadata service first.

- 1 Start the MDM and Tomcat servers.

For more information on starting MDM server and application, check [“MDM Server & Application Server Startup Process”](#).

- 2 Navigate to <MDM_INSTALL_LOC>/MDM4.9.0.0/shared_services.



- 3 Open <shared_services_name>/bin folder and run start_<shared_services_name>_server.bat file. Shared services start.

PC > Local Disk (C:) > MDM > MDM4.7.0.0 > shared_services > connected-identity > bin

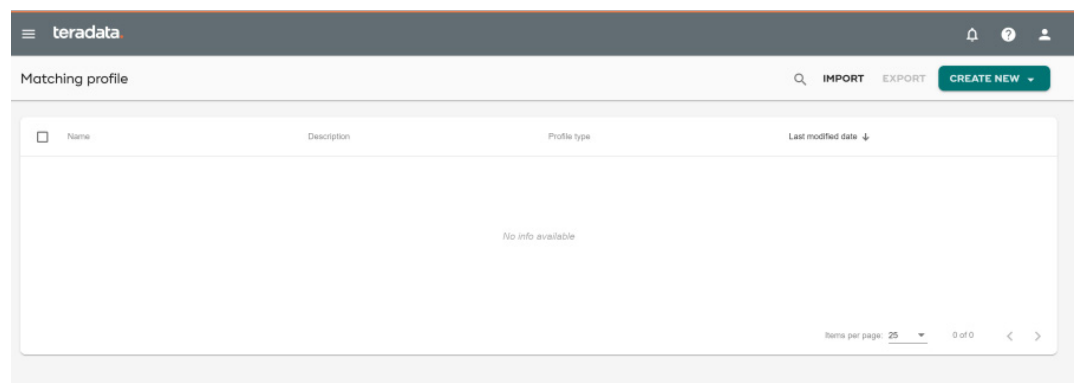
Name	Date modified	Type	Size
connected-identity	02-12-2021 10:04	Application	17,053 KB
connected-identity	02-12-2021 10:04	XML Document	1 KB
install_service	02-12-2021 10:04	Windows Batch File	1 KB
restart_service	02-12-2021 10:04	Windows Batch File	1 KB
service_reresh	02-12-2021 10:04	Windows Batch File	1 KB
service_status	02-12-2021 10:04	Windows Batch File	1 KB
start_connected-identity_server	02-12-2021 10:04	Windows Batch File	1 KB
start_service	02-12-2021 10:04	Windows Batch File	1 KB
stop_service	02-12-2021 10:04	Windows Batch File	1 KB
uninstall_service	02-12-2021 10:04	Windows Batch File	1 KB

Similarly repeat the step 2 and 3 for customer-360 and metadata as well to bring the corresponding services up.

Note: Shared services are prepackaged with Tomcat.

- 4 Log in with the URL <http://localhost/mdm/bcm/framework/login.jsp> (**Note: do not include the port in URL if you have started your nginx server**).

You should be able to see below Matching screen under **Data harmonization** --> **Matching**



- 5 Ensure that shared services are running. For example: check for Matching or Survivorship.

Start Shared Services in Linux

Follow below steps to start shared services in linux:

- 1 Copy the shared services from `<MDM_Install_dir>/shared_services/<service_name>/bin/<service_name>.service` to the location `/etc/systemd/system`



`<MDM_Install_dir>/shared_services/<service_name>/bin/<service_name>.service` creates only in Linux environment through MDM/RDM Installer.

- 2 `chmod 777 /etc/systemd/system/<service_name>.service`
- 3 `systemctl daemon-reload`
- 4 Start the service using command: **`systemctl start <service_name>`** [Ex : connected-identity `systemctl start`]

Start nginx

Start nginx using command: *start nginx*



For more information on starting nginx, see [Installation and Configuration of Nginx](#).

Launch MDM Web UI

Once the MDM server and application server are up, perform the following steps to launch MDM web UI:

- 1 Launch MDM web UI by starting a browser and entering the following URL: *http://<hostname>:<port>/mdm*
Where host is the IP address/host Machine name and port is the port of the application server.
- 2 Login to MDM Web UI with user name and password.
- 3 In both the MDM and Web application log files, ensure that there are no errors.

Log Files

- 1 MDM logs for Colocated Deployment, available at: `<MDM_INSTALL_LOC>\log`
 - 2 MDM logs for Collapsed Deployment, available at: `<App_Server_Home>\log`
 - 3 WebSphere logs available at: `<WebSphere_Home>/profiles/AppSrv01/server/logs`
 - 4 Weblogic logs available at:
`<WEBLOGIC>\user_projects\domains\<DOMAIN_NAME>\server\Adminserver\logs`
 - 5 Tomcat log is available at `<Tomcat_Install_Directory>\bin\logs` file.
-



To get MDM logs in the collapsed mode, you need to create the log folder in the following location:

For Tomcat: `<Tomcat_Install_Directory>`

For Weblogic: `<Weblogic>\user_projects\domains\<domain_name>`

For WebSphere: `<Websphere_home>\usr\servers`

For Connected Identity Services: `<mdm_install>\shared_services\connected-identity\logs`

For Metadata Services: `<mdm_install>\shared_services\metadata\logs`

CHAPTER 6 MDM Deployment Manager

What's In This Chapter

The MDM Deployment Manager provides the ability to create a runtime deployment package from a Studio MDM application project and move/deploy this code from 'development' to 'QA' to 'production' environments.

MDM Deployment Manager addresses the deployment of applications built in Studio to other servers within a customer's Teradata environment. It does not address the installation of MDM Studio, Server or Database, as that is still handled by the MDM installation and schema generation.

Topics include:

- [Deploying Custom Application using the properties file](#)
- [Debugging Deployment Process](#)

Deploying Custom Application using the properties file

The MDM Deployment Manager consists of two parts:

- 1 A function within Studio to custom application/project in Studio for an MDM application and then insert the CLOB into the MDM Deployment database and table(s).
- 2 A script based DeploymentMgr is used for retrieving the application package from the database/filesystem and deploying it to an existing MDM installed location. Script based Deployment Manager is fully supported on Windows and in all UNIX platform.
- 3 A Script Based DM installation enables an installer to run without any user interaction using a properties file (usually available at <MDM_Install_Directory>\deply_custom\dm.properties).

For details on the 1st part, refer to *Master Data Management Studio User Guide*.

Script Based DM Installation Process

Script Based Deployment Manager installation process has two main steps to retrieve an application package from the Deployment database/filesystem and to deploy it to an existing MDM installed location:

- 1 Create a properties file with settings for properties as [Section : “Creating Property File for Deployment Manager.”](#).
- 2 Start the script based installation process and use the values specified in the property file. For the detailed procedure, see [“Starting the Script Based Installation Process for DM”](#)

Creating Property File for Deployment Manager

Deploying the Custom Application using the script based installer, the installation program uses a property file to determine the installation options. Before you run the installation program you must create a property file for the installation as in [Table 7](#)

To create property file for DM installation:

In the dm.properties file, modify the values for the parameters as in [Table 7](#).

Table 7: DM Script Based Installation Property Values

Variable	Description	Example
* Indicates required field.		
* CUSTOM_APP_SOURCE	Set the value to DATABASE/FILE_SYSTEM	CUSTOM_APP_SOURCE=DATABASE
* CUSTOM_APP_DB_HOST	Database Hostname/IP Address, Where the Custom Application to be deployed is stored	CUSTOM_APP_DB_HOST=*****
* CUSTOM_APP_DB_USER	Database Username	CUSTOM_APP_DB_USER=****
*CUSTOM_APP_DB_PASSWORD	Database Password	CUSTOM_APP_DB_PASSWORD=*****
*CUSTOM_APP_DB_NAME	Database Name, Name of the Database where the Custom Application is stored	CUSTOM_APP_DB_NAME=CustomAppName
*PROJ_ID	Project Id of the Custom Application to be deployed, To be used only if CUSTOM_APP_SOURCE=DATABASE	PROJ_ID=1
PATH_OF_CUST_APP_JAR (optional)	Set the below property only if CUSTOM_APP_SOURCE=FILE_SYSTEM, Location of the Custom Application jar file in FILE_SYSTEM.	PATH_OF_CUST_APP_JAR= /C/CustomAppJar/CustomApp_v2.jar
*DEPLOYER_NAME	Used for Auditing purpose, Provide any name of person/process details.	DEPLOYER_NAME=QAUSER

Table 7: DM Script Based Installation Property Values

Variable	Description	Example
* GENERATE_ISG	Generate Incremental Database Schema details Set the value to YES/NO	GENERATE_ISG=YES
* IMPORT_ROLLEDUP_DATA	Import Rolled Up Data details Set the value to YES/NO.	IMPORT_ROLLEDUP_DATA=NO
*CUSTOM_APP_MODE_OF_DEPLOYMENT	Deployment Mode details Set the Value to COLLAPSED/COLOCATED.	CUSTOM_APP_MODE_OF_DEPLOYMENT=COLOCATED

Sample deployment Manager script based installer properties file is provided below for reference:

If there are any space in the Paths used then update those paths within quotes.

All Path information should be mentioned in linux format

Custom Application Source details

Set the value to DATABASE/FILE_SYSTEM. Eg. Set
CUSTOM_APP_SOURCE=DATABASE, if the Custom Application is stored in a Database.
CUSTOM_APP_SOURCE=DATABASE

Teradata Database details to be used for Custom Application Deployment

CUSTOM_APP_DB_HOST=Database Hostname/IP Address, Where the Custom Application to be deployed is stored

CUSTOM_APP_DB_USER=Database Username

CUSTOM_APP_DB_PASSWORD=Database Password

CUSTOM_APP_DB_NAME=Database Name, Name of the Database where the Custom Application is stored

CUSTOM_APP_DB_HOST=*****

CUSTOM_APP_DB_USER=****

CUSTOM_APP_DB_PASSWORD=*****

CUSTOM_APP_DB_NAME=CustomAppName

Custom Application details

```
# Set PROJ_ID=Project Id of the Custom Application to be deployed, To be used only if  
CUSTOM_APP_SOURCE=DATABASE
```

```
PROJ_ID=1
```

```
# Set the below property only if CUSTOM_APP_SOURCE=FILE_SYSTEM, Location of  
the Custom Application jar file in FILE_SYSTEM. Eg. PATH_OF_CUST_APP_JAR=/C/  
CustomAppJar/CustomApp_v2.jar
```

```
PATH_OF_CUST_APP_JAR=
```

```
# Custom Application Deployment details
```

```
# Used for Auditing purpose, Provide any name of person/process details.
```

```
DEPLOYER_NAME=QAUSER
```

```
# Generate Incremental Database Schema details
```

```
# Set the value to YES/NO. Eg. For Generating the Incremental Database Schema during the  
Custom App Deployment set GENERATE_ISG=YES.
```

```
GENERATE_ISG=YES
```

```
# Import Rolled Up Data details
```

```
# Set the value to YES/NO. Eg. Set IMPORT_ROLLEDUP_DATA=YES, If Custom  
Application is packaged with any Custom Data for any of its Tables and if it needs to be  
uploaded into the Target.
```

```
IMPORT_ROLLEDUP_DATA=NO
```

```
# Deployment Mode details
```

```
# Set the Value to COLLAPSED/COLOCATED. Eg. For Collapsed Mode deployment, SET  
MODE_OF_DEPLOYMENT=COLLAPSED
```

```
CUSTOM_APP_MODE_OF_DEPLOYMENT=COLOCATED
```



The DM Install location path should not contain any spaces.

Starting the Script Based Installation Process for DM

To start the Script Based installation process on Windows/Linux system:

- 1 Update the installation properties in the dm.properties file available at
<MDM_Install_Directory>\deploy_custom
- 2 Run the installation using deploy.sh file available at : <MDM_Install_Directory>\
deploy_custom

Windows :

From Git Bash command line

sh deploy.sh

Linux:

From terminal run

sh deploy.sh



- You can have your custom projects under different Databases (can be staging database as well). Deployment Manager refers to this database to fetch archived/stored (using MDM Studio) MDM custom Projects.
- For re-deployment, use the same existing deployment mode. If the custom application is deployed in co-located mode, it can still be re-deployed in collapsed mode. But if the custom application is deployed in collapsed mode, it cannot be redeployed in colocated mode.
- On the production environment, if the incoming web service/WSDL node is used in the workflows, it is the responsibility of the user to manually Activate the wsdl and add an entry in the xserver.xml and xserverweb.xml file under <wsdl-client-config> as shown in the code snippet below:

```
<wsdlLocations>
```

```
<wsdlLocation Value="..\wsdl\Test\GoogleSearchService.wsdl"  
Group="Test" Version="2"/>  
</wsdlLocations>
```

The above changes must be done before you do the custom application deployment using deployment manager, so that when coloc.jar is generated for collapsed mode it will always have the wsdl added/activated.

- If the deployment mode is collapsed then the wsdl files needs to be copied to the Application Server. For Tomcat and Weblogic deployments, copy the whole wsdl folder from the development environment to the production environment under <AppServer_Install_Directory>.

The wsdl folder will be created in <MDM_Install_Directory>/wsdl only if WSDL is available.

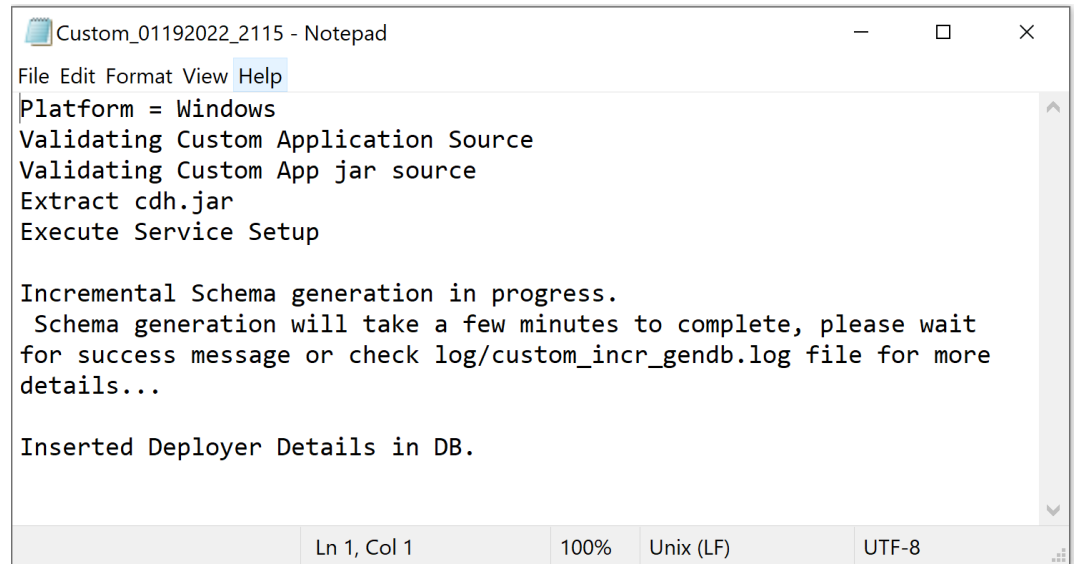
- When scripts are migrated across operating systems (Windows to Linux), use Dos2Unix command to compile the scripts to convert plain text files in DOS to UNIX format.
- Deployment Manager always deploys an application incrementally. The assumption is that base MDM has been installed and the base MDM schema is already present.
- For verification, refer to Teradata MDM Deployment Manager Install.log in the <MDM_Install_Directory>/deploy_custom, once the deployment is complete.

Debugging Deployment Process

While deploying the application, a deployment log file (Custom_<timestamp>.log) file is created in the MDM deploy_custom folder. The deploy log file lists the deployment stages along with the errors and warnings if any as below:

- Displays error if any while changing permissions for custom_project and its sub folder.
- Displays the status of schema generation.
- Displays error if any while copying cfg/properties to /web/mdmclient/WEB-INF/bcm/cfg/properties
- Displays error if any while copying mdm-spec-gen.xml meta-enabled-srv-c-filenames.xml from bin to WEB-INF\bcm\cfg\properties
- Displays error if any while copying cfg/xservice to /web/mdmclient/WEB-INF/bcm/cfg/xservice
- Displays error if any while copying x2.properties as x2_coloc.properties
- Displays error if any while calling mkcolocjar.bat /.sh. prior to execution.
- Displays error if any while calling mkcustomcolocjar.bat /.sh prior to execution.
- Displays the status of inserting deployment details in database.
- Displays the status of MDM Client WAR file creation for Websphere Deployment and any error if encountered.

- Displays installation successful if there is no error as below.



The screenshot shows a Notepad window titled "Custom_01192022_2115 - Notepad". The text inside the window is as follows:

```
File Edit Format View Help
Platform = Windows
Validating Custom Application Source
Validating Custom App jar source
Extract cdh.jar
Execute Service Setup

Incremental Schema generation in progress.
  Schema generation will take a few minutes to complete, please wait
  for success message or check log/custom_incr_gendb.log file for more
  details...

Inserted Deployer Details in DB.
```

The status bar at the bottom of the Notepad window displays: "Ln 1, Col 1", "100%", "Unix (LF)", and "UTF-8".

CHAPTER 7 MDM Upgrade

What's In This Chapter

This chapter provides information on MDM server software and database upgrade process.

Topics include:

- [Introduction](#)
- [Upgrade Process](#)
- [Troubleshooting](#)
- [Backdown Procedure for MDM](#)
- [Sync Up Custom Application](#)

Introduction

The MDM upgrade process is an easy and simple single click process. It allows upgrading MDM from a previous version (supported version) to current version. The upgrade process consists of multiple steps which are executed sequentially based on a configuration file. Once all steps of upgrade are completed successfully, the source version gets upgraded to latest version.

The upgrade process captures the upgrade execution log details in a predefined log file and also in database tables. The upgrade process also supports the restart ability feature. Using the restart ability feature, the upgrade execution process restarts from where the previous upgrade process failed. The upgrade process reads the status of the previously failed/partial-run upgrade from the database table and then restart the upgrade execution from that point.

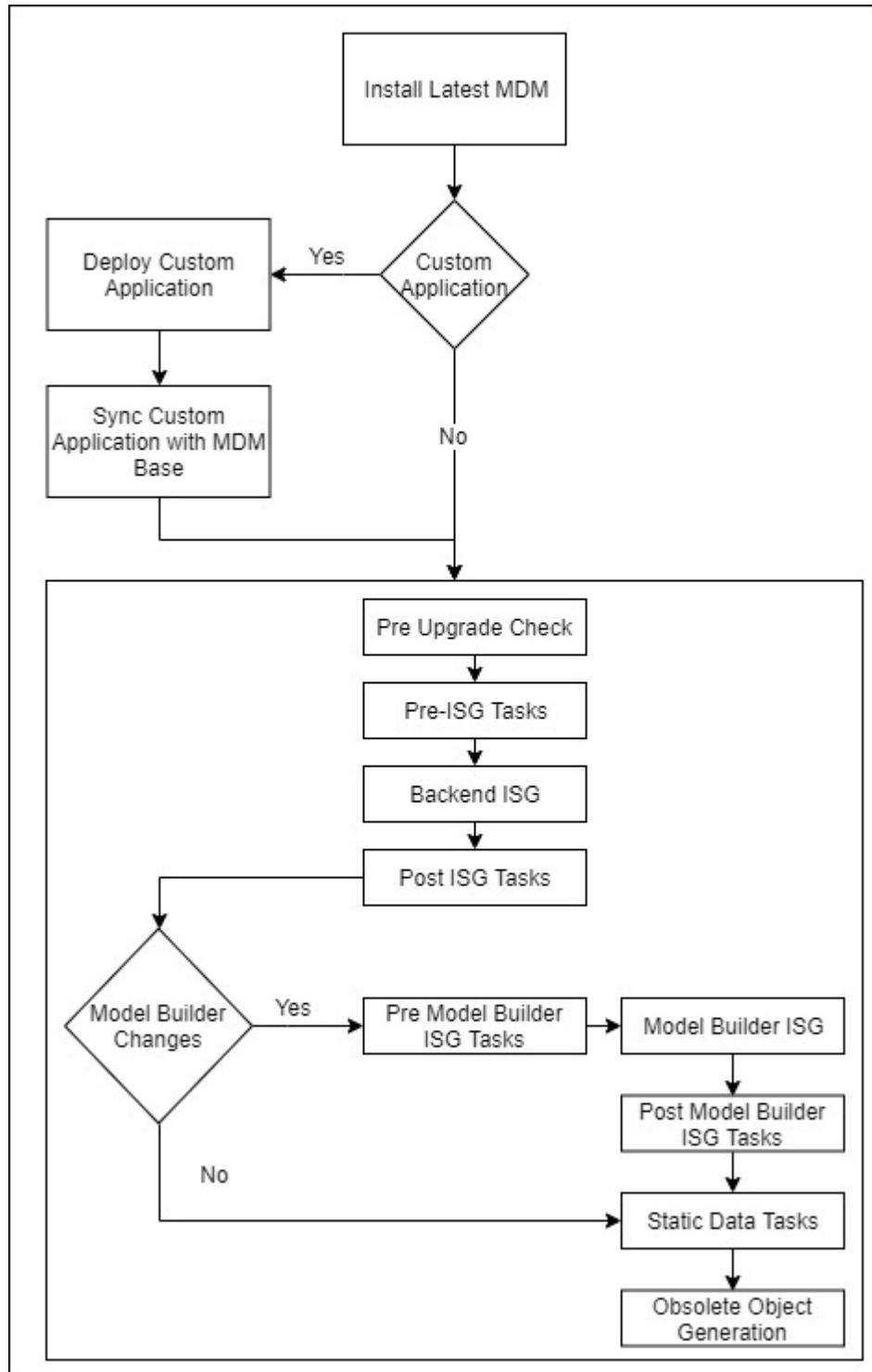
Upgrade Process

Upgrade process consists of several steps which includes pre-upgrade checks, pre-isg (incremental schema generation) steps, isg steps, static data load steps & post-isg steps. All these steps are executed in a pre-defined sequence which is required for successful completion of upgrade. The main program to perform upgrade reads a configuration file which contains the instructions to execute the upgrade. The process to upgrade database of MDM from older supported version to current version is a single click operation. Deployment of custom application & syncing up of related files is pre-requisite for the database upgrade.

The upgrade of MDM to current version(4.9) is supported from 4.6, 4.7.1, and 4.8 releases inclusive of any patch applied.

Figure 4 displays the consistent upgrade process flow diagram.

Figure 4: Upgrade Process Flow Diagram





- Before running the upgrade, check [Section : “Backdown Procedure for MDM,”](#) .
- For database user access rights, check [Chapter 1: “MDM System Requirements,”](#) in Master Data Management Installation Guide.
- For deploying MDM in collapsed mode on Websphere, the separator in x2 properties should be a semicolon (;), not a colon (:).
- In case of temporal tables, configurable UI needs to be regenerated manually after the upgrade to see editable Bi Temporal Timestamp column.
- In case of input staging tables, configurable UI needs to be regenerated to use excel upload functionality after upgrade.

Prerequisite:

- 1 Ensure that there are no pending approvals on business rules.
- 2 Ensure that there are no models in pending approval state or validated state.
- 3 Ensure that RLS and CLS are not enabled on the Code set table.
- 4 Ensure to run ISG for all the migrated profiles/for changes done in models before starting upgrade.

Upgrade Steps

Perform the following steps to upgrade MDM version:

- 1 Install MDM 4.09.01.00 and use database details of already existing installation. While installing MDM, set Schema Generation option as “NO” in the mdm.properties to use the database of the existing installation. For details on MDM Installation, see [Section : “Installing MDM,”](#) in [Chapter 2: “MDM Installation.”](#)



The versions 4.7.0.0 & 4.7.1.0 are same. But we should pick correct installer 4.7.1.0 for 4.7.0.0 and 4.7.1.0 installation purpose. When we are upgrading from 4.7.0.0 to higher version the Source Version should be 4.7.0.0 in the configuration file.

- 2 In case of Custom Application, deploy the custom application using Deployment Manager. Choose Incremental Schema Generation option as “no” and complete the installation. This step is mandatory. For details on Deployment Manager, see [Chapter 6: “MDM Deployment Manager.”](#)
- 3 Make sure that Custom application is in sync with the base installation.
 - For detail instructions on sync up of scripts, service property files and models refer [Sync Up Custom Application](#)

<MDM_INSTALL_HOME>/custom/<CUSTOM_APPLICATION>/cfg/
custom_SchemaGen_Incr_MetaInfo.xml must be in sync with
<MDM_INSTALL_HOME>/cfg/SchemaGen_Incr_MetaInfo.xml from base.

- 4 In case of custom installation copy Upgrade.bat/sh from <MDM_INSTALL_HOME>/bin to <MDM_INSTALL_HOME>/custom/<CUSTOM_APPLICATION>/bin
- 5 Modify Upgrade.bat/.sh to include the source MDM Upgrade Version to be upgraded to latest version as below. Update the value of the property called “MDM_SOURCE_VERSION” with the correct source version of MDM from which upgrade needs to be performed. You need to use the exact source version names as specified in the script. For Custom project, set the value of the property called CUSTOM_PROJECT to “true” as in the below script.
- 6 Execute Upgrade.bat/sh from <MDM_INSTALL_HOME>/bin and in case of Custom installation from <MDM_INSTALL_HOME>/custom/<CUSTOM_APPLICATION>/bin

```
File Edit Format View Help
@echo off
REM Copyright (c) 2006-2021 by Teradata Corporation.
REM All Rights Reserved.
REM Teradata CONFIDENTIAL AND TRADE SECRET

title Teradata Platform Upgrade
echo Initialising Teradata MDM Upgrade
echo %DATE:~4,10% %TIME%
echo Running Teradata MDM Upgrade

call .\platformEnv.bat
set CLASSPATH=.;%PLATFORM_ROOT%/upgrade;%CLASSPATH%;
rem Supported MDM Upgrade Versions: 4.6.0.0 || 4.7.0.0 || 4.8.0.0
rem 4.7 and 4.7.1 versions are the same and have to provide the same source number(i.e, 4.7.0.0) for either of these

set SOURCE_VERSION=
set PLATFORM_PROJECT=MDM
rem Supported CUSTOM_PROJECT values: true || false
set CUSTOM_PROJECT=false
%JAVA_HOME%\bin\java %JAVA_OPTIONS% com.teradata.platform.core.upgrade.UpgradeMerger %SOURCE_VERSION% %PLATFORM_PROJECT%
%CUSTOM_PROJECT%
```

- 7 Logs can be checked in Upgrade.log and UpgradeIsg.log(only for ISG process) from <MDM_INSTALL_HOME>/log and in case of Custom installation from <MDM_INSTALL_HOME>/custom/<CUSTOM_APPLICATION>/log. The Upgrade.log file will contain all steps of upgrade process and UpgradeIsg.log will contain the logs of ISG (Incremental Schema Generation) performed during upgrade process.
- 8 Bring up the dependent services by following the steps mentioned below:

B connected-identity Service:

In order to bring up the server go to the below specified path and run start_connected-identity_server.bat file <MDM_Install_Location_Home>\shared_services\connected-identity\bin\start_connected-identity_server.bat/sh Server comes up once we get message in console like "Started Application launcher in secs.."

C Customer 360(Optional):

In order to bring up the server got to the specified path and run start_customer-360_server.bat/sh

<MDM_Install_Location_Home>\shared_services\customer-360\bin\start_customer-360_server.bat

Server comes up once we get message in console like "Started Application launcher in secs.."

- 9 After successful upgrade, bring up MDM Server from <MDM_INSTALL_HOME>/bin/startServer.bat/.sh only in case of colocated setup. In case of custom installation, to bring up the server run <MDM_INSTALL_HOME>/custom/<CUSTOM_PROJECT>/bin/startServer.bat/.sh
- 10 Bring up Web Server (i.e. Tomcat, Weblogic, Websphere etc.).
- 11 Go to the MDM UI homepage and Navigate to the Cleansing and Standardization profile page then click **Import**.
- 12 On import UI, select the profile and click **Configure** to select master staging of instance for both target database and source database.

Select the CSUPGRADEDUMMYPROFIEL.cleansing profile for importing.

Profile selection path: Go to MDM installed location folder-->upgrade--> <MDM_4.6> --> staticdata\CSUPGRADEDUMMYPROFIEL.cleansing.

- 13 Click **Import** after configuration.
- 14 Execute the following SQLs and SPs under MDM user:

```
UPDATE SV_SURV_MAP_PROFILE
SET CONNECTION_ID =(sel CONNECTION_ID from CI_DEFAULT_CONNECTION
WHERE CONNECTION_NAME='Default CI-MDM Connection');
```

```
UPDATE CL_CAS_PROFILE
SET CONNECTION_ID =(sel CONNECTION_ID from CI_DEFAULT_CONNECTION
WHERE CONNECTION_NAME='Default CI-MDM Connection');
```

```
UPDATE FM_PROFILE
SET CONNECTION_ID =(sel CONNECTION_ID from CI_DEFAULT_CONNECTION
WHERE CONNECTION_NAME='Default CI-MDM Connection');
```

```
call UPD_CI_PKG_UPD ();
call SV_SURV_REFRESH_PROFILES (msg);
```

- 15 Add the request to RunTest.xml file in <MDM_Home>/cfg/xservice/test/requests and execute runTest.bat/sh present in <MDM_Home>/bin.

Execute below request to view pre-upgrade Approval History data in UI. <REQUEST Name="transferClobDataToHistoryTableUpgrade" ServiceName="MDMServices"/>



For upgraded MDM to run successfully Metadata, Connected-identity, MDM and Web servers (all 4 servers) should be up and running.

- 16 After upgrade, restart the MDM Server.

Obsolete Objects:

Once the upgrade process is completed successfully, the following obsolete files will be created at <MDM_Install_Directory> and in case of custom application upgrade, the files will be created at <MDM_Install_Directory> /custom/<CUSTOM_APPLICATION>:

- OBSOLETE_OBJECTS.txt

This file provides a list of Obsolete Database Objects from MDM user database.

- **OBSOLETE_OBJECTS_DROP_SQL.sql**

This file provides the drop sqls to drop the Obsolete Database Objects from MDM user database. MDM Admin can verify the list and ensure no useful tables are used and if any object is required from list, you can delete the object and execute the drop sqls as required.

Upgrade: Restart and Logs

The upgrade process captures status of each steps of each executions performed during upgrade. Any error encountered during upgrade will be logged. Once an error is encountered, the upgrade process will terminate. After rectifying the error, the upgrade can be re-started again. On re-start, upgrade will start from the step where it failed.

The ability to restart upgrade process helps to make the upgrade process fast since you need not start upgrade fresh each time an error is encountered. Also, the detail logs of upgrade steps will help to identify the errors fast.


Upgrade process will be logged in upgradeLog.log file available at:

<MDM_INSTALL_HOME>/Upgrade and in case if the process is started again, the existing upgradeLog.log file will be backed up with timestamp and a new file will be generated. The UpgradeISG.log file is backed-up only if any process within ISG fails. If the ISG is successful, log details are added continually as data is logged. The log file names and locations are user configurable in the configurable file available at:

<MDM_INSTALL_HOME>/upgrade/

MDM_<source_version>\MDM_<source_version>_Config.xml as below.

```
<Upgrade TargetVersion="4.6.0.0" SourceVersion="4.5.0.0">
  <Configuration>
    <System>
      <IsCustomProject Value="false"/>
      <LogFilePath Value="..\log/upgrade.log"/>
      <ISGLogFilePath Value="..\log/upgradeISG.log"/>
      <ObsoleteObjectsListPath Value="..\OBSOLETE_OBJECTS.txt"/>
      <ObsoleteObjectsDropSQLListPath Value="..\OBSOLETE_OBJECTS_DROP_SQL.sql"/>
    </System>
  </Configuration>
```

 **Log name and location are user configurable**

Complete Upgrade process will be logged in multiple tables as in below list:

- SYS_UPGRADE_PROCESS_LOG: Captures overall upgrade process status.
- SYS_UPGRADE_STEP_LOG: Captures upgrade process Steps of each executions.
- SYS_UPGRADE_TASK_LOG: Captures upgrade process detail Tasks of each steps.
- SYS_UPGRADE_PRECHECK_LOG: Capture Pre-Upgrade check entries
- SYS_UPGRADE_POSTCHECK_LOG: Capture Post-Upgrade check entries

The upgrade process can be re-started by executing the Upgarde.bat/.sh script as mentioned above.

Troubleshooting

Solution to common issues related to MDM server post upgrade:

1. For colocated installation, ensure that the MDM server port number is same in all the following files:

- <listenerPort Value="14444"/> in xserver.xml available at <MDM_Install_Directory>\cfg\properties
- <soapAddr Value="http://localhost:7001/"> in xserverweb.xml available at <MDM_Install_Directory>\cfg\properties
- locatorURL=http://localhost:14444/ in xcoreweb.properties available at <MDM_Install_Directory>\web\mdmclient\WEB-INF\classes
- <property name="locatorURL" value="http://localhost:14444/"> in xcore-init.cnd available at <MDM_Install_Directory>\web\mdmclient\WEB-INF\bcm\model

2. For both collapsed and colocated installation, make sure that the following points are satisfied:

- Tomcat port and context name are in sync in xserver.xml and xserverweb.xml for the parameter <restEndPointURL Value="http://localhost:8080/mdm/xcorexml/"> available at <MDM_Install_Directory>\cfg\properties
- <restBase64AuthString Value="YWRtaW46YWRtaW4="> is in sync in xserver.xml and xserverweb.xml available at <MDM_Install_Directory>\cfg\properties. For more details on how to Encrypt the restBase64AuthString parameter value, refer to section “*Poster Call through Rest Web Service*” in *Appendix A in Master Data Management Server Guide.pdf*.
- Encryption can be done with help of online open source tools.

To verify if changes are correct, use restEndPointURL to login and you should see REST API Documentation.

In case of Weblogic server, clear cache from:

<WL_HOME>\user_projects\domains\<my_domain>\servers\AdminServer\tmp

Clean Weblogic/WebSphere/Tomcat application cache and start application server.

[Table 8](#) provides the list of error messages in SYS_UPGRADE_TASK_LOG table.

Table 8: Error Messages in SYS_UPGRADE_TASK_LOG Table

TASK_NAME	STATUS	MESSAGE	REMARKS
PRE UPGRADE CHECK	ERROR	Data present in APPROVED_DATE or APPROVED_BY. These columns will be dropped as part of upgrade. Contact MDM Support before proceeding	Data present in Columns to be dropped.

Table 8: Error Messages in SYS_UPGRADE_TASK_LOG Table

TASK_NAME	STATUS	MESSAGE	REMARKS
PRE UPGRADE CHECK	WARNING	NODE_NAME part of UNIQUE CONSTRAINT.This may lead to inconsistency after upgrade	In Hierarchy Node tables, NODE_NAME is made Unique. In Current database, Node Name is part of Composite Key which is different from expected Index for Templates.
PRE UPGRADE CHECK	ERROR	NODE_NAME IS not Unique.This will lead to upgrade failure	In Hierarchy Node tables, NODE_NAME is made Unique. In Current database, Node Name is not unique. This may lead to inconsistency after upgrade.
PRE UPGRADE CHECK	ERROR	Backup table already exist. Make sure backup is in sync with the latest tables	Make sure Backup exist for corresponding tables to avoid any further issues
PRE UPGRADE CHECK	ERROR	USER MAY NOT HAVE SUFFICIENT RIGHTS OR RIGHTS MAY HAVE BEEN PROVIDED THROUGH A DATABASE ROLE	Make sure all recommended rights are present
PRE UPGRADE CHECK	WARNING	YOU MAY NEED TO PROVIDE SOME ADDITIONAL PERM SPACE TO CONTINUE WITH THE UPGRADE	For performing Upgrade, MDM expects 150% more space in corresponding database. Make sure sufficient space is provided
PRE UPGRADE CHECK	ERROR	THE BELOW MASTER TABLES DO NOT HAVE PRIMARY KEY/PRIMARY INDEX. YOU WILL SEE FAILURE MESSAGE IN ISG	Master tables do not have PK. This may lead to upgrade failure
UPD_BACKUP_MDM_TABLE	WARNING	Nothing Backed up! Either Table got backed up already Or Target Table not present in DB	Make sure Backup exist for corresponding tables to avoid any further issues
'UPD_RESTORE_VAL IDATION'	'MISMATCH-ERROR'	Restored Table Data not matching <Table Name>	Restoration did not work as expected. Make sure Restored table contains correct data
'UPD_RESTORE_VAL IDATION'	ERROR'	MAIN TABLE NOT CREATED <Table Name>	ISG did not run as expected

3. Upgrade Pre-Check to find PK columns not set as required in MB Attribute Details table

Symptom:

If the system shows following error

"preUpgradeCheck: Execute Pre-Check Error: Following Columns which are PK [983#Language_Id] have their required field as false. Pls make required field as true and proceed."

Solution:

You need to set the columns as Required. Use the following script to set required field as true for the PK columns.

```
UPDATE TRGT FROM
MB_ATTRIBUTE_DETAILS TRGT ,
(SELECT objAtt.*
FROM MB_ATTRIBUTE_DETAILS AS objAtt INNER JOIN
(
SELECT RES.OBJECT_ID, RES.OBJECT_NAME ,
ATTR.KEY_NAME, ATTR.ATTRIBUTE_NAME
FROM MB_OBJECT_INDICES_ATTRIBUTES AS ATTR RIGHT JOIN (
SELECT OBJ.OBJECT_NAME , IND.OBJECT_ID , IND.KEY_NAME
FROM MB_OBJECT_DETAILS AS OBJ INNER JOIN MB_OBJECT_INDICES AS IND
ON OBJ.OBJECT_ID = IND.OBJECT_ID
WHERE OBJ.STATUS = 'GENERATED'
AND OBJ.SERVICE_NAME = 'MDM'
AND IND.KIND = 'PRIMARY')
AS RES
ON RES.KEY_NAME = ATTR.KEY_NAME
AND RES.OBJECT_ID = ATTR.OBJECT_ID )
AS res1
ON res1.OBJECT_ID = objAtt.OBJECT_ID
AND res1.ATTRIBUTE_NAME = objAtt.ATTRIBUTE_NAME
WHERE objAtt.required = '0'
) Tab1
SET REQUIRED = '1'
WHERE TRGT.OBJECT_ID=TAB1.OBJECT_ID AND TAB1.ATTRIBUTE_NAME =
TRGT.ATTRIBUTE_NAME;
```

4. Getting an error as backup already exist while running upgrade in pre-upgrade.

Solution:

You can either of the two solutions for avoiding the error:

- 1 Drop below tables using the following command:

```
drop table SYS_UPGRADE_PROCESS_LOG__BKP4800;
drop table SYS_UPGRADE_TASK_LOG__BKP4800;
drop table SYS_UPGRADE_STEP_LOG__BKP4800;
drop table SYS_UPGRADE_PROCESS_LOG;
drop table SYS_UPGRADE_TASK_LOG;
drop table SYS_UPGRADE_STEP_LOG;
```

- 2 Drop the below tables and rename remaining tables with a different suffix or prefix other than _BKP4800

Drop below tables using the following command:

```
drop table SYS_UPGRADE_PROCESS_LOG__BKP4800;
```

drop table SYS_UPGRADE_TASK_LOG__BKP4800;

drop table SYS_UPGRADE_STEP_LOG__BKP4800;

Rename following tables with a different suffix or prefix instead _BKP4800

SYS_UPGRADE_PROCESS_LOG,

SYS_UPGRADE_TASK_LOG,

SYS_UPGRADE_STEP_LOG

Sync Up Custom Application

Following changes need to be done to sync up the custom application with latest base MDM

- 1 The classpath in <CUSTOM_PROJECT>.project file at: <MDM_INSTALL_HOME>/custom/<CUSTOM_PROJECT>/eclipseproject/ must be in sync with the classpath in tk.project or tk_unix.project file and available at: <MDM_INSTALL_HOME>/studio/resources/config/project/
- 2 Script changes to be done in <MDM_INSTALL_HOME>/custom/<Custom-Application>/bin
 - Move all your custom CLASSPATH settings to <MDM_INSTALL_HOME>/custom/<Custom-Application>/bin/customServicesEnv.bat/.sh file (if applicable)
 - Copy and replace platformEnv.bat/.sh, platformServicesEnv.bat/.sh, mdmServicesEnv.bat/.sh file from <MDM_INSTALL_HOME>/bin to <MDM_INSTALL_HOME>/custom/<Custom-Application>/bin



The CLASSPATH and files in the bin folder must be synced with the latest customizations while upgrading from older version to newer version.

- In <MDM_INSTALL_HOME>/custom/<Custom_Application_Name>/bin/platformEnv.bat/.sh file, update PLATFORM_ROOT parameter as PLATFORM_ROOT=../..
 - Update JAVA_HOME in <MDM_INSTALL_HOME>/custom/<Custom_Application_Name>/bin/PlatformEnv.bat/.sh file
- 3 Changes to be done in <MDM_INSTALL_HOME>/custom/<Custom-Application>/cfg/properties

Open custom service files and make sure that all the entries for handler, extensionFile, dataPersistSpecFile and param are in sync with the base service file(s).
 - 4 Changes for <MDM_INSTALL_HOME>\custom\<Custom-Application>\cfg\xservice\<Custom-Application>\model-instance and <MDM_INSTALL_HOME>\custom\<Custom-Application>\models

Note: The OOTB mtts are deprecated and converted to model builder models. The following steps are required if you have any mtts.

- Make sure that no OOTB model is present in <MDM_INSTALL_HOME>/custom/<Custom-Application>/model. In case there is a customization done, make sure all the base OOTB columns of that models are present in custom also.
- In case of addition/removal of .mtt files in <MDM_INSTALL_HOME>/models, make sure that model-instance for same is also added/removed from <MDM_Install_Home>/custom/<Custom-Application>/cfg/xservice/<custom-staging-service>/model-instance
- Copy model, model instance VersionInfo.mtt file source installation(<MDM_INSTALL_HOME>\models\Toolkit\models\Toolkit Models\System\VersionInfo.mtt , <MDM_INSTALL_HOME>\cfg\xservice\toolkit\model-instance\models\Toolkit Models\System\VersionInfo.mtt) file to target installation (<MDM_INSTALL_HOME>/custom/<Custom-Application>/model/<CustomModels>/VersionInfo.mtt, <MDM_INSTALL_HOME>/custom/<Custom-Application>/cfg/XService/<Custom-Application>/model-instance/models/VersionInfo.mtt)
- Copy the cfg.xml file of models (<MDM_INSTALL_HOME>\models\Toolkit\cfg.xml) and model instance (<MDM_INSTALL_HOME>\cfg\xservice\toolkit\model-instance\cfg.xml) from source installation to target installation. This step is needed only when base MDM without custom application is upgraded.
- If any link to OOTB.mtt/.mdt file is made in custom mtt files make sure that all linked base.mtt/.mtd files are also present.
- Run serviceSetup.bat/.sh from <MDM_INSTALL_HOME>/custom/<Custom-Application>/bin/

In collapsed set-up Run mkColoc.bat/.sh from <MDM_INSTALL_HOME>/web/mdmclient/bin/ for base MDM upgrade and <MDM_INSTALL_HOME>/web/mdmclient/bin/custom/<Custom-Application>/mkcustomcolocjar.bat/.sh for Base MDM with Custom Application upgrade

- If all MTTs are converted to Model Builder tables, remove the entry ../../models/Toolkit under XML tag <PackageRoots> from mdm-spec-gen.xml located at (<MDM_INSTALL_HOME>/custom/<Custom-Application>/cfg/



- If publication objects are created on base MDM tables, then while performing upgrade to newer version of MDM, publication objects are required to be recreated to the newer MDM version.
 - In any service, if DEFAULT_MAX_ROWS_FETCH parameter value is less then the total count of MU_LEFT_NAV_URL and MU_LEFT_NAV_STRUCTURE, the value has to be replaced with a higher value then the total count.
-

Backdown Procedure for MDM

This section describes how to restore the MDM database in case of any upgrade failures or otherwise.

Before every major change to the database or as a standard practice of data warehousing maintenance, any Applications database including MDM will generally be backed-up periodically using Teradata B.A.R solutions.

As the solution described below is a Teradata Arcmain solution, so before performing this step, refer the Arcmain manuals for more details.



Use the same MDM database user for Restore/Copy process unlike the standard practice of using ADMIN Id's. It is assumed that, the user id used for Archive/restore has been given the archive and restore access rights.

For detailed information on database user requirements, see [“Chapter 7 MDM Upgrade”](#).

- 1 Using Arcmain, archive all the data tables belonging to the MDM user (including the underlying database's if database topology is used, you could use the "ALL" option, if the database's used in the MDM installation are descendents of the MDM USER). The backup media could be data disks, tapes or any other industry standard solutions.
- 2 Ensure that step 1 is successful by verifying backed up media for its correctness.
- 3 Perform an Arcmain RESTORE for all the archived database's after cleaning up the database's (you could use delete database command in the script as well).

Note that RESTORE is different from COPY. When a database is restored, all the database objects, including triggers and PK/FK links are copied unlike copy, which DOESN'T COPY the Trigger's and PK-FK links. If ARC copy is employed then, you will have to manually copy or write scripts to create triggers and referential integrity constraints.

- 4 Run Revalidate references command to validate the restore/copy, this could be done from the Restore/copy script as well.
- 5 Verify that the restore is successful by comparing the archived and restored database.
- 6 Verify the entries in SYS_DB_MAP are right, if not update appropriately.
- 7 If everything runs correctly till step 6, try to point the MDM application to this restored database.

All the MDM functionality should continue to work as before.

Troubleshooting

Common issues that user may face if MDM database is Archived and Restored/Copy.

- On restored database, Master tables that are temporal get restored without Primary key definition.

Solution: after restore, rename Master tables that are temporal and recreate Master table with original DDLs. Original DDLs can be regenerated from MTTs or can be copied from Archived database.

- Foreign Key links of tables may be lost during copy.

Solution: after restore, you can manually alter tables that had FKs with actual FK definitions. SYS_DBC_INDICES will provide details of FK definitions from original database. Can create some script to generate Alter table definitions from using SYS_DBC_INDICES table.

- Stored procedures will not work as expected or displays errors.

Solution: recompile stored procedures using bin/compile_MDM_SP.bat/sh

- ISG may fail with complaints on No Reference access on Tables.

Solution: run following Grant access command.

```
SELECT 'GRANT ALL ON ' || TRIM(DATABASENAME) || '.' ||  
TRIM(TABLENAME) || ' TO ' || USER || ';' (TITLE '') FROM DBC.TABLES  
WHERE DATABASENAME IN (SELECT DISTINCT(PHYSICAL_DB) FROM  
SYS_DB_MAP) AND TABLEKIND = 'T';
```

Copy the result of above query in SQL Assistant and run.

- Starting Connected Identity shared services is failing if Database UDF jars are already present.

Login to MDM user and use below statements to drop UDFs and Jars already present.

```
DROP FUNCTION UF_ADDRSIM ;  
DROP FUNCTION UF_BIRTHDATE_VALIDATE ;  
DROP FUNCTION UF_DOUBLEMETAPHONE ;  
DROP FUNCTION UF_DOUBLEMETAPHONE_P75 ;  
DROP FUNCTION UF_DROPALPHA ;  
DROP FUNCTION UF_DROPNONALPNUM ;  
DROP FUNCTION UF_DROPNUMBER ;  
DROP FUNCTION UF_DTEDIF1 ;  
DROP FUNCTION UF_DTEDIF2 ;  
DROP FUNCTION UF_EDITDISTANCE_U ;  
DROP FUNCTION UF_EDITDISTANCE_U_OPT ;  
DROP FUNCTION UF_EXDA ;  
DROP FUNCTION UF_EXMO ;  
DROP FUNCTION UF_EXYR ;  
DROP FUNCTION UF_FRST2CHAR ;  
DROP FUNCTION UF_FRST3CHAR ;  
DROP FUNCTION UF_FRSTCHAR ;  
DROP FUNCTION UF_GENDER ;  
DROP FUNCTION UF_GENDER_CODE_VALIDATE ;  
DROP FUNCTION UF_GETNAME ;  
DROP FUNCTION UF_JAROC ;  
DROP FUNCTION UF_JAROIDX ;  
DROP FUNCTION UF_METAPHONE ;  
DROP FUNCTION UF_METAPHONE_P75 ;  
DROP FUNCTION UF_SDX ;  
DROP FUNCTION UF_SNDX ;  
DROP FUNCTION UF_SNDX3 ;  
DROP FUNCTION UF_SQUEEZE ;
```

```
DROP FUNCTION UF_SQUISH ;
DROP FUNCTION UF_STRINGCOMP ;
DROP FUNCTION UF_STRSIM ;
DROP FUNCTION UF_STRSIM2 ;
DROP FUNCTION UF_STRSIM3 ;
DROP FUNCTION UF_STRSIM4 ;
DROP FUNCTION UF_VALIDEMAIL ;
CALL SQLJ.REMOVE_JAR('MetaphoneUDF',0);
CALL SQLJ.REMOVE_JAR('DoubleMetaphoneUDF',0);
CALL SQLJ.REMOVE_JAR('phoneStandardization',0);
```

APPENDIX A **WebClient War Deployment on Weblogic**

What's In This Appendix

This appendix provides information on WebClient war deployment on Weblogic.

Topics include:

- [WebClient War Deployment on Weblogic](#)

WebClient War Deployment on Weblogic

Perform the following steps to deploy webclient as an archived war file:

- 1 In the weblogic.xml at `<MDM_Install_Directory>/web/mdmclient/WEB-INF`, set
`<container-descriptor>`
`<show-archived-real-path-enabled>true</show-archived-real-path-enabled>`
`</container-descriptor>`
- 2 Create a war file of the Web component.
- 3 Deploy the war file on Weblogic.
 - Navigate to Weblogic console in any browser.
For example: `http://localhost:7001/console`
The Weblogic console is displayed as in [Figure 5](#).

Figure 5: Weblogic Console

The screenshot shows the Oracle WebLogic Server Administration Console login interface. At the top, the Oracle logo is followed by 'WebLogic Server®' and 'Administration Console'. Below this is a 'Welcome' header. The main content area contains the text 'Log in to work with the WebLogic Server domain'. There are two input fields: 'Username:' with the value 'weblogic' and 'Password:' with masked characters. A 'Log In' button is positioned to the right of the password field. Below the login fields, a message states: 'It is recommended that users log out when finished with the Administration Console or when visiting an untrusted site.' At the bottom, the version 'WebLogic Server Version: 10.3.0.0' and copyright information are displayed.

ORACLE® WebLogic Server®
Administration Console

Welcome

Log in to work with the WebLogic Server domain

Username: weblogic

Password: ●●●●●●●●

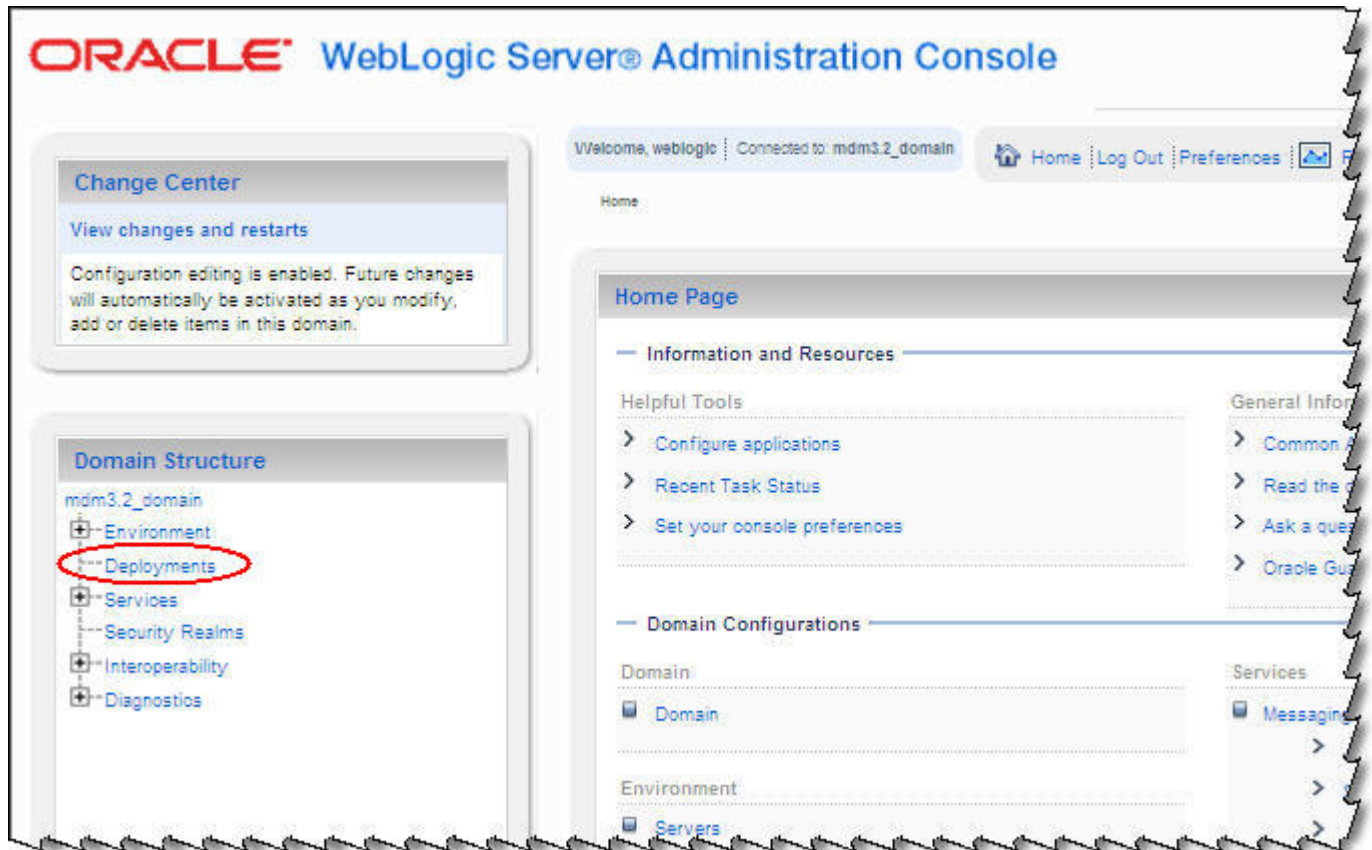
Log In

It is recommended that users log out when finished with the Administration Console or when visiting an untrusted site.

WebLogic Server Version: 10.3.0.0
Copyright © 1996, 2008, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

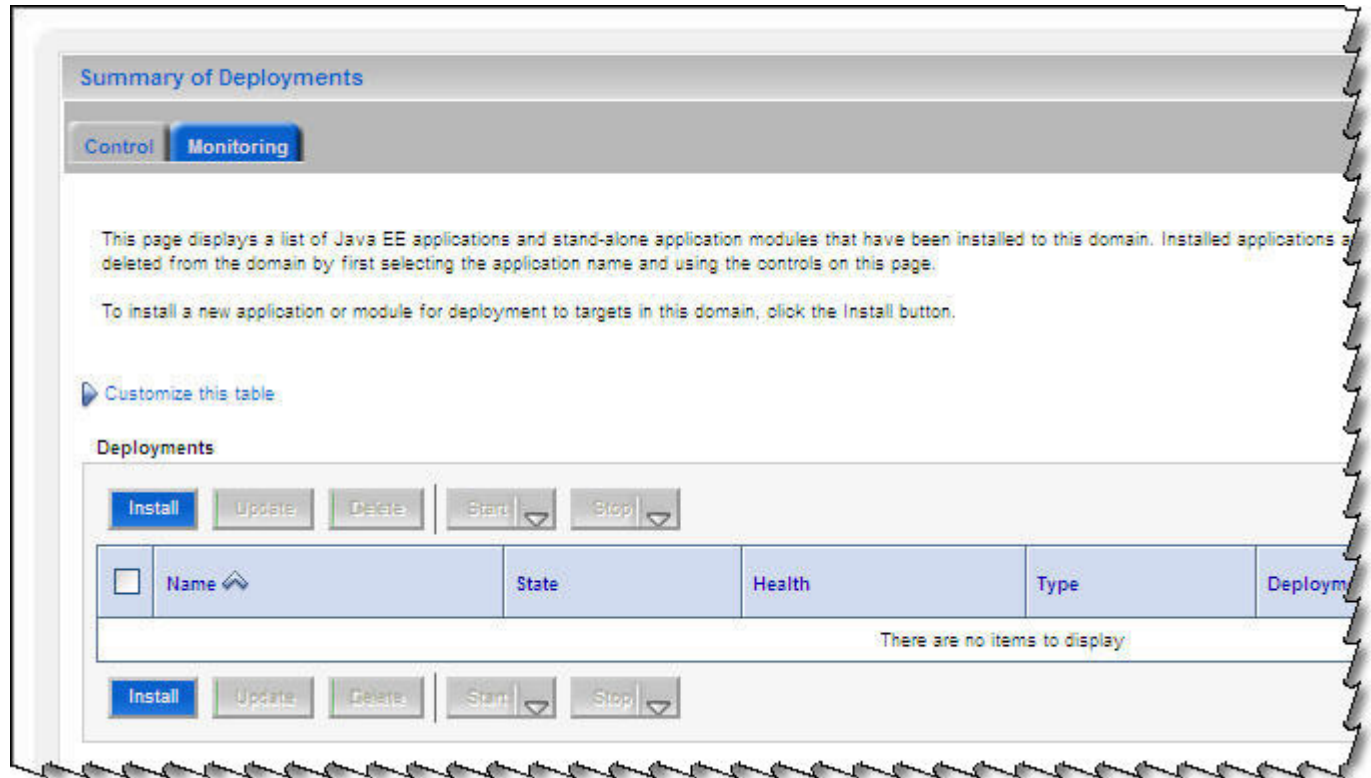
- On the Weblogic Console, enter login credentials and click **Log In**.
The **Weblogic Console Home Page** (Figure 6) is displayed.

Figure 6: Weblogic Console Home Page



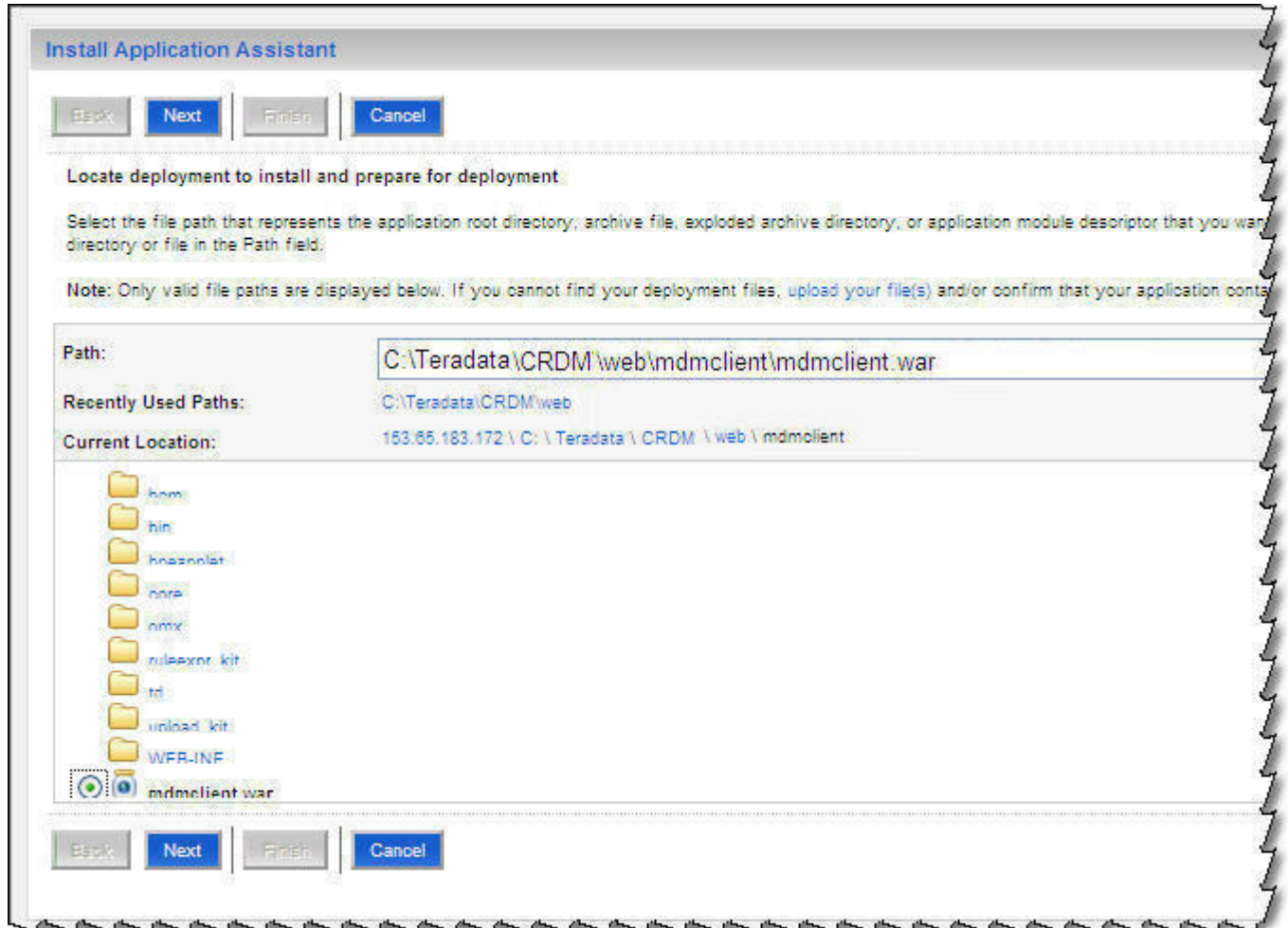
- On the **Weblogic Console Home Page**, in the **Domain Structure**, click **Deployments** [Figure 6](#).
The **Summary of Deployments** page ([Figure 7](#)) is displayed.

Figure 7: Summary of Deployments



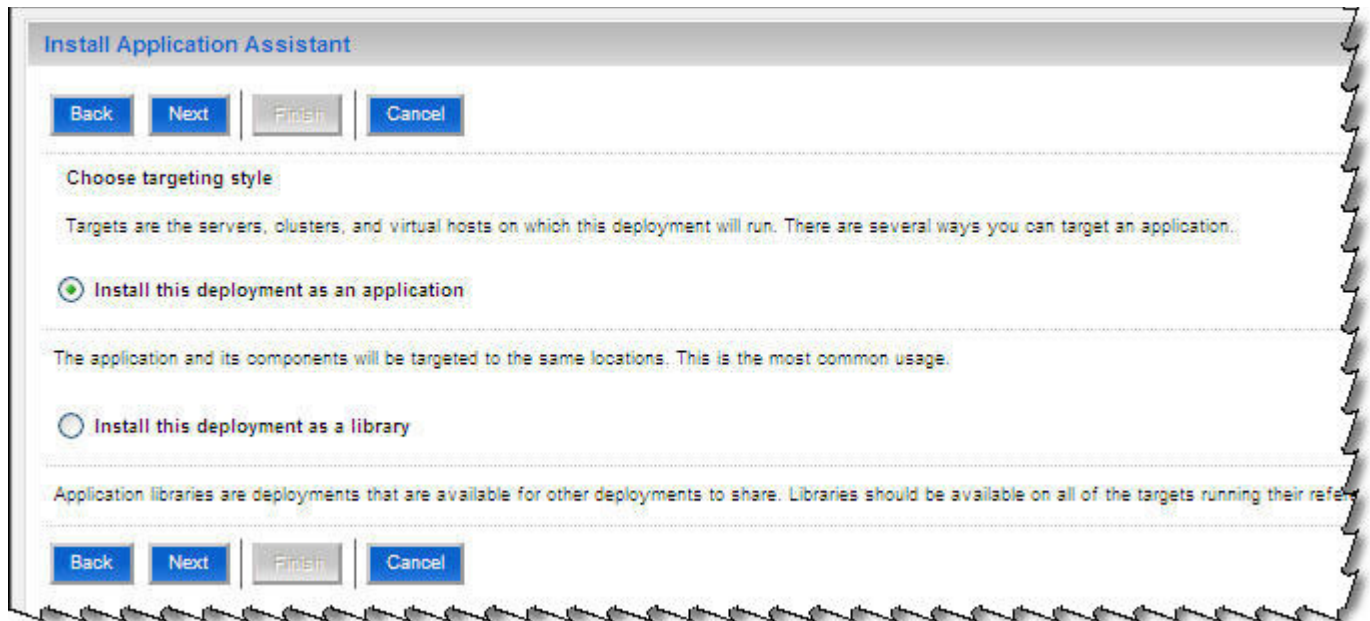
- The **Summary of Deployments** page (Figure 7), click **Install**.
The **Install Application Assistant** page (Figure 8) is displayed.

Figure 8: Install Application Assistant



- On the **Install Application Assistant** page (Figure 8), enter the path of mdmclient.war file and click **Next**.
The **Install Application Assistant—Choose Targeting Style** page (Figure 9) is displayed.

Figure 9: Install Application Assistant—Choose Targeting Style



- On the **Install Application Assistant—Choose Targeting Style** page (Figure 9), select the option “Install this deployment as an application” and click **Next**.
The **Install Application Assistant—Optional Settings** page (Figure 10) is displayed.

Figure 10: Install Application Assistant—Choose Targeting Style

Install Application Assistant

Back Next Finish Cancel

Optional Settings

You can modify these settings or accept the defaults.

General

What do you want to name this deployment?

Name:

Security

What security model do you want to use with this application?

☒ DD Only: Use only roles and policies that are defined in the deployment descriptors.

☐ Custom Roles: Use roles that are defined in the Administration Console; use policies that are defined in the deployment descriptor.

☐ Custom Roles and Policies: Use only roles and policies that are defined in the Administration Console.

☐ Advanced: Use a custom model that you have configured on the realm's configuration page.

Source accessibility

How should the source files be made accessible?

☒ Use the defaults defined by the deployment's targets

Recommended selection.

☐ Copy this application onto every target for me.

During deployment, the files will be copied automatically to the managed servers to which the application is targeted.

☐ I will make the deployment accessible from the following location

Location:

Provide the location from where all targets will access this application's files. This is often a shared directory. You must ensure the application files are accessible to all targets.

Back Next Finish Cancel

- On the **Install Application Assistant—Optional Settings** page (Figure 10), retain the options as is and click **Finish**.
The Deployment Summary page displays the success message and the name of the application deployed.

4 Start MDM and WebLogic Servers.

Note: You can also deploy webclient as an archived war file from backend by adding mdmclient.war in config.xml file as below.

```
<source-path><MDM_Install_Dir>\web\mdmclient.war</source-path>
```

APPENDIX B SSL Configuration with MDM

What's In This Appendix

This appendix provides information on configuring SSL with MDM guidelines.

Topics include:

- [Configure SSL with MDM](#)
- [Setup SSL Certificate in MDM and Shared Services](#)

Configure SSL with MDM

This section contains information for setting up the SSL with MDM. To enable SSL, you must generate the nginx certificate.

You need not to download a Root CA certificate, as they are available in web browsers trust stores and sometimes pre-installed on some operating systems.

However, if you download Root CA certificate (such as starting your own CA or self-signing), you can download the other necessary certificates on [Comodo's Support Page](#).

Your computer can identify whether a certificate is valid if the root certificate is not on their trusted root CA list. You get a warning message confirming that the certificate is not a trusted one.

Self-signed certificate should be added in the java Keystore.

Services are interacting using an inter-process communication protocol “http”. Nginx acts as proxy server and accepts only encrypted data if SSL is configured. Since the service pass the encrypted data, so certificate adds in the java key store.

Prerequisite:

Ensure that you have installed openssl.

Generate nginx Certificate

If you already have an SSL certificate in pfx format, then do the following:

- 1 Convert SSL certificate from pfx format to pem format using following command:
`openssl pkcs12 -in \D\Teradata\Certs\dtdataappt01.txhealth.org.pfx -clcerts -nokeys -out mdm.crt.pem`
- 2 Enter keystore password from server.xml available at web application server (Tomcat) using following command:

```
openssl pkcs12 -in D:\Teradata\Certs\tddataappt01.txhealth.org.pfx -nocerts -nodes -out
mdm.key.pem
```

If you do not have an SSL certificate, follow below steps to generate nginx certificate:

- 1 Download and install root certificates on your machine.
 - a Create a new directory as *certs* under */opt/teradata/*
 - b Set an environment variable, *CA_CERTS=/opt/teradata/certs*
 - c Copy all the downloaded certificates to this directory (i.e. unzip all the files from */installer/bash-scripts/tdaa-dev-root-ca.tar/tdaa-dev-root-ca*)
 - d Install the certificate *ca-root.crt.pem* into the system using following steps:
 - i Open Run dialog prompt.
 - ii Type *mmc* in Run and press enter key to open the Root console.
 - iii Click **File** and then select **Add/Remove Snap-ins** to open the window in the snapshot.
 - iv Select **Certificates** from **Available snap-ins** and click **Add>** button.
 - e Make a copy of *ca-root.crt.pem* and change the extension to *ca-root.crt*
 - f Add the same certificate into your java KeyStore. Open command prompt as administrator in the location *\$JAVA_HOME\lib\security* and run the command :
`keytool -import -alias ca-root -keystore cacerts -file C:\opt\teradata\certs\ca-root.crt.pem`
 For JAVA 11 / Linux machine, the command for the same will be `$JAVA_HOME/bin/keytool -importcert -trustcacerts -file /opt/teradata/certs/ca-root.crt.pem -alias ca_root -keystore $JAVA_HOME/lib/security/cacerts`



Password for adding the certificate is “changeit”.

- 2 Create a new folder 'mdm' inside '\etc' folder.
- 3 Generate certificate signing request configuration for nginx. A sample can be found below
- 4 Generate private key for nginx using the command:
`openssl genrsa -out \etc\{folder_name}\{file_name}.key.pem 2048`
- 5 Generate certificate signing request using the command:
`openssl req -new -x509 -key \etc\{name}\{name}.key.pem -config \etc\{name}\{name}.cnf -out \etc\{name}\{name}.csr.pem`
- 6 Generate signed intermediate certificate using the root CA:
`openssl x509 -in \etc\{name}\{name}.csr.pem -CA \opt\teradata\certs\ca-int.crt.pem -CAkey \opt\teradata\certs\ca-int.key.pem -CAcreateserial -CAserial \opt\teradata\certs\ca-int.{name}.{hostname}.srl -days 750 -out \etc\{name}\{name}.crt.pem`

- 7 Generate Certificate Bundle using the command:

```
cat \etc\{name}\{name}.crt.pem \opt\teradata\certs\ca-int.crt.pem > \etc\{name}\{name}-  
bundle.crt.pem
```

- 8 Add the generated certificate to your java KeyStore. Open command prompt as administrator in the location \$JAVA_HOME\lib\security and run the command:

```
keytool -import -alias ca-root1 -keystore cacerts -file C:\etc\{name}\{name}.pem
```

For JAVA 11 / Linux machine, use the command:

```
$JAVA_HOME/bin/keytool -import -alias ca-root1 -keystore cacerts -file  
C:\etc\{name}\{name}.crt.pem
```



Ensure that you provide a different alias name to the CA root for adding root certificate in the keystore.

Setup SSL Certificate in MDM and Shared Services

Follow below steps to set up SSL certificate in MDM and shared services:

- 1 Go to nginx.conf and add the certificate under HTTPS server. A sample can be found.



nginx.conf

Ensure that the below two parameters are added in the nginx config file:

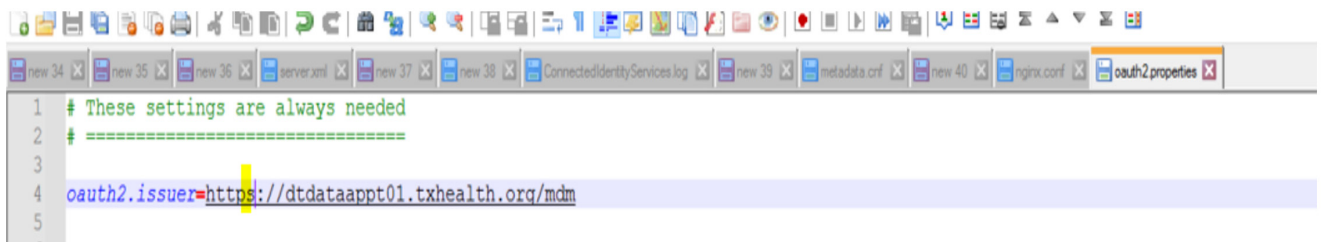
- proxy_set_header X-Forwarded-Proto https;
- proxy_set_header X-Forwarded-Host localhost;

```
#error_log logs/error.log notice;
#error_log logs/error.log info;
#pid logs/nginx.pid;
events {
    worker_connections 1024;
}
http {
    include mime.types;
    default_type application/octet-stream;
    #log_format main '$remote_addr - $remote_user [$time_local] "$request" '
    # '$status $body_bytes_sent "$http_referer" '
    # '"$http_user_agent" "$http_x_forwarded_for"';
    #access_log logs/access.log main;
    sendfile on;
    #tcp_nopush on;
    #keepalive_timeout 0;
    keepalive_timeout 65;
    #gzip on;
    #HTTPS servers
    #
    server {
        listen      80;
        server_name  dtdataappt01.txhealth.org;
        return       301 https://$server_name$request_uri;
    }

    server {
        listen      443 ssl;
        server_name  dtdataappt01.txhealth.org;
        ssl_certificate      D:\SSL\CERT\mdm.crt.pem;
        ssl_certificate_key  D:\SSL\CERT\mdm.key.pem;

        proxy_buffer_size    256k;
        proxy_buffers         4 256k;
        proxy_busy_buffers_size 256k;
        proxy_cache off;
        proxy_redirect off;
    }
}
```


- 2 Edit oauth2.properties file located at the location *<MDM-HOME>\web\mdmclient\WEB-INF\spring\config* and change the protocol from 'http' to 'https' in the url.



```
1 # These settings are always needed
2 # =====
3
4 oauth2.issuer=https://dtdataappt01.txhealth.org/mdm
5
6
```

- 3 Edit td-workflow-connected-identity.properties file located at the location *<MDM-Home>\cfg* and change the protocol from 'http' to 'https' in the url.

For collapsed, change td-workflow-connected-identity.properties at the location *<MDM-Home>\web\mdmclient\WEB-INF\bcm\cfg* and run the colloc.jar



```
1 connected.identity.base.uri : dtdataappt01.txhealth.org
2 connected.identity.context.path : connected-identity
3 oauth.token.endpoint : https://dtdataappt01.txhealth.org/mdm/oauth/token
4
```




This is not applicable to collocated (DEV).

- 4 Edit the application.properties file located inside shared service folder and change **authentication-service.protocol** properties protocol scheme to 'https' for all the shared services(both Connected-Identity and Metadata).

```
# Authentication service configuration
#services.auth.context.prefix=authentication
services.auth.context.prefix=mdm
authentication-service.protocol=https
authentication-service.url=${authentication-service.protocol}://${server.hostname:localhost}/${services.auth.context.prefix}
authentication-service.clientId=${security.client.id:connected-identity}
authentication-service.clientSecret=${security.client.secret:secret}
```

The properties service.metadata.url and service.mdm.url property are present only in CI application.properties.

```
#below one not required for metadata service and cust-360
service.metadata.url=https://${server.hostname:localhost}/${services.metadata.context.prefix}
service.mdm.url=https://${server.hostname:localhost}/${services.mdm.context.prefix}
```



Ensure that server.protocol is mentioned as 'http' only.

- 5 Open mdm url with 'https'.

APPENDIX C Table Name Changes During Upgrade

What's In This Appendix

This appendix provides information on MDM database sizing guidelines.

Topics include:

- [Introduction](#)
- [List of Tables Upgrading During Upgrade](#)

Introduction

When you upgrade MDM versions lower than 4.7 to the upper version, the list of tables rename in the process of upgrade. Earlier, these tables name were not getting upgrade as per the generic bkp naming convention which was occurring an issue or error while running the upgrade script.

When you run the upgrade script and custom rename task start functioning, the mentioned list of table get rename as per the format: <table_name>_bkp<version_no>

For example if you are upgrading to MDM 4.7.0.0(BKP table name is version specific) then it will be <table_name>_bkp4700

List of Tables Upgrading During Upgrade

TABLES WITH SUFFIX AS _BKP4700		
WF_CONFIG_HEADER	BR_BUSINESS_RULE	MU_LEFT_NAV_URL
ID_SQR	US_ACTIVITY	MU_LEFT_NAV_URL
MU_LEFT_NAV_STRUCTURE	US_ROLE_ACTIVITY_MAP	GROUP
GROUP_JOB_RUN	JOB_GROUP	JOB_MASTER
JOB_RUN_DETAILS	SOURCE_INTAKE	CL_CLEANINGPAD_COL_MAP
CL_CLEANINGPAD_MAP	CL_PROFILE_RULE_MAP	FM_ATTR_PAIRING
FM_ATTRIBUTES	FM_DEDUPE_RUN_HISTORY	FM_KEYS

TABLES WITH SUFFIX AS _BKP4700		
SV_GT_SURV_RUN_ID	SV_GT_SURV_SCORE	SV_SURV_EXEC_DETAIL_LOG
SV_SURV_EXEC_LOG	SV_SURV_MAP_MATCH_ASSOC	SV_SURV_MAPPING
SV_SURV_REL_MAPPING	SV_SURV_TRG_CNSLDLTN_RULES	SV_SURV_ID_GENERATOR
FM_PARAMETERS	SV_SURV_RULES_DEFN	SV_SURV_LOOKUP
CL_CAS_PROFILE_EXEC_LOG	FM_PROFILE	SV_SURV_LOOKUP
FM_GT_LOG	FM_GT_MATCHING_SCORES	FM_GT_PER_CNT
FM_GT_SPMLM_MATCH_VALUES	FM_GT_TABLE_NAMES	FM_GT_VEC
FM_GT_NICKNAME_TEMP	SV_EXE_RUN_ID	SV_GT_PF_HOLD_CNT
SV_GT_SURV_EXEC_DETAIL_LOG	SV_GT_SURV_EXEC_LOG	SV_GT_SURV_INTERIM_SQL
SYS_GT_ERR_SVRTY_CNT_TEMP	CL_GT_CAS_RULE_EXEC_LOG	CL_GT_CAS_SOURCE_COUNT
CL_SOURCE	CL_GT_PF_PK_IDX	SV_SURV_AUTO_ID_GEN
SV_SURV_MAPPING_CONDITION	SV_SURV_MAP_PROFILE	SV_SURV_SQL
SV_SURV_RULE	SV_SURV_TRANSFORMATION	SV_SURV_INTERIM_SQL
FM_SPMLM_ATTRIBUTE_SETS	FM_RECON_KEY_MAP	FM_KEY_MATCH_RECON_MAP
FM_RUN_HISTORY	FM_BLOCK	FM_CROSS_REF_KEY_MAP
SYS_DB_MAP	FM_NICKNAMES	FM_GT_NICKNAMES



These tables are part of CI shared services.

Below are some tables mentioned with their before and after name of upgrade:

TABLERNAME BEFORE RENAME	TABLERNAME AFTER RENAME OPERATION
SC_SCHEDULER_JOBS	SC_SCHEDULER_JOBS_bkp_old
NOS_ATTRIBUTES	NOS_ATTRIBUTES_bkp_old
NOS_OBJECTS	NOS_OBJECTS_bkp_old

APPENDIX D Spring Security

What's In This Appendix

This document describes the integration of spring security in MDM framework focusing only on namespace configuration introduced in spring 3.x.

Topics include:

- [Overview](#)
- [MDM Spring Security Architecture](#)
- [Spring Security Integration with MDM](#)
- [Troubleshooting](#)

Overview

Spring Security is a Java/J2EE framework that provides advanced security features for the enterprise application. Spring Security targets two areas namely, Authentication and Authorization. The Authorization is the process of giving permission to user to access the resources for which the user is authorized to use. The Authentication is the process that ensures and confirms a user's identity. For example, when you logon to any application and enter the login credentials, you authenticate yourself. MDM supports spring authentication. Spring authorization is currently not supported in MDM.

At the authentication level, spring security supports various authentication models. MDM supports the below authentication models of spring security with default implementations that are used by namespace configuration. This appendix demonstrates the configurations of using the following authentication models in MDM.

- User service authentication
- LDAP authentication
- Active Directory authentication
- Kerberos authentication
- SAML authentication
- ADFS authentication

Namespace Configuration

The namespace configuration of the spring security provides a simpler and concise way of configuring by hiding the underlying complexity of the framework. To start the namespace configuration, security filter needs to be defined in web.xml as shown below:

```

<filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter
class>org.springframework.web.filter.DelegatingFilterProxy</filter-
class>
</filter>
<filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>

```

In the above configuration, *DelegatingFilterProxy* delegates the control to a filter implementation, which is defined as a bean called *springSecurityFilterChain*. The *springSecurityFilterChain* is an infrastructure internal bean to handle namespace configurations. Once this configuration is done, all the incoming requests to MDM enters the spring framework for security checks. Spring security scans the entire request for authentication and communicates the authenticated user details to MDM framework.

Security Configuration

The MDM spring security configuration is performed in the below XML files located at:
<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring.

- 1 user-service-authentication-provider.xml
- 2 ldap-authentication-provider.xml
- 3 ads-authentication-provider.xml
- 4 kerberos-authentication-provider.xml
- 5 saml-authentication-provider.xml
- 6 adfs-authentication-provider.xml

Apart from user-service-authentication-provider.xml, all other files are dynamic and all the configurations are preconfigured. The user specific configuration are acquired from a property file located under <MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config folder.

MDM Spring Security Login and Logout

Spring DispatcherServlet defined in web.xml is mapped with two URL patterns responsible for login and logout. All MDM spring security login and logout are passed through this servlet.

```

<servlet>
    <servlet-name>mvc-dispatcher</servlet-name>
    <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>mvc-dispatcher</servlet-name>
    <url-pattern>/j_spring_security_check</url-pattern>
</servlet-mapping>
<servlet-mapping>

```

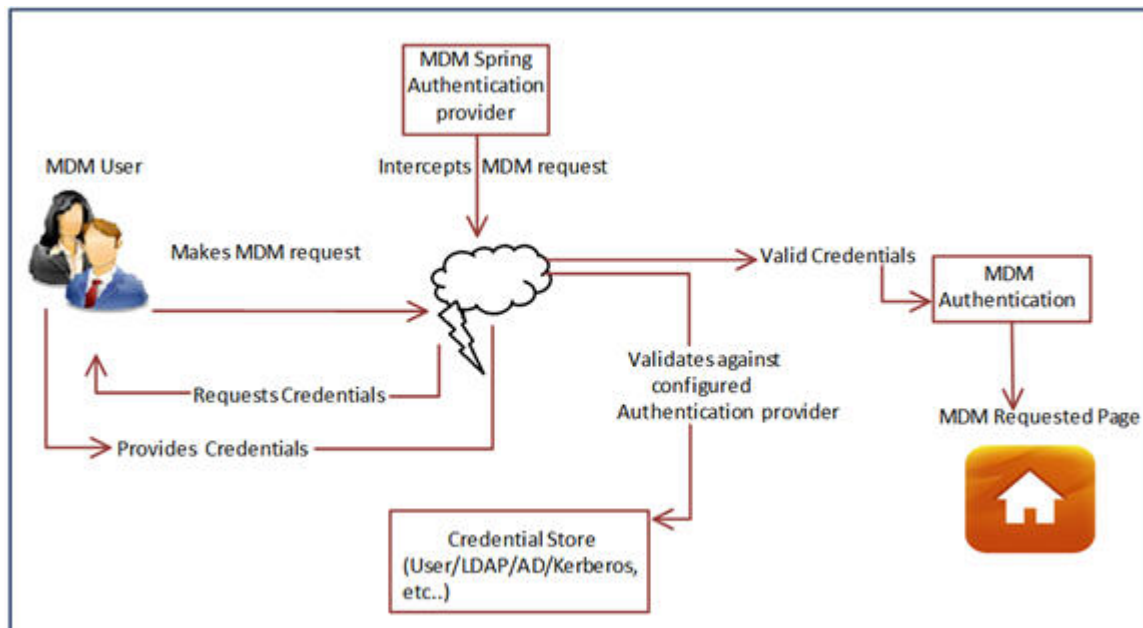
```
<servlet-name>mvc-dispatcher</servlet-name>
<url-pattern>/j_spring_security_logout</url-pattern>
</servlet-mapping>
```

MDM Spring Security Architecture

The MDM spring security architecture relies heavily on the use of delegates and servlet filters to provide layers of functionality around the context of a MDM request. Servlet Filters (classes that implement the `javax.servlet.Filter` interface) are used to intercept user requests and perform pre or post-processing, or redirect the request altogether, depending on the function of the servlet filter. The final destination servlet is the spring dispatcher servlet.

The [Figure 11](#) displays the architecture of MDM spring security.

Figure 11: MDM Spring Security Architecture



Spring Security Integration with MDM

The integration of spring security in MDM involves the below step:

- [Configure Authentication Provider](#)

Configure Authentication Provider

MDM spring security is certified with the below four types of authentication providers:

Note: Rename `<MDM_Install_Directory>\web\mdmclient\bcm\frameworksecure_login.jsp` to `login.jsp` for ADS, LDAP and other authentications mechanism which does not require “Forgot Password” and other buttons.

- [User Service Authentication Provider](#)
- [LDAP Authentication Provider](#)
- [ADS Authentication Provider](#)
- [Kerberos Authentication Provider](#)
- [SiteMinder Authentication Provider](#)
- [SAML Authentication Provider](#)
- [ADFS Authentication Provider](#)
- [HTTP Strict Transport Security \(HSTS\)](#)
- [OAuth2 Authentication](#)

User Service Authentication Provider

The following section demonstrates the unsecured user service authentication process of enabling spring security in MDM. The MDM login in turn authenticates spring secured authentication with a fundamental username and password. The user service authentication provider should not be used in production environment as the username and passwords are available in plain text.

Perform the following steps to enable user service authentication provider in MDM:

- 1 Open web.xml from `<MDM_Install_Dir>\web\mdmclient\WEB-INF` and ensure that user service authentication provider.xml is configured under contextConfigLocation section as below:

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/user-service-authentication-
provider.xml</param-value>
</context-param>
```

- 2 Open user-service-authentication-provider.xml from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring` and in the `<sec:user-service>` section, add or modify MDM users and passwords. For example, to add "crdm_user" with password "secret", use the below code line.

```
<sec:user name="crdm_user" password="secret" authorities="user" />
```

- 3 Restart servers.
- 4 Now login to MDM using the URL `http://HOSTNAME:PORT/mdm/bcm/framework/secure_login.jsp` and with the above configured username and password.

LDAP Authentication Provider

LDAP is mostly used as a way to centralize corporate user information, partitioning thousands of users into logical groups, and allowing unified sharing of user information across many distinct systems. For security purposes, LDAP is commonly used to facilitate centralized username and password authentication. The user's credentials are stored in the LDAP directory and authentication requests are made against the directory on behalf of the user. This simplifies the management process for administrators, as user credentials like login, password, and other details are stored in a single location in the LDAP. The below

section demonstrates the configuration of MDM spring security LDAP authentication provider that checks user credentials against the LDAP provider.

Perform the following steps to enable LDAP authentication provider in MDM:

- 1 Open web.xml from `<MDM_Install_Dir>\web\mdmclient\WEB-INF` and add ldap-authentication-provider.xml under contextConfigLocation section as below. Generally, it not required to modify this file as it is preconfigured.

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/ldap-authentication-
provider.xml</param-value>
</context-param>
```

- 2 Open ldap.properties file from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config` and configure the entries for the following properties:

```
ldap.URL=ldap://serve.domain.com:10389/ou=system (LDAP server URL)
ldap.manager-dn=uid=admin,ou=system (manager dn)
ldap.manager-password=secret (manager password)
```

- 3 Other setting are required for advanced configurations and mostly not required to alter.

Note: The other below marked entries such as search filter and search base for user group and role-prefix can be commented if not used based on the requirement.

```
<sec:authentication-manager>
    <sec:ldap-authentication-provider
        user-search-filter="${ldap.user-search-filter}"
        user-search-base="${ldap.user-search-base}"
        group-search-filter="${ldap.group-search-filter}"
        group-search-base="${ldap.group-search-base}"
        group-role-attribute="${ldap.group-role-attribute}"
        role-prefix="${ldap.role-prefix}">
    </sec:ldap-authentication-provider>
</sec:authentication-manager>
```

- 4 Restart servers.
- 5 Now login with `http://HOSTNAME:PORT/mdm/bcm/framework/secure_login.jsp` URL and use LDAP username and password.
- 6 To configure logout success page, open ldap.properties file from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config` and modify the below default setting as required.

```
ldap.logout-success-url=/bcm/framework/
login.jsp?FROM=logout&NonSSOLogin=yes
For example: ldap.logout-success-url=/bcm/framework/
<customlogout.html>
```


Enabling LDAP Authorization for Soap based Web Services :

Note: Soap based web service is deprecated from MDM 4.9.0.0 release.

Using Multi Domain Authentication with MDM

Perform the following steps to configure multiple domains to connect to MDM through LDAP authentication:

- 1 Stop all servers
- 2 Navigate to `<MDM_Install_Directory>\web\mdmclient\WEB-INF\web.xml` and open the authentication provider file.

```

</listener>
<listener>
  <listener-class>org.springframework.web.context.request.RequestContextListener</listener-class>
</listener>
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>/WEB-INF/spring/ldap-authentication-provider.xml</param-value>
</context-param>
<filter>
  <filter-name>springSecurityFilterChain</filter-name>
  <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>
<filter-mapping>
  <filter-name>springSecurityFilterChain</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<!-- ***** Spring ***** -->

```

- 3 In the above example (authentication provider file), LDAP configuration file is at : `<MDM_Install_Directory>\web\mdmclient\WEB-INF\spring\ldap-authentication-provider.xml`
- 4 Open the ldap-authentication-provider.xml file and under the `<sec:authentication-manager>` tag, add `<sec:ldap-authentication-provider block>` as follows:

```

<sec:authentication-manager>
  <sec:ldap-authentication-provider server-ref="ldapServer"
    user-search-filter="${ldap.user-search-filter}"
    user-search-base="${ldap.user-search-base}"
    group-search-filter="${ldap.group-search-filter}"
    group-search-base="${ldap.group-search-base}"
    group-role-attribute="${ldap.group-role-attribute}"
    role-prefix="${ldap.role-prefix}" />
  <sec:ldap-authentication-provider server-ref="ldapServer1"
    user-search-filter="${ldap1.user-search-filter}"
    user-search-base="${ldap1.user-search-base}"
    group-search-filter="${ldap1.group-search-filter}"
    group-search-base="${ldap1.group-search-base}"
    group-role-attribute="${ldap1.group-role-attribute}"
    role-prefix="${ldap1.role-prefix}" />
</sec:authentication-manager>

```

- 5 To configure ldap-server, in the above ldap-authentication-provider block, add id tag for mapping ldap server as below in server-ref tag.

```

<sec:ldap-server id="ldapServer" url="${ldap.URL}" manager-dn="${ldap.manager-dn}" manager-password="${ldap.manager-password}" />
<sec:ldap-server id="ldapServer1" url="${ldap1.URL}" manager-dn="${ldap1.manager-dn}" manager-password="${ldap1.manager-password}" />

```


Any number of authentication provider and ldap-server can be configured under <sec:authentication-manager> tag.

- 6 Navigate to <MDM_Install_Directory>\web\mdmclient\WEB-INF\spring\config and open ldap.properties and add the new properties for second ldap server.

```
ldap1.user-search-filter=(uid={0})
ldap1.user-search-base=ou=users
ldap1.group-search-filter=(uniqueMember={0})
ldap1.group-search-base=ou=groups
ldap1.group-role-attribute=cn
ldap1.role-prefix=ROLE_
```

- 7 Clear all cache & restart servers.

With above configuration, initially user will be authenticated using first LDAP, if it fails, then user will be authenticated using second LDAP, if both fails then error page will be displayed.

Authorization through Active Directory

MDM authenticate users via LDAP, authorization is accomplished using MDM. For enabling group assignment from LDAP, the parameter "AD_AUTHORIZATION_ENABLED" must be set to "true" in xserver.xml and xserverweb.xml as below. By default, this parameter is set to "false".

```
<param Value="true" Name="AD_AUTHORIZATION_ENABLED"/>
```

Once this parameter is enabled, group assignment from Active directory will be honored and on successful login, the user will be able to view and access the left navigation entries in MDM UI.

When a new user is created, the user is preferably assigned to MDM Default user group. If user tries to login only with MDM default group, there will not be any left navigation entries available to that user and user cannot perform any MDM operations on MDM web UI. To get left navigation entries admin must assign respective MDM Group/s to the user in Active Directory.

On login, if AD_AUTHORIZATION_ENABLED flag is true and once user authorization is completed successfully, LDAP authentication populater will be called, LDAP authentication populater is responsible to get the list of Groups assigned to current logged-in user from LDAP. Assigned Group details will be persisted in session and LDAP group detail information will be added to MDM Identity object and left navigation will be rendered accordingly.

The LDAP/AD group needs to exist in MDM and vice versa, if not available the corresponding user group has to be manually created in MDM or LDAP/AD.

ADS Authentication Provider

Active Directory (AD) is a directory service implemented by Microsoft for Windows domain networks. It is included in most Windows server operating systems. An AD domain controller authenticates and authorizes all users and computers in a Windows domain type network

assigning and enforcing security policies for all computers and installing or updating software. For example, when a user logs into a computer that is part of a Windows domain, Active Directory checks the submitted password and determines whether the user is a system administrator or normal user. The below section demonstrates the process of configuring MDM spring security ADS authentication provider that checks user credentials against Microsoft Active Directory service.

Perform the following steps to enable ADS authentication provider in MDM:

- 1 Open web.xml from `<MDM_Install_Dir>\web\mdmclient\WEB-INF` and add ads-authentication-provider.xml under contextConfigLocation section as below. Generally, it not required to modify this file as it is preconfigured.

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/ads-authentication-provider.xml</param-value>
</context-param>
```

- 2 Open ads.properties file from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config` and configure the entries for the following properties:

```
ads.domain=ads.domain.com (domain fully qualified name)
ads.URL=ldap://ads.domain.com (URL of ADS server)
```

- 3 Restart servers.
- 4 Now login with `http://HOSTNAME:PORT/mdm/bcm/framework/secure_login.jsp` URL and use ADS username and password for login.
- 5 To configure logout success page, open ads.properties file from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config` and modify the below default setting as required.

```
ads.logout-success-url=/bcm/framework/
login.jsp?FROM=logout&NonSSOLogin=yes
```

For example: `ads.logout-success-url=/bcm/framework/`
`<customlogout.html>`

Kerberos Authentication Provider

Kerberos is a mutual authentication protocol used for authenticating clients either individual users or network resources against a centralized credentials repository known as the key distribution center (KDC). The negotiation between the client and KDC is well documented in several internet standards (primarily RFC 4120, The Kerberos Network Authentication Service (V5), available at <http://tools.ietf.org/html/rfc4120>) and several books are available covering Kerberos in a high level of detail. The general purpose of Kerberos authentication and Kerberos infrastructure is to provide secure and trustworthy authentication of principals for either individual users, network resources, or software applications. Kerberos describes both the security protocol and a software implementation originally developed at the Massachusetts Institute of Technology (MIT). The following section demonstrates the configuration of MDM spring security Kerberos authentication provider that checks user credentials against Kerberos.

Perform the following steps to enable Kerberos authentication provider in MDM:

- 1 Open web.xml from `<MDM_Install_Dir>\web\mdmclient\WEB-INF` and add kerberos-authentication-provider.xml under contextConfigLocation section as below. Generally, it not required to modify this file as it is preconfigured.

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/kerberos-authentication-
provider.xml</param-value>
</context-param>
```

- 2 Open kerberos.properties file from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config` and configure the entries for the following properties:

```
krb.debug=false (flag for debug, values can be true or false)
krb.service.principal=HTTP/server.domain.com (service principal of
application server)
krb.keytab.location=file:C:\\config\\myserver.keytab (key tab file
location)
krb.conf.location=C:\\config\\krb5.conf (krb5.config)
```

- 3 Open `<MDM_Install_Dir>\web\mdmclient\WEB-INF\web.xml`

Change welcome-file-list section

```
<welcome-file-list>
<welcome-file>bcm/framework/redirect.jsp</welcome-file>
</welcome-file-list>
```

as

```
<welcome-file-list>
<welcome-file>bcm/framework/target.jsp</welcome-file>
</welcome-file-list>
```

- 4 Restart servers.
- 5 Now login with `http://HOSTNAME:PORT/mdm/bcm/framework/target.jsp` URL, once you enter this URL, it will automatically login with domain user name.

Note: Steps to configure Kerberos are available separately.

- 6 To configure logout success page, open kerberos.properties file from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config` and modify the below default setting as required.

```
krb.logout-success-url=/bcm/framework/logout.html
```

For example: `krb.logout-success-url=/bcm/framework/
<customlogout.html>`

SiteMinder Authentication Provider

SiteMinder is a centralized web access management system that enables user authentication and single sign-on, policy-based authorization, identity federation, and auditing of access to Web applications.

In Pre-Authenticated scenarios (that is, situations where MDM users want to use MDM for authorization, but the user has already been reliably authenticated by SiteMinder system prior to accessing MDM application), MDM spring security has to:

- Identify the user making the request.
- Obtain the authorities for the user.

MDM spring security identifies SiteMinder user by an HTTP request header. By default, SiteMinder user header contains authenticated user detail.

Perform the following steps to enable SiteMinder authentication provider in MDM:

- 1 Open web.xml from <MDM_Install_Directory>\web\mdmclient\WEB-INF and add siteminder-authentication-provider.xml under contextConfigLocation section as below. Generally, it is not required to modify this file as it is preconfigured.

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/siteminder-authentication-
provider.xml</param-value>
</context-param>
```

- 2 Open siteminder.properties file from <MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config and configure the entries for the following properties:

```
siteminder.principalRequestHeader=SM_USER
(HTTP header name containing user name detail)
siteminder.exceptionIfHeaderMissing=true
(If header value is missing exception will be thrown if value is set to true.)
```

- 3 Open <MDM_Install_Dir>\web\mdmclient\WEB-INF\web.xml

Change welcome-file-list section

```
<welcome-file-list>
<welcome-file>bcm/framework/redirect.jsp</welcome-file>
</welcome-file-list>
```

as

```
<welcome-file-list>
<welcome-file>bcm/framework/target.jsp</welcome-file>
</welcome-file-list>
```

- 4 Restart servers.
- 5 Now login with http://HOSTNAME:PORT/mdm URL, once you enter this URL, MDM will automatically login with SiteMinder user name.
- 6 To configure logout success page, open siteminder.properties file from <MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config and modify the below default setting as required.

```
siteminder.logout-success-url=/bcm/framework/logout.html
```

For example: siteminder.logout-success-url=/bcm/framework/
<customlogout.html>

Note: When using a system like above, the framework performs no authentication checks and it is important that the external system is configured properly and protects all access to MDM application. If a hacker is able to forge the headers in their original request without being detected, they could potentially select any username as wished.

SAML Authentication Provider

MDM 4.9 support 2FA (second factor authentication) using SAML or ADFS authentication protocol.

SAML (Security Assertion Mark-up Language) is an XML-based, open-standard data format for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider.

SAML is a standard for logging users into applications based on their sessions in another context. This single sign-on (SSO) login standard provides the following advantages over logging in using a username/password:

- Standardization
- Support for Multi Factor Authentication
- Increased security
- No weak passwords
- Platform neutrality
- Loose coupling of directories
- Improved online experience for end users
- Reduced administrative costs for service providers

The [Figure 12](#) displays the simple SAML flow and [Figure 13](#) displays the high level authentication of SAML.

Figure 12: Simple SAML Flow

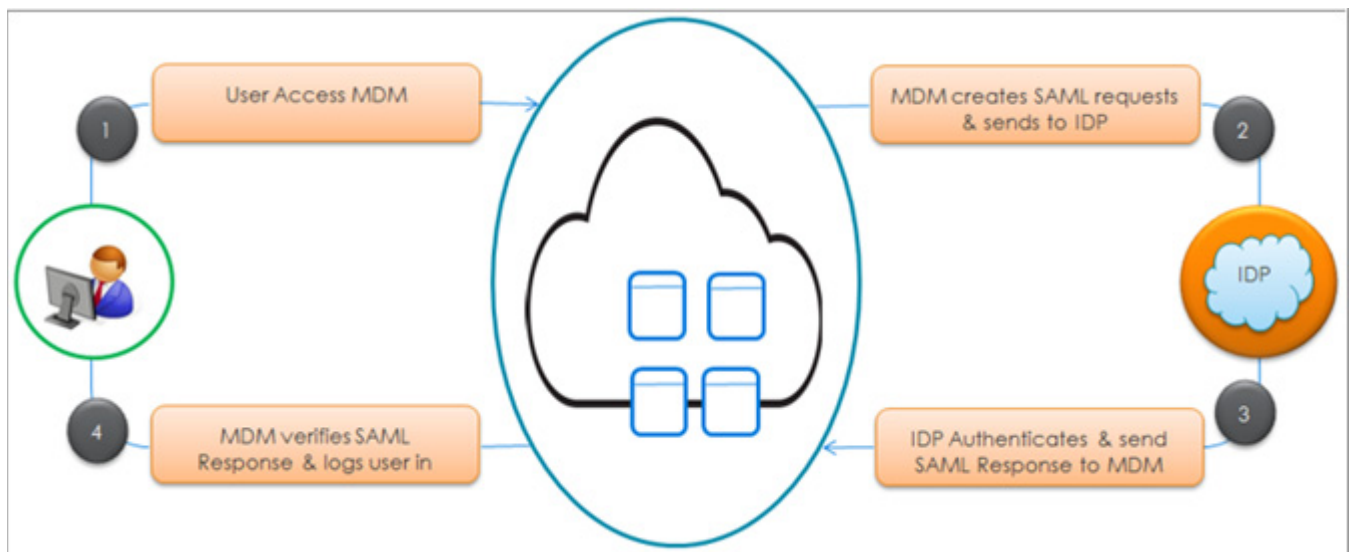
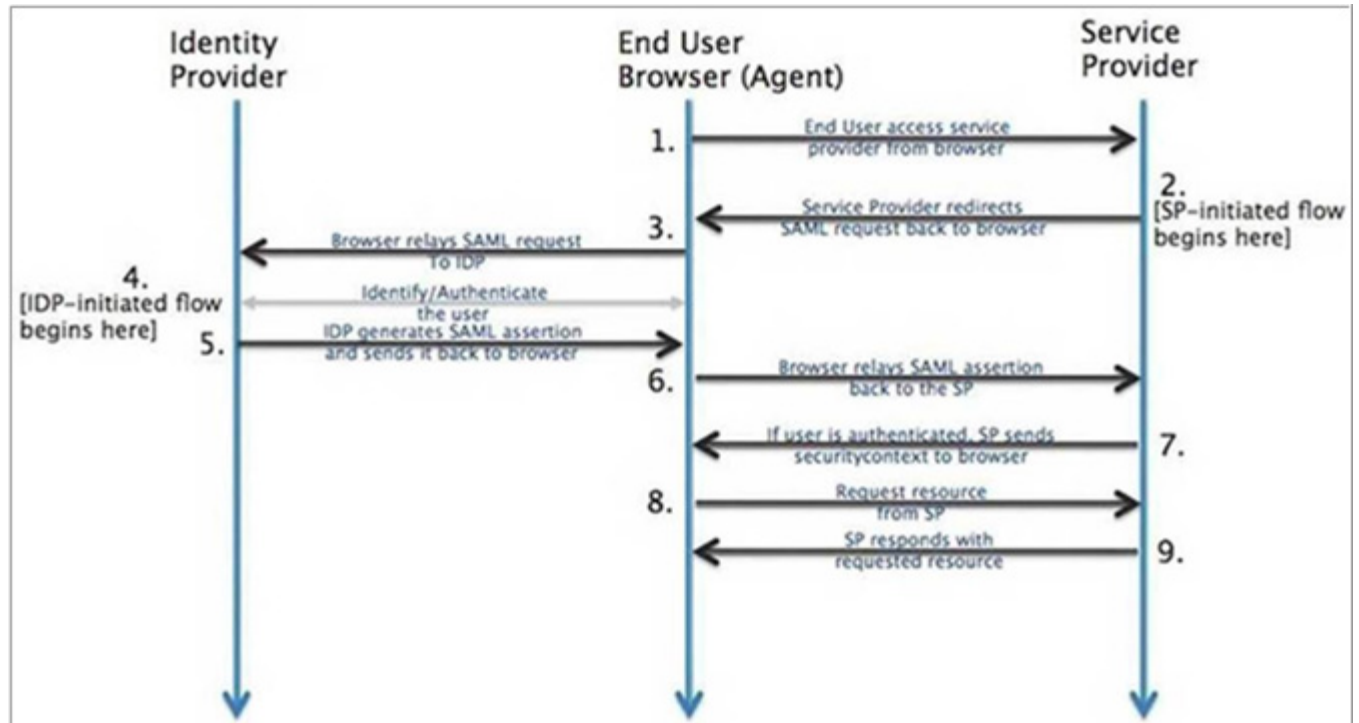


Figure 13: High Level Authentication of SAML



To enable SSO, you can use an Identity Management Service such as Ping Federate or Okta.

Prerequisite

- Ensure that you have performed Okta Configuration.
- Ensure that you have performed Ping Configuration.

Perform the following steps to enable SAML authentication provider in MDM:

1. Open and modify web.xml from \web\mdmclient\WEB-INF and do the following:
 - a. Comment the line platform-authentication-provider.xml
 - b. add saml-authentication-provider.xml under contextConfigLocation


```

<context-param>
<param-name>contextConfigLocation</param-name>
<param-value>/WEB-INF/spring/saml-authentication-provider.xml</param-value>
</context-param>
                    
```
2. Generate keystore using the keytool. Use below command to generate keystore:


```
keytool -genkey -v -keystore samlKeystore.jks -alias apollo -keyalg RSA -keysize 2048 -validity 60000
```
3. Create security folder at web\mdmclient\WEB-INF\classes location and place generated keystore in the security folder.
4. Open saml.properties file from \web\mdmclient\WEB-INF\spring\config and configure following properties:

- saml.metadataURL= https://auth.pingone.asia/80432d84-7728-4082-8481-5ca1073dc4e1/saml20/metadata/5676573f-0713-437b-9524-9f47c3af534b (Metadata URL grabbed from Ping Identity application)
 - saml.metadataURL.requestTimeout=5000 (No change, Timeout, if metadata url is used)
 - saml.jks.path=classpath:security//samlKeystore.jks (Keystore path)
 - saml.jks.password=Java Key store password
 - saml.jks.private.key.name=Private Key name
 - saml.jks.private.key.password=Private Key password
 - saml.jks.defaultKey.key=apollo Private Key Value
 - saml.sp.metadata.entityId=Metadata entity Id grabbed from Ping Identity application
 - saml.logout-success-url=https://trial-3867328.okta.com/login/signout
 - saml.defaultFailureUrl=/bcm/framework/login.jsp?ERROR=true&DESCRIPTION=INVALID_SPRING_LOGIN
 - oauth2.issuer=http://<host_name>mdm
- 5 Unzip commons-httpclient-3.1.jar from 3rdPartyLibJar zip and place it in web\mdmclient\WEB-INF\lib folder.

Ensure that all the Ping user Ids are generated in MDM application.
Note: Add the following entry in catalina.bat file at <TOMCAT_HOME>\bin\ folder location

```
set "JAVA_OPTS=%JAVA_OPTS% -Dorg.owasp.esapi.SecurityConfiguration=org.owasp.esapi.reference.DefaultSecurityConfiguration".
```
 - 6 Restart MDM and Tomcat servers.
 - 7 To obtain MDM metadata, use http://HOSTNAME/mdm/saml/metadata
 - 8 Navigate to http://HOSTNAME/mdm/ URL which redirects to configured IDP login page.
 - 9 Enter authentication information from IDP and the response redirects to MDM page.

ADFS Authentication Provider

The Active Directory Federation Services (ADFS), a software component can run on Windows Server operating systems to provide users with single sign-on access to systems and applications located across organizational boundaries. It uses a claims-based access-control authorization model to maintain application security and to implement federated identity. Claims-based authentication involves authenticating a user based on a set of claims about that user's identity contained in a trusted token. Such a token is often issued and signed by an entity that is able to authenticate the user by other means, and that is trusted by the entity doing the claims-based authentication. It is part of the Active Directory Services.

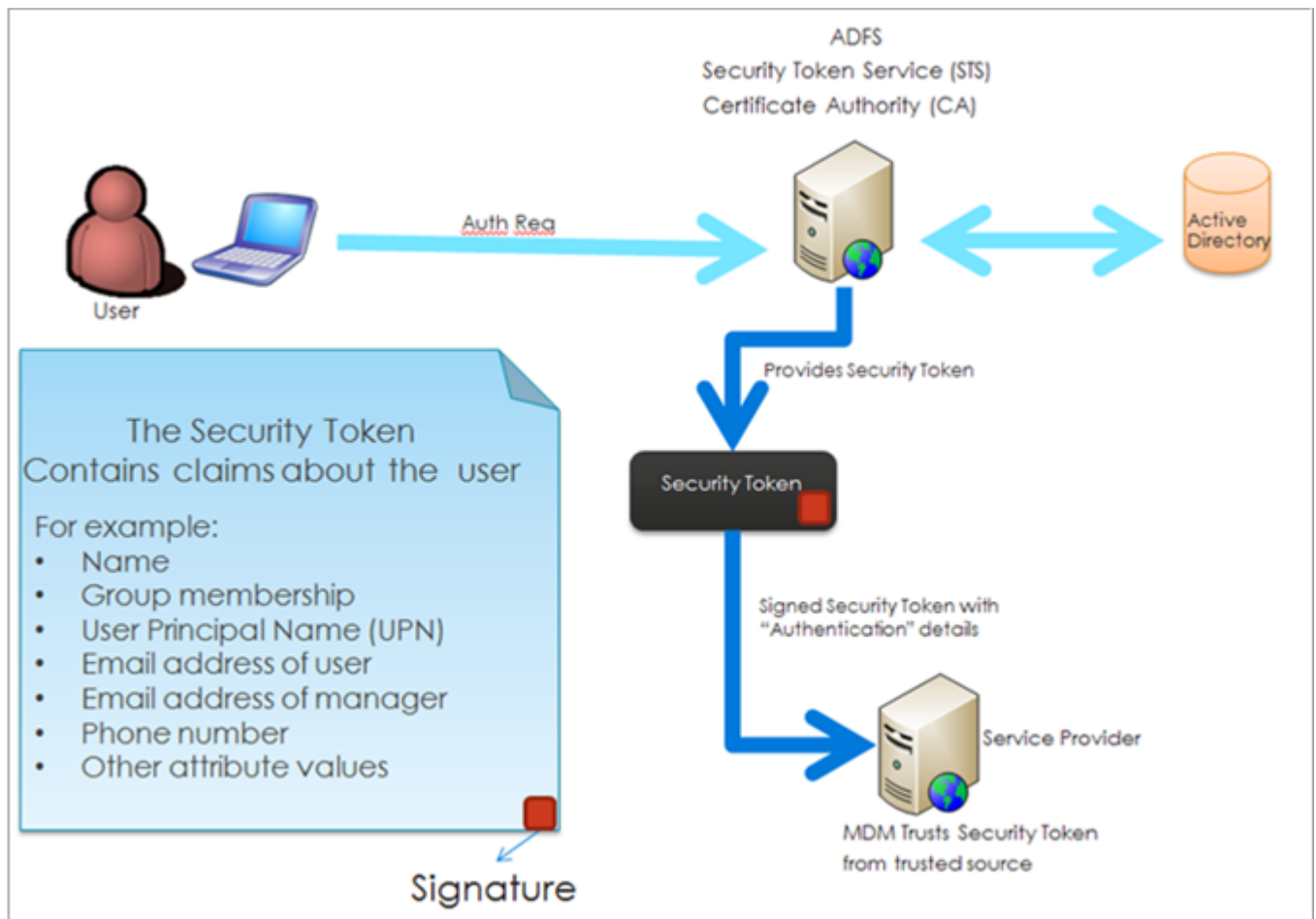
In ADFS, identity federation is established between two organizations by establishing trust between two security realms. A federation server on one side (the Accounts side) authenticates the user through the standard means in Active Directory Domain Services and then issues a token containing a series of claims about the user, including its identity. On the

Resources side, another federation server validates the token and issues another token for the local servers to accept the claimed identity. This allows a system to provide controlled access to its resources or services to a user that belongs to another security realm without requiring the user to authenticate directly to the system and without the two systems sharing a database of user identities or passwords.

ADFS integrates with Active Directory Domain Services, using it as an identity provider. ADFS can interact with other Web services and SAML 2.0-compliant federation services as federation partners.

The [Figure 14](#) displays the high level authentication of ADFS.

Figure 14: ADFS High Level Concept



Perform the following steps to enable ADFS authentication provider in MDM:

- 1 Open web.xml from `<MDM_Install_Dir>\web\mdmclient\WEB-INF` and add adfs-authentication-provider.xml under contextConfigLocation section as below.

Generally, it not required to modify this file as it is pre-configured.

```
<context-param>
  <param-name>contextConfigLocation</param-name>
```



```
<param-value>/WEB-INF/spring/adfs-authentication-provider.xml</
param-value>
</context-param>
```

- 2 Open adfs.properties file from `<MDM_Install_Dir>\web\mdmclient\WEBINF\spring\config` and configure the entries for the following properties:
 - adfs.metadataURL—Metadata URL
 - adfs.metadataURL.requestTimeout—Timeout, (if metadata url is used)
 - adfs.jks.path—Java Key store path
 - adfs.jks.password—Java Key store password
 - adfs.jks.private.key.name—Private Key name
 - adfs.jks.private.key.password—Private Key password
 - adfs.jks.defaultKey.key—Default key value
 - adfs.sp.metadata.entityId—Metadata entity Id
 - adfs.logout-success-url—Logout success URL
- 3 Restart servers.
- 4 Navigate to `http://HOSTNAME:PORT/mdm/ URL` and page will be redirected to configured IDP login page. Enter authentication information from IDP and the response will be redirected to MDM page.

HTTP Strict Transport Security (HSTS)

HTTP Strict Transport Security (HSTS) is a web security policy mechanism that allows web servers to declare that web browsers should interact with it using only HTTPS connections, and not the insecure HTTP protocol used alone. HSTS is an IETF standards track protocol and is specified in RFC 6797.

When you type in your MDM's URL, enter `https://mdm.example.com`? If you omit the https protocol, you are potentially vulnerable to Man in the Middle attacks. Even if the URL performs a redirect to `https://mdm.example.com`, a malicious user could intercept the initial HTTP request and manipulate the response (that is redirect to `https://mimdm.example.com` and steal their credentials).

Many users omit the https protocol and this is why HTTP Strict Transport Security (HSTS) was created. Once `mdm.example.com` is added as a HSTS host, a browser can know ahead of time that any request to `mdm.example.com` should be interpreted as `https://mdm.example.com`. This greatly reduces the possibility of a Man in the Middle attack occurring.

Note: In accordance with RFC6797, the HSTS header is only injected into HTTPS responses. In order for the browser to acknowledge the header, the browser must first trust the CA that signed the SSL certificate used to make the connection (not just the SSL certificate).

One way for a site to be marked as a HSTS host is to have the host preloaded into the browser. Another is to add the "Strict-Transport-Security" header to the response. For example, the

following would instruct the browser to treat the domain as an HSTS host for a year (there are approximately 31536000 seconds in a year).

```
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

The optional includeSubDomains directive instructs Spring Security that subdomains (i.e. secure.mybank.example.com) should also be treated as an HSTS domain.

As with the other headers, MDM Spring Security adds HSTS by default. You can customize HSTS headers with the <hsts> element as shown below:

Open <MDM_INSTALL_LOCATION>\web\mdmclient\WEB-INF\spring\platform-authentication-provider.xml (or configured authentication provider)

```
<http>
<!-- .... -->
  <headers>
    <hsts>
      include-subdomains="true"
      max-age-seconds="31536000"/>
    </headers>
  </http>
```

OAuth2 Authentication

OAuth is an open standard for access delegation, commonly used as a way for Internet users to grant websites or applications access to their information on other websites without giving them the passwords. This mechanism is generally used to permit the users to share information about their accounts with third party applications or websites.

Generally, OAuth provides to clients a "secure delegated access" to server resources on behalf of a resource owner. It specifies a process for resource owners to authorize third-party access to their server resources without sharing their credentials. Designed specifically to work with Hypertext Transfer Protocol (HTTP), OAuth essentially allows access tokens to be issued to third-party clients by an OAuth2 authorization server, with the approval of the resource owner. The third party then uses the access token to access the protected resources hosted by the MDM server.

Typically, it is best practice to use JSON Web Token (JWT) instead of http basic authentication for authentication. In MDM JWT token-based authentication, the OAuth2 server creates JWT token with a secret and sends the JWT token to the client. The REST client can store the JWT (usually in local storage) and includes JWT in the header with every MDM REST API request. The MDM server would then read and validate the JWT with every request from the client and sends back the MDM API response.

OAuth2 is enabled by default for new JSON REST webservice framework.

Pre requisite

- OAuth2 server is required for token generation.
- MDM username and OAuth2 username should be identical.

Supported Servers

Oauth2 tokens can be obtained from IAM server like key clock server.

Configuration Parameters

To enable Oauth2 authentication for REST webservice in MDM, open oauth2.properties file from `<MDM_Install_Dir>\web\mdmclient\WEB-INF\spring\config` and configure the entries for the following properties:

Table 9: OAuth Configuration Parameters

Property	Description
oauth2.issuer	The JWT claims set is validated to ensure the token is not expired and matches the expected issuer. The issuer specified in this parameter is matched with the one available in JWT claim name "iss".

Limitations

- Only new REST based web service will be supported for Oauth2 authentication.
- Customer will be providing an OAuth2 identity provider which can generate JWT tokens from token endpoint.
- Token refresh must be taken care by the REST client.
- Only username will be obtained for JWT token, authentication data will be obtained from MDM.
- If Key source is not available, then error will be logged.
- Signature will be verified using configured JWKS URL.
- If Bearer token not available in request header exception will be thrown.
- If there is mismatch in issuer, respective exceptions will be thrown.
- If token is expired or invalid or corrupted or tampered, then "Invalid Jwt" exception will be thrown details message will be logged.
- On successful authentication "ROLE_AUTHENTICATED_JWT" role will be auto assigned.
- User name will be picked from "user_name" claim available in JWT.
- Supported Grant type is only "Password Credentials"

Troubleshooting

1 Problems related to spring security issues:

To debug spring security related issues, perform the follow steps:

- Stop all servers.
- Navigate to web.xml from `<MDM_Install_Directory>\web\mdmclient\WEB-INF` and uncomment the below section:

```
<context-param>
  <param-name>log4jConfigLocation</param-name>
  <param-value>/WEB-INF/spring/config/log/log4j.xml</param-value>
```

```
</context-param>
<listener>
    <listener-class>org.springframework.web.util.Log4jConfigListener</
listener-class>
</listener>
```

- c Clear cache and restart all servers.

The spring security log is available at

<MDM_Install_Directory>\web\mdmclient\WEB-INF\spring\config\log\mdm-spring-security-detail.log

- d The default log location and name can be modified by configuring springAppender section in <MDM_Install_Directory>\web\mdmclient\WEB-INF\spring\config\log\log4j.xml

It is recommended to comment above section after debugging in production environment.

2 MDM application server not having access to IDP triggers the "Metadata Provider Exception" issue in SAML authentication.

If the following error occurs in SAML authentication like "No IDP was configured, please update included metadata with at least one IDP] with root cause" then the root cause, maybe the MDM application server doesn't have network access to the IDP server.

Sample error log:

" org.apache.catalina.core.StandardWrapperValve.invoke Servlet.service() for servlet [jsp] in context with path [/mdm] threw exception [javax.servlet.ServletException: org.opensaml.saml2.metadata.provider.MetadataProviderException: No IDP was configured, please update included metadata with at least one IDP] with root cause org.opensaml.saml2.metadata.provider.MetadataProviderException: No IDP was configured, please update included metadata with at least one IDP"

To fix this issue make sure all IDP URL's mentioned in the metadata file/URL are accessible from the MDM Application server machine.

3 SSO with SSL: You have configured SSL into your MDM setup using SSL Development Setup procedure and wants to configure SSL in MDM setup with SSO

- Ensure that you update the protocol from http to https in the ACS url which you have configured in the SSO solution using Ping/Oktta.
- Open web.xml from /web/mdmclient/WEB-INF. Comment the line platform-authentication-provider.xml and add saml-authentication-provider-ssl.xml under contextConfigLocation section as below:

```
<context-param>
<param-name>contextConfigLocation</param-name>
<param-value>/WEB-INF/spring/saml-authentication-provider-
ssl.xml</param-value>
</context-param>
```

- Edit samlssl.properties present in <MDM_INSTALL_HOME>/web/mdmclient/WEB-INF/spring/config/ and do the following changes:
 - saml.metadataURL= Metatadata URL
 - saml.metadataURL.requestTimeout=Timeout (if metadata url is used)

- saml.jks.path= Java Keystore path
- saml.jks.password= Java Key store password
- saml.jks.private.key.name= Private Key name
- saml.jks.private.key.password=Private Key password
- saml.jks.defaultKey.key= Private Key Value
- saml.sp.metadata.entityId= Metadata entity Id (grabbed from Ping Identity application)
- saml.serverName = Server Name
- saml.contextPath=/mdm (/mdm or /rdm based on your instance)
- saml.logout-success-url= Logout success URL
- oauth2.issuer=Oauth2 Issuer URL

4 Issue with Ping Federate using ACS url with default Port 80

When you provide an ACS url in Ping which uses default port 80, sometimes Ping fails to recognize the URL using this port and an error message displays as "Invalid ACS url".

To fix this issue, you can configure nginx to listen at a distinct port/address, access the MDM url at that port and configure the ACS url in Ping accordingly.

To configure nginx to listen at port 85, check [Configure Nginx Port](#).

5 Issues with SSL

- **If you face an issue with setting up of openssl and running the openssl command prompt**

You can use gitbash for generation of certificates as openssl which comes with the gitbash package. You can use version of gitbash version git version 2.27.0.windows.1 and openssl version OpenSSL 1.1.1g 21 Apr 2020.

- If you are running command from gitbash, you should take care of the slashes as its opposite to how command prompt works. Below are the sample commands for generation of Nginx certs through gitbash.

Note: Commands run from the location where mdm.cnf file is located and are of linux based convention, so use ./

- openssl genrsa -out ./mdm.key.pem 2048
- openssl req -new -x509 -key ./mdm.key.pem -config ./mdm.cnf -out ./mdm.csr.pem
- openssl x509 -in ./mdm.csr.pem -CA C:/opt/teradata/certs/ca-int.crt.pem -CAkey C:/opt/teradata/certs/ca-int.key.pem -CAcreateserial -CAserial C:/opt/teradata/certs/ca-int.mdm.srl -days 750 -out ./mdm.crt.pem
- cat ./mdm.crt.pem C:/opt/teradata/certs/ca-int.crt.pem > ./mdm-bundle.crt.pem
- Add the generated certificate into your java KeyStore : Open gitbash as administrator in the location %JAVA_HOME%\lib\security and run the command:
keytool -import -alias ca-root1 -keystore cacerts -file C:/etc/mdm/mdm.crt.pem

Note: Create etc and opt folder in C drive.

6 Q. SSO timeout session redirects to the MDM login page instead Windows login page

- Customize the SAML:
 - Update `saml.logout-success-url` with Single Logout Service URL (slo) in `saml.properties` file.
 - Update URL for logout as stored in table `MU_LEFT_NAV_URL` to:
`/saml/logout?local=true`.

You can try to run the below sql on your database:

```
update MU_LEFT_NAV_URL SET PAGE_URL='/saml/logout?local=true' where  
PAGE_NAME='logout';
```



To preserve previous data, it is recommended to take a backup of the original table and run the SQL on the new table,

APPENDIX E Install Nginx as Windows Service

What's In This Appendix

This appendix provides information about installing nginx as a Windows Service. MDM.

Topics include:

- “Create Nginx as Windows Service”

Create Nginx as Windows Service

The following section provides the steps to create nginx as windows service:

- 1 Download WinSW-x64.exe file from the location <https://github.com/winsw/winsw/releases/download/v2.12.0/WinSW-x64.exe> and copy the file to nginx home location.
- 2 Rename WinSW-x64.exe as nginxservice.exe
- 3 Save below content as nginxservice.xml. (update c:\nginx as per your path):

```
<service>
<id>nginx</id>
<name>nginx</name>
<description>nginx</description>
<executable>c:\nginx\nginx.exe</executable>
<logpath>c:\nginx\</logpath>
<logmode>roll</logmode>
<depend></depend>
<startargument>-p</startargument>
<startargument>c:\nginx</startargument>
<stopexecutable>c:\nginx\nginx.exe</stopexecutable>
<stopargument>-p</stopargument>
<stopargument>c:\nginx</stopargument>
<stopargument>-s</stopargument>
<stopargument>stop</stopargument>
</service>
```

- 4 Open command prompt and execute below commands to install windows as service:

```
cd <nginx_path>
nginxservice.exe
```


APPENDIX D **Configure Nginx Port**

What's In This Appendix

This document describes the steps for configuring nginx port when the defined port 80 for nginx is not free to use.

Topics include:

- [Configure Nginx Port Except Port 80](#)

Configure Nginx Port Except Port 80

The default port of Tomcat server is 8080 and the default port for shared services are 8103, 8116 and 8115. Ensure that the ports for Tomcat server and shared services are available before starting the installation process. If any of these ports are already in use or cannot be opened due to security restrictions, Teradata recommends you to consult the network team for assistance.

To modify the port numbers, you need to make changes in the service properties file of the services and the nginx configuration file to listen on alternative ports.

Follow below steps to configure nginx port if port 80 is not available to use:

- 1 Navigate to the nginx installation folder and open to edit nginx.conf under conf folder.
- 2 Change the port number from **listen 80;** to **listen 9090;**
- 3 Append:9090 to the following lines
 - proxy_set_header Host \$host:9090;
 - proxy_set_header X-Real-IP \$remote_addr:9090;
 - proxy_set_header X-Forwarded-For \$remote_addr:9090;
 - proxy_set_header X-Forwarded-Host \$host:9090;
 - proxy_set_header X-Forwarded-Server \$host:9090;
- 4 Restart nginx servers.
- 5 Add the port to the url in oauth2.properties file.
- 6 In application.properties file for shared service, add the port name to server.hostname as below:
`server.hostname=${tdaa.proxy.hostname:localhost}:9090`
- 7 Change the port in all script.js files in covalent folder from 8080 to 9090.

- 8 Add the port to `oauth.token.endpoint` in `td-workflow-connected-identity.properties` file in `Installer\cfg`.
For Collapsed setup add the port to same file in `Installer\web\mdmclient\WEB-INF\bcm\cfg`.
- 9 Add the port number '9090' in the '`connected.identity.base.url`' field of the '`td-workflow-connected-identity.properties`' file in `Installer\cfg`.
For Collapsed setup add the port to same file in `Installer\web\mdmclient\WEB-INF\bcm\cfg`.
- 10 Run `mkcoloc jar`.
- 11 Restart the servers.

Index

- A
 - Application Server
 - Startup 56
- D
 - Database
 - Geospatial Access Right 8
 - Preparing database for MDM 46
 - User requirements 4
 - Deployment
 - CRDM Application 62
 - Custom Application 62
 - Deployment Manager 62
 - Debug deployment process 69
 - Documentation xiv
- F
 - Fallback
 - Database Tables 40, 43
- H
 - Hardware Specification
 - Development Environment 1
 - Production Environment 2, 4
- I
 - Installing
 - MDM 28
- M
 - MDM
 - Backdown Procedure 87
 - DBC Object Details 41
 - Installation 28
 - Packaging Overview 15
- O
 - Override Changes
 - Overriding Scripts 84
- P
 - Purpose vii
- S
 - Starting
 - Server 56
 - System requirements 1
 - Windows 3, 82
- T
 - Temporal 88
 - Troubleshooting 52
- U
 - Upgrade
 - Process 72
 - Troubleshooting 88
- W
 - Web Services 52
 - Webclient Deployment
 - IBM WebSphere 48
 - Tomcat 50
 - War 91
 - WebSphere 48