TIBCO® Data Virtualization

Business Directory Guide

Version 8.8.1 | April 2025

Copyright © 2002-2025. Cloud Software Group, Inc. All Rights Reserved.

Contents

Contents **2**

Install Business Directory **6**
Business Directory Requirements 6
Business Directory Limitations 10
Installing Business Directory 10
Preparing for Installation on UNIX 11
Running the Business Directory Installer for Windows and UNIX 11
Setting the Business Directory Server to Start Automatically on UNIX 13
Running Silent Mode Installation 15
Keystore and Truststore Files for Business Directory 20
Setting up a Keystore File for Business Directory 20
Customizing the Business Directory Server Startup Scripts on UNIX 22
Tips from an Expert if the Server Does Not Start 24
Tips from an Expert on Controlling Cipher Suite Information 24
Removing Business Directory Service Files on UNIX 25
Removing Files before Uninstalling on UNIX 25
Uninstalling Business Directory On UNIX 26
Business Directory Log and Out File Reference 26

Business Directory Introduction **30**
Overview of Business Directory 30
Configuring Business Directory 31
Starting and Stopping Business Directory Services 32
Configuring Published Studio Locations for use within Business Directory 33
Configuring the Business Directory Server 34
Configuring LDAP Access for Business Directory 36
Understanding How Business Directory Works with LDAP Domains and 37

Passwords
Defining LDAP Domain Access for Business Directory 38
Changing the Repository Password 39
How Business Directory Works with the composite Domain 41
How Business Directory Works with Escape Characters 42
Changing the Language from User Profile 42

Using Business Directory 43
Logging into Business Directory 43
Sites 44
Adding a Site 45
Removing a Site 46
Site Metadata 47
Editing a Site 49
Resources 49
Searching Resources 50
Refining the Search with Filters 50
Browsing and Watching Published TDV Resources 51
Adding Comments to Objects 53
Previewing Resource Data in Business Directory 54
Categories 55
Defining Categories and Values 55
Adding Bulk Classifications for Resources 55
Adding Categories to a Resource 56
Custom Properties 56
Defining and Editing Custom Property Groups and Properties 57
Defining and Editing Custom Groups and Properties 58
Specifying the Custom Property Value for a Resource Tab. 60
Resource Data Lineage 60
What Data Lineage Reveals 60
Viewing Data Lineage 61
Access Rights 62

Viewing Users on the Access Control Page 63
Managing User Access 64
Changing the Admin Password After Installation 65

Business Directory API and System Tables 66
Using the Business Directory REST API 66
Using Business Directory to Run the REST API Methods 66
Using cURL Commands to Run the REST API Methods 68
Setting Up Secure cURL Execution Example 68
Characteristics of Programmatic Use of the REST API 69
Backing Up and Restoring Business Directory Information 71
Backing Up Business Directory 71
Restoring Business Directory 72
Backing up Business Directory Encryption settings 74
Restoring Business Directory Encryption settings 75
Accessing Business Directory System Table Information 75

Business Directory REST API 77
Business Directory REST APIs 77
Categories 78
Comments 96
Configs 102
Data 105
Domains 109
Groups 118
Metadata 125
propertyGroups 129
Resources 152
Security 170
Session 173
Sites 176
userProfiles 188

Users 193
Watches 200

TIBCO Documentation and Support Services 209

Legal and Third-Party Notices 212

Install Business Directory

This topic describes the installation of Business Directory.

- Business Directory Requirements
- Business Directory Limitations
- Installing Business Directory
- Keystore and Truststore Files for Business Directory
- Customizing the Business Directory Server Startup Scripts on UNIX
- Tips from an Expert if the Server Does Not Start
- Removing Business Directory Service Files on UNIX
- Business Directory Log and Out File Reference

Business Directory Requirements

- A minimum of 500MB of free disk space before installation.
- Port number availability for TDV and Business Directory:
  - TDV ports

TDV Ports Default	Description
9400	Web services HTTP port <– port needs to be exposed for non SSL TDV http access
9401	JDBC, ODBC, and ADO.NET <– port needs to be exposed for non SSL TDV client access
9402	Web services HTTP SSL <– port needs to be exposed for SSL TDV http access
9403	JDBC SSL, ODBC SSL, and ADO.NET SSL <– port needs to

TDV Ports Default	Description
	be exposed for SSL TDV client access
9404	Default caching database port
9405	[reserved]
9406	Monitor Daemon
9407	Active Cluster - JGroups (when installed) <– used for internal cluster communication only (more details in ActiveClusterGuide.pdf), needs to be exposed if cluster nodes are in different networks though
9408	Repository
9409	Monitor (when installed)

– BD ports

Business Directory Ports Default	Description
9500	Web services HTTP port
9501	JDBC, ODBC, and ADO.NET
9502	Web services HTTP SSL <– expose this port to enable HTTP access
9503	JDBC SSL, ODBC SSL, and ADO.NET SSL
9504	[reserved]
9505	[reserved]
9506	Monitor Daemon

Business Directory Ports Default	Description
9507	[reserved]
9508	Repository
9509	Monitor (when installed)

For operating systems requirements, see the *TDV Installation and Upgrade Guide*, Operating System Support for Server topic.

TDV web-based products runs on these Web browsers:

Web Browsers	TDV Support	Notes
Microsoft Internet Explorer 10 and 11	Active	Business Directory, Deployment Manager and Web Manager support IE 11.0 on Windows 7 and IE 11.590 on Windows 10. Business Directory and Deployment Manager do not work in compatibility mode on IE, and compatibility mode must not be enabled for these applications.
Mozilla Firefox 56	Active	Business Directory, Deployment Manager, Web Manager and Web UI support Mozilla Firefox v94.0 on Windows 10. Business Directory, Deployment Manager, and Web Manager support Mozilla Firefox v94.0 on macOS Catalina. Monitor supports the same on Windows 10 and does not run on macOS
Chrome 62	Active	Business Directory, Deployment Manager, Web Manager and Web UI support Chrome v95.0 on Windows 10. Business Directory, Deployment Manager, and Web Manager support Chrome v95.0 on macOS Catalina.Monitor supports the same on Windows 10

Web Browsers	TDV Support	Notes
		and does not run on macOS
Safari 10.1	Active	Not supported for web service API calls. Not supported for TDV Web UI. Business Directory, Deployment Manager and Web Manager support v13.1.
Edge	Active	Business Directory, Deployment Manager, Web Manager and Web UI support v95.0 on Windows 10.

Monitor requires a Web browser running Adobe Flash Player Version 10 or greater.

For the Monitor and Deployment Manager client web applications to function properly, the machine that is running a compatible browser must be running on a machine with Windows 7 or higher. Occasionally the login screen for these web applications does not close automatically, you can close it and continue using the product or you can choose to run in a different browser.

For best results, when running Business Directory and Deployment Manager concurrently, use different browsers.

You can switch to another browser to resolve the issue.

Online help (and long lists in Manager) might not display as expected in Chrome. You can switch to another browser to resolve the issue.
The TDV and Business Directory servers require a secure connection. So when you first connect a browser to any TDV web-based application, you might get a warning about connecting to an untrusted site.

Depending on your browser:

- You might be asked to allow the connection process to continue.
- You might want to configure the browser to trust the site so that warning messages no longer appear. For some site configurations this might require configuration of SSL connections for your entire TDV environment.

TDV provides a configuration option that can be tuned to override the “Select a certificate” popup. In TDV Studio, choose Administration > Configuration > Server > Communications > Want Client Authentication.

Default value for this setting is True. When True, Server will send client certificate request for SSL authentication.

You can set this configuration to false to avoid the popup "Select a Certificate" in Web Manager. TDV server must be restarted to take effect.

Note: If you have mutual authentication configured in any of your published web services, setting this configuration to false will disable it.

To set up Business Directory for SSL communication with other TDV components, see Keystore and Truststore Files for Business Directory.

Business Directory Limitations

You can sometimes mix versions of Business Directory and TDV as follows.

BD Version	TDV Version	Support
older	newer	Not Supported
newer	older	Supported

For example:

- Business Directory 8.1, 8.2 clients are not compatible with published resources from TDV 8.3.
- The use of Business Directory 8.3 clients with published resources from TDV 8.2 is supported.

Installing Business Directory

This section describes how to install Business Directory:

- Preparing for Installation on UNIX

- Running the Business Directory Installer for Windows and UNIX
- Setting the Business Directory Server to Start Automatically on UNIX
- Running Silent Mode Installation

Preparing for Installation on UNIX

You can skip these steps if you are installing on Windows.

To prepare for installation on a UNIX computer

1. Make sure you have reviewed and completed any necessary preparation.
2. All Redhat OS Linux variants that have SELinux support can utilize it enabled (i.e. SELinux = enabled). If one wants to run TDV and/or Business Directory with SELinux enabled then an appropriate security policy that allows read/write access to the TDV installation directory and TDV ports is required before installation.
3. If necessary, log into the installation machine as a non-root user. Change your working directory to the user's home directory.
4. Run the following command for your platform:

   ```
   chmod 755 <installer file name>
   ```

   For example:

   ```
   chmod 755 install TIB_tdv_bd_<version>_linux_x86_64.bin
   ```

5. Make sure that you have READ and WRITE permissions on the installation directory.

Running the Business Directory Installer for Windows and UNIX

To run the installer

1. Make sure you have enough space for the temporary installation files. If you get an error, see the *TDV Installation and Upgrade Guide*, for how to resolve it.

2. Run the installer executable for your platform.

Platform	Instructions

UNIX	Run the following command
	`./<installer file name>`
	For example:
	`./TIB_tdv_bd_<version>_linux_x86_64.bin`

Windows	Double-click the following file: `TIB_tdv_bd_<version>_win_x86_64.exe` **Note**: TDV Version 8.3 installer and patch are packaged with VC++ redistributable and they will be installed along with TDV.

3. Follow the prompts on the screen.

   You will be prompted for the following passwords.

TDV Password Type	Description
BD Application Password	The BD application password is used to login to Business Directory.
Repository Password	Repository is the database that is used to store all of the data and metadata about the items that you create within TDV and BD. The Repository also stores your configuration and other environment settings. The Repository password is used to protect the Repository.

Note: TDV will generate a new encryption key when the installation is in a new destination. If it is an existing installation TDV uses the existing encryption key. In

case of any errors encountered, the administrator may have to investigate if the encryption key file location and content are correct and then contact TDV support team for assistance.

The installation process might take a few minutes, during which progress windows are displayed.

4. Finish to exit the installer when the installation is completed.

   The Server starts automatically at the completion of the installation process.

   Install and uninstall logs are called bitrock_installer_<number>.log while the installer is running. After installation is complete, the logs are named <product>_BD_install or <product>_BD_uninstall.log. The log files can be found in the following directories:

Platform	Default Location of Log Files
Unix	/tmp
Windows Server 2012, 2016, 2019, Windows 10.	%HOMEDRIVE%\Users\<username>\AppData\Local\Temp

5. Optionally, download and install the latest Business Directory patch as described in *TDV Release Notes*.

Setting the Business Directory Server to Start Automatically on UNIX

If at any time after installing the software, you restart the UNIX installation machine, Server and the metadata repository do NOT start automatically (unlike when they start automatically after a successful installation of the software).

To configure the Business Directory service files bd.repository and bd.server

1. Log into the installation machine as root.
2. Change the working directory to <bd_install_dir>/bin.
3. Run the following command as the root user:

   ```
   bd_install_services.sh
   ```

   This command prompts for a user name, and other details to install and configure the service files bd.repository and bd.server.
4. Enter the name of the user to start Business Directory (not the root user) and the other information requested.

   The script then installs bd.repository and bd.server into an appropriate location on the installation machine and configures them. The location will be printed on your screen when the configuration is successful, so make note of this location, because you need this to perform verification of the service files.

   Do not run the bd.repository or bd.server scripts in the <bd_install_dir>/bin/ directory. These are template files used by bd_install_services.sh only and are not meant to be run.

   Running bd_install_services.sh does not interrupt any repository or server processes that are running, but prepares the machine for automatically starting those processes during restart of the UNIX-based computer.

To verify the Business Directory service files configuration

5. Go to the location noted previously from running bd_install_services.sh.
6. Enter these commands:

   ```
   ./bd.repository restart
   ```

   ```
   ./bd.server restart
   ```
7. Optionally, you might need to restart your TDV Server.

If the machine is rebooted, the monitor, server, and repository processes should automatically start once the machine is ready to go.

Running Silent Mode Installation

You can install in silent mode. A silent mode installation does not require any user input at the time of installation to complete the installation process. It does not have a graphical user interface (GUI) but instead uses the values from a response file to perform the installation.

Topics covered include:

- Creating the Options File for a Silent Installation
- Running the Installer in Silent Mode

Creating the Options File for a Silent Installation

Optionally, when running a silent mode installation you can use an options file that has specific key-value pairs.

To create the options file for a silent install

1. In a text editor, create a options file similar to the following:

Business Directory	# Modify install directory and all port number references # mode=unattended install_directory=/opt/TIBCO/BD server_port=9500 repository_admin_password=password bd_admin_password=password

```
----------------------------------------
```

TDV Server

```
# Modify install directory and all port number references
#
mode=unattended
install_directory=/opt/TIBCO/TDV
server_port=9400
repository_admin_password=password
database_admin_password=password
server_admin_password=password
----------------------------------------
```

2. Edit the values within the file for your installation.

The following table describes the variables in the response file:

Variable	Description and Value
INSTALL_ DIRECTORY	Directory in which to install the software referred to as <TDV_ Installdir>. The value can be empty, or the directory can be non-existent. On UNIX, there can be no space in the directory name. Examples: `install_directory=/opt/TIBCO/TDV`

Variable	Description and Value
	install_directory=C:\Program Files\TIBCO\Studio install_directory=/opt/TIBCO/BD
REPOSITORY_ ADMIN_ PASSWORD	Password to access the repository database, which is automatically installed during the installation. PostgreSQL requires that the password you choose cannot contain a # or $.
SERVER_PORT	Defaults to 9400 for TDV and 9500 for Business Directory.
DATABASE_ ADMIN_ PASSWORD	The password used to access the default caching database, which is automatically created during installation. PostgreSQL requires that the password you choose cannot contain a # or $. This password is used only during TDV Server installation and is not required during the Business Directory installation.
SERVER_ADMIN_ PASSWORD	The password used to login to the web manager and the client applications.
BD_ADMIN_ PASSWORD	Business Directory Server Application password.

3. Save the file as <installer.properties>.

Running the Installer in Silent Mode

To run the installer in silent mode

4. Option file method:

   a. Create the options file. See Creating the Options File for a Silent Installation .

   b. Run the installer with the following option: <instFile>.exe/bin --optionfile <OPTION_FILE>

5. Command line (no options file) method - See examples below:

6. Run one of the following commands:

Component	Command Options
TDV Server	• Windows Installation with all input parameters: <instFILE>.exe --mode unattended --install_ directory <TDV_Installdir> --server_port "6400" --server _admin_password "admin1" --repository_ admin_password "password" --database_admin_ password "password" *Note*: database_admin_password is only valid for TDV Server. BD does not use this variable. • Windows Installation with only the required parameters: <instFILE>.exe --mode unattended --server _ admin_password "admin1" --repository_admin_ password "password" --database_admin_ password "password" This command installs TDV Server in the default directory C:\Program Files\TIBCO\TDV Server <version> on default port 9400. • Linux/AIX Installation with all input parameters: <instFILE>.bin --mode unattended --install_ directory <TDV_Installdir> --server_port "6400" --server _admin_password "admin1" --repository_ admin_password "password" --database_admin_ password "password" • Linux/AIX Installation with only the required parameters: <instFILE>.bin --mode unattended --server _ admin_password "admin1" --repository_admin_ password "password" --database_admin_ password "password" This command installs TDV Server in the default directory /opt/TIBCO/TDV_Server_<version> on default port 9400. *Note*: User should have rwx permissions on /opt
Business Directory	• Windows Installation with all input parameters: <instFILE>.exe --mode unattended --install_ directory “<TDV_Installdir>” --bd_admin_ password "admin1" --repository_admin_password "password" --server_port 9500 • Windows Installation with only the required parameters:

Component	Command Options
	<instFILE>.exe --mode unattended --bd_admin_password "admin1" --repository_admin_password "password" This command installs BD in the default directory C:\Program Files\TIBCO\BD Server<version> on default port 9500. • Linux/AIX Installation with all input parameters: <instFILE>.bin --mode unattended --install_directory “<TDV_Installdir>" --bd_admin_password "admin1" --repository_admin_password "password" --server_port 9500 • Linux/AIX Installation with only the required parameters: <instFILE>.bin --mode unattended --bd_admin_password "admin1" --repository_admin_password "password" The above command installs BD in the default directory: /opt/TIBCO/TDV_BD_Server_8.0 on port 9500 *Note*: User should have rwx permissions on /opt

Note: Run the command, <instFile>.exe --help for a list of all valid options.

The variables are as follows:

- <instFILE> is the file name. For example, TIB_tdv_bd_8.0.0_win_x86_64 for a Windows TDV Business Directory.
- <OPTION_FILE> is the name of the file where the input parameters are stored.

7. Verify that the installation was successful by looking for the TDV installation directory. You can also view success or failure messages in:
   - %HOMEDRIVE%\TDV_BD_install.log (Windows) or /tmp/TDV_BD_install.log (UNIX)
   - %HOMEDRIVE%\TDV_install.log (Windows) or /tmp/TDV_install.log (UNIX)
8. For CentOS, Red Hat Enterprise Linux, and Oracle Red Hat Enterprise Linux systems Security-Enhanced Linux (SELinux) must be enabled. See Configuring Security Enhanced Linux Environments, page 7 in the Security Features Guide.

Keystore and Truststore Files for Business Directory

For TDV components (Studio, all TDV server instances, and so on) to use SSL for communications among themselves, each component must have a keystore file containing its own SSL key, and a truststore file containing the SSL key for each authenticated component with which it will communicate. You can use keytool to generate a key for each component, or store an existing key that has a chain of Certificate Authority behind it. If you do not generate a key for a given component, it uses a self-signed certificate by default, which is unsafe.

The following procedure describes the setup procedure:

- Setting up a Keystore File for Business Directory

Setting up a Keystore File for Business Directory

The Java key and certificate management tool, keytool, is available for administering public/private key pairs and certificate authorities.

This topic describes how to set up a keystore file for Business Directory. A generalized description of the procedure for setting up keystore and truststore files for all TDV components is in the *TDV Administration Guide*.

To set up a keystore file for Business Directory

1. Verify that a key exists in the keystore you want to use:

   ```
   cd <BD_install_dir>/jdk/bin
   ```

   ```
   keytool -list -keystore <keystore_location> -storepass <password> -v > keystore.txt
   ```

   If no Business Directory key exists, obtain one or generate one using the keytool utility.

2. Search the text file for `Entry type: PrivateKeyEntry`.

   Make note of the value from `Alias name: <alias_name>`.

3. Copy the keystore to a directory of your choice.

4. Open <BD_install_dir>/bd/conf/server/server_values.xml in a simple editor like Wordpad.

   Note: If you open the file in Word, it will display an interpreted form of the XML file instead of the raw file.

5. Change the values of three attributes:

   Keystore Key Alias (On Server Restart)
   For example, the alias name might be cis_server

```
<common:attribute>
<common:name>/server/communications/KeystoreKeyAliasOnServerRestart</common:name>
<common:type>STRING</common:type>
<common:value>[alias_name]</common:value>
</common:attribute>
```

   Keystore File Location (On Server Restart)
   For example, C:/Program Files/TDV/BD 7.0/conf/server/security/cis_server_keystore.jks

```
<common:attribute>
<common:name>/server/communications/KeystoreLocationOnServerRestart</common:name>
<common:type>STRING</common:type>
<common:value>[keystore_location <fulldirectorypath>/mykeystore.jks]</common:value>
</common:attribute>
```

   Keystore Password (On Server Restart), which will be encrypted automatically when the server restarts

```
<common:attribute>
<common:name>/server/communications/KeystorePasswordOnServerRestart</common:name>
<common:type>PASSWORD_STRING</common:type>
<common:value>[unencrypted_password_string]</common:value>
</common:attribute>
```

6. If necessary, change the value of one other attribute:

   Keystore File Type (On Server Restart)
   For example, JKS or PKCS12

```
<common:attribute>
<common:name>/server/communications/KeystoreTypeOnServerRestart</common:name>
<common:type>STRING</common:type>
<common:value>[JKS]</common:value>
</common:attribute>
```

7. Restart Business Directory.

Customizing the Business Directory Server Startup Scripts on UNIX

The Business Directory installation provides a startup script for UNIX that you can customize for your own purposes; but to maintain customizations you might have made to this script across hotfix or patch updates, you must activate an environment variable.

Note: You can also use this functionality to add JRE VM arguments or run commands automatically before launching Business Directory.

The procedure below describes how to configure the startup script for UNIX.

To configure the Business Directory startup scripts for UNIX

1. Stop the Business Directory Server.
2. Navigate to the <bd_install_dir>/bd/conf directory.

   ```
   cd <bd_install_dir>/bd/conf
   ```

3. Copy script_env.sh.sample to script_env.sh.
4. Open script_env.sh with a text editor and uncomment the last two lines:

   ```
   # CIS_SERVER_VM_ARGS=
   ```

   ```
   # export CIS_SERVER_VM_ARGS
   ```

5. Change the value of CIS_SERVER_VM_ARGS to include all the values in VM_ARGS from <bd_install_dir>/bin/bd_server.sh.

   Retrieve the platform-specific value of VM_ARGS from <bd_install_dir>/bin/bd_server.sh. Locate the line for your platform:

   Make sure you add double-quotes around the value specified for CIS_SERVER_VM_ARGS.

   If you want to run an executable or command-line utility at this point, make sure the command returns control to bd_server.sh. If it does not, Business Directory does not start correctly.

   Each command you add should be on a new line.

6. Start the Business Directory Server.
7. Check the end of the newest <bd_install_dir>/logs/cs_bd_server.out.<timestamp> file to ensure the script environment functionality is working.

Tips from an Expert if the Server Does Not Start

If the TDV server does not start and the log files indicate that the cause is not enough heap memory, you can modify the default max memory setting.

To modify the max memory setting

1. Stop the server.
2. Increase the MAX_MEMORY value in the one of the following location depending on your server:
   - <TDV_install_dir>/conf/server/server.properties
   - <bd_install_dir>/bd/conf/server/server.properties
3. If adjusting the heap size with MAX_MEMORY is not enough to allow large CAR files to load, you can try setting the following Studio configuration parameters back to their default values:
   - Default Bytes to Fetch—Default value is 100.
   - Default Rows to Fetch—Default value is 1000.
4. From the process manager for your platform, shut down and restart all TDV processes (such as the TDV Server and monitor).

Tips from an Expert on Controlling Cipher Suite Information

By default, Business Directory displays information about cipher suites that are a part of your data sources. You can control this behavior by editing TDV configuration parameters through the TDV Studio user interface.

To modify the SSL cipher suites setting

1. Stop the server.
2. Open Studio > Administration > Configuration.
3. Locate the Disabled Cipher Suites for SSL Connectors configuration parameter.
4. Adjust the value.

5. Sets the cipher suites to disable when creating an SSL connector.

   The string should be a comma separated list of cipher suites, that can be fed by the exact cipher suite name used in the JDK or by using a regular expressions.

   Default values are ".*NULL.*", ".*RC4.*", ".*MD5.*", ".*DES.*", ".*DSS.*" for eliminating old/insecure/anonymous ciphers.

   Removing the string will cause the default JRE settings to take effect.

6. From the process manager for your platform, shut down and restart all TDV processes (such as the TDV Server and monitor).

Removing Business Directory Service Files on UNIX

You can use the bd_remove_services.sh script from a command line to uninstall the Business Directory services files that are used to restart the server and repository automatically on UNIX. This command does not interrupt any repository or server processes that are running, but removes the Business Directory Service files.

This section includes the following:

- Removing Files before Uninstalling on UNIX
- Uninstalling Business Directory On UNIX

Removing Files before Uninstalling on UNIX

Before you uninstall Business Directory, remove the service files from the installation machine, because the uninstaller does not remove these files automatically.

To remove the service files bd.repository and bd.server

1. Log into the installation machine as root.
2. Change the working directory to <bd_install_dir>/bin.
3. Run the following command:

```
bd_remove_services.sh
```

4. Navigate up the directory structure and delete <bd_install_dir>.

Uninstalling Business Directory On UNIX

During the uninstallation process, all the components from the previous installation are removed. You cannot uninstall the components individually.

To remove the Business Directory service files

1. Log into the installation machine as the user that installed the software.
2. Run the following command:

   ```
   <bd_install_dir>/uninstall
   ```

   For a silent uninstall, a message is displayed indicating that the uninstall is occurring. For an interactive uninstall, go to step 3.
3. Press the Enter key.

   You will see a warning about loss of data.
4. Press the Enter key to complete the uninstallation process and leave the uninstaller.
5. Make sure to delete <bd_install_dir>.

Business Directory Log and Out File Reference

The log files for Business Directory are stored in <BD_install_dir>\logs. The table below lists the BD log files. For the most accurate and current list of files, see the *TDV Administration Guide* chapter *About Log Files*.

File Name	Description
cs_bd.out	Business Directory log. Lists the current user and actions specified by VM_ARGS, such as installing, starting, stopping, and uninstalling Windows services, the BD repository, and the BD Monitor Daemon.

File Name	Description
cs_bd_csmonitor_daemon.log	Tracks the Business Directory Monitor if it is running as a daemon.
cs_bd_bundles.log	Tracks the activity of Business Directory when they are installed as a bundle.
cs_bd_cluster.log	Records log messages related to the fact that clustering option is not installed in BD. This file resides in the cluster directory under the logs directory. The Cluster Logging Detail Level parameter in log4j2.properties determine what to include.
cs_bd_csmonitor_collector.log	Log for the TDV Monitor Server (which is distinct from the Monitor Daemon process, MonitorBoot). Collectors hosted within the monitored TDV instances periodically take snapshots of the current state of the host instance, and keep track of general activity such as requests, sessions, transactions, and events.
cs_bd_csmonitor_server.log	Tracks Business Directory Monitor activities.
cs_data_cache-<day>.log	Data cache logs, each with a 3-letter day of the week (Mon, Tue, Wed, and so on) in its name. These reside in the cs_data_cache directory under the logs directory.
cs_bd_monitor_events.log	Monitor Daemon events log. Records the categories of events selected through configuration parameters.
cs_bd_monitor.log	Monitor Daemon main log. If the Server does not start or stops responding, this log and cs_bd_server.log are the files to check for errors.
cs_bd_monitor.out	Combines stdout and stderror for the Monitor Daemon (MonitorBoot) process. Any thread dumps of the Monitor Daemon process are written to this file.

File Name	Description
	Note: In the windows platform, additional log files are created with a numeric suffix that increments by 1 (for example cs_bd_monitor.out1, cs_bd_monitor.out2, etc), once a defined limit is reached. This limit is specified in the file <BD_Install_ Dir>/bd/conf/monitor/wrapper.conf The default limit is 10MB.
cs_repository-<day>.log	Repository logs, each with a 3-letter day of the week (Mon, Tue, Wed, and so on) in its name. These reside in the cs_repository directory under the logs directory. These files record repository database events and status for TDV and BD. **Note:** If you use “composite.sh monitor stop” from the command-line, ServerBoot posts the message, “LOG: could not receive data from client: No connection could be made because the target machine actively refused it.” Because ServerBoot is a child process, it cannot be prevented from posting this message. However, if this message is not logged twice in a row, No error has actually occurred.
cs_bd_server_ client.log	Log of activities of BD Server clients.
cs_bd_server_ dsrc.log	Log of data source functionality.
cs_bd_server_ events.log	Server events log. Records the categories of events selected through configuration parameters.
cs_bd_server_file_ cache.log	Tracks Monitor file-cache activities.
cs_bd_server_ metadata.log	Records what objects are being written to the repository, or changes to it.

File Name	Description
cs_bd_server_ status.log	Server Status log files can be used to determine software license conformance and help with corporate asset management. This log keeps data from each server session that is initiated.
cs_bd_server_ task.log	Supplements cs_bd_server.log with exceptions that occur outside of the main execution thread (for example, in background threads).
cs_bd_server.log	Main log. Nearly every error that occurs is logged here. A notable exception is unexpected Server crashes. If the Server does not start or stops responding, this log and cs_monitor.log are the files to check for errors. This file also includes data source type and version information.
cs_bd_server.out	Standard output and error log for BD Server (BDServerBoot) processes. Any thread dumps of a ServerBoot process are written to this file.
cs_bd_tools.log	Tracks the errors that occur from the command line utilities (for eg. bd_encryption_util.sh, bd_server.sh, etc.).

Business Directory Introduction

This section describes how to use Business Directory regardless of what type of Business Directory user you are.

- Overview of Business Directory
- Configuring Business Directory
- Configuring LDAP Access for Business Directory
- Changing the Repository Password
- How Business Directory Works with the composite Domain
- How Business Directory Works with Escape Characters
- Changing the Language from the User Profile

Overview of Business Directory

Business Directory is a business-friendly interface that provides a catalog of the published resources contained in one or more instances of TDV. The interface provides a seamless way to communicate and distribute information about data across groups within an organization to facilitate collaboration. Business Directory facilitates collaboration, and supports governance of data as it moves through its life cycle.

Core Capabilities	Supporting Features
Governance	Resource lineage can be displayed. Metadata information can be added to the resources and their parent containers. That metadata is saved only within the Business Directory.
Collaboration	Business Directory works across all published resources in a TDV instance, and across any number of TDV instances in an enterprise.

Core Capabilities	Supporting Features
	Resources within the Business Directory can be commented on. Resources and their parent containers can be followed, allowing for passive communication of resource updates.
Communication	Shared instances of TDV outside of team boundaries. Browse and analyze database resources from disparate groups for potential usage. Carry out simple searches and apply post-search filters. Communicate with other stakeholders. Provide feedback to developers.

Depending on your role and relationship to the data, the value of Business Directory may differ. You could find value simply from knowing what data is available within your larger organization so that you can optimize its usage for your purposes. Or you could provide the metadata information that links the raw data to the business decisions that are critical to the success of your organization.

Exactly how you and your teams implement and use Business Directory will be up to you. This document strives to cover all the potential tasks that you need to understand when creating the custom workflows for your organization.

Configuring Business Directory

Business Directory has a server and repository that are separate from your TDV Server and repository. Installation of Business Directory starts them automatically for you.

- Starting and Stopping Business Directory Services
- Configuring Published Studio Locations for use within Business Directory
- Configuring the Business Directory Server

Starting and Stopping Business Directory Services

The Business Directory Server and Repository are started automatically after installation and set to restart automatically when the server machine is restarted.

To start the Business Directory Services

1. Navigate to <bd_install_dir>/bin.
2. Run one of the following from the command line, depending on your platform:

Component	Platform	Command
Server	UNIX	bd_server.sh [run|stop]
	Windows	bd_server.bat [run|stop]
Repository	UNIX	bd.sh repo [start|stop|restart]
	Windows	bd.bat repo [start|stop|restart]

   Or use the Windows Task Manager Services tab to start or stop any of the BD services.

3. Log into the Business Directory web application to validate that the Business Directory server started. See Logging into Business Directory.

Tips From an Expert on the Sequence to Start/Stop Business Directory Services

- In general, to stop the TDV processes - Stop the TDV Server first, followed by TDV Repo and then TDV Cache. When Starting the TDV processes, start the TDV Cache first, followed by TDV Repo and then TDV Server.

  Run these scripts in the given sequence:

  Stop BD in WINDOWS:

  bd_server.bat stop

  bd.bat repo stop

Stop BD in UNIX:

bd_server.sh stop

bd.sh repo stop

Start BD Server in WINDOWS:

bd.bat repo start

bd_server.bat start

Start BD in UNIX:

bd.sh repo start

bd_server.sh start

Note: If you stop cache before stopping the server, it is possible that the server could still try to access the repo and you may encounter errors.

- To stop TDV BD Service in Windows, open the Windows services and stop the Windows Service "TDV BD Server <version>" fallowed by "TDV BD Repository <version>". To start the TDV BD Service, open the Windows services and start the Service "TDV BD Server <version>"

Configuring Published Studio Locations for use within Business Directory

Business Directory cannot display resources published to locations that have slashes in their names.

To configure your Studio data sources for use with Business Directory

1. Open Studio.
2. Navigate to the database or web service portion of the Studio navigation tree.
3. Validate that the name of the database or web service container that owns the resources you want to see displayed within Business Directory contains no slashes (/).

4. Remove slashes (/) from the names of any published database or web services that you want to be able to view within Business Directory.

Configuring the Business Directory Server

Using Business Directory requires some minimal configuration of the server. This server configuration is simply to allow for email notifications.

To configure the Business Directory server

1. Start Business Directory. See Logging into Business Directory.
2. Navigate to Admin > Server Configuration.
3. Select or type values for the following:

Field	Description
Refresh on server start-up	Enable or disable the reintrospection of each site defined within Business Directory when the server is started. Depending on how many sites are registered within Business Directory, you might consider disabling this for better performance.
SMTP Server address and Port number	Simple Mail Transfer Protocol (SMTP) address, which is typically a server name, and the port, which defaults to 25.
SMTP Authentication Required	Enable or disable this requirement. If enabled, you must enter values for the SMTP user name and password.
SMTP Authentication User Name	
SMTP Authentication User Password	
Send watch notifications from this email address	Email address that you want assigned and attributed to the messages sent from Business Directory to the people who have signed up to watch various

Field	Description
	resources.
Server max memory size (in KB)	The Current value is a read-only field that displays the current setting. Use the On Server Restart value field to select a new value to set max memory at after the Business Directory server is restarted.
Keystore File Location	Read-only fields used to alert you to the location of the TDV Server keystore and truststore files.
Truststore File Location	
Keystore File Location (On Server Restart)	The location of the keystore/truststore file used in SSL authentication to establish the identity of the server to external clients. The file must contain exactly one entry (i.e. private key/certificate pair). It may also contain zero or more trusted certificate entries from trusted certificate authorities that are used to validate the certificates that are presented by external clients.
Truststore File Location (On Server Restart)	
Keystore Alias	The alias name of the key entry (i.e public certificate) used in SSL authentication to establish the identity of the server to external clients.
Keystore Alias (On Server Restart)	The alias name mentioned here will take effect on the next server start.
Keystore Type	The type of the keystore/truststore file. It must be a valid type such as "JKS" or "PKCS12".
Truststore Type	
Keystore Type (On Server Restart)	The file type mentioned here will take effect on the next server start.
Truststore Type (On Server	

Field	Description
Restart)	
Keystore Password (On Server Restart)	The password of the keystore/truststore file and the entries within it. All password protected entries in the file must use the same password as the file itself.
Truststore Password (On Server Restart)	

4. Click Save.

Configuring LDAP Access for Business Directory

Business Directory automatically imports the names of all of the domains, users, and groups from the TDV sites to which it connects. This includes any LDAP server information being used by the TDV site.

Note: Azure Domain is not supported in Business Directory.

LDAP servers are created when adding a TDV Site to the Business Directory that is using LDAP. The user credentials are stored in the domain server and the domain server needs to be available for them to be verified.The connection information is imported from the TDV Site, with the exception of the password.

You can restrict what resources Business Directory users see, by using Studio. LDAP or TDV user profiles. Typically, if you define the user profile through Studio so that a particular user can not see a resource, then that same user in Business Directory would not be able to see that resource. For example, LDAP user profile MMEG in Studio can see compositeView, but not SWsalesView. When user MMEG logs into Business Directory, SWsalesView would not be visible through the web application.

For more information, see LDAP Domain Administration in the *TDV Administration Guide* topic.

All TDV sites that are connected to Business Directory must share the same LDAP server.

This section includes:

- Understanding How Business Directory Works with LDAP Domains and Passwords
- Defining LDAP Domain Access for Business Directory

Understanding How Business Directory Works with LDAP Domains and Passwords

LDAP domains are imported automatically if they exist for the TDV sites that you are exposing through Business Directory. Multiple LDAP domains in different TDV instances are consolidated into one LDAP domain in Business Directory.

During a site refresh, new LDAP users and groups are updated to Business Directory, but none are deleted. For example, if an LDAP domain is deleted from within TDV, after refreshing this site in Business Directory, this LDAP domain is not removed from Business Directory.

Password information is never transferred to the Business Directory server. When the Business Directory server needs to authenticate a user in the composite domain of a TDV site, it uses that TDV site to authenticate the user.

When the Business Directory needs to authenticate a user in an LDAP domain, it relies on the LDAP server to authenticate that user. Authentication of LDAP users in Business Directory is done directly against the LDAP server, without involving the TDV site, using the original LDAP user name.

When you import LDAP server information from a remote TDV server, the Business Directory imports everything needed to connect with the LDAP server except the following:

- LDAP connection password
- ldap.properties file

Depending on how many different instances of TDV you plan to exposed through Business Directory, the domains that you can use to log into Business Directory varies as follows:

For use with	Domains you can use
A single TDV	• The composite domain

For use with	Domains you can use
site	• Separately defined LDAP domains • A combination of the composite and LDAP domains
Multiple TDV sites	The imported LDAP domains from the TDV sites that you define within Business Directory

Defining LDAP Domain Access for Business Directory

If the TDV sites that you want to expose within Business Directory have LDAP users defined, you must complete their configuration.

If you want to use LDAP domains with Business Directory, you should configure the ldap.properties file. You should also use the properties file to indicate whether you want permissions granted to nested groups.

The following instructions provide one method for completing the configuration using an ldap.properties file that is distributed with TDV.

To configure the necessary LDAP access for Business Directory

1. Locate your TDV LDAP properties file in the following directory:

   ```
   <CIS_install_dir>/conf/server/ldap.properties
   ```

2. Copy the file to the following Business Directory location:

   ```
   <bd_install_dir>\conf\server\ldap.properties
   ```

3. Edit the file as necessary. For details, see LDAP Domain Administration in the *TDV Administration Guide* topic.
4. Start Business Directory. See Logging into Business Directory.
5. Add TDV sites to Business Directory.

   The following things happen:

- The LDAP connection information that the TDV site uses is imported into the Business Directory server. This includes everything except the LDAP password.
- The LDAP domain is imported into the Business Directory server using the same domain name as the one that is used in the TDV server.
- All LDAP-based users and groups registered with TDV are imported to the Business Directory server.
- By default all imported LDAP users (not groups) get the Access Directory right.

6. Open the Sites page and provide the LDAP password for any sites with the message:

   ```
   The ldap password is missing. Please supply the password. [bd-200671]
   ```

7. Optionally, grant additional Business Directory access rights for the additional LDAP users.

LDAP users can now use their configured domain and username/password combinations to log into Business Directory.

Changing the Repository Password

After installation you might periodically need to change your TDV or Business Directory repository password.

In these instructions, <install_dir> means <BD_install_dir> or <TDV_install_dir>.

To change the repository password

1. Stop the repository.
2. Locate and open the ph_hba.conf file. The file is typically at:

   ```
   <install_dir>\repository\data\pg_hba.conf
   ```

3. Find and change all lines with "password" to "trust" for the METHOD column. For example:

   ```
   TYPE DATABASE USER ADDRESS METHOD
   ```

```
"local" is for Unix domain socket connections only

local all all password

IPv4 local connections:

host all all 127.0.0.1/32 password

IPv6 local connections:

host all all ::1/128 password
```

4. Start the repository. For example, on Windows:

```
composite.bat repo start
```

5. Login to the PostgreSQL database using one of the following commands:

Platform	Command	Notes
Windows	./bin/psql -hlocalhost -p9508 -Uroot -dpostgres	
UNIX	cd <install_dir>/repository; export LD_LIBRARY_PATH=<install_dir>/repository/lib; ./bin/psql -hlocalhost -p9508 -Uroot -dpostgres	Use SHLIB for HPUX and LIBPATH for AIX platforms instead of LD_LIBRARY_PATH, which is only for Linux platforms.

6. Run the psql ALTER USER command.

```
postgres=# ALTER USER root with password '<NEW_DBA_PASSWORD>';
```

```
postgres=# \q
```

7. Stop the repository.
8. Locate and open the ph_hba.conf file. The file is typically at:

   ```
   <install_dir>\repository\data\pg_hba.conf
   ```

9. Find and change all lines with "trust" to "password"for the METHOD column.
10. Start the repository.
11. Log in to the PostgreSQL database with the new password.

How Business Directory Works with the composite Domain

Composite domain users logging into Business Directory are authenticated against the remote TDV Server. For this to work, the remote TDV Server must be active.

Changes to the user access profiles made on the composite domain are not immediately applied to Business Directory. For example, if a user is removed on the remote TDV, this is not reflected in the Business Directory server until the next site refresh. Users might still be able to log into the Business Directory server.

When multiple sites are defined in Business Directory, you can also use the site name as the domain to log in with a composite domain for a particular site. For example, if you have site_a and site_b, and you want to log in as browse_user with the composite domain for site_b, you can use site_ as the domain when logging in to Business Directory.

How Business Directory Works with Escape Characters

Because Business Directory uses a JSON parser and stores data in a PostgreSQL database, there are some ways that your data will be saved and redisplayed when you search it. For example, a comment with the text "bdjonComment World!" will be returned as "E'bdjonComment World\\\\!'".

Changing the Language from User Profile

You can change the language setting for the UI display. To do this,

1. Click on the Admin Menu.
2. Choose Profile.
3. Change the language from the Language drop down.
4. Save your changes. A confirmation dialog is displayed. Once confirmed, the user is re-logged into Business Directory and the User Interface is displayed in the language chosen.

   Note: Currently Online help is provided in English and Japanese.

Using Business Directory

This section describes how to use Business Directory. However, you need to have the appropriate rights to perform most of these actions. These rights are discussed in the last of the following list of topics:

- Logging into Business Directory
- Sites
- Resources
- Categories
- Custom Properties
- Resource Data Lineage
- Access Rights

Logging into Business Directory

When you start Business Directory, you need to know the Business Directory server instance to which you want to connect.

To log into Business Directory

1. Type the following URL in your browser:

   ```
   http://<hostname>:9500/directory
   ```

   The <hostname> is a fully qualified domain name (FQDN) or specific IP address for your Business Directory instance. If there are published TDV databases on the host, they are immediately registered as a site. Sites added automatically will need to be introspected using the instructions in Site Metadata.

2. Enter your user credentials and information for the Business Directory server to which you are connecting. For example:

Field	Example Value	Notes
Domain	finswest	Use the Business Directory **site name** in this field if you, the user logging in, belong to the composite domain as defined for that TDV site. You have to use the site name, because there can be one or more composite domain registered within Business Directory depending on how many sites are registered for use. If each site has a composite domain and composite is specified as the domain name, it is unclear which site to use to authenticate the user. To properly authenticate the user, TDV needs the site information, TDV assumes composite for the domain, and then authenticates the user based on username and password.
User	admin	
Password	**&%	

3. Click Login.

   Business Directory opens.

Sites

In Business Directory, a site is a TDV instance. Business Directory can view and manipulate published resources from multiple sites. If there are published TDV databases on the host used in your Business Directory URL, they are immediately registered as a site.

This section includes:

- Adding a Site
- Removing a Site
- Site Metadata
- Editing a Site

Adding a Site

Add sites to Business Directory to indicate which instances of TDV you want to view.

When you add a site to Business Directory, you are also adding that site's users to Business Directory. This includes LDAP domains and users, and certain composite domain users. The following composite domain users are not added to Business Directory: anonymous, unknown, monitor, nobody, system.

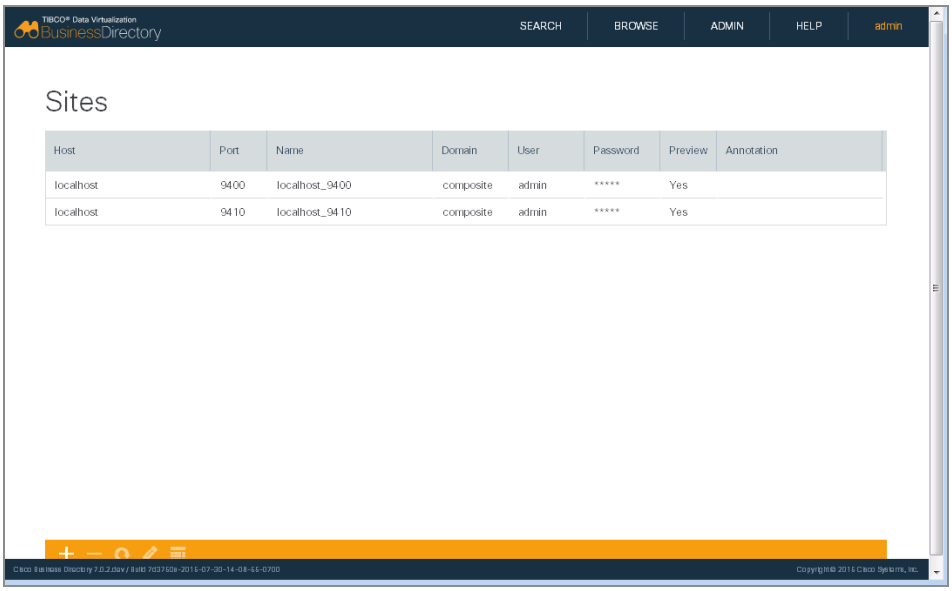
Note: TDV Server must be up and running when adding/editing sites in Business Directory.

To add a site

1. Make sure the user you are logged in as Admin, or as a user with the Admin access right.

   For details, see Access Rights.

2. Select ADMIN > SITES.



3. Click the Add (plus-sign) icon.
4. Type values for the following fields:

Field	Description of Value to Enter
Host	Enter the name or IP address of the machine where the TDV server is installed.
Port	Enter the port number for the TDV server.
Name	Type a name for this site to use within Business Directory. If you want to change the name of a site you have defined and saved, you need to delete and redefine it with a new name.
Domain	Enter a domain name that exists in this TDV instance.
User	Enter the name of a user who can access the information stored under this domain.
Password	Enter the user's password.
Preview	Enable or disable data preview. You can change this after adding a site, after entering the password.
Annotation	Enter a description of the site that can be used to help you find it among similarly named sites.

5. Click Save.

   While the site is being added, it is disabled in the Site grid, and a blue spinning icon appears to the left of the host name.

   Note: You can also click Cancel if you want to quit without saving the information.

6. Provide the LDAP password for any sites that return this message:

   > The ldap password is missing. Please supply the password. [bd-200671]

Removing a Site

You can remove a site from Business Directory. This does nothing to the TDV instance itself; it affects only the site's visibility in Business Directory.

To remove a site

1. Select ADMIN > SITES.
2. Select one or more of the registered sites.
3. Click the Remove Site (minus-sign) icon.
4. Click Yes in the Remove Site(s) dialog box.

   While the site is being removed, it is disabled in the Site grid, and a red spinning icon appears to the left of the host name.

Site Metadata

Introspection is the process of collecting published resource metadata from a TDV site. When you add a site to Business Directory, introspection is run automatically. You can choose to refresh (update or “reintrospect”) the published metadata on demand, or automatically at scheduled intervals.

Note: This introspection or reintrospection process pertains to the availability of the published metadata in Business Directory, not in the TDV instance.

This section includes:

- Business Directory Refresh Limitations
- Refreshing Site Data on Demand
- Refreshing Site Data Automatically
- Canceling Automatic Site Data Refresh

Business Directory Refresh Limitations

When introspecting your TDV sites, there are a few limitations to consider:

- The data that you want to view through Business Directory must exist as valid published data from the TDV site that you have defined.
- If problems or errors occur in the resources that you have published and want to introspect with Business Directory, the introspection process in Business Directory fails.
- Business Directory does not introspect or display published *legacy* web services.
- Web service operations cannot contain commas or parentheses.

Refreshing Site Data on Demand

You can refresh site data at any time. While the refresh process is running, Business Directory will be placed in a read-only mode.

To refresh site data

1. Select ADMIN > SITES.
2. Select one or more of the registered sites.
3. Click the Refresh Sites button.

   While the site is being refreshed, it is disabled in the Site grid, and rotating refresh arrows appear to the left of the host name. If the refresh process is brief, however, the icon might not appear.

Refreshing Site Data Automatically

Depending on the volatility of the published TDV resources that you want to browse with Business Directory, reintrospection might need to be a daily activity.

If a refresh schedule has previously been defined, you can use this same procedure to edit the definition. Deleting the refresh schedule can also be done from within the same edit dialog.

To schedule site refreshes

4. Select ADMIN > SITES.
5. Select a registered site.
6. Click the Schedule Site Refresh button.
7. Select the interval at which you want the sites to be refreshed (daily or weekly).
8. From the drop-down list, select the time (for the TDV instance in its time zone), at which you want the refresh to occur.
9. Click Save or Cancel to return to the main sites page.

Canceling Automatic Site Data Refresh

You can delete an existing scheduled site refresh.

To cancel a scheduled site refresh

10. Select ADMIN > SITES.
11. Select a registered site.
12. Click the Schedule Site Refresh button.
13. Click the X in the upper right corner of the Schedule Refresh dialog box.

Editing a Site

You can change a number of characteristics of a site.

Note: TDV Server must be up and running when adding/editing sites in Business Directory.

To edit a site

1. Select ADMIN > SITES.
2. Select one of the registered sites.
3. Click the Edit Site button.

   The site information is displayed with the fields you cannot change grayed out.
4. Type your password in the appropriate field in the row.
5. Select each characteristic you want to change, and type or select a new value for it.
6. Click Save.

Resources

You can search, browse, or watch resources, add comments, and preview data:

- Searching Resources
- Refining the Search with Filters
- Browsing and Watching Published TDV Resources

- Adding Comments to Objects
- Previewing Resource Data in Business Directory

Searching Resources

Searching the Business Directory can help you quickly find resources of interest.

Some items to consider when working with searches and resources that are displayed in Business Directory:

- Searches are simple. Searches do not support complex expressions, such as those containing AND and OR.
- Searching is case-insensitive. It does not matter what case you use when typing search terms.
- Sorting in the Data Preview grid is case-sensitive; that is, items are ordered A, B, ..., Y, Z, a, b... .
- You can search on a custom property *value*, but not on a custom property name.
- When you search on a custom property value, you must escape any slash with a backslash, and any backslash with another backslash. So “/” becomes “\\/” in the search string.
- Which resources you can see depends on your access rights. See Access Rights.

To search resources within Business Directory

1. Select BROWSE.
2. Type the value for which you want to search in the Search field at the top of the page.
3. Review the data that is returned.
4. Determine if you want to narrow the search. See Refining the Search with Filters.

Refining the Search with Filters

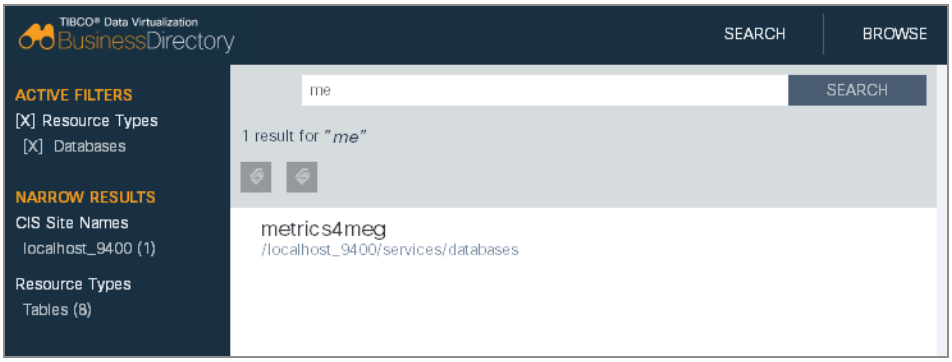
After doing an initial search, you can refine your search using a variety of filters including:

- Category or label

- Site name
- Resource type
- Service name
- Watches

To search resources within Business Directory

1. Select BROWSE.
2. Type the value for which you want to search in the Search field at the top of the page.
3. Select an item under NARROW RESULTS to further narrow your search for resources.



If a site is removed and you immediately navigate back to this page, the filters are still active. Refresh the page or navigate to the home page to enable searching and filtering again.

Browsing and Watching Published TDV Resources

You can browse through all of the published TDV resources for the sites registered with Business Directory.

If you decide to watch a resource, you are sent emails anytime a comment is added or if any of the resource metadata changes in any way. For example, if a site refresh is run and a resource has three new columns, you are sent an email.

To browse published TDV resources

1. Make sure that the site you are interested in browsing has been recently refreshed for the most current view of published resources.
2. Select BROWSE.
3. Select one or more of the registered sites.
4. Expand the site to view the published Database and Web Service resources.
5. Select a specific resource to view properties and details of the resource.

   When you select a resource in the left pane, details for the resource are displayed to the right and include some of the following:

   - A navigation bar showing the nodes leading to the resource currently selected. You can click any node to view that part of the resource tree.
   - The name of the resource.
   - WSDL URLs.
   - The type of resource (procedure, table, and so on).
   - Primary and foreign key information.
   - Index.
   - The host name and port name, or a name assigned to the resource.

   There is also a set of tabs, which vary depending on the type of object you are viewing, with further resource information:

   - Parameters (procedure)—Parameter name, data type, direction.
   - Resources (folder)—Resource name and type (web service, database, folder, and so on).
   - Columns (table)— Column name, data type, whether it is a primary or a foreign key, whether it is indexed, and annotations. If a primary key column is displayed, you can navigate to the tables that use this primary key as a foreign key. Additionally, any columns that are foreign keys have a link back to the table in which this foreign key is a primary key.
   - Operations (Web Service)—Operation name and type, and annotations.
   - Properties—Property name, default value, current value.
   - Data—Preview of the data that this resource can retrieve.

- Lineage–Diagram of the data lineage for the object. For details of what data lineage is within TDV, see the *TDV User Guide*.

6. To watch the object, click Watch.
   - You might be asked to type your name and email address so that notifications can be sent to you.
   - To watch the associated child objects, click Watch Children.

Adding Comments to Objects

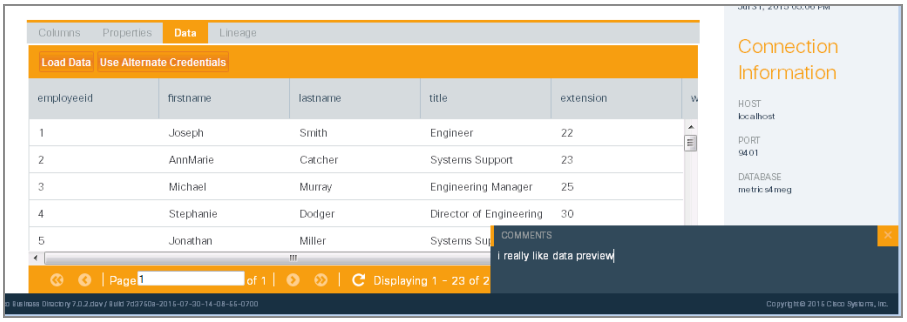
You can comment on any object viewable through the Business Directory Browse feature. The comments are stored in the Business Directory repository.

Note: Comments are not saved to the TDV repository, and they are not viewable through Studio.

For information on the access needed to edit or delete comments, see Access Rights.

To add comments to an object

1. Select BROWSE.
2. Navigate and select a published object or container.
3. Click the comment link on the page or the icon in the lower right portion of the screen.



4. Type your comment and click Return.

   Your comment is added to the object, and now others can see and reply to your comment. Anyone who is watching the resource or container is notified of your comment.

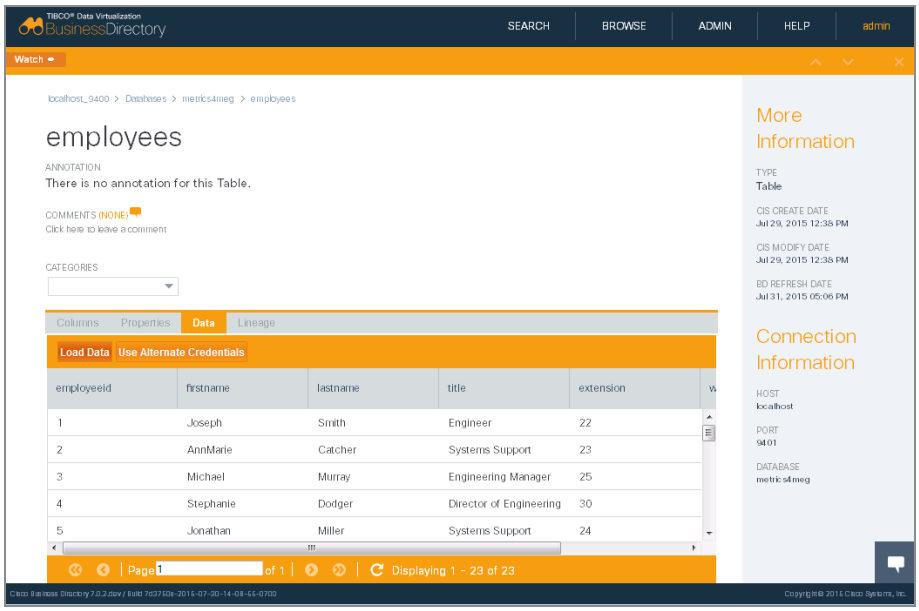
Previewing Resource Data in Business Directory

Previewing resource data within Business Directory lets you determine whether the resource you are working with is really the one you are interested in.

Note: Data preview must be enabled. See Adding a Site.

To preview data

1. Follow the instructions in Searching Resources.
2. After selecting the resource you are interested in, select the Data tab.
3. Click Load Data to get data to display.



4. Optionally, to view data you might need to Use Alternate Credentials to authenticate as a user with rights to view the data.

   There are many reasons that this might be required, depending on your corporate security requirements. Use the LDAP domain name from the TDV instance rather than the consolidated LDAP domain name that is visible within Business Directory.

5. Optionally, you can use column filtering or write your own filter WHERE clause. This is particularly useful when you are viewing large amounts of data.

Categories

This section includes:

- Defining Categories and Values
- Adding Bulk Classifications for Resources
- Adding Categories to a Resource

Defining Categories and Values

Each category that you define must have at least one label so that it can be associated with a resource on the Browse page. For required permissions, see Access Rights.

Double quotes in the category name are not supported.

To define categories and values

1. Select ADMIN > CATEGORIES.
2. Select Add Category (the plus-sign icon).
3. Type a category name.
4. Click Save.
5. Hover over the wrench icon and click Add Value.
6. Type a value.

Adding Bulk Classifications for Resources

Bulk classifications are a quick way to associate many resources with a category or name of a business process so that you and others can easily find them.

During classification, if a resource already has the category applied to it, the category is not reapplied. Similarly, when clearing the classification from resources, if the category is not present on the resource there is no change to the resource.

For information about permissions, see Access Rights.

To add or clear classifications

1. Select BROWSE or use SEARCH.
2. Locate and select a resource or container to which you want the classification to apply. Use multi-select options to select more than one resource or container.
3. Click the Add icon above the resource tree pane to classify the resource with the category.

   Click the Remove icon above the resource tree pane to clear the classification from the resources with the category.
4. Choose your classification from the list of categories. See Defining Categories and Values, for how categories are defined.

Adding Categories to a Resource

Categories associate resources with a name that you or others can easily find. For information on permissions, see Access Rights.

To add categories to a resource

1. Select BROWSE or use SEARCH.
2. Expand the tree and navigate to the resources to which you want to associate a category.
3. Select a specific resource to view properties and details of the resource.
4. Below Categories, select the category that you want to associate with the resource.
5. Add more categories or delete them as necessary.

Custom Properties

You can add groups and properties within Business Directory. The custom properties can be used to facilitate additional logic or reporting that can be driven from the information stored in Business Directory.

- Defining and Editing Custom Property Groups and Properties
- Specifying the Custom Property Value for a Resource Tab.

Defining and Editing Custom Property Groups and Properties

Before Property groups are available to the resources within Business Directory, a user with administration privileges must add them to the custom properties page. A Property Group must have at least one property defined for it to be viewable from the Properties Tab for a resource.

Note: On Firefox, blank custom properties occasionally disappear from the UI; but after a refresh of the browser, the properties appear as expected.

To add and manage custom properties

1. Select ADMIN > CUSTOM PROPERTIES.
2. Click the Add Group plus-sign icon at the lower left of the page.
3. Type the name of the Group of properties you want to define.
4. Optionally, type a description of the Group in the Annotation field. This description is only visible to other administrators who are adding or editing a Property Group. It does not appear on the resource page.
5. Select a navigation option to determine where the Property Group appears on the resource detail page:

Option	Description
PROPERTIES_TAB	(Default) Lists the Property Groups.
CUSTOM_TAB	Adds a tab with the Property Group name.
SHARED_AREA	Displays the Property Group and its properties in the shared area above the detail table.

6. Navigate to and select a location.

   This location links the resources to this custom property. You can map more than one location to the property. For example, if you want the ACCT property available to all the views that you have published for SITE! and SITE@, then you need to add both SITE! and SITE@ to the location field.

7. Scroll over the Group that you just added.
8. Scroll over the tool icon that appears to the left of the group or property and select one of the following:

Option	Description
Add	Type a name for the property. Select the data type for the property. Type a default value for the property. It must be a valid value for the data type you have specified. The value can later be overridden for a specific resource.
Cut	
Paste	
Delete	
Edit	Open the property editor.

9. Navigate back to the Browse page and select a resource that belongs to the site location that you defined.
10. Select the Properties tab.

Defining and Editing Custom Groups and Properties

Note: On Firefox, blank custom properties occasionally disappear from the UI; but after a refresh of the browser, the properties appear as expected.

To add custom properties

1. Select ADMIN > CUSTOM PROPERTIES.
2. Highlight a Property Group Name
3. Hover over the wrench icon and click Add Property.
4. Type the name of the property you want to define.

5. Select a property type you want from the drop-down list.

   Properties options are:

Option	Note	Option	Note
• STRING_ TYPE	Default	• TIME_TYPE	
• BOOLEAN_ TYPE		• TIMESTAMP_ TYPE	
• YES_NO_ TYPE		• URL_TYPE	
• INTEGER_ TYPE		• SINGLE_ ENUMERATOR	Complete Valid Values field
• DECIMAL_ TYPE		• MULTIPLE_ ENUMERATOR	Complete Valid Values field
• DATE_TYPE		• RICH_TEXT_ TYPE	

6. Type the default value.

Note: for Single and Multiple enumerators you must complete the valid values field in addition to the default value.

7. Click Save.

To edit custom properties

8. Select ADMIN > CUSTOM PROPERTIES.
9. Highlight a Property Name
10. Hover over the wrench icon and click Edit Property.
11. Type the modified name of the property you want to define.
12. Select a property type from the drop-down list.
13. Enter a default value.

Note: For Single_ and Multiple_Enumerator types you must complete the valid values field in addition to the default value.

14. Click Save.

Specifying the Custom Property Value for a Resource Tab.

After a custom property has been defined for resources associated with the locations specified in the location field and a default value is defined, you can specify exact values for the property of each resource.

Note: On Firefox, blank custom properties sometimes disappear from the UI; but after a refresh of the browser display, the properties appear as expected.

To manipulate custom properties

1. Select BROWSE and open the resource for which you want to edit custom properties.
2. Select the Properties tab.
3. Double-click a Property Name row that contains a value.
4. Edit the Value and click Save.

Resource Data Lineage

Business Directory provides a tab that let you view a resource's data lineage. This section contains:

- What Data Lineage Reveals
- Viewing Data Lineage

What Data Lineage Reveals

Data Lineage tells you which data sources, tables, and columns provided the data found in a specific resource. This information can help you understand:

- What will be affected by a proposed change to a column or resource.
- How a published resource is provided to a consuming application, so that you can trace dependencies back to their origins.
- What resources you might need to have privileges for, because of resource dependencies.

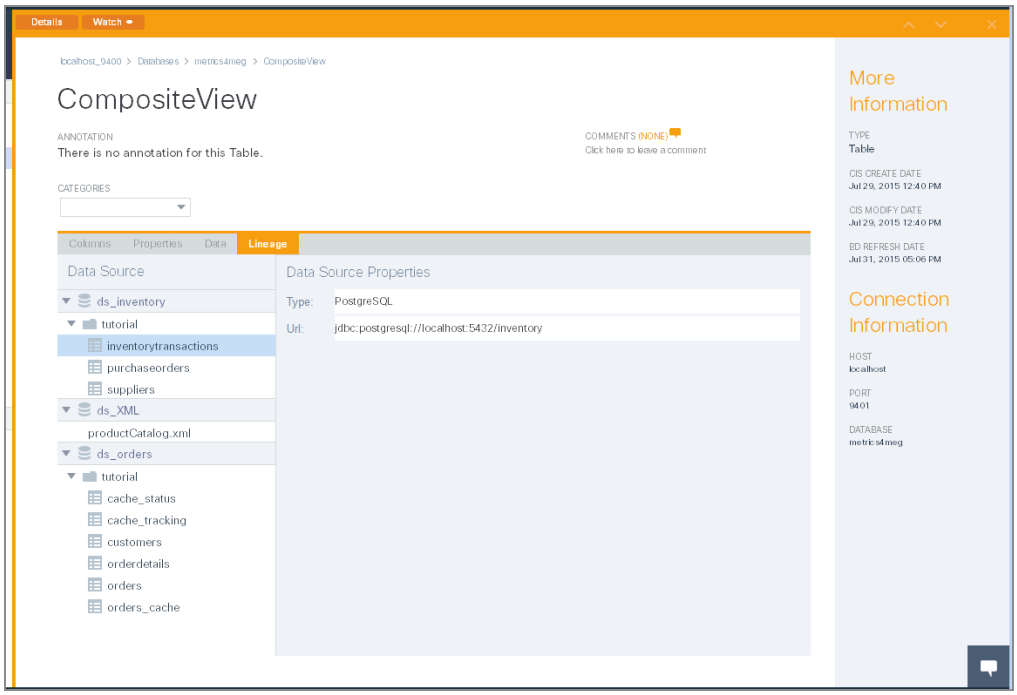
For further details on how TDV collects and displays data lineage information, see the *TDV User Guide*.

Viewing Data Lineage

You can view data lineage information in Business Directory. This information reflects the dependencies present in the underlying TDV instance.

To view data lineage information

1. Select BROWSE and open a resource.
2. Select the Data Lineage tab.
3. Expand the tree under the Data Source column to see which data sources contribute to creating the resource.
4. Expand each of the data sources further to see what procedures or tables and columns are used in the creation of the resource in which you are interested.



Access Rights

The users and groups listed under Access Control are specific to Business Directory. Rights that are displayed in gray are inherited from the group to which the user or group belongs.

This section contains:

- Viewing Users on the Access Control Page
- Managing User Access
- Changing the Admin Password After Installation

Business Directory users can be assigned rights that control what content users and groups can add, edit, or delete. The following rights are available:

Access Level	Description
Access Directory	Access Business Directory and view all information. This right is required to log into the Business Directory.

Access Level	Description
Apply Comment	Create a comment. Edit or delete one's own comments.
Moderate Comment	Remove any comment or comment thread. Only the author of a comment can edit that comment.
Apply Category	Assign or unassign category values for any resource.
Manage Category	Create or remove a category. Create, edit, or remove a category's list of values.
Apply Property	Assign the *value* of a custom property on a resource.
Manage Property	Create, edit, or delete custom properties and property groups. Assign or remove custom properties from resources.
Read All	Read all resource information listed in the Business Directory. Users who do not also have the Access Directory rights remain in the system, but they are not allowed access beyond the login screen.
Admin	Add, remove, or refresh sites. Administer users and groups, including LDAP. Read and manage all Business Directory content. If this right is assigned, the user has **all** rights regardless of other selections.

Viewing Users on the Access Control Page

As the number of Business directory users grows, being able to sort how you view them can help you manage users and their rights.

To view and sort Business Directory users

1. Select ADMIN > ACCESS CONTROL.
2. In the drop-down field, select Users or Groups
3. Optionally, to filter data, in the text field, type criteria to refine how the information is filtered.

4. Determine which of the following columns you want to use to sort the data:

User	Domain	Sites
Apply Comment	Moderate Comment	Apply Category
Manage Category	Apply Property	Manage Property
Read All	Access Directory	Admin
Member Of		

5. Click a heading again to change sort order.

Managing User Access

If you have Admin rights for a Business Directory instance, you can manage all of its users and groups.

The resources that users can view depend on their rights as defined in both Business Directory and the TDV sites.

To mange business directory user access

1. Select ADMIN > ACCESS CONTROL.
2. Select the row that contains the user whose rights you want to modify.
3. Determine which rights you want to grant to the user.

Common Tasks	Rights Needed
Add a comment	Access Directory plus Apply Comment
Create a category and value	Access Directory plus Manage Category

Common Tasks	Rights Needed
Associate a category value with a resource	Access Directory plus Apply Category
Create a custom property or group	Access Directory plus Manage Property
Associate a custom property with a resource	Access Directory plus Apply Property

4. Select or clear check marks depending on which rights you want to grant to the user.

Changes are saved automatically.

Changing the Admin Password After Installation

Business Directory manages the password for the Admin user (but for no other users).

To change the Admin user password

1. Log in to Business Directory as the Admin user.
2. Select Admin > PROFILE in the area next to the HELP button.
3. On the User Profile page, select Change Password.

   You can also type a first name, last name, and email address on the User Profile pane.
4. Follow the prompts to change the password.

   If you typed a first name, it is displayed in place of “admin” next to the HELP button.

Business Directory API and System Tables

This section describes how to use Business Directory REST API, and how to explore the system tables associated with Business Directory.

- Using the Business Directory REST API
- Backing Up and Restoring Business Directory Information
- Accessing Business Directory System Table Information

Using the Business Directory REST API

Business Directory provides access to the REST API methods through a Swagger web interface. From the API web page you can access information about each method, including what it does and what parameters are required. You can also run the methods directly from the web page.

You can implement the REST API methods in two ways:

- Using Business Directory to Run the REST API Methods
- Using cURL Commands to Run the REST API Methods
- Characteristics of Programmatic Use of the REST API

Using Business Directory to Run the REST API Methods

To use the Business Directory API methods

1. Select Help > REST API.

   A new browser tab opens with a list of REST methods, grouped by what they act on and what they do.

2. Ensure that you have an SSL connection. For example, your URL should be:

```
https://localhost:9502/directory/api-docs/
```

3. Click the text on the method you want to look at or use.

   The method description expands to display its description, the rights required, examples of its use, and a Try it out! button.

4. To use a method, type values for required parameters and click Try it out!.

   Note: Although the button says Try it out, the method executes “live” on the instance to which it points.
   Commands the require a file input or output cannot be run from the Try it out! button.

5. Review the information.

Using cURL Commands to Run the REST API Methods

The REST API page displays several cURL examples for each method on the page. You can use a command window to run the cURL command and get results or modify your Business Directory instance.

Typically, UNIX and Cygwin command line windows come with the cURL tool installed, For Windows, you might need to obtain and install the cURL tool.

These instructions assume that running in insecure mode. If you want to configure secure execution of the cURL commands, see an example set up in Setting Up Secure cURL Execution Example

To use a command window to run the Business Directory REST API methods

1. Select Help > REST API.

   A new browser tab opens with a list of REST methods, grouped by what they act on and what they do.

2. Click the text on the method you want to look at or use.

   The method description expands to display its description, the rights required, examples of its use, and a Try it out! button.

3. Copy the text of the cURL example that appears closest to what you might need to use.

4. Open your favorite command window.

5. Paste the cURL command, optionally add a '-k' to the command to run in insecure mode, and click Return.

6. Review the results.

Setting Up Secure cURL Execution Example

The cis_server.pem is a certificate which is exported from <TDV_install_ dir>conf\server\security\cis_server_keystore.jks.

To troubleshoot secure cURL execution

1. If you run a curl command similar to:

   ```
   curl -X GET -u admin:admin "https://localhost:9502/rest/userProfiles" - -cacert cis_
   server.pem
   ```

   You will get output similar to:

   ```
   curl: (51) SSL: certificate subject name 'cis_server' does not match target host name
   'localhost'
   ```

2. Make your host name match the certificate, then you can run the cURL command without errors.

For example, update your /etc/hosts file as follows:

```
127.0.0.1 cis_server
```

cURL command:

```
curl -X GET -u admin:admin "https://cis_server:9502/rest/userProfiles" --cacert cis_server.pem
```

Characteristics of Programmatic Use of the REST API

When using REST API methods programatically, be aware of the following.

SSL

SSL is required. Do not attempt to invoke the API over a non-secured HTTP channel.

Method Overrides

If your REST client library does not support PUT, PATCH, and DELETE methods, you can include an "X-HTTP-Method-Override" header and use the POST method. For example:

```
curl -X POST -u admin:admin "http://localhost:9500/rest/v1/sites/test_site" -H"X-HTTP-Method-Override:DELETE"
```

POST

POST methods that create single entities return a Location header with the path to the new entity. POST methods that create multiple entities do not return the paths to the new entities.

PUT

PUT methods completely update entities. Omitting a property when using a PUT method results either in an error or in setting the entity's property to a null or default value.

GET

Many GET methods have equivalent system (TDV SQL) queries. The Swagger documentation provides examples.

If a GET method lists items, it supports several query parameters:

- The "limit" parameter specifies the number of records to return.
- The "offset" parameter specifies the starting record.
- The "orderBy" parameter returns results in a specific order. It uses SQL syntax.
  - Encode the "orderBy" parameter as you would encode a standard URL query string.
  - Escape a field name that contains spaces by surrounding the field name with double quotes.
  - In a field escaped for spaces, escape a double quote with two double quotes.
  - To determine field names for the orderBy clause, refer to Swagger documentation for equivalent system query syntax.
  - Use HELP > SYSTEM TABLES to view documentation of columns in those tables.
- Some orderBy examples:

  orderBy=City, State, Zip

  orderBy=City DESC, State ASC, Zip

orderBy="First Name" ASC, City, Zip ASC

Backing Up and Restoring Business Directory Information

The REST API provides export and import commands that you can use to back up and restore the Business Directory system.

The commands as they apply to backup and restore are described, with examples, in the following sections:

- Backing Up Business Directory
- Restoring Business Directory
- Backing up Business Directory Encryption settings
- Backing up Business Directory Encryption settings

Backing Up Business Directory

You can back up Business Directory metadata (custom properties, catalogs, and other data) by exporting it to a password protected CAR file.

To back up Business Directory

1. Select Help > REST API.

   A new browser tab opens with a list of REST methods, grouped by what they act on and what they do.
2. Scroll down to the “metadata: System Metadata” group and click “Export metadata to a file.”
3. Make sure you have the required rights.
4. Type values for required parameters, including an appropriate name and location for the CAR file.
5. Copy the cURL command that is the closest to what you think you will need.
6. Open a command window and run the cURL command. See Using cURL Commands to Run the REST API Methods.

Export as Admin user Example

Here is an example of the CURL invocation you would use as the Admin user.

```
curl -u "admin:admin" -X GET
"https://localhost:9502/rest/v2/metadata?encryptionPassword=testPassword" -o export001.car
```

Export as an LDAP user Example

Here is an example of the CURL invocation you would use as an LDAP user with BD_ ADMIN and ACCESS_DIRECTORY rights.

```
curl -u "user@ldapDomain:password" -X GET
"https://localhost:9502/rest/v2/metadata?encryptionPassword=testPassword" -o export001.car
```

Restoring Business Directory

You can restore Business Directory metadata (custom properties, catalogs, and other data) by importing a previously exported CAR file. You must know the password that was used to protect the backup file.

To restore Business Directory

1. Select Help > REST API.

   A new browser tab opens with a list of REST methods, grouped by what they act on and what they do.
2. Scroll down to the “metadata: System Metadata” group and click “Import full metadata from a file.”
3. Make sure you have the required rights.
4. Type values for required parameters.
5. Copy the cURL command that is the closest to what you think you will need.
6. Open a command window and run the cURL command. See Using cURL Commands to Run the REST API Methods.

Import as Admin user Example

Here is an example of the CURL invocation you would use as the Admin user.

```
curl -u "admin:admin"  -i  -F "encryptionPassword=testPassword" -F "overwrite=true" -F
"file=@export001.car" -X PUT "https://localhost:9502/rest/v2/metadata"
```

Import as Admin user, with method override Example

Here is an example of the CURL invocation you would use as the Admin user, using the X-HTTP-Method-Override option. (See Method Overrides.)

```
curl -u "admin:admin"  -i -H "X-HTTP-Method-Override:PUT"  -F
"encryptionPassword=testPassword" -F "overwrite=true" -F "file=@export001.car" -X POST
"https://localhost:9502/rest/v2/metadata"
```

Import as an LDAP user Example

Here is an example of the CURL invocation you would use as an LDAP user with BD_ ADMIN and ACCESS_DIRECTORY rights.

```
curl -u "user@ldapDomain:password"  -i  -F "encryptionPassword=testPassword" -F
"overwrite=true" -F "file=@export001.car" -X PUT "https://localhost:9502/rest/v2/metadata"
```

Import as Admin user (ignoring encryption errors) Example

Here is an example of the CURL invocation you would use as the Admin user (ignoring the encryption errors)

```
curl -u "admin:admin"  -i  -F "encryptionPassword=testPassword" -F "overwrite=true" -F
"ignoreEncryption=true" -F "file=@export001.car" -X PUT "https://localhost:9502/rest/v2/metadata"
```

Note: If -ignoreEncyption option is used, then all backup data will be imported regardless of whether a valid encryption key was provided. This means that the import will not fail. This option can be used to allow partially importing any backed up data. However, the import process will only import data that is not encrypted or can be decrypted using the provided encryption key. All encrypted portions of the backup data that cannot be decrypted will be imported as empty values and the import will otherwise succeed.

This affects all encrypted values in the backup data, which includes, but is not limited to data source and LDAP domain connection passwords.

Backing up Business Directory Encryption settings

You can back up Business Directory Encryption settings to a password protected file for server recovery in case of emergency.

To back up Business Directory encryption settings

1. Select Help > REST API.

   A new browser tab opens with a list of REST methods, grouped by what they act on and what they do.
2. Scroll down to the “security” group and click “Backup the encryption settings to a password protected file.”
3. Make sure you have the required rights.
4. Type values for required parameters, including an appropriate name and location for the CAR file.
5. Copy the cURL command that is the closest to what you think you will need.
6. Open a command window and run the cURL command. See Using cURL Commands to Run the REST API Methods.

Backup as Admin user Example

Here is an example of the CURL invocation you would use as the Admin user.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/security/backup_encryption_
settings?encryptionPassword=testPassword" -o backup_encryption_settings.txt
```

Manual Backup of Encryption Settings

After the installation of BD, manually backup the files boot.properties and encryption.properties to avoid loss of data during unforeseen system issues.

Restoring Business Directory Encryption settings

You can restore the encryption settings from the backup file. You must know the password that was used to protect the backup file.

To restore Business Directory Encryption Settings

1. Select Help > REST API.

   A new browser tab opens with a list of REST methods, grouped by what they act on and what they do.
2. Scroll down to the “security” group and click “Restore the encryption settings from the backup file.”
3. Make sure you have the required rights.
4. Type values for required parameters.
5. Copy the cURL command that is the closest to what you think you will need.
6. Open a command window and run the cURL command. See Using cURL Commands to Run the REST API Methods.

Import as Admin user Example

Here is an example of the CURL invocation you would use as the Admin user.

```
curl -u "admin:admin" -i -F "encryptionPassword=testPassword" -F "file=@backup_encryption_
settings.txt" -X POST "https://localhost:9502/rest/v2/security/import_encryption_settings"
```

Accessing Business Directory System Table Information

You can access system table information at any time.

About System Tables

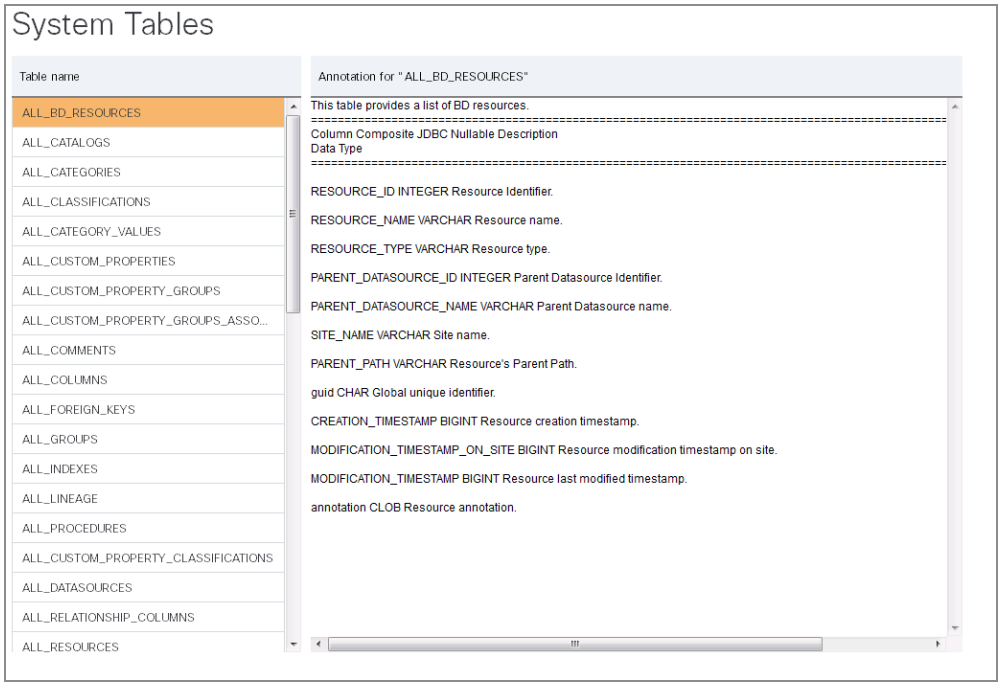
Business Directory can be used to view system tables. You could then use the information to run an API method that can show its contents.

Note: System tables are *virtual tables*. They map to a physical database table, a view, a structure in server memory, or a combination of these. TIBCO reserves the right to change the system tables at any time.

For system tables, what you see depends on the rights and privileges you have. Studio users are limited to executing SQL SELECT statements on these tables. The rights and privileges to change system tables are locked, to prevent changes that could compromise functionality and performance.

To access system table information

1. Select Help > System Tables.
2. Select one of the tables listed.



3. Review the Annotation information.

Business Directory REST API

Business Directory provides access to the REST API methods through a Swagger web interface. From the API web page you can access information about each method, including what it does and what parameters are required. You can also run the methods directly from the web page. This chapter is a quick reference of all the Business Directory REST APIs.

Business Directory REST APIs

TDV provides a list of REST APIs to manage the Business Directory resources. This sections explains all the operations that can be performed on the resources by using the REST APIs provided for the following Business Directory functions:

- Categories
- Comments
- Configs
- Data
- Domains
- Groups
- Metadata
- propertyGroups
- Resources
- Security
- Session
- Sites
- userProfiles
- Users
- Watches

Categories

The following operations can be performed on the different categories of the BD resources:

- GET /categories
- POST /categories
- DELETE /categories
- PATCH /categories
- POST /categories/classifications
- GET /categories/CategoryName
- PATCH /categories/CategoryName
- GET /categories/categoryName/values
- POST /categories/categoryName/values
- DELETE /categories/categoryName/values
- GET /categories/{categoryName}/values/{categoryValueName}/resources
- POST /categories/{categoryName}/values/{categoryValueName}/resources
- PUT /categories/{categoryName}/values/{valueName}
- DELETE /categories/{categoryName}/values/{valueName}/resources
- DELETE /categories/{categoryName}/values/{valueName}/resources/{resourceType}/{resourcePath}
- DELETE /categories/{categoryName}/values/{value}

GET /categories

This API is used to get all categories.

Parameters

None

Example to get all categories

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/categories"
```

Example to get all categories as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/categories"
```

Example to get all categories count

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/categories?count"
```

POST /categories

This API is used to create new categories and values. The provided JSON should be a list of CategoryBeans that you'd like to add or update. A CategoryBean consists of a categoryName and List of category values. If the set of values is empty, then the category will be added without any values. If the category already exists, then the given values will be added to the category.

Parameters

None

Request Body

```
[
 {
  "categoryName": "string",
  "categoryValues": [
```

```
    "string"
  ]
 }
]
```

Example to create a new category "category1" with empty values

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/categories" -H "Content-Type:application/json" -d "[{\"categoryName\":\"category1\", \"categoryValues\" : []}]"
```

Example to create a new category "category1" with empty values as ldap user

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/categories" -H "Content-Type:application/json" -d "[{\"categoryName\":\"category1\", \"categoryValues\" : []}]"
```

Example to create a new category "category2" with values categoryValue1, categoryValue2, categoryValue3

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/categories" -H "Content-Type:application/json" -d "[{\"categoryName\":\"category2\",\"categoryValues\":[\"categoryValue1\",\"categoryValue2\",\"categoryValue3\"]}]"
```

Example to add values to existing category "category2"

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/categories" -H "Content-Type:application/json" -d "[{\"categoryName\":\"category2\",\"categoryValues\":[\"categoryValue4\",\"categoryValue5\",\"categoryValue6\"]}]"
```

DELETE /categories

This API is used to delete categories and values. The provided JSON should be a map whose keys are the categories that you'd like to delete. The values of the map are a set of values to delete within the category. If the set of values is empty, then the entire category will be deleted.

Parameters

None

Request Body

```
[
 {
  "categoryName": "string",
  "categoryValues": [
   "string"
  ]
 }
]
```

Example to delete categories category1 and category2

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/categories" -H "Content-Type:application/json" -d "[{\"categoryName\" : \"category1\" }, { \"categoryName\" : \"category2\" }]"
```

Example to delete values "categoryValue1", "categoryValue2", "categoryValue3" from category category2

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/categories" -H "Content-Type:application/json" -d "[{\"categoryName\" : \"category2\", \"categoryValues\" : [\"categoryValue1\",\"categoryValue2\",\"categoryValue3\"]}]"
```

Example to delete values "categoryValue1", "categoryValue2", "categoryValue3" from category category2 as an ldap user

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/categories" -H "Content-Type:application/json" -d "[{\"categoryName\" : \"category2\", \"categoryValues\" : [\"categoryValue1\",\"categoryValue2\",\"categoryValue3\"]}]"
```

PATCH /categories

This API is used to rename categories and values.

Parameters

None

Request Body

```
[
 {
   "op": "RENAME_CATEGORY",
   "oldCategoryValue": "string",
   "oldCategoryName": "string",
   "newCategoryOrCategoryValueName": "string"
```

```
}
```

Example to Rename category "originalCategoryName1" to "category1"

```
curl -X PATCH -u admin:admin "https://localhost:9502/rest/v2/categories" -H "Content-
Type:application/json" -d "[{\"op\":\"RENAME_CATEGORY\",
\"oldCategoryName\":\"originalCategoryName1\",
\"newCategoryOrCategoryValueName\":\"category1\"}]"
```

POST /categories/classifications

This API is used to classify resources by associating them with categories and values. The classifications you specify in the request body (JSON) for each resource replace any current classifications for that resource, so you can use this method both to add and remove classifications.

Parameters

None.

Request Body

```
[
 {
  "resourcePath": "string",
  "resourceType": "string",
  "classifications": [
   {
```

```
      "categoryName": "string",
      "categoryValues": [
        "string"
      ]
    }
  ]
 }
]
```

Example to classify a resource with Multiple categories/values

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/categories/classifications" -H
"Content-Type:application/json" -d "[{\"resourcePath\":\"/localhost_
9400/services/databases/foo/bar/orders\", \"resourceType\":\"database_table\", \"classifications\":
[{\"categoryName\":\"sales\",\"categoryValues\":[\"hrsales\",\"prodsales\"]}]}]"
```

Example to classify a resource with Multiple categories/values as ldap user

```
curl -X POST -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/classifications" -H "Content-Type:application/json" -d "
[{\"resourcePath\":\"/localhost_9400/services/databases/foo/bar/orders\",
\"resourceType\":\"database_table\", \"classifications\":
[{\"categoryName\":\"sales\",\"categoryValues\":[\"hrsales\",\"prodsales\"]}]}]"
```

Example to remove classifications for a resource

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/categories/classifications" -H
"Content-Type:application/json" -d "[{\"resourcePath\":\"/localhost_
9400/services/databases/foo/bar/orders\", \"resourceType\":\"database_table\", \"classifications\":
null }]"
```

GET /categories/CategoryName

This API is used to get a category.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category to get.	path	string

Example to get category "category2"

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/categories/category2"
```

Example to get category "category2" as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/categories/category2"
```

PATCH /categories/CategoryName

This API is used to rename a single category and/or its values.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category to rename.	path	string

Example to rename category "category1" to "oldCategoryName1"

```
curl -X PATCH -u admin:admin "https://localhost:9502/rest/v2/categories/oldCategoryName1" -H "Content-Type:application/json" -d "[{\"op\":\"RENAME_CATEGORY\", \"newCategoryOrCategoryValueName\":\"category2\"}]"
```

Example to rename category "category1" to "oldCategoryName1" as ldap user

```
curl -X PATCH -u user@ldapDomain:password "https://localhost:9502/rest/v2/categories/oldCategoryName1" -H "Content-Type:application/json" -d "[{\"op\":\"RENAME_CATEGORY\", \"newCategoryOrCategoryValueName\":\"category2\"}]"
```

Example to Rename category value name "category1/value1" to "category1/value2"

```
curl -X PATCH -u admin:admin "https://localhost:9502/rest/v2/categories/category1" -H "Content-Type:application/json" -d "[{\"op\":\"RENAME_CATEGORY_VALUE\", \"oldCategoryValue\":\"value1\", \"newCategoryOrCategoryValueName\":\"value2\"}]"
```

Example to Rename category name and value "category1/value1" to "category2/value2"

```
curl -X PATCH -u admin:admin "https://localhost:9502/rest/v2/categories/category1" -H "Content-Type:application/json" -d "[{\"op\":\"RENAME_CATEGORY\", \"newCategoryOrCategoryValueName\":\"category2\"}, {\"op\":\"RENAME_CATEGORY_VALUE\", \"oldCategoryValue\":\"value1\", \"newCategoryOrCategoryValueName\":\"value2\"}]"
```

GET /categories/categoryName/values

This API is used to get values for a category.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category for which to get values..	path	string

Example to get values for category "category2"

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/categories/category2/values"
```

Example to get values for category "category2" as ldap user

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/category2/values"
```

Example to get value count for category "category2"

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/categories/category2/values?count"
```

POST /categories/categoryName/values

This API is used to create new values in a category. The provided JSON should be a list of string values that you'd like to add to the category.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category to which to add new values	path	string

Request Body

```
[
  "string"
]
```

Example to update category "category2" with new values categoryValue1, categoryValue2, categoryValue3

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/categories/category2/values" -H "Content-Type:application/json" -d "[\"categoryValue1\",\"categoryValue2\",\"categoryValue3\"]"
```

Example to update category "category2" with new values categoryValue1, categoryValue2, categoryValue3 as ldap user

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/categories/category2/values" -H "Content-Type:application/json" -d "[\"categoryValue1\",\"categoryValue2\",\"categoryValue3\"]"
```

DELETE /categories/categoryName/values

This API is used to drop all values from a category.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category to which to add new values	path	string

Example to drop all values from category "category2"

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/categories/category2/values"
```

Example to drop all values from category "category2" as ldap user

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/category2/values"
```

GET /categories/{categoryName}/values/{categoryValueName}/resources

This API is used to fetch classifications for a particular category value.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category	path	string
categoryValueName	The name of the category value	path	string

Example to list classifications for the products/bestseller category/value pair

```
curl -X GET -u admin:admin
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources"
```

Example to list classifications for the products/bestseller category/value pair as ldap user

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources"
```

Equivalent system query:

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/data/typed" -H "Content-
Type:application/json" -d "{\"query\":\"select * from ALL_CLASSIFICATIONS where CATEGORY_
VALUE_ID IN (select CATEGORY_VALUE_ID from ALL_CATEGORY_VALUES where
CATEGORY_NAME = 'products' and CATEGORY_VALUE_NAME = 'bestseller'
)\",\"standardSQL\":true}"
```

Example to get the count of classifications for the products/bestseller category/value pair

```
curl -X GET -u admin:admin
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources?count"
```

POST /categories/{categoryName}/values/{categoryValueName}/resources

This API is used to classify resources by associating them with a category/value pair.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	Category name	path	string

Name	Description	Parameter Type	Data Type
	with which to classify the resource		
categoryValueName	Category value with which to classify the resource	path	string

Request Body

```
[
 {
  "resourcePath": "string",
  "resourceType": "string"
 }
]
```

Example to Classify a resource with the products/bestseller category/value pair

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/categories/products/values/bestseller/resources" -H "Content-Type:application/json" -d "[{\"resourcePath\":\"/localhost_9400/services/databases/ds/foo\", \"resourceType\":\"database_table\"}]"
```

Example to Classify a resource with the products/bestseller category/value pair as ldap user

```
curl -X POST -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources" -H "Content-
Type:application/json" -d "[{\"resourcePath\":\"/localhost_9400/services/databases/ds/foo\",
\"resourceType\":\"database_table\"}]"
```

PUT /categories/{categoryName}/values/{valueName}

This API is used to rename a single category value.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	Category under which a value is to be renamed.	path	string
categoryValueName	Category value to rename	path	string
body	The new category value	body	string

Example to rename category value "category1/Value1" to "oldCategoryName1/RenamedValue1"

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/categories/category1/values/Value1" -H
"Content-Type:application/json" -d "RenamedValue1"
```

Example to rename category value "category1/Value1" to "oldCategoryName1/RenamedValue1" as ldap user

```
curl -X PUT -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/category1/values/Value1" -H "Content-
Type:application/json" -d "RenamedValue1"
```

DELETE /categories/{categoryName}/values/{valueName}/resources

This API is used to remove resource classifications.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category associated with the classification to remove.	path	string
categoryValueName	The name of the category value associated with the classification to remove	path	string

Example to remove the products/bestseller classification from a resource

```
curl -X DELETE -u admin:admin
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources" -H "Content-
Type:application/json" -d "[{\"resourcePath\":\"/localhost_9400/services/databases/ds/foo\",
\"resourceType\":\"database_table\"}]"
```

Example to remove the products/bestseller classification from a resource as ldap user

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources" -H "Content-
Type:application/json" -d "[{\"resourcePath\":\"/localhost_9400/services/databases/ds/foo\",
\"resourceType\":\"database_table\"}]"
```

DELETE /categories/{categoryName}/values/{valueName}/resources/{resourceType}/{resourcePath}

This API is used to remove a resource classification.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category associated with the classification to remove.	path	string
valueName	The name of the value to unclassify	path	string
resourceType	The type of the resource from which the category value is to be unclassified	path	string
resourcePath	The path of the resource from which the category value is to be unclassified. URL Encode any slashes in the path	path	string

Example to remove classifications from a resource

```
curl -X DELETE -u admin:admin
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Ffoo%2Fbar%2Forders"
```

Example to remove classifications from a resource as ldap user

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/products/values/bestseller/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Ffoo%2Fbar%2Forders"
```

DELETE /categories/{categoryName}/values/{value}

This API is used to drop a value from a category.

Parameters

Name	Description	Parameter Type	Data Type
categoryName	The name of the category from which to delete a value	path	string
value	The name of the value to delete from the category	path	string

Example to drop from category "category2" the value "categoryValue3"

```
curl -X DELETE -u admin:admin
"https://localhost:9502/rest/v2/categories/category2/values/categoryValue3"
```

Example to drop from category "category2" the value "categoryValue3" as ldap user

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/categories/category2/values/categoryValue3"
```

Comments

The operations that can be performed on the comments of a resource are:

- GET /comments
- POST /comments
- PUT /comments
- DELETE /comments
- GET /comments/resource/{resourceId}
- GET /comments/{commentId}

GET /comments

This API is used to fetch all comments.

Parameters

None

Example to retrieve all comments

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/comments"
```

Example to retrieve all comments as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/ comments"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
COMMENTS" -H "standardSQL:true" -H "system:true"
```

Example to retrieve count of all comments

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/comments?count"
```

POST /comments

This API is used to add a new comment for a resource. The provided JSON should be a list of map entries whose keys are resourcePath, resourceType and comment. The values of the map are values associated with the keys. Special characters must be escaped where required.

Parameters

None

Request Body

```
[
 {
   "resourcePath": "string",
   "resourceType": "string",
   "comment": "string"
 }
]
```

Example to create a new comment on a published resource

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/comments" -H "Content-Type:application/json" -d '[{"resourcePath":"/localhost_9400/services/databases/ds/foo", "resourceType":"TABLE", "comment":"Hello World!"}]'
```

Example to create comments for multiple published resources

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/comments" -H "Content-Type:application/json" -d "[{\"resourcePath\":\"/localhost_9400/services/databases/foo/bar/orders\", \"resourceType\":\"TABLE\", \"comment\":\"alpha\"},{\"resourcePath\":\"/localhost_9400/services/databases/foo/bar/employees\", \"resourceType\":\"TABLE\", \"comment\":\"beta\"}]"
```

Example to create comments for multiple published resources as ldap user

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/comments" -H "Content-Type:application/json" -d "[{\"resourcePath\":\"/localhost_9400/services/databases/foo/bar/orders\", \"resourceType\":\"TABLE\", \"comment\":\"alpha\"}, {\"resourcePath\":\"/localhost_9400/services/databases/foo/bar/employees\", \"resourceType\":\"TABLE\", \"comment\":\"beta\"}]"
```

PUT /comments

This API is used to update comments. The provided JSON should be a map whose keys are the comment IDs for which comment needs to be updated. The corresponding values of the map should contain the updated comments. Special characters must be escaped where required.

Parameters

None

Request Body

```
{
```

```
  "additionalProp1": "string",
  "additionalProp2": "string",
  "additionalProp3": "string"
}
```

Example to update comment on a specific comment ID

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/comments" -H "Content-Type:application/json" -d "{\"1001\":\"changed comment\"}"
```

Example to update comment on a specific comment ID as ldap user

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/comments" -H "Content-Type:application/json" -d "{\"1001\":\"changed comment\"}"
```

Example to update comment on a specific comment ID with X-HTTP-Method-Override

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/comments" -H "X-HTTP-Method-Override:PUT" -H "Content-Type:application/json" -d "{\"1001\":\"changed comment\"}"
```

DELETE /comments

This API is used to delete comments. The provided JSON should be a list of comment IDs that are to be deleted.Special characters must be escaped where required.

Parameters

None

Request Body

```
[
  “string”
]
```

Example to delete comments

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/comments" -H "Content-Type:application/json" -d "[1001]"
```

Example to delete comments as ldap user

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/comments" -H "Content-Type:application/json" -d "[1001]"
```

Example to delete comments with X-HTTP-Method-Override

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/comments" -H "X-HTTP-Method-Override:DELETE" -H "Content-Type:application/json" -d "[1001]"
```

GET /comments/resource/{resourceId}

This API is used to fetch comments for a resource.

Parameters

Name	Description	Parameter Type	Data Type
resourceId	The resource identifier	path	integer

Example to retrieve comments for a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/comments/resource/4567"
```

Example to retrieve comments for a resource as ldap user

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/comments/resource/4567"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
COMMENTS where RESOURCE_ID = 4567" -H "standardSQL:true" -H "system:true"
```

Example to retrieve count of comments for a resource.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/comments/resource/4567?count"
```

GET /comments/{commentId}

This API is used to fetch a specific comment.

Parameters

Name	Description	Parameter Type	Data Type
commentId	The comment id	path	integer

Example to retrieve a comment

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/comments/1234"
```

Example to retrieve a comment as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/comments/1234"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
COMMENTS where COMMENT_ID = 1234" -H "standardSQL:true" -H "system:true"
```

Configs

The following operations can be performed on the different configurations items:

- GET /configs/verify/{item}
- GET /configs/version
- GET /configs/verify/{item}
- PUT /configs/{item}

GET /configs/verify/{item}

This API is used to verify a config item value.

Parameters

Name	Description	Parameter Type	Data Type
item	The config item name to verify	path	string

Example to verify email config items

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/configs/verify/_server_config_email_
smtpHost"
```

Example to verify email config items (as ldap user who has BD_ADMIN rights)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/configs/verify/_server_config_email_smtpHost"
```

GET /configs/version

This API is used to get server version information.

Parameters

None

Example to get server version information

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/configs/version"
```

GET /configs/verify/{item}

This API is used to get a config item value. Only the BD Admin can invoke this

Parameters

Name	Description	Parameter Type	Data Type
item	the config item name to be modified	path	string

Example to get if enable refresh sites metadata when bd server starts up

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/configs/refreshSitesOnStartup"
```

Example to get if enable refresh sites metadata when bd server starts up (as ldap user who has BD_ADMIN rights)

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/configs/refreshSitesOnStartup"
```

PUT /configs/{item}

This API is used to change a config item value. Only the BD Admin can invoke this

Parameters

Name	Description	Parameter Type	Data Type
item	the config item name.	path	string
value		formData	string

Example to enable refresh sites metadata when bd server starts up

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/configs/refreshSitesOnStartup" -d
"value=true"
```

Example to enable refresh sites metadata when bd server starts up (as ldap user who has BD_ADMIN rights)

```
curl -X PUT -u user@ldapDomain:password
"https://localhost:9502/rest/v2/configs/refreshSitesOnStartup" -d "value=true"
```

Example to enable refresh sites metadata when bd server starts up (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/configs/refreshSitesOnStartup" -H "X-
HTTP-Method-Override:PUT" -d "value=true"
```

Data

The following operations can be performed on the SQL queries against the BD server:

- GET /data
- GET /data/query
- POST /data/typed

GET /data

This API is used to execute a SQL query against the BD server. Queries issued for previewing data must be composite SQL queries i.e., non-standard.Returns a set of tuples in a 2-dimensional JSON array: [[row1col1value, row1col2value], [row2col1value, row2col2value]].

Parameters

Name	Description	Parameter Type	Data Type
query	Contains the sql statement. Proper URL encoding is required	query	string
standardSQL	If true, then SQL is standard. If false, then Composite SQL is assumed. Default = true.	query	boolean
system	If true, query is assumed to be a system query. If false, then query is for previewing data. Default = true.	query	boolean

Example to query a system table

```
curl -X GET -u admin:admin
"https://localhost:9502/rest/v2/data?q=select+1%2C+2%2C+%27yo%27%2C+date+%272002-2-
2%27"
```

SQL: select 1, 2, 'yo', date '2002-2-2'

Example to Query a system table (as ldap user)

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/data?q=select+1%2C+2%2C+%27yo%27%2C+date+%272002-2-
2%27"
```

SQL: select 1, 2, 'yo', date '2002-2-2'

Example to Preview data from a published database table.

```
curl -X GET -u admin:admin
"https://localhost:9502/rest/v2/data?q=select+%2A+from+%2Flocalhost_
9400%2Fservices%2Fdatabases%2Ftutorial%2Forders&standardSQL=false&system=false"
```

SQL: select * from /localhost_9400/services/databases/tutorial/orders

GET /data/query

This API is used to execute a SQL query against the BD server. Queries issued for previewing data must be composite SQL queries i.e., non-standard.Returns a set of tuples in a 2-dimensional JSON array: [[row1col1value, row1col2value], [row2col1value, row2col2value]].

Parameters

Name	Description	Parameter Type	Data Type
query	Contains the sql statement. Proper	header	string

Name	Description	Parameter Type	Data Type
	URL encoding is required		
standardSQL	If true, then SQL is standard. If false, then Composite SQL is assumed. Default = true.	header	boolean
system	If true, query is assumed to be a system query. If false, then query is for previewing data. Default = true.	header	boolean

Example to query a system table

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" --header "query:select 1, 2, 'yo', date '2002-2-2'"
```

Example to Query a system table (as ldap user)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/data/query" --header "query:select 1, 2, 'yo', date '2002-2-2'"
```

Example to Preview data from a published database table.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from /localhost_9400/services/databases/tutorial/orders" -H "standardSQL:false" -H "system:false"
```

POST /data/typed

This API is used to execute a SQL query against the BD server. Returns a set of tuples in a 2-dimensional JSON array: [[row1col1value, row1col2value], [row2col1value, row2col2value]].

Parameters

Name	Description	Parameter Type	Data Type
body	Contains the sql statement, an optional 'standardSQL' flag that defaults to true and an optional 'isSystem' flag that defaults to true. Set the 'standardSQL' and 'isSystem' flags to false for performing a data preview, as the data preview query must be a composite query i.e., non-standard.	body	Modelschema

Request Body

```
{
  "standardSQL": true,
  "query": "string",
  "system": true
}
```

Example to query a system table

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/data/typed" -H "Content-
Type:application/json" -d '{"query":"SELECT * from ALL_COMMENTS","standardSQL":true}'
```

Example to Query a system table (as ldap user)

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/data/typed" -H
"Content-Type:application/json" -d '{"query":"SELECT * from ALL_
COMMENTS","standardSQL":true}'
```

Example to Preview data from a published database table.

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/data/typed" -H "Content-
Type:application/json" -d "{\"query\":\"SELECT * from /localhost_
9400/services/databases/tutorial/orders\",\"standardSQL\":false,\"system\":false}"
```

Domains

The following operations can be performed against the ldap domains defined in the Business Directory:

- GET /domains
- POST /domains
- GET /domains/{domainNameString}/groups
- POST /domains/{domainNameString}/groups
- DELETE /domains/{domainNameString}/groups
- PUT /domains/{domainName}
- DELETE /domains/{domainName}

GET /domains

This API is used to retrieve the domains filtered by site or retrieve ldap domains defined by the BD administrator of this business directory instance.

Parameters

Name	Description	Parameter Type	Data Type
site	Filter domains by site. If not specified, then only ldap domains will be returned.	query	string

Example to list domains for site 'test_site'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/domains?site=test_site"
```

Example to list domains for site 'test_site' as ldap user with BD_ADMIN rights

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/domains?site=test_site"
```

Example to only get ldap domains defined within business directory

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/domains"
```

Example to get count of all domains for site 'test_site'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/domains?site=test_site&count"
```

Example to only get count of ldap domains defined within business directory

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/domains?count"
```

POST /domains

This API is used to create a new ldap domain within Business Directory.

Parameters

None

Request Body

```
{
 "name": "string",
 "type": "OTHER",
 "url": "string",
 "login": "string",
 "password": "string",
 "clearpassword": "string",
 "authentication": "string",
 "validation": "TRUE",
 "annotation": "string",
 "typeString": "string",
 "authenticationType": "Simple"
}
```

Example to add a ldap domain named my_bd_ldap_domain

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/domains" -H "Content-Type:application/json" -d "{ \"name\" : \"my_bd_ldap_domain\", \"type\" : \"Other\", \"url\" : \"ldap://172.23.7.75:389/dc=composite,dc=com\", \"login\" : \"cn=Directory Manager\", \"password\" : \"DB63C710940F3BC2045CEDEFFF506F24\", \"clearpassword\" : \"\", \"authentication\" : \"Simple\", \"validation\" : \"TRUE\" }"
```

Example to add a ldap domain named my_bd_ldap_domain (as preexisting ldap user of domain 'ldapDomain' with BD_ADMIN rights)

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/domains" -H "Content-Type:application/json" -d "{ \"name\" : \"my_bd_ldap_domain\", \"type\" : \"Other\", \"url\" : \"ldap://172.23.7.75:389/dc=composite,dc=com\", \"login\" : \"cn=Directory Manager\", \"password\" : \"DB63C710940F3BC2045CEDEFFF506F24\", \"clearpassword\" : \"\", \"authentication\" : \"Simple\", \"validation\" : \"TRUE\" }"
```

Example to add a ldap domain with a password in the clear

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/domains" -H "Content-Type:application/json" -d "{ \"name\" : \"my_bd_ldap_domain\", \"type\" : \"Other\", \"url\" : \"ldap://172.23.7.75:389/dc=composite,dc=com\", \"login\" : \"cn=Directory Manager\", \"password\" : \"\", \"clearpassword\" : \"directory\", \"authentication\" : \"Simple\", \"validation\" : \"TRUE\" }"
```

GET /domains/{domainNameString}/groups

This API is used to contact the ldap server and retrieve all groups. This will retrieve all groups regardless if the group already exists in Business Directory or not.

Parameters

Name	Description	Parameter Type	Data Type
domainNameString	the name of the ldap domain	path	string

Example to retrieve all ldap groups in an existing ldap domain 'my_bd_ldap_domain'.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain/groups"
```

Example to retrieve all ldap groups in an existing ldap domain 'my_bd_ldap_domain' as ldap user with BD_ADMIN rights

```
curl -X GET -u user@my_bd_ldap_domain:password "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain/groups"
```

POST /domains/{domainNameString}/groups

This API is used to add external groups to a ldap domain. If the specified group has already been added, an error will result. If the specified group does not exist in the remote ldap server, an error will result. The provided JSON should be a list of group names to add to the ldap domain.

Parameters

Name	Description	Parameter Type	Data Type
domainNameString	the ldap domainbe modifiedto	path	string

Request Body

```
[
  "string"
]
```

Example to add ldap groups in an existing ldap domain 'my_bd_ldap_domain'.

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain/groups" -H "Content-Type:application/json" -d "[\"buffy\", \"smallville\" ]"
```

Example to add ldap groups in an existing ldap domain 'my_bd_ldap_domain' as ldap user with BD_ADMIN rights.

```
curl -X POST -u user@my_bd_ldap_domain:password "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain/groups" -H "Content-Type:application/json" -d "[\"buffy\", \"smallville\" ]"
```

DELETE /domains/{domainNameString}/groups

This API is used to delete external groups from a ldap domain. If the specified group has not already been added, an error will result. The provided JSON should be a list of groups to remove from the domain.

Parameters

Name	Description	Parameter Type	Data Type
domainNameString	the ldap domain from which groups are to be deleted	path	string

Request Body

```
[
  "string"
]
```

Example to remove ldap groups from an existing ldap domain 'my_bd_ldap_domain'.

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain/groups" -H "Content-Type:application/json" -d "[\"buffy\", \"smallville\" ]"
```

Example to remove ldap groups from an existing ldap domain 'my_bd_ldap_domain' as ldap user with BD_ADMIN rights

```
curl -X DELETE -u user@my_bd_ldap_domain:password "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain/groups" -H "Content-Type:application/json" -d "[\"buffy\", \"smallville\" ]"
```

Example to remove ldap groups from an existing ldap domain 'my_bd_ldap_domain' with X-HTTP-Method-Override

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain/groups" -H "X-HTTP-Method-Override:DELETE" -H "Content-Type:application/json" -d "[\"buffy\", \"smallville\" ]"
```

PUT /domains/{domainName}

This API is used to update an LDAP domain within Business Directory.

Parameters

Name	Description	Parameter Type	Data Type
domainName	name of the ldap domain	path	string

Request Body

```
{
```

```
"name": "string",
"type": "OTHER",
"url": "string",
"login": "string",
"password": "string",
"clearpassword": "string",
"authentication": "string",
"validation": "TRUE",
"annotation": "string",
"typeString": "string",
"authenticationType": "Simple"
}
```


Example to update a ldap domain named my_bd_ldap_domain

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain" -H "Content-Type:application/json" -d "{ \"name\" : \"my_bd_ldap_domain\", \"type\" : \"Other\", \"url\" : \"ldap://172.23.7.75:389/dc=composite,dc=com\", \"login\" : \"cn=Directory Manager\", \"password\" : \"DB63C710940F3BC2045CEDEFFF506F24\", \"clearpassword\" : \"\", \"authentication\" : \"Simple\", \"validation\" : \"TRUE\" }"
```

Example to update a ldap domain named my_bd_ldap_domain as ldap user with BD_ADMIN rights

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain" -H "Content-Type:application/json" -d "{ \"name\" : \"my_bd_ldap_domain\", \"type\" : \"Other\", \"url\" : \"ldap://172.23.7.75:389/dc=composite,dc=com\", \"login\" : \"cn=Directory Manager\", \"password\" : \"DB63C710940F3BC2045CEDEFFF506F24\", \"clearpassword\" : \"\", \"authentication\" : \"Simple\", \"validation\" : \"TRUE\" }"
```

Example to update a ldap domain with a password in the clear; do not validate

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain" -H "Content-Type:application/json" -d "{ \"name\" : \"my_bd_ldap_domain\", \"type\" : \"Other\", \"url\" : \"ldap://172.23.7.75:389/dc=composite,dc=com\", \"login\" : \"cn=Directory Manager\", \"password\" : \"\", \"clearpassword\" : \"directory\", \"authentication\" : \"Simple\", \"validation\" : \"FALSE\" }"
```

DELETE /domains/{domainName}

This API is used to remove an ldap domain.

Parameters

Name	Description	Parameter Type	Data Type
domainName	the name of the ldap domain	path	string

Example to remove an existing ldap domain 'my_bd_ldap_domain'.

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/domains/my_bd_ldap_domain"
```

Example to remove an existing ldap domain 'my_bd_ldap_domain' as ldap user with BD_ADMIN rights

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/domains/my_bd_
ldap_domain"
```

Example to remove an existing ldap domain 'my_bd_ldap_domain' with X-HTTP-Method-Override

```
curl -X POST -u admin:admin -H "X-HTTP-Method-Override:DELETE"
"https://localhost:9502/rest/v2/domains/my_bd_ldap_domain"
```

Groups

The operations that can be performed on the groups defined in the Business Directory are:

- GET /groups
- GET /groups/{groupParam}
- PUT /groups/{groupParam}
- POST /groups/{groupParam}/roles
- DELETE /groups/{groupParam}/roles
- DELETE /groups/{groupParam}/roles/{role}

GET /groups

This API is used to retrieve groups defined in Business Directory. Filter by group, domain or roles.

Parameters

Name	Description	Parameter Type	Data Type
groups		query	Array(string)
domain		query	string
rolefilter	Optional. Filter groups by the specified roles.	query	Array(string)

Example to retrieve groups from ldap domain 'my_bd_ldap_domain'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/groups?domain=my_bd_ldap_domain"
```

Example to retrieve groups from ldap domain 'my_bd_ldap_domain' as ldap user with BD_ADMIN rights

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/groups?domain=ldapDomain"
```

Example to retrieve groups 'xyz' and 'abcd'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/groups?groups=xyz&groups=abcd"
```

Example to retrieve groups that have roles BD_ADMIN and MANAGE_CATEGORY

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/groups?roleFilter=BD_
ADMIN&roleFilter=MANAGE_CATEGORY"
```

Example to retrieve count of all groups from ldap domain 'my_bd_ldap_domain'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/groups?domain=my_bd_ldap_domain&count"
```

Example to retrieve count of all groups that have roles BD_ADMIN and MANAGE_CATEGORY

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/groups?roleFilter=BD_ADMIN&roleFilter=MANAGE_CATEGORY&count"
```

GET /groups/{groupParam}

This API is used to retrieve a group defined in Business Directory.

Parameters

Name	Description	Parameter Type	Data Type
groupParam	group in group@domain format.	path	string

Example to retrieve groups from ldap domain 'my_bd_ldap_domain'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/groups?domain=my_bd_ldap_domain"
```

Example to retrieve groups from ldap domain 'my_bd_ldap_domain' as ldap user with BD_ADMIN rights

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/groups/myGroup@ldapDomain"
```

PUT /groups/{groupParam}

This API is used to add or remove roles from groups. The roles include READ_ALL_RESOURCES, APPLY_COMMENT, MODERATE_COMMENT, MANAGE_CATEGORY, APPLY_CATEGORY, APPLY_CUSTOM_PROPERTIES, MANAGE_CUSTOM_PROPERTIES, BD_ADMIN, and ACCESS_DIRECTORY.Additionally, you can modify group annotations

Parameters

Name	Description	Parameter Type	Data Type
groupParam	group in group@domain format.	path	string

Request Body

```
{
  "groupName": "string",
  "domain": "string",
  "annotation": "string",
  "roles": [
    "string"
  ]
}
```

Example to set BD_ADMIN and MODERATE_COMMENT as only roles for group 'g2' in 'ad2003' domain; set annotation

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/groups/g2@ad2003" -H "Content-Type:application/json" -d "{ \"annotation\" : \"new annotation\", \"roles\" : [ \"BD_ADMIN\", \"MODERATE_COMMENT\" ]}"
```

Example to set BD_ADMIN and MODERATE_COMMENT as only roles for group 'g2' in 'ad2003' domain; set annotation as ldap user with BD_ADMIN rights

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/groups/g2@ad2003" -H "Content-Type:application/json" -d "{ \"annotation\" : \"new annotation\", \"roles\" : [ \"BD_ADMIN\", \"MODERATE_COMMENT\" ]}"
```

Example to set BD_ADMIN and MODERATE_COMMENT as only roles for group 'g2' in 'ad2003' domain; set annotation awith X-HTTP-Method-Override

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/groups/g2@ad2003" -H "X-HTTP-Method-Override:PUT" -H "Content-Type:application/json" -d "{ \"annotation\" : \"new annotation\", \"roles\" : [ \"BD_ADMIN\", \"MODERATE_COMMENT\" ]}"
```

POST /groups/{groupParam}/roles

This API is used to add roles to a group. The roles include READ_ALL_RESOURCES, APPLY_COMMENT, MODERATE_COMMENT, MANAGE_CATEGORY, APPLY_CATEGORY, APPLY_CUSTOM_PROPERTIES, MANAGE_CUSTOM_PROPERTIES, BD_ADMIN, and ACCESS_DIRECTORY.

Parameters

Name	Description	Parameter Type	Data Type
groupParam	group in group@domain format.	path	string

Request Body

```
[
  "string"
]
```

Example to add BD_ADMIN and MODERATE_COMMENT to group 'g2'

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/groups/g2@ad2003/roles" -H "Content-Type:application/json" -d "[ \"BD_ADMIN\", \"MODERATE_COMMENT\" ]"
```

Example to add BD_ADMIN and MODERATE_COMMENT to group 'g2' as ldap user with BD_ADMIN rights

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/groups/g2@ad2003/roles" -H "Content-Type:application/json" -d "[ \"BD_ADMIN\", \"MODERATE_COMMENT\" ]"
```

DELETE /groups/{groupParam}/roles

This API is used to remove all roles from a group.

Parameters

Name	Description	Parameter Type	Data Type
groupParam	group in group@domain format.	path	string

Example to remove MODERATE_COMMENT from group 'g2'

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/groups/g2@ad2003/roles"
```

Example to remove MODERATE_COMMENT from group 'g2' as ldap user with BD_ADMIN rights

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/groups/g2@ad2003/roles"
```

Example to remove MODERATE_COMMENT from group 'g2' with X-HTTP-Method-Override:DELETE

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/groups/g2@ad2003/roles" -H "X-HTTP-Method-Override:DELETE"
```

DELETE /groups/{groupParam}/roles/{role}

This API is used to remove role from a group. The roles include READ_ALL_ RESOURCES, APPLY_COMMENT, MODERATE_COMMENT, MANAGE_ CATEGORY, APPLY_CATEGORY, APPLY_CUSTOM_PROPERTIES, MANAGE_ CUSTOM_PROPERTIES, BD_ADMIN, and ACCESS_DIRECTORY.

Parameters

Name	Description	Parameter Type	Data Type
groupParam	group in group@domain format.	path	string
role	role	path	string

Example to remove MODERATE_COMMENT from group 'g2'

```
curl -X DELETE -u admin:admin
"https://localhost:9502/rest/v2/groups/g2@ad2003/roles/MODERATE_COMMENT"
```

Example to remove MODERATE_COMMENT from group 'g2' as ldap user with BD_ADMIN rights

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/groups/g2@ad2003/roles/MODERATE_COMMENT"
```

Example to remove MODERATE_COMMENT from group 'g2' (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin
"https://localhost:9502/rest/v2/groups/g2@ad2003/roles/MODERATE_COMMENT" -H "X-HTTP-
Method-Override:DELETE"
```

Metadata

The following operations can be performed on the metadata of the Business Directory:

- GET /metadata
- PUT /metadata
- GET /metadata/annotation/{tableName}

- GET /metadata/ws

GET /metadata

This API is used to export full metadata of Business Directory to a file.

Parameters

Name	Description	Parameter Type	Data Type
encryptionPassword	Encryption password	query	string

Example to export metadata

```
curl -u "admin:admin" -X GET
"https://localhost:9502/rest/v2/metadata?encryptionPassword=testPassword" -o export001.car
```

Example to export metadata as ldap user with BD_ADMIN and ACCESS_DIRECTORY rights

```
curl -u "user@ldapDomain:password" -X GET
"https://localhost:9502/rest/v2/metadata?encryptionPassword=testPassword" -o export001.car
```

PUT /metadata

This API is used to import full metadata of Business Directory from a file.

Parameters

Name	Description	Parameter Type	Data Type
file	Name of the file to be imported.	query	object

Name	Description	Parameter Type	Data Type
encryptionPassword	Encryption Password	query	string
overwrite	Option to indicate whether to overwrite	query	boolean
ignoreEncryption	Option to indicate whether encryption errors can be ignored	query	boolean

Note: If the option ***ignoreEncryption*** is used, then all backup data will be imported regardless of whether a valid encryption key was provided. This means that the import will not fail. This option can be used to allow partially importing any backed up data. However, the import process will only import data that is not encrypted or can be decrypted using the provided encryption key. All encrypted portions of the backup data that cannot be decrypted will be imported as empty values and the import will otherwise succeed.

This affects all encrypted values in the backup data, which includes, but is not limited to data source and LDAP domain connection passwords.

Example to import metadata

```
curl -u "admin:admin"  -i  -F "encryptionPassword=testPassword" -F "overwrite=true" -F
"file=@export001.car" -X PUT "https://localhost:9502/rest/v2/metadata"
```

Example to import metadata as ldap user with BD_ADMIN and ACCESS_DIRECTORY rights

```
curl -u "user@ldapDomain:password"  -i  -F "encryptionPassword=testPassword" -F
"overwrite=true" -F "file=@export001.car" -X PUT "https://localhost:9502/rest/v2/metadata"
```

Example to import metadata (X-HTTP-Method-Override)

```
curl -u "admin:admin"  -i -H "X-HTTP-Method-Override:PUT"  -F
"encryptionPassword=testPassword" -F "overwrite=true" -F "file=@export001.car" -X POST
"https://localhost:9502/rest/v2/metadata"
```

Example to import metadata (ignoring encryption errors)

```
curl -u "admin:admin"  -i  -F "encryptionPassword=testPassword" -F "overwrite=true" -F
"ignoreEncryption=true" -F "file=@export001.car" -X PUT "https://localhost:9502/rest/v2/metadata"
```

GET /metadata/annotation/{tableName}

This API is used to retrieve annotation for a given table

Parameters

Name	Description	Parameter Type	Data Type
tableName		path	string

Example

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/metadata/annotation/SYS_SITES"
```

GET /metadata/ws

This API is used to get web service metadata, such as url, soap version, protocol etc. Returns a set of tuples in a JSON array: [{key:value, key:value, key:value}, {key:value, key:value, key:value}].

Parameters

Name	Description	Parameter Type	Data Type
clientWSPath	client web service path	query	string

Example

```
curl -X GET -u admin:admin
"https://localhost:9502/rest/v2/metadata/ws?clientWSPath=/services/webservices/localhost_
9400/soap_test/binding_test"
```

propertyGroups

Following are the REST API operations for the propertyGroups:

- GET /propertyGroups
- POST /propertyGroups
- POST /propertyGroups/properties
- DELETE /propertyGroups/properties
- GET /propertyGroups/properties/sorted
- GET /propertyGroups/sorted
- GET /propertyGroups/{groupName}
- PUT /propertyGroups/{groupName}
- DELETE /propertyGroups/{groupName}
- GET /propertyGroups/{groupName}/associations
- POST /propertyGroups/{groupName}/associations
- DELETE /propertyGroups/{groupName}/associations/{association}
- GET /propertyGroups/{groupName}/properties
- GET /propertyGroups/{groupName}/properties/sorted
- GET /propertyGroups/{groupName}/properties/{propertyName}

- PUT /propertyGroups/{groupName}/properties/{propertyName}
- DELETE /propertyGroups/{groupName}/properties/{propertyName}
- GET /propertyGroups/{groupName}/properties/{propertyName}/resources
- POST /propertyGroups/{groupName}/properties/{propertyName}/resources
- DELETE /propertyGroups/{groupName}/properties/{propertyName}/ resources/{resourceType}/{resourcePath}
- PUT /propertyGroups/{groupName}/properties/{propertyName}/ {propertyOrder}
- GET /propertyGroups/{groupName}/resources
- PUT /propertyGroups/{groupName}/{groupOrder}

GET /propertyGroups

This API is used to fetch all property groups.

Parameters

None

Example to list all property groups

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups"
```

Example to list all property groups as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups"
```

Example to get count all property groups

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups?count"
```

POST /propertyGroups

This API is used to create a new custom property group or groups.

Parameters

None

Request Body

```
[
 {
  "name": "string",
  "annotation": "string",
  "associations": [
   "string"
  ],
  "location": "PROPERTIES_TAB"
 }
]
```

Example to create a new custom property group named custom_ propert_group1

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/propertyGroups" -H "Content-Type:application/json" -d "[{ \"name\" : \"custom_property_group1\" }]"
```

Example to create a new custom property group named custom_ propert_group1 as ldap user

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups" -H "Content-Type:application/json" -d "[{ \"name\" : \"custom_property_group1\" }]"
```

Example to create a new custom property group named custom_ property_group2 with an association to /test_site/services/databases/ds & /test_site/services/databases/examples.

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/propertyGroups" -H "Content-Type:application/json" -d "[{ \"name\" : \"custom_property_group2\", \"associations\" : [ \"/test_ site/services/databases/ds\", \"/test_site/services/databases/examples\"] }]"
```

POST /propertyGroups/properties

This API is used to create a new custom property by specifying its name, type, extended type, and default value(s).

Parameters

None

Request Body

```
[
 {
   "Property Name": "string",
   "Property Type": "STRING_TYPE",
   "Property Extended Type": [
```

```
    "string"
  ],
  "Property Group": "string",
  "Property Default Value": [
    "string"
  ]
 }
]
```

Example to create a new property for property group foo

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/properties" -H
"Content-Type:application/json" -d "[{\"name\" : \"custom1\", \"propertyGroup\" : \"foo\", \"type\" :
\"STRING_TYPE\" }]"
```

Example to create a new SINGLE_ENUMERATION_TYPE property

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/properties" -H
"Content-Type:application/json" -d "[{\"name\" : \"custom2\", \"propertyGroup\" : \"foo\", \"type\" :
\"SINGLE_ENUMERATION_TYPE\", \"extendedType\" : [ \"a\", \"b\", \"b2\" ], \"defaultValue\" : [
\"b2\" ] }]"
```

Example to create a new MULTI_ENUMERATION_TYPE property with multiple default values

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/properties" -H "Content-Type:application/json" -d "[{\"name\" : \"custom3\", \"propertyGroup\" : \"foo\", \"type\" : \"MULTI_ENUMERATION_TYPE\", \"extendedType\" : [ \"a\", \"b\", \"b2\" ], \"defaultValue\" : [ \"b\", \"b2\" ] }]"
```

Example to retrieve custom properties

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/data/typed" -H "Content-Type:application/json" -d "{ \"query\" : \"SELECT CUSTOM_PROPERTY_ID, CUSTOM_PROPERTY_NAME, CUSTOM_PROPERTY_TYPE, CUSTOM_PROPERTY_EXTENDED_TYPE, CUSTOM_PROPERTY_GROUP, CUSTOM_PROPERTY_DEFAULT_VALUE FROM /services/databases/system/all_custom_properties\" }"
```

Example to retrieve custom properties as ldap user

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/data/typed" -H "Content-Type:application/json" -d "{ \"query\" : \"SELECT CUSTOM_PROPERTY_ID, CUSTOM_PROPERTY_NAME, CUSTOM_PROPERTY_TYPE, CUSTOM_PROPERTY_EXTENDED_TYPE, CUSTOM_PROPERTY_GROUP, CUSTOM_PROPERTY_DEFAULT_VALUE FROM /services/databases/system/all_custom_properties\" }"
```

DELETE /propertyGroups/properties

This API is used to delete custom properties from the Business Directory.

Parameters

None

Request Body

```
[
```

```
 {
```

```
    "Property Name": "string",
    "Property Type": "STRING_TYPE",
    "Property Extended Type": [
      "string"
    ],
    "Property Group": "string",
    "Property Default Value": [
      "string"
    ]
  }
]
```

Example to delete a custom property named 'custom1' in custom property group 'foo'; also delete property named 'custom2' in group 'bar'

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/properties" -H "Content-Type:application/json" -d "[{ \"name\" : \"custom1\", \"propertyGroup\" : \"foo\" }, { \"name\" : \"custom2\", \"propertyGroup\" : \"bar\" }]"
```

Example to delete a custom property named 'custom1' in custom property group 'foo'; also delete property named 'custom2' in group 'bar' (as ldap user)

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/propertyGroups/properties" -H "Content-Type:application/json" -d "[{
\"name\" : \"custom1\", \"propertyGroup\" : \"foo\" }, { \"name\" : \"custom2\", \"propertyGroup\" :
\"bar\" }]"
```

GET /propertyGroups/properties/sorted

This API is used to get all sorted property ids.

Parameters

None

Example to get all sorted property ids in all groups

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/properties/sorted"
```

GET /propertyGroups/sorted

This API is used to get all sorted group ids.

Parameters

None

Example to get all sorted group ids

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/sorted"
```

GET /propertyGroups/{groupName}

This API is used to fetch a single property group.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property Group	path	string

Example to fetch 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1"
```

Example to Fetch 'custom_property_group1' property group (as ldap user)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1"
```

PUT /propertyGroups/{groupName}

This API is used to update a group's annotation and associations.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property Group	path	string

Request Body

```
{
```

```
  "name": "string",
```

```
 "annotation": "string",
 "associations": [
  "string"
 ],
 "location": "PROPERTIES_TAB"
}
```

Example to update a custom property group named custom_property_group2 with an association to /test_site/services/databases/ds & /test_site/services/databases/examples.

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group2" -H "Content-Type:application/json" -d "{ \"name\" : \"custom_property_group2\", \"associations\" : [ \"/test_site/services/databases/ds\", \"/test_site/services/databases/examples\"] }"
```

Example to update a custom property group named custom_property_group2 with an association to /test_site/services/databases/ds & /test_site/services/databases/examples as ldap user

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group2" -H "Content-Type:application/json" -d "{ \"name\" : \"custom_property_group2\", \"associations\" : [ \"/test_site/services/databases/ds\", \"/test_site/services/databases/examples\"] }"
```

DELETE /propertyGroups/{groupName}

This API is used to remove a custom property group.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property Group	path	string

Example to remove a custom property group named custom_propert_group1

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1"
```

Example to remove a custom property group named custom_propert_group1 (as ldap user)

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1"
```

GET /propertyGroups/{groupName}/associations

This API is used to fetch resource path associations for a single property group.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property Group	path	string

Example to fetch resource path associations for 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/associations"
```

Example to fetch resource path associations for 'custom_property_group1' property group (as ldap user)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/associations"
```

POST /propertyGroups/{groupName}/associations

This API is used to add an association to a custom property group.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The name of the property group to which to add an association	path	string

Request Body

```
[
  "string"
]
```

Example to update a custom property group named custom_property_group2 with an association to /test_site/services/databases/ds & /test_site/services/databases/examples.

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group2/associations" -H "Content-Type:application/json" -d "[\"/test_site/services/databases/ds\", \"/test_site/services/databases/examples\" ]"
```

Example to update a custom property group named custom_property_group2 with an association to /test_site/services/databases/ds & /test_site/services/databases/examples as ldap user

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group2/associations" -H "Content-Type:application/json" -d "[\"/test_site/services/databases/ds\", \"/test_site/services/databases/examples\" ]"
```

ociations"

DELETE /propertyGroups/{groupName}/associations/{association}

This API is used to remove an association from a custom property group.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The name of the property group from which to remove an association	path	string
association	The association to remove. URL encode the path	path	string

Example to update a custom property group named custom_property_group2 by removing its association with /test_site/services/databases/ds

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group2/associations/%2Ftest_site%2Fservices%2Fdatabases%2Fds"
```

Example to update a custom property group named custom_property_group2 by removing its association with /test_site/services/databases/ds as ldap user

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group2/associations/%2Ftest_site%2Fservices%2Fdatabases%2Fds"
```

GET /propertyGroups/{groupName}/properties

This API is used to fetch all properties for a single property group.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property Group	path	string

Example to fetch properties for 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/properties"
```

Example to fetch count of all properties for 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/properties?count"
```

Example to Fetch count of all properties for 'custom_property_group1' property group (as ldap user)

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/properties?count"
```

GET /propertyGroups/{groupName}/properties/sorted

This API is used to get all sorted property ids in a group.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property Group	path	string

Example to get all sorted property ids in custom_property_group1

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_
group1/properties/sorted"
```

GET /propertyGroups/{groupName}/properties/{propertyName}

This API is used to fetch a single custom property.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property Group	path	string
propertyName	The Property Name	path	string

Example to fetch property 'custom1' for the 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_
group1/properties/custom1"
```

Example to Fetch property 'custom1' for the 'custom_property_group1' property group (as ldap user)

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/properties/custom1"
```

PUT /propertyGroups/{groupName}/properties/{propertyName}

This API is used to update a custom property's type, extended type, and default value.

Parameters

Name	Description	Parameter Type	Data Type
groupName	group that contains the property	path	string
propertyName	Name of the custom property to be updated.	path	string

Example to update custom property 'custom1' type, rename to 'custom1alpha', update enumeration, and set default value

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1"
-H "Content-Type:application/json" -d "{\"name\" : \"custom1alpha\", \"propertyGroup\" : \"foo\",
\"type\" : \"MULTI_ENUMERATION_TYPE\", \"extendedType\" : [ \"a\", \"b\", \"b2\" ], \"defaultValue\"
: [ \"b\", \"b2\" ] }"
```

Example to update custom property 'custom1' type, rename to 'custom1alpha', update enumeration, and set default value (as ldap user)

```
curl -X PUT -u user@ldapDomain:password
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1" -H "Content-
Type:application/json" -d "{\"name\" : \"custom1alpha\", \"propertyGroup\" : \"foo\", \"type\" :
\"MULTI_ENUMERATION_TYPE\", \"extendedType\" : [ \"a\", \"b\", \"b2\" ], \"defaultValue\" : [ \"b\",
\"b2\" ] }"
```

DELETE /propertyGroups/{groupName}/properties/{propertyName}

This API is used to delete a single custom property from the Business Directory.

Parameters

Name	Description	Parameter Type	Data Type
groupName	group that contains the property	path	string
propertyName	Name of the custom property to be removed.	path	string

Example to delete a custom property named 'custom1' from custom property group 'foo'

```
curl -X DELETE -u admin:admin
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1"
```

Example to delete a custom property named 'custom1' from custom property group 'foo' as ldap user

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1"
```

GET /propertyGroups/{groupName}/properties/{propertyName}/resources

This API is used to Fetch recorded classifications for a single custom property. Note resources classified with the default value for a property may or may not appear in the results.

Parameters

Name	Description	Parameter Type	Data Type
groupName	group that contains the property	path	string
propertyName	Name of the property.	path	string

Example to fetch classifications for the 'custom1' property in the 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/properties/custom1/resources"
```

Example to fetch count of all classifications for the 'custom1' property in the 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/properties/custom1/resources?count"
```

Example to fetch count of all classifications for the 'custom1' property in the 'custom_property_group1' property group (as ldap user)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/properties/custom1/resources?count"
```

POST /propertyGroups/{groupName}/properties/{propertyName}/resources

This API is used to apply a custom property onto a Business Directory Resource. Set the custom property value for a resource.

Parameters

Name	Description	Parameter Type	Data Type
groupName	group that contains the property	path	string
propertyName	Name of the custom property to be applied.	path	string

Request Body

```
{
 "resourceBeans": [
  {
   "resourcePath": "string",
   "resourceType": "string"
  }
 ],
 "propertyValue": [
```

```
  "string"
 ]
}
```

Example to set custom property 'custom1' to resource 'orders'

```
curl -X POST -u admin:admin
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1/resources" -H "Content-
Type:application/json" -d "{ \"resourceBeans\" : [ { \"resourcePath\" : \"/test_
site/services/databases/tutorial/orders\", \"resourceType\" : \"TABLE\" }], \"propertyValue\" : [
\"One\", \"Two\" ] }"
```

Example to set custom property 'custom1' to resource 'orders' as ldap user

```
curl -X POST -u user@ldapDomain:password
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1/resources" -H "Content-
Type:application/json" -d "{ \"resourceBeans\" : [ { \"resourcePath\" : \"/test_
site/services/databases/tutorial/orders\", \"resourceType\" : \"TABLE\" }], \"propertyValue\" : [
\"One\", \"Two\" ] }"
```

DELETE /propertyGroups/{groupName}/properties/{propertyName}/resources/{resourceType}/{resourcePath}

This API is used to remove a custom property value from a Business Directory Resource.

Parameters

Name	Description	Parameter Type	Data Type
groupName	group that contains the	path	string

Name	Description	Parameter Type	Data Type
	property		
propertyName	Name of the custom property to be removed.	path	string
resourceType	The resource type for the resource from which to remove the classification.	path	string
resourcePath	The resource path for the resource from which to remove the classification.	path	string

Example to clear custom property 'custom1' from resource 'orders'

```
curl -X DELETE -u admin:admin
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1/resources/TABLE/%2Ftest_
site%2Fservices%2Fdatabases%2Ftutorial%2Forders"
```

Example to clear custom property 'custom1' from resource 'orders' (as ldap user)

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1/resources/TABLE/%2Ftest_
site%2Fservices%2Fdatabases%2Ftutorial%2Forders"
```

PUT /propertyGroups/{groupName}/properties/{propertyName}/{propertyOrder}

This API is used to update a custom property's order within the group. The order numbers of properties start from 0.

Parameters

Name	Description	Parameter Type	Data Type
groupName	group that contains the property	path	string
propertyName	Name of the custom property to be updated.	path	string
propertyOrder	A new order number for the existing custom property in the group.	path	integer

Example to update custom property 'custom1' within group 'foo', to set its ordinal number to 5.

```
curl -X PUT -u admin:admin
"https://localhost:9502/rest/v2/propertyGroups/foo/properties/custom1/5" -H "Content-
Type:application/json"
```

GET /propertyGroups/{groupName}/resources

This API is used to fetch recorded classifications for a single custom property group. Please note resources classified with the default value for a property may or may not appear in the results.

Parameters

Name	Description	Parameter Type	Data Type
groupName	The Property group	path	string

Example to fetch classifications for the 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/resources"
```

Example to fetch count of all classifications for the 'custom_property_group1' property group

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/resources?count"
```

Example to fetch count of all classifications for the 'custom_property_group1' property group (as ldap user)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/propertyGroups/custom_property_group1/resources?count"
```

PUT /propertyGroups/{groupName}/{groupOrder}

This API is used to update a group's ordinal number.

Parameters

Name	Description	Parameter Type	Data Type
groupName	group that contains the property	path	string
groupOrder	A new order number for the existing custom property group.	path	integer

Example to update a custom property group named custom_property_group2 with a new order number

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/propertyGroups/custom_property_group2/4" -H "Content-Type:application/json"
```

Resources

The different operations that can be performed on the Business Directory resources are:

- GET /resources
- GET /resources/{resourceType}/guid/{resourceGuid}
- GET /resources/{resourceType}/guid/{resourceGuid}/columns
- GET /resources/{resourceType}/guid/{resourceGuid}/columns/id/{columnId}
- GET /resources/{resourceType}/guid/{resourceGuid}/columns/ {columnName}
- GET /resources/{resourceType}/guid/{resourceGuid}
- GET /resources/{resourceType}/guid/{resourceGuid}/columns
- GET /resources/{resourceType}/guid/{resourceGuid}/columns/id/{columnId}
- GET /resources/{resourceType}/guid/{resourceGuid}/columns/ {columnName}
- GET /resources/{resourceType}/{resourcePath}
- GET /resources/{resourceType}/{resourcePath}/categories
- POST /resources/{resourceType}/{resourcePath}/categories
- DELETE /resources/{resourceType}/{resourcePath}/categories
- DELETE /resources/{resourceType}/{resourcePath}/categories/{categoryName}/values/{valueName}
- GET /resources/{resourceType}/{resourcePath}/columns
- GET /resources/{resourceType}/{resourcePath}/columns/id/{columnId}
- GET /resources/{resourceType}/{resourcePath}/columns/{columnName}

GET /resources

This API is used to retrieve a resource by type and path, id, or guid. Execute a free-text search for resources. Returns a set of tuples in a 2-dimensional JSON array: [[row1col1value, row1col2value], [row2col1value, row2col2value]].If the 'includeFilterCounts' parameter is true and the 'q' parameter is included (even if empty), then the SearchCount procedure is used to fetch a list of valid filters for the search, and the result includes 'resources' and 'counts' objects.

Parameters

Name	Description	Parameter Type	Data Type
q	Free-text search criteria	query	string
resourceTypeFilter	Filter by resource types	query	string
siteNameFilter	filter by site names	query	string
datasourceIdFilter	Filter by datasource ids	query	string
categoryValueIdFilter	Filter by category value ids	query	string
watchFilter	Filter by user's watches	query	boolean
includeFilterCounts	Include filter counts	query	boolean
offset	Offset of first result entry to return. Default = 0.	query	integer
limit	Maximum number of entries to return.	query	integer

Name	Description	Parameter Type	Data Type
	Default = 2147483647.		
type	The type of the resource to retrieve	query	string
path	The path of the resource to retrieve. URL Encode any slashes in the path.	query	string
id		query	integer
guid		query	string

Example to search for resources

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/?q=foo&limit=10&offset=0"
```

Example to search for resources as ldap user

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/resources/?q=foo&limit=10&offset=0"
```

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources?type=database_
table&id=410266"
```

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources?type=database_
table&path=%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo"
```

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources?type=database_
table&path=%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo"
```

Example to get a count of resources searched for

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/?q=foo&count"
```

GET /resources/{resourceType}/guid/{resourceGuid}

This API is used to retrieve a resource by guid.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource to retrieve.	path	string
resourceGuid	The guid of the resource to retrieve.	path	integer

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce"
```

GET /resources/{resourceType}/guid/{resourceGuid}/columns

This API is used to retrieve columns of a table by table type and guid.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource for which to retrieve columns	path	string
resourceGuid	The guid of the resource	path	string

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce/columns"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce/columns"
```

GET /resources/{resourceType}/guid/{resourceGuid}/columns/id/{columnId}

This API is used to retrieve a table column by table type and guid and column ID.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource for which to retrieve columns	path	string
resourceGuid	The guid of the resource	path	string
columnId	The column ID	path	integer

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce/columns/id/11515"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce/columns/id/11515"
```

GET /resources/{resourceType}/guid/{resourceGuid}/columns/{columnName}

This API is used to retrieve a table column by table type and guid and column name.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource for which to retrieve	path	string

Name	Description	Parameter Type	Data Type
	columns		
resourceGuid	The guid of the resource	path	string
columnName	The column name	path	string

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce/columns/m_integer"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/guid/09c50755-f0cb-46c3-83a0-b68d2d1c9cce/columns/m_integer"
```

GET /resources/{resourceType}/id/{resourceId}

This API is used to retrieve a Retrieve a resource by ID.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource to retrieve	path	string
resourceId	The ID of the resource to retrieve.	path	integer

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_table/id/410266"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/id/410266"
```

GET /resources/{resourceType}/id/{resourceId}/columns

This API is used to retrieve columns of a table by table type and ID.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource to retrieve	path	string
resourceId	The ID of the resource to retrieve.	path	integer

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/id/410266/columns"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/id/410266/columns"
```

GET /resources/{resourceType}/id/{resourceId}/columns/id/{columnId}

Retrieve a table column by table type and ID and column ID.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource to retrieve	path	string
resourceId	The ID of the resource to retrieve.	path	integer
columnId	The column ID	path	integer

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/id/410266/columns/id/11515"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/id/410266/columns/id/11515"
```

GET /resources/{resourceType}/id/{resourceId}/columns/{columnName}

Retrieve a table column by table type and id and column name.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource to retrieve	path	string
resourceId	The ID of the resource to retrieve.	path	integer
columnName	The column name	path	string

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/id/410266/columns/m_integer"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/id/410266/columns/m_integer"
```

GET /resources/{resourceType}/{resourcePath}

This API is used to retrieve a resource by type and path.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource to retrieve	path	string

Name	Description	Parameter Type	Data Type
resourcePath	The path of the resource to retrieve. URL Encode any slashes in the path.	path	string

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo"
```

GET /resources/{resourceType}/{resourcePath}/categories

This API is used to fetch category classifications for a particular resource.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource for which to retrieve classifications	path	string
resourcePath	The path of the resource for which to retrieve classifications. URL Encode any slashes in the path.	path	string

Example to list classifications for the resource "/localhost_9400/services/ databases/ ds/ foo"

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories"
```

Example to list classifications for the resource "/localhost_9400/services/ databases/ ds/ foo" as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories"
```

Equivalent system query

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/data/typed" -H "Content-
Type:application/json" -d "{\"query\":\"select * from ALL_CLASSIFICATIONS where RESOURCE_
ID in (select RESOURCE_ID from ALL_BD_RESOURCES where RESOURCE_TYPE =
'database_table' and PARENT_PATH || '/' || RESOURCE_NAME = '/localhost_
9400/services/databases/ds/foo' )\",\"standardSQL\":true}"
```

Example to count of all classifications for the resource "/localhost_9400/ services/ databases/ds/foo"

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories&count"
```

POST /resources/{resourceType}/{resourcePath}/categories

This API is used to classify a resource by associating them with category/value pairs.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource to classify	path	string
resourcePath	The path of the resource to classify. URL Encode any slashes in the path.	path	string

Request Body

```
[
 {
  "categoryName": "string",
  "categoryValues": [
   "string"
  ]
 }
]
```

Example to classify a resource with the products/bestseller and products/favorite category/value pairs

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories" -H "Content-
Type:application/json" -d "[{\"categoryName\":\"products\", \"categoryValues\": [\"bestseller\",
\"favorite\"]}]"
```

Example to classify a resource with the products/bestseller and products/favorite category/value pairs (as ldap user)

```
curl -X POST -u user@ldapDomain:password
"https://localhost:9502/rest/v2/resources/database_table/%2Flocalhost_
9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories" -H "Content-
Type:application/json" -d "[{\"categoryName\":\"products\", \"categoryValues\":
[\"bestseller\", \"favorite\"]}]"
```

DELETE /resources/{resourceType}/{resourcePath}/categories

This API is used to remove resource classifications.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource from which a classification is to be removed.	path	string
resourcePath	The path of the resource from which a classification is to be removed. URL Encode any slashes in the path	path	string

Example to remove the products/bestseller and products/favorite category/value pair classification from a resource

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories" -H "Content-
Type:application/json" -d "[{\"categoryName\":\"products\", \"categoryValues\": [\"bestseller\",
\"favorite\"]}]"
```

Example to remove the products/bestseller and products/favorite category/value pair classification from a resource (as ldap user)

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/resources/database_table/%2Flocalhost_
9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories" -H "Content-Type:application/json" -d
"[{\"categoryName\":\"products\", \"categoryValues\": [\"bestseller\", \"favorite\"]}]"
```

DELETE /resources/{resourceType}/{resourcePath}/categories/{categoryName}/values/{valueName}

This API is used to remove a single resource classification.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource from which a classification is to be removed.	path	string
resourcePath	The path of the resource from which a classification is to be removed. URL Encode any slashes in the path.	path	string
categoryName	The name of the category	path	string

Name	Description	Parameter Type	Data Type
	associated with the classification to remove.		
valueName	The name of the category value associated with the classification to remove.	path	string

Example to remove a classification from a resource

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_
9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories/products/values/bestseller"
```

Example to remove a classification from a resource as ldap user

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/resources/database_table/%2Flocalhost_
9400%2Fservices%2Fdatabases%2Fds%2Ffoo/categories/products/values/bestseller"
```

GET /resources/{resourceType}/{resourcePath}/columns

This API is used to retrieve columns of a table by table type and path.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource for which to retrieve columns	path	string
resourcePath	The path of the resource.	path	string

Name	Description	Parameter Type	Data Type
	URL Encode any slashes in the path.		

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/columns"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/columns"
```

GET /resources/{resourceType}/{resourcePath}/columns/id/{columnId}

This API is used to retrieve a table column by table type and path and column ID.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource for which to retrieve columns	path	string
resourcePath	The path of the resource. URL Encode any slashes in the path.	path	string
columnId	The column ID	path	integer

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/columns/id/11515"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/columns/id/11515"
```

GET /resources/{resourceType}/{resourcePath}/columns/{columnName}

This API is used to retrieve a table column by table type and path and column name.

Parameters

Name	Description	Parameter Type	Data Type
resourceType	The type of the resource for which to retrieve columns	path	string
resourcePath	The path of the resource. URL Encode any slashes in the path.	path	string
columnName	The column name	path	string

Example to retrieve a resource

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/columns/m_integer"
```

Example to retrieve a resource as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/resources/database_
table/%2Flocalhost_9400%2Fservices%2Fdatabases%2Fds%2Ffoo/columns/m_integer"
```

Security

The Security operations that can be performed on the Business Directory resources are:

- GET /security/backup_encryption_settings
- GET /security/generateUUID
- POST /security/import_encryption_settings
- GET /security/systemEncryption
- PUT /security/systemEncryption

GET /security/backup_encryption_settings

This API is used to backup the encryption settings to a password protected file for server recovery in case of emergency.

Parameters

Name	Description	Parameter Type	Data Type
encryptionPassword	Password used by the encryption utility	query	string

Example to backup the encryption settings

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/security/backup_encryption_
settings?encryptionPassword=testPassword" -o backup_encryption_settings.txt
```

GET /security/generateUUID

This API is used to get the system randomly generated UUID.

Parameters

None

Example to get the system randomly generated UUID.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/security/generateUUID"
```

POST /security/import_encryption_settings

This API is used to restore the encryption settings from the backup file. You must know the password that was used to protect the backup file.

Parameters

Name	Description	Parameter Type	Data Type
file			object
encryptionPassword		body	string

Example to restore the encryption settings

```
curl -u "admin:admin" -i -F "encryptionPassword=testPassword" -F "file=@backup_encryption_
settings.txt" -X POST "https://localhost:9502/rest/v2/security/import_encryption_settings"
```

GET /security/systemEncryption

This API is used to get the system encryption settings.

Parameters

None

Example to get the system encryption settings.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/security/systemEncryption"
```

PUT /security/systemEncryption

This API is used to update the system encryption settings. Can take a long time in large database to re-encrypt data.

Parameters

None

Request Body

```
{
  "algorithm": "string",
  "password": "string",
  "uuid": "string",
  "keySize": "string"
}
```

Example to update the system encryption settings

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/security/systemEncryption" -H "Content-Type:application/json" -d "{\"password\":\"MyTestEncryptionPassword\",\"uuid\":\"0b352e1e-ab56-4271-a813-31183df63788\"}"
```

Session

The session operations that can be performed are:

- GET /session
- PUT /session
- DELETE /session

GET /session

This API is used to get information about the currently open session. This may include updates to a user's rights.

Parameters

None

Example to get session information

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/session"
```

Example to get session information as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/ rest/v2/session"
```

PUT /session

This API is used to initiate a long running session with the BD server. Returns information about the current session including a session token and the current user

object. The session token should be used for all following Rest API calls and be placed within the session HTTP cookie.

Parameters

None

Request Body

```
{
  "user": {
    "name": "string",
    "domainName": "string",
    "id": 0,
    "annotation": "string",
    "memberReferences": [
      {
        "memberName": "string",
        "domainName": "string"
      }
    ],
```

```
    "rights": 0,
    "effectiveRights": 0,
    "inheritedRights": 0,
    "attributes": {
      "empty": true
    },
    "locked": true
  },
  "sessionToken": "string",
  "autoCloseMode": true
}
```


Example to begin a new session

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/session"
```

Example to begin a new session as ldap user

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/ rest/v2/session"
```

DELETE /session

This API is used to end the current session and invalidates the session token that was previously returned when creating the session.

Parameters

None

Example to end new session

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/session"
```

Example to end new session as ldap user

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/ rest/v2/session"
```

Sites

The different REST API operations that can be performed on the Business Directory sites are:

- GET /sites
- POST /sites
- GET /sites/{siteName}
- PUT /sites/{siteName}
- DELETE /sites/{siteName}
- GET /sites/{siteName}/dataPreviewPermission
- POST /sites/{siteName}/dataPreviewPermission
- DELETE /sites/{siteName}/dataPreviewPermission
- POST /sites/{siteName}/refresh
- GET /sites/{siteName}/scheduledRefresh
- PUT /sites/{siteName}/scheduledRefresh

- DELETE /sites/{siteName}/scheduledRefresh

GET /sites

This API is used to fetch all sites.

Parameters

None

Example to list all sites

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/sites/"
```

Example to list all sites as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/ rest/v2/sites/"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from SYS_SITES" -H "standardSQL:true" -H "system:true"
```

Example to retrieve count of all sites.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/sites?count"
```

POST /sites

This API is used to add a site or sites.

Parameters

None

Request Body

```
[
 {
  "siteName": "string",
  "uri": "string",
  "domain": "string",
  "userName": "string",
  "password": "string",
  "annotation": "string",
  "status": "string",
  "modifyTime": 0,
  "taskId": "string",
  "async": true,
  "dataPreviewEnabled": true
 }
]
```

Example to create a new site

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites" -H "Content-Type:application/json" -d "[{\"siteName\" : \"test_site\", \"uri\" : \"http://admin@localhost:9400\", \"domain\" : \"composite\", \"password\" : \"admin\", \"annotation\" : \"test site\", \"dataPreviewEnabled\" : false }]"
```

Example to create a new site as ldap user with BD_ADMIN rights

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites" -H "Content-Type:application/json" -d "[{\"siteName\" : \"test_site\", \"uri\" : \"http://admin@localhost:9400\", \"domain\" : \"composite\", \"password\" : \"admin\", \"annotation\" : \"test site\", \"dataPreviewEnabled\" : false }]"
```

GET /sites/{siteName}

This API is used to fetch a single site.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string

Example to list a single site

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/"
```

Example to list a single site as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/ rest/v2/sites/test_site/"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from SYS_SITES where SITE_NAME = 'test_site'" -H "standardSQL:true" -H "system:true"
```

PUT /sites/{siteName}

This API is update connection information for a site.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string

Request Body

```
{
  "siteName": "string",
  "uri": "string",
  "domain": "string",
  "userName": "string",
  "password": "string",
  "annotation": "string",
  "status": "string",
```

```
  "modifyTime": 0,

  "taskId": "string",

  "async": true,

  "dataPreviewEnabled": true

}
```

Example to Update a site

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/sites/test_site" -H "Content-Type:application/json" -d "{\"siteName\" : \"test_site\", \"uri\" : \"http://admin@localhost:9400\", \"domain\" : \"composite\", \"password\" : \"admin\", \"annotation\" : \"new annotation (updated)\", \"dataPreviewEnabled\" : false }"
```

Example to Update a site as ldap user with BD_ADMIN rights

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_site" -H "Content-Type:application/json" -d "{\"siteName\" : \"test_site\", \"uri\" : \"http://admin@localhost:9400\", \"domain\" : \"composite\", \"password\" : \"admin\", \"annotation\" : \"new annotation (updated)\", \"dataPreviewEnabled\" : false }"
```

Example to Update a site (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites/test_site" -H "X-HTTP-Method-Override:PUT" -H "Content-Type:application/json" -d "{\"siteName\" : \"test_site\", \"uri\" : \"http://admin@localhost:9400\", \"domain\" : \"composite\", \"password\" : \"admin\", \"annotation\" : \"new annotation (updated)\", \"dataPreviewEnabled\" : false }"
```

DELETE /sites/{siteName}

This API is delete a site from Business Directory.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string

Example to delete site 'test_site'

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/sites/test_site"
```

Example to delete site 'test_site' (as ldap user with BD_ADMIN rights)

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_site"
```

Example to delete site 'test_site' (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites/test_site" -H "X-HTTP-Method-Override:DELETE"
```

GET /sites/{siteName}/dataPreviewPermission

This API is get data preview enabled setting for a site.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string

Example to get data preview enabled setting for a site

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/dataPreviewPermission"
```

Example to get data preview enabled setting for a site as ldap user with BD_ADMIN rights

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_
site/dataPreviewPermission"
```

POST /sites/{siteName}/dataPreviewPermission

This API is enable data preview for a site.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string
async		query	string

Example to enable data preview for a site

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites/test_
site/dataPreviewPermission"
```

Example to enable data preview for a site as ldap user with BD_ADMIN rights

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_
site/dataPreviewPermission"
```

DELETE /sites/{siteName}/dataPreviewPermission

This API is disable data preview for a site.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string
async		query	string

Example to disable data preview for a site

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/sites/test_
site/dataPreviewPermission"
```

Example to disable data preview for a site as ldap user with BD_ADMIN rights

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_
site/dataPreviewPermission"
```

Example to Disable data preview for a site (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites/test_
site/dataPreviewPermission" -H "X-HTTP-Method-Override:DELETE"
```

POST /sites/{siteName}/refresh

This API is refresh a site by contacting the site.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string
async		query	string

Example to refresh site 'test_site'

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/refresh"
```

Example to Refresh site 'test_site' (as ldap user with BD_ADMIN rights)

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_site/refresh"
```

GET /sites/{siteName}/scheduledRefresh

This API is get the refresh schedule settings of a site from Business Directory.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string

Example to get refresh schedule info for site 'test_site'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/scheduledRefresh"
```

Example to get refresh schedule info for site 'test_site' (as ldap user with BD_ADMIN rights)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_
site/scheduledRefresh"
```

PUT /sites/{siteName}/scheduledRefresh

This API is set the refresh schedule settings of a site from Business Directory.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string

Request Body

```
{
  "enabled": true,
  "clusterAware": true,
  "startTime": "string",
  "endTime": "string",
  "interval": "string"
}
```

Example to set refresh schedule info for site 'test_site'.

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/scheduledRefresh" -H "Content-Type:application/json" -d "{\"enabled\" : \"true\", \"clusterAware\" : \"false\", \"startTime\" : \"2014-06-20 12:00:00\", \"endTime\" : \"2014-06-21 12:00:00\", \"interval\" : \"1d\" }"
```

Example to set refresh schedule info for site 'test_site' as ldap user with BD_ADMIN rights

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_site/scheduledRefresh" -H "Content-Type:application/json" -d "{\"enabled\" : \"true\", \"clusterAware\" : \"false\", \"startTime\" : \"2014-06-20 12:00:00\", \"endTime\" : \"2014-06-21 12:00:00\", \"interval\" : \"1d\" }"
```

Example to Set refresh schedule info for site 'test_site' (X-HTTP-Method-Override):

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/scheduledRefresh" -H "X-HTTP-Method-Override:PUT" -H "Content-Type:application/json" -d "{\"enabled\" : \"true\", \"clusterAware\" : \"false\", \"startTime\" : \"2014-06-20 12:00:00\", \"endTime\" : \"2014-06-21 12:00:00\", \"interval\" : \"1d\" }"
```

DELETE /sites/{siteName}/scheduledRefresh

This API is used to delete a site schedule refresh from Business Directory.

Parameters

Name	Description	Parameter Type	Data Type
siteName		path	string

Example to delete a site refresh schedule for 'test_site'

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/scheduledRefresh"
```

Example to delete a site refresh schedule for 'test_site' as ldap user with BD_ADMIN rights

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/sites/test_
site/scheduledRefresh"
```

Example to Delete a site refresh schedule for 'test_site' (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/sites/test_site/scheduledRefresh" -H
"X-HTTP-Method-Override:DELETE"
```

userProfiles

The following operations can be performed on the userProfiles:

- GET /userProfiles
- PUT /userProfiles
- DELETE /userProfiles
- GET /userProfiles/locale
- GET /userProfiles/{userParam}

GET /userProfiles

This API is used to Fetch all user profiles.

Parameters

None

Example to list all profiles

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/userProfiles"
```

Example to list all profiles as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/ userProfiles"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
USER_PROFILES" -H "standardSQL:true" -H "system:true"
```

Example to retrieve count of all profiles

```
curl -X GET -u admin:admin "https://localhost:9502/rest v2/ userProfiles?count"
```

PUT /userProfiles

This API is used to create or update a user profile for a user. The provided JSON should be a UserProfileBean containing firstName, lastName and email.Special characters must be escaped where required.

Parameters

None

Request Body

```
{
  "firstName": "string",
  "lastName": "string",
  "email": "string",
  "locale": "string",
```

```
  "profileEmpty": true
```

```
}
```

Example to create a user profile

```
curl -X PUT -u user:password "https://localhost:9502/rest/v2/userProfiles" -H "Content-Type:application/json" -d "{\"firstName\":\"myFirstName\", \"lastName\":\"myLastName\", \"email\":\"myemail@gmail.com\"}"
```

Example to update user profile with a different email

```
curl -X PUT -u user:password "https://localhost:9502/rest/v2/userProfiles" -H "Content-Type:application/json" -d "{\"firstName\":\"myFirstName\", \"lastName\":\"myLastName\", \"email\":\"differentemail@gmail.com\"}"
```

Example to update user profile with a different email as ldap user

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/userProfiles" -H "Content-Type:application/json" -d "{\"firstName\":\"myFirstName\", \"lastName\":\"myLastName\", \"email\":\"differentemail@gmail.com\"}"
```

Example to update user profile with a different email (X-HTTP-Method-Override)

```
curl -X POST -u user:password "https://localhost:9502/rest/v2/userProfiles" -H "X-HTTP-Method-Override:PUT" -H "Content-Type:application/json" -d "{\"firstName\":\"myFirstName\", \"lastName\":\"myLastName\", \"email\":\"differentemail@gmail.com\"}"
```

DELETE /userProfiles

This API is used to delete the caller's user profile, if it exists.

Parameters

None

Example to delete User profile

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/userProfiles" -H "Content-Type:application/json"
```

Example to delete User profile as ldap user

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/ v2/ userProfiles" -H "Content-Type:application/json"
```

Example to delete User profile (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/userProfiles" -H "X-HTTP-Method-Override:DELETE" -H "Content-Type:application/json"
```

GET /userProfiles/locale

This API is used to get the caller's locale from user profile, if it exists.

Parameters

None

Example to get User Locale

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v1/userProfiles/ locale"
```

Example to get User Locale as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v1/userProfiles/locale"
```

GET /userProfiles/{userParam}

This API is used to fetch a user profile by name and domain or by id.

Parameters

Name	Description	Parameter Type	Data Type
userParam	A string representing the user name and domain in user@domain format or a string representing the user's id.	path	string

Example to fetch one user profile by user name and domain

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/userProfiles/admin@composite"
```

Example to fetch one user profile by id

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/ userProfiles/-1973"
```

Equivalent system query (by name and domain)

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
USER_PROFILES WHERE USER_ID = (SELECT USER_ID FROM ALL_USERS WHERE
USERNAME='admin' AND DOMAIN_NAME='composite')" -H "standardSQL:true" -H "system:true"
```

Equivalent system query (by ID)

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
USER_PROFILES WHERE USER_ID = -1973" -H "standardSQL:true" -H "system:true"
```

Equivalent system query (as ldap user)

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/data/query" -H
"query:select * from ALL_USER_PROFILES WHERE USER_ID = -1973" -H "standardSQL:true" -H
"system:true"
```

Users

The following operations can be performed on the users of the Business Directory:

- GET /users
- GET /users/{userParam}
- PUT /users/{userParam}
- PUT /users/{userParam}/password
- POST /users/{userParam}/roles
- DELETE /users/{userParam}/roles
- DELETE /users/{userParam}/roles/{role}

GET /users

This API is used to get BD users.

Parameters

Name	Description	Parameter Type	Data Type
domain	Filter by domain.	query	string
userName	Filter by this user name.	query	string
roleFilter	Optional. Filter users by the specified roles.	query	Array[string]

Example to get all users

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/users"
```

Example to get all users as ldap user with BD_ADMIN rights

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/ v2/users"
```

Example to return users that only have BD_ADMIN & MANAGE_ CUSTOM_PROPERTIES right

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/users?site=test1& roleFilter=BD_
ADMIN&role_filter=MANAGE_CUSTOM_PROPERTIES"
```

Example to return user information for all users in remote site 'test1'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/users?domain= test1"
```

Example to return user information for ldap domains 'ad2003'

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/users?domain=ad2003"
```

GET /users/{userParam}

This API is used to get a BD user.

Parameters

Name	Description	Parameter Type	Data Type
userParam	User in user@domain format.	path	string

Example to get all users

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/users/admin@composite"
```

Example to get all users as ldap user with BD_ADMIN rights

```
curl -X GET -u user@ldapDomain:password
"https://localhost:9502/rest/v2/users/admin@ldapDomain"
```

PUT /users/{userParam}

This API is used to update roles or annotation for a BD user. Only the BD Admin can invoke this.

Parameters

Name	Description	Parameter Type	Data Type
userParam	User in user@domain format.	path	string

Request Body

```
{
  "userName": "string",
  "domain": "string",
  "annotation": "string",
  "roles": [
```

```
  "string"
 ]
}
```

Example to set ldap user 'adenisof' to only have BD_ADMIN and MANAGE_CUSTOM_PROPERTY roles

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/users/adenisof@ldapDomain" -H "Content-Type:application/json" -d "{ \"roles\" : [ \"BD_ADMIN\", \"MANAGE_CUSTOM_PROPERTIES\" ]}"
```

Example to remove all roles from ldap user 'adenisof' and update the annotation (as ldap user with BD_ADMIN rights)

```
curl -X PUT -u user@ldapDomain:password "https://localhost:9502/rest/v2/users/adenisof@ldapDomain" -H "Content-Type:application/json" -d "{ \"annotation\" : \"test new annotation\" }"
```

Example to remove all roles from ldap user 'adenisof' and update the annotation (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/users/adenisof@ldapDomain" -H "X-HTTP-Method-Override:PUT" -H "Content-Type:application/json" -d "{ \"annotation\" : \"test new annotation\" }"
```

PUT /users/{userParam}/password

This API is used to change the password for a BD user. Only the BD Admin can invoke this

Parameters

Name	Description	Parameter Type	Data Type
userParam	User in user@domain format.	path	string

Request Body

```
{
  "oldPassword": "string",
  "newPassword": "string"
}
```

Example to update password for user 'admin'

```
curl -X PUT -u admin:admin "https://localhost:9502/rest/v2/users/admin@composite/password" -H "Content-Type:application/json" -d "{ \"oldPassword\" : \"admin\", \"newPassword\" : \"admin2\" }"
```

Example to update password for user 'admin' (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/users/admin@composite/password" -H "X-HTTP-Method-Override:PUT" -H "Content-Type:application/json" -d "{ \"oldPassword\" : \"admin\", \"newPassword\" : \"admin2\" }"
```

POST /users/{userParam}/roles

This API is used to add roles to a BD user. Only the BD Admin can invoke this

Parameters

Name	Description	Parameter Type	Data Type
userParam	User in user@domain format.	path	string

Request Body

```
{
 "Roles": [
  "string"
 ]
}
```

Example to add BD_ADMIN and MANAGE_CUSTOM_PROPERTY to ldap user 'adenisof'

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/users/adenisof@ldapDomain/roles" -H "Content-Type:application/json" -d "{ \"roles\" : [ \"BD_ADMIN\", \"MANAGE_CUSTOM_PROPERTIES\" ]}"
```

Example to add BD_ADMIN and MANAGE_CUSTOM_PROPERTY to ldap user 'adenisof' (as ldap user with BD_ADMIN rights)

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/users/adenisof@ldapDomain/roles" -H "Content-Type:application/json" -d "{ \"roles\" : [ \"BD_ADMIN\", \"MANAGE_CUSTOM_PROPERTIES\" ]}"
```

DELETE /users/{userParam}/roles

This API is used to remove all roles from a BD user. Only the BD Admin can invoke this

Parameters

Name	Description	Parameter Type	Data Type
userParam	User in user@domain format.	path	string

Example to remove all roles from ldap user 'adenisof'

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/users/adenisof@ldapDomain/roles"
```

Example to remove all roles from ldap user 'adenisof' (as ldap user with BD_ADMIN rights)

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/users/adenisof@ldapDomain/roles"
```

Example to remove all roles from ldap user 'adenisof' (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/users/adenisof@my_bd_ldap_
domain/roles" -H "X-HTTP-Method-Override:DELETE"
```

DELETE /users/{userParam}/roles/{role}

This API is used to remove a role from a BD user. Only the BD Admin can invoke this

Parameters

Name	Description	Parameter Type	Data Type
userParam	User in user@domain format.	path	string
role	Role to remove.	path	string

Example to remove all roles from ldap user 'adenisof'

```
curl -X DELETE -u admin:admin
"https://localhost:9502/rest/v2/users/adenisof@ldapDomain/roles/BD_ADMIN"
```

Example to remove BD_ADMIN role from ldap user 'adenisof' (as ldap user with BD_ADMIN rights)

```
curl -X DELETE -u user@ldapDomain:password
"https://localhost:9502/rest/v2/users/adenisof@ldapDomain/roles/BD_ADMIN"
```

Example to remove BD_ADMIN role from ldap user 'adenisof' (X-HTTP-Method-Override)

```
curl -X POST -u admin:admin
"https://localhost:9502/rest/v2/users/adenisof@ldapDomain/roles/BD_ADMIN" -H "X-HTTP-
Method-Override:DELETE"
```

Watches

Following are the operations that can be performed on the watches set up in Business Directory:

- GET /watches
- POST /watches

- DELETE /watches
- PATCH /watches
- DELETE /watches/users
- GET /watches/users/{ownerId}
- GET /watches/{watchId}

GET /watches

This API is used to fetch all watches.

Parameters

None

Example to list all watches

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/watches"
```

Example to list all watches as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/ watches"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
WATCHES" -H "standardSQL:true" -H "system:true"
```

Example to retrieve count of all watches.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/watches?count"
```

POST /watches

This API is used to add a new watch for a resource. The provided JSON should be a list of map entries whose keys are resourcePath, resourceType and includeChildren. The values of the map are values associated with the keys.

Parameters

None

Request Body

```
[
 {
   "resourcePath": "string",
   "resourceType": "string",
   "includeChildren": true
 }
]
```

Example to create a new watch on a published resource

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/watches" -H "Content-
Type:application/json" -d "[{\"resourcePath\":\"/localhost_
9400/services/databases/sources/Categories\", \"resourceType\":\"TABLE\",
\"includeChildren\":false}]"
```

Example to create watches for multiple published resources

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/watches" -H "Content-
Type:application/json" -d "[{\"resourcePath\":\"/localhost_
9400/services/databases/sources/Customers\", \"resourceType\":\"TABLE\",
\"includeChildren\":false},{\"resourcePath\":\"/localhost_9400/services/databases/ds\",
\"resourceType\":\"DATABASE\", \"includeChildren\":true}]"
```

Example to create watches for multiple published resources as ldap user

```
curl -X POST -u user@ldapDomain:password "https://localhost:9502/rest/v2/watches" -H "Content-
Type:application/json" -d "[{\"resourcePath\":\"/localhost_9400/services/databases/sources/
Customers\", \"resourceType\":\"TABLE\", \"includeChildren\":false},{\"resourcePath\":\"/localhost_
9400/services/databases/ds\", \"resourceType\":\"DATABASE\", \"includeChildren\":true}]"
```

DELETE /watches

This API is used to delete a watch on a resource. The provided JSON should be a list of watch IDs that are to be deleted.

Parameters

None

Request Body

```
[
```

```
 0
```

```
]
```

Example to delete a watch on a resource

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/watches" -H "Content-
Type:application/json" -d "[1001]"
```

Example to delete a watch on a resource as ldap user

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/watches" -H
"Content-Type:application/json" -d "[1001]"
```

Example to delete a watch on a resource (X-HTTP-Method-Override):

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/watches" -H "X-HTTP-Method-
Override:DELETE" -H "Content-Type:application/json" -d "[1001]"
```

PATCH /watches

This API is used to update watches. The provided JSON should be a map whose keys are the watch IDs for which includeChildren parameter needs to be updated. The corresponding values of the map should contain the updated includeChildren boolean value.

Parameters

None

Request Body

```
{
  "additionalProp1": true,
  "additionalProp2": true,
  "additionalProp3": true
}
```

Example to update includeChildren on a specific watch ID

```
curl -X PATCH -u admin:admin "https://localhost:9502/rest/v2/watches" -H "Content-Type:application/json" -d "{\"1001\":\"true\"}"
```

Example to update includeChildren on a specific watch ID as ldap user

```
curl -X PATCH -u user@ldapDomain:password "https://localhost:9502/rest/v2/watches" -H "Content-Type:application/json" -d "{\"1001\":\"true\"}"
```

Example to update includeChildren on a specific watch ID (X-HTTP-Method-Override):

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/watches" -H "X-HTTP-Method-Override:PATCH" -H "Content-Type:application/json" -d "{\"1001\":\"true\"}"
```

DELETE /watches/users

This API is used to delete all watches based on user IDs. A BD admin provided JSON should be a list of user IDs whose watches are to be deleted. Any user executed JSON should be a list of size 1 containing user ID for whom watches are to be deleted.

Parameters

None

Request Body

```
[
```

```
 0
```

```
]
```

Example for a user deleting his own watches

```
curl -X DELETE -u user:password "https://localhost:9502/rest/v2/watches/ users" -H "Content-Type:application/json" -d "[300]"
```

Example for a user deleting his own watches as ldap user

```
curl -X DELETE -u user@ldapDomain:password "https://localhost:9502/rest/v2/watches/users" -H "Content-Type:application/json" -d "[300]"
```

Example of BD admin deleting other users watches

```
curl -X DELETE -u admin:admin "https://localhost:9502/rest/v2/watches/users" -H "Content-Type:application/ json" -d "[300,400]"
```

Example for a user deleting his own watches (X-HTTP-Method-Override):

```
curl -X POST -u admin:admin "https://localhost:9502/rest/v2/watches/users" -H "X-HTTP-Method-Override:DELETE" -H "Content-Type:application/json" -d "[300,400]"
```

GET /watches/users/{ownerId}

This API is used to fetch watches for a user.

Parameters

Name	Description	Parameter Type	Data Type
ownerId	The owner's user id	path	integer

Example to retrieve a watch

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/watches/ users/-1973"
```

Example to retrieve a watch as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/ watches/ users/-1973"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
WATCHES where OWNER_ID = -1973" -H "standardSQL:true" -H "system:true"
```

Example to retrieve count of a watches for a user.

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/watches/ users/-1973?count"
```

GET /watches/{watchId}

This API is used to fetch a specific watch.

Parameters

Name	Description	Parameter Type	Data Type
watchId	The watch id or ids	path	integer

Example to retrieve a watch

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/watches/1234"
```

Example to retrieve a watch as ldap user

```
curl -X GET -u user@ldapDomain:password "https://localhost:9502/rest/v2/ watches/1234"
```

Equivalent system query

```
curl -X GET -u admin:admin "https://localhost:9502/rest/v2/data/query" -H "query:select * from ALL_
WATCHES where WATCH_ID = 1234" -H "standardSQL:true" -H "system:true"
```

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the Product Documentation website, mainly in HTML and PDF formats.

The Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the TIBCO® Data Virtualization page.

Users

- TDV Getting Started Guide
- TDV User Guide
- TDV Web UI User Guide
- TDV Client Interfaces Guide
- TDV Tutorial Guide
- TDV Northbay Example

Administration

- TDV Installation and Upgrade Guide
- TDV Administration Guide
- TDV Active Cluster Guide
- TDV Security Features Guide

Data Sources

- TDV Adapter Guides

TDV Data Source Toolkit Guide (Formerly Extensibility Guide)

References

TDV Reference Guide

TDV Application Programming Interface Guide

Other

TDV Business Directory Guide

TDV Discovery Guide

TDV and Business Directory Release Notes - Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Release Version Support

TDV 8.5 and 8.8 are designated as Long Term Support (LTS) versions. Some release versions of TIBCO® Data Virtualization products are selected to be long-term support (LTS) versions. Defect corrections will typically be delivered in a new release version and as hotfixes or service packs to one or more LTS versions. See also Long Term Support.

How to Contact Support for TIBCO Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our product Support website.
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the product Support website. If you do not have a username, you can request one by clicking **Register** on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote

on feature requests from within the TIBCO Ideas Portal. For a free registration, go to TIBCO Community.

Legal and Third-Party Notices

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, TIBCO logo, TIBCO O logo, ActiveSpaces, Enterprise Messaging Service, Spotfire, TERR, S-PLUS, and S+ are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (https://www.cloud.com/legal) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: https://scripts.sil.org/OFL

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at https://www.cloud.com/legal.

Copyright © 2002-2025. Cloud Software Group, Inc. All Rights Reserved.