webMethods Adapter for MongoDB Installation and User's Guide

Version 9.12

November 2019

WEBMETHODS

This document applies to webMethods Adapter for MongoDB 9.12 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: ADAPTER-MAW-IUG-912-20240112

Table of Contents

About this Guide

This guide describes how to configure and use webMethods Adapter for MongoDB. It contains information for administrators and application developers who want to exchange data with MongoDB.

To use this guide effectively, you should be familiar with:

- The basic concepts and tasks for working with MongoDB
- Creating flow or Java services
- Terminology and basic operations of your operating system
- The setup and operation of webMethods Integration Server.
- How to perform basic tasks with Software AG Designer.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies service names and locations in the format *folder.subfolder.service*, APIs, Java classes, methods, properties.
Italic	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at https://documentation.softwareag.com.

In addition, you can also access the cloud product documentation via https://www.softwareag.cloud. Navigate to the desired product and then, depending on your solution, go to "Developer Center", "User Center" or "Documentation".

Product Training

You can find helpful product training material on our Learning Portal at https://knowledge.softwareag.com.

Tech Community

You can collaborate with Software AG experts on our Tech Community website at https://techcommunity.softwareag.com. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at https://github.com/softwareag and https://hub.docker.com/u/softwareag and discover additional Software AG resources.

Product Support

Support for Software AG products is provided to licensed customers via our Empower Portal at https://empower.softwareag.com. Many services on this portal require that you have an account. If you do not yet have one, you can request it at https://empower.softwareag.com/register. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

1 Overview of the Adapter

About Adapter for MongoDB

webMethods Adapter for MongoDB is an add on to webMethods Integration Server that enables you to exchange data with MongoDB with the help of a MongoDB Java driver. The adapter provides seamless and real-time communication with the database without requiring changes to existing application infrastructure.

Using Adapter for MongoDB, Integration Server clients can create and run services that executes the transactions to retrieve data from, and insert and update data in, MongoDB.

For example, you can use Adapter for MongoDB to add a customer to a MongoDB database based on data from another system that is connected to Integration Server. Or you can use Adapter for MongoDB to poll a MongoDB database for customers that have been added to the database, and to send that data to Integration Server to be inserted into another resource.

Architecture Overview

Adapter for MongoDB provides a set of user interface, services, and templates that enable you to create integrations with databases using a MongoDB driver. The MongoDB adapter is provided as a single package that must be installed on Integration Server.

As Adapter for MongoDB uses a MongoDB driver to perform operations on databases, the adapter requires a supported MongoDB driver to be installed and loaded in the packages directory of Integration Server. For more details see "Installing a MongoDB java driver on Integration Server " on page 30.

Adapter for MongoDB enables you to configure the following components:

- **Adapter Connections.** Enable Integration Server to connect to database systems at runtime. You must configure an adapter connection before you can configure adapter services.
- **Adapter Services.** Enable Integration Server to initiate and perform database operations on a database. For example, an adapter service could enable a trading partner to query your inventory database to determine whether a particular item is currently in stock. You can configure adapter services using adapter services templates, which are provided with Adapter for MongoDB.
- **Adapter Listeners and Listener Notifications.** Registers item to subscription and notifies the Integration Server when a notification is generated by MongoDB server. For more information, see "Adapter Listeners and Listener Notifications" on page 14.

Package Management

Adapter for MongoDB is provided as a package called WmMongoDBAdapter that you manage like any package on Integration Server. There are several considerations regarding how you set up and effectively manage your packages on Integration Server:

- You must create user-defined packages for your connections, adapter services, and notifications.

- You should understand how package dependencies work so you make the best decisions regarding how you manage your adapter services and notifications.
- You control which development groups have access to which adapter services and notifications.
- You should understand how clustering, an advanced feature of Integration Server, works to effectively manage your adapter services.

Adapter Connections

Adapter for MongoDB connects to a database through MongoDB driver at run time. You create one or more connections at design time to use in integrations. The number of connections you create, and the types of those connections depend on the types of databases you are connecting to and your integration needs. For example, if you have multiple installations of the same kinds of databases, you access each using different connections.

For example, if you have a data warehouse system and an ERP system that uses your Adapter for MongoDB connections containing parameters that Integration Server uses to manage connections to the database, so that they can be used by the adapter to provide services. You can configure connections using Integration Server Administrator. You must have Integration Server Administrator privileges to access Adapter for MongoDB's administrative screens.

Connection Pools

Integration Server includes a connection management service that dynamically manages connections and connection pools based on configuration settings that you specify for the connection. All adapter services use connection pooling.

A connection pool is a collection of connections with the same set of attributes. Integration Server maintains connection pools in memory. Connection pools improve performance by enabling adapter services to re-use open connections instead of opening new connections.

Run-Time Behavior of Connection Pools

When you enable a connection, Integration Server initializes the connection pool, creating the number of connection instances you specified in the connection's **Minimum Pool Size** field when you configured the connection. Whenever an adapter service needs a connection, Integration Server provides a connection from the pool. If no connections are available in the pool, and the maximum pool size has not been reached, the server creates one or more new connections (according to the number specified in the **Pool Increment Size** field) and adds them to the connection pool. If the pool is full (as specified in **Maximum Pool Size** field), the requesting service will wait for Integration Server to obtain a connection, up to the length of time specified in the **Block Timeout** field, until a connection becomes available. Periodically, Integration Server inspects the pool and removes inactive connections that have exceeded the expiration period that you specified in the **Expire Timeout** field.

If initialization of the connection pool fails because of a network connection failure or some other type of exception, you can enable the system to retry the initialization any number of times, at specified intervals.

Built-In Services for Connections

Integration Server provides built-in services that enable you to programmatically control connections. You can use them to enable and disable a connection, and to return usage statistics and the current state (Enabled or Disabled) and error status for a connection. These services are located in the WmART package, in the pub.art.connection folder.

The setAdapterServiceNodeConnection and setPollingNotificationNodeConnection built-in services enable you to change the connection associated with an adapter service or notification respectively.

Adapter Services

To use Adapter for MongoDB, you create adapter services. Adapter services allow you to connect to the adapter's resource and initiate an operation on the resource from Integration Server.

You call adapter services from flow or Java services to interact with database tables. The adapter services perform database operations by calling MongoDB APIs. Integration Server then uses adapter connections that you defined earlier to execute the adapter services.

Adapter services are based on templates provided with Adapter for MongoDB. Each template represents a specific technique for doing work on a resource, such as using the Select Document template to retrieve specified information from a database.

An adapter service template contains all the code necessary for interacting with the resource but without the data specifications. You provide these specifications when you create a new adapter service.

Creating a new service from an adapter service template is straightforward. Using Software AG Designer, you assign the service a default adapter connection.

After you select the connection for the adapter service, you select the adapter service template and supply the data specifications using Designer. Some familiarity with using Designer is required. For more information, see the *webMethods Service Development Help* for your release.

Adapter for MongoDB provides the following adapter service templates:

Adapter Service Type	Adapter Service Template	Description
Insert Operation	Insert Document	Inserts new information into a collection.
Find Operation	Select Document	Retrieves the information from collection.
Delete Operation	Delete Document	Deletes documents from collection and includes a mapping for an output field that stores the number of affected rows.

Adapter Service Type	Adapter Service Template	Description
Update Operation	Update Document	Updates the existing document in a collection and includes a mapping for an output field that stores the number of affected rows.
Aggregation Operation	Aggregate Query	Run aggregation queries on MongoDB.
Dynamic Operation	Dynamic Query Executor	Runs a raw MongoDB query

Changing the Connection Associated with an Adapter Service at Design Time

Integration Server provides built-in services that you can use at design time to change the connection associated with an adapter service. The built-in services, setAdapterServiceNodeConnection are provided in the WmART package's pub.art.service folder. Using this function, you can change the specific connection associated with an adapter service at design time so that you do not need to create and maintain multiple adapter services.

Note:
The setAdapterServiceNodeConnection services can be run at design time only. Do not use them within an Integration Server flow or Java service. You must run the services directly from Designer by selecting a service and running it.

For details, see the *webMethods Integration Server Built-In Services Reference* for your release.

Changing the Connection Associated with an Adapter Service at Run Time

Adapter for MongoDB enables you to dynamically change the user credentials of a connection associated with an adapter service at run time. This feature enables you to interact with a database with different user privileges.

For example, consider a service is associated adapter connection that uses an administrator's credentials at design time to define a connection to a database. At run time, you can override the administrator's account credentials with the individual user's credentials to limit the access to the database according to the permission level each user has. This capability also enables you to keep track of the database operations by the user initiating the service.

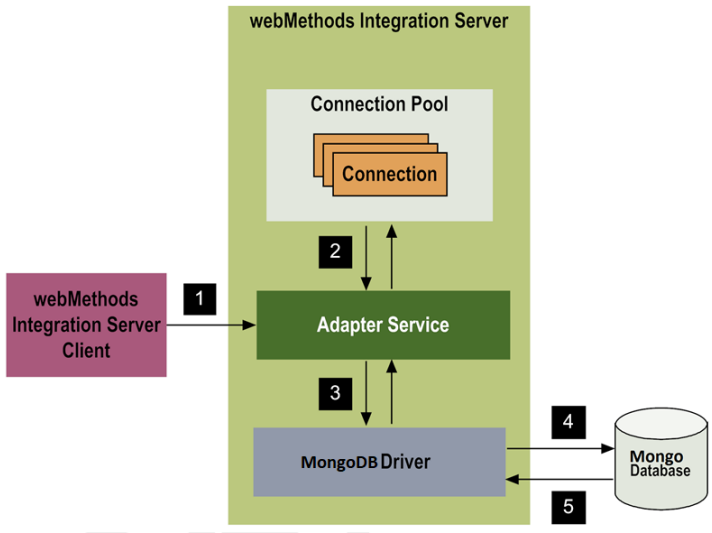
Changing the User Credentials of a Service's Associated Connection at Run Time

Adapter for MongoDB enables you to dynamically change the user credentials of a connection associated with an adapter service at run time. This feature enables you to interact with a database with different user privileges.

For example, consider a service's associated adapter connection that uses an administrator's credentials at design time to define a connection to a database. At run time, you can override the administrator's account credentials with individual user credentials to limit access to the database according to the permission level each user has. This capability also enables you to keep track of the database operations by the user initiating the service.

For more information, see "Dynamically Changing the User Credentials of a Service's Connection at Run Time" on page 36.

Adapter Service Transaction Processing



Adapter Listeners and Listener Notifications

Adapter for MongoDB provides listeners and listener notifications, there are two types of listeners:

- **MongoDB Database Listener.** Database Change Stream Listener.
- **MongoDB Collection Listener.** Collection Change Stream Listener.

Listeners

The listener in the Adapter for MongoDB represents a change stream. When a MongoDB Adapter listener is created, a database or a collection change stream is also created.

When you enable the listener, it waits for a message from the change stream. When a message appears in the change stream, the listener then passes the message to a listener notification.

You must never invoke a listener directly from a service or client. Instead, use the Integration Server Administrator to configure, enable, and disable the services.

All listeners function in the following manner:

- All listeners stop functioning when the package containing the listener node is disabled or when Integration Server shuts down.
- All listeners start functioning when the package containing the listener node is enabled or when Integration Server restarts.

Listener Notification

A listener notification works in conjunction with a listener to process messages in the change stream. When a listener receives a message from the change stream, the listener passes the message to an enabled listener notification that you associated with the listener. For more information about enabling listener notification, see "Enabling Listeners" on page 61.

Note:
The message is lost, if you do not configure any notifications with a listener, or if you do not enable any of the already configured notifications. Software AG recommends that you always start a listener notification before a listener. Create separate listeners to receive notifications of different types. you should configure only one notification per listener. If you configured multiple notifications with a listener, enable only the notification that you require.

When you create a listener notification, Adapter for MongoDB creates a publishable document type. At run time, after the listener receives a message from the server, the listener invokes the notification. The notification publishes the document created with the notification. A listener notification can publish a document in either of the following ways:

- Publish to a JMS queue or topic when Integration Server is connected to a JMS provider.
- To a local instance of Integration Server when it is not connected to a JMS provider.

Note:
The insert, update, delete, replace, and invalidate notifications are applicable for **MongoDB Collection Listener**. The rename, drop, and invalidate notifications are applicable for **MongoDB Database Listener**.

When you subscribe to a specific listener notification, a publishable document type is created. You must subscribe to an Integration Server trigger to receive these notifications. For more information about using triggers with services, see the *Publish-Subscribe Developer's Guide*.

Note:
To use the JMS protocol with listener notifications, you must first configure a JMS connection alias on Integration Server. For more information, see the *webMethods Integration Server Administrator's Guide*.

Controlling Pagination

When using the adapter on Integration Server 9.10 and later, you can control the number of items that are displayed on the adapter Connections screen and Notifications screen. By default, 10 items are displayed per page. Click **Next** and **Previous** to move through the pages, or click a page number to go directly to a page.

To change the number of items displayed per page, set the `watt.art.page.size` property and specify a different number of items.

To set the number of items per page

1. From Integration Server Administrator, click **Settings > Extended**.

2. Click **Edit Extended Settings**. In the Extended Settings editor, add or update the `watt.art.page.size` property to specify the preferred number of items to display per page. For example, to display 50 items per page, specify:

   ```
   watt.art.page.size=50
   ```

3. Click **Save Changes**. The property appears in the Extended Settings list.

   For more information about working with extended configuration settings, see *webMethods Integration Server Administrator's Guide* for your release.

2 Installing and Uninstalling Adapter for MongoDB

Overview of Installing, and Uninstalling Adapter for MongoDB

This chapter explains how to install, and uninstall webMethods Adapter for MongoDB 9.12. The instructions use the Software AG Installer and the Software AG Uninstaller wizards. For complete information about the wizards or other installation methods, or to install other webMethods products, see *Installing webMethods Products On Premises* for your release.

Requirements

For a list of operating systems, and webMethods products supported by Adapter for MongoDB, see *webMethods Adapters System Requirements* . Adapter for MongoDB has no hardware requirements beyond those of its host Integration Server.

The Integration Server Home Directory

Beginning with Integration Server 9.6, you can create and run multiple Integration Server instances under a single installation directory. Each Integration Server instance has a home directory under Integration Server_directory \instances\ instance_name, that contains the packages, configuration files, log files, and updates for the instance.

For more information about running multiple Integration Server instances, see the *webMethods Integration Server Administrator's Guide* for your release.

If you are using Integration Server 9.5 and lower, the Integration Server home directory is Integration Server_directory . For example, on Integration Server 9.5 the adapter package is installed in the Integration Server_directory \packages_directory.

This guide uses the packages_directory as the home directory in Integration Server classpaths. For Integration Server 9.6 and above, the packages_directory is Integration Server_directory \instances\instance_name\packages directory. For Integration Server 9.5 and lower, the packages_directory is Integration Server_directory \packages directory.

Installing Adapter for MongoDB

Note:
If you are installing Adapter for MongoDB in a clustered environment, you must install the adapter on each Integration Server in the cluster, and each installation must be identical. For more information about working with Adapter for MongoDB in a clustered environment, see "Adapter for MongoDB in a Clustered Environment" on page 25.

To install Adapter for MongoDB

1. Download Installer from the Empower Product Support website.

2. If you are installing the adapter on an existing Integration Server, shut down the Integration Server.

3. Start the Installer wizard.

4. Choose the webMethods release that includes Integration Server on which you want to install the adapter. For example, if you want to install the adapter on Integration Server 10.3, choose the 10.3 release.

5. Specify the installation directory as follows:

   - If you are installing on an existing Integration Server, specify the webMethods installation directory that contains the host Integration Server.
   - If you are installing both the host Integration Server and the adapter, specify the installation directory to use.

6. In the product selection list, select **Adapters > webMethods Adapter 9.12 for MongoDB**.

   If you are using Integration Server 9.6 and above, you can choose to install the package in the default instance. In this case, Software AG Installer installs the adapter in both the locations, *Integration Server_directory* \packages and the default instance packages directory located in *Integration Server_directory* \instances\default\packages.

7. To download the documentation for the adapter, go to Software AG Documentation website.

8. After completing the installation , close the Installer and start the host Integration Server.

9. Adapter for MongoDB uses MongoDB Java driver to communicate with MongoDB Sever. Copy the MongoDB Java driver.jar files to Integration Server_directory \instances\ instance_name \packages\ WmMongoDBAdapter\code\jars directory.

   Note:
   To enable the adapter connection, it is necessary to add the required jar files. Refer the *Adapter System Requirement* document for the information regarding the supported MongoDB Java driver.

10. See "Installing a MongoDB java driver on Integration Server " on page 30 for instructions on installing a compatible MongoDB driver.

Uninstalling Adapter for MongoDB

To uninstall Adapter for MongoDB

1. Shut down the host Integration Server. You do not need to shut down any other webMethods products or applications that are running on your machine.

2. Start Software AG Uninstaller, selecting the webMethods installation directory that contains the host Integration Server.

3. In the product selection list, select **Adapters > webMethdods Adapter 9.12 for MongoDB**. You can also choose to uninstall the documentation.

4. After Uninstaller completes, restart the host Integration Server.

Uninstaller removes all Adapter for MongoDB-related files that were installed. However, Uninstaller does not delete files created after you installed the adapter (for example, user-created or configuration files), nor does it delete the adapter directory structure. You can go to the *Integration Server_directory* \packages directory and *Integration Server_directory* \instances\default\packages directory. Delete the WmMongoDBAdapter directory.

3 Package Management

Overview of Package Management

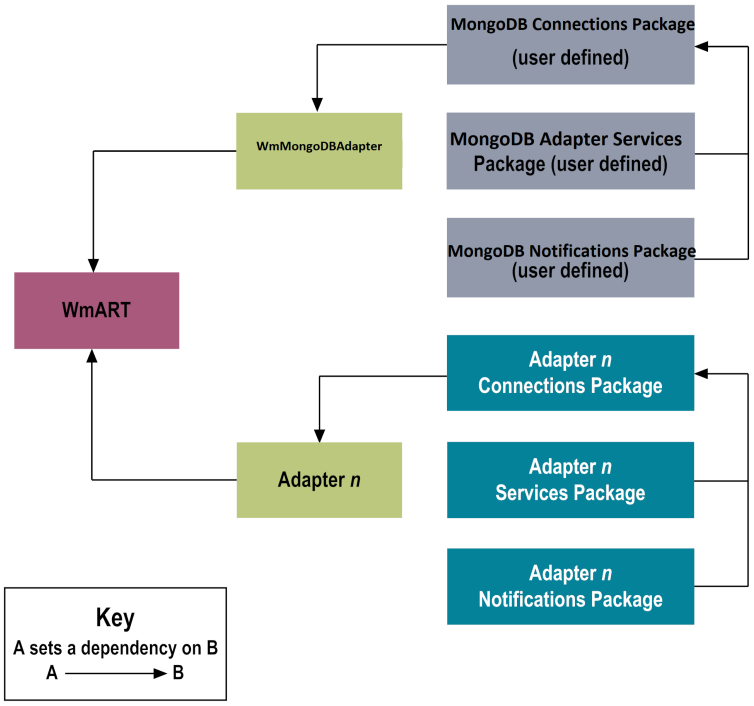
The following sections describe how to set up and manage your Adapter for MongoDB packages, set up Access Control Lists (ACLs), and use the adapter in a clustered environment.

Adapter for MongoDB Package Management

Adapter for MongoDB is provided as a package called WmMongoDBAdapter. You manage the WmMongoDBAdapter package as you would manage any package on webMethods Integration Server.

When you create connections, adapter services, and listener notifications, define them in user-defined packages rather than in the WmMongoDBAdapter package. Doing so will allow you to manage the package more easily.

As you create user-defined packages in which to store connections, adapter services, and listener notifications, use the package management functionality provided in Software AG Designer and set the user-defined packages to have a dependency on the WmMongoDBAdapter package. That way, when the WmMongoDBAdapter package loads or reloads, the user-defined packages load automatically. See the following diagram:



Package management tasks include:

- Setting package dependencies (see "Package Dependency Requirements and Guidelines" on page 23).
- "Enabling Packages" on page 23.
- "Importing and Exporting Packages" on page 24.

- "Group Access Control" on page 25.

Package Dependency Requirements and Guidelines

This section contains a list of dependency requirements and guidelines for user-defined packages. For instructions for setting package dependencies, see the *webMethods Service Development Help* for your release.

- A user-defined package must have a dependency on its associated adapter package, WmMongoDBAdapter. (The WmMongoDBAdapter package has a dependency on the WmART package.)
- Package dependencies ensure that at startup the Integration Server automatically loads or reloads all packages in the proper order: the WmART package first, the adapter package next, and the user-defined packages last. The WmART package is automatically installed when you install Integration Server. You should not need to manually reload the WmART package.
- If the connections and adapter services of an adapter are defined in different packages, then:
  - Package that contains the connections must have a dependency on the adapter package.
  - Packages that contain adapter services must have a dependency on their associated connection package.
- Keep connections for different adapters in separate packages so that you do not create interdependencies between adapters. If a package contains connections for two different adapters, and you reload one of the adapter packages, the connections for both adapters will reload automatically.
- Integration Server will not allow you to enable a package if it has a dependency on another package that is disabled. That is, before you can enable your package, you must enable all packages on which your package depends. For information about enabling packages, see "Enabling Packages" on page 23.
- Integration Server will allow you to disable a package even if another package that is enabled has a dependency on it. Therefore, you must manually disable any user-defined packages that have a dependency on the adapter package before you disable the adapter package. For information about disabling packages, see "Disabling Packages" on page 24.
- You can name connections, adapter services, and notifications the same name provided that they are in different folders and packages.

Enabling Packages

All packages are automatically enabled by default. Use the following procedure when you want to enable a package that was previously disabled.

To enable a package

1. Open Integration Server, if it is not already open.

2. In the **Packages** menu of the navigation area, click **Management**.

3. Click **No** in the **Enabled** column. The server displays a ✔ and **Yes** in the **Enabled** column.

Note:
Enabling an adapter package will not cause its associated user-defined packages to be reloaded. For information about reloading packages, see the *webMethods Service Development Help* for your release.

Important:
Before you manually enable a user-defined package, you must first enable its associated adapter package (WmMongoDBAdapter).

Disabling Packages

When you want to temporarily prohibit access to the elements in a package, disable the package. When you disable a package, the server unloads all of its elements from memory. Disabling a package prevents Integration Server from loading that package at startup.

Important:
If your adapter has multiple user-defined packages, and you want to disable some of them, disable the adapter package first (WmMongoDBAdapter). Otherwise, errors will be issued when you try to access the remaining enabled user-defined packages.

To disable a package

1. Open Integration Server Administrator, if it is not already open.

2. In the **Packages** menu of the navigation area, click **Management**.

3. Click **Yes** in the **Enabled** column for the package that you want to disable. The server issues a prompt to verify that you want to disable the package. Click **OK** to disable the package. When the package is disabled, the server displays **No** in the **Enabled** column.

A disabled adapter will:

- Remain disabled until you explicitly enable it using Integration Server Administrator.
- Not be listed in Designer.

Importing and Exporting Packages

You import and export packages using Designer. Exporting allows you to export the package to a .zip file and save it to your hard drive. The .zip file can then be imported for use by another package.

Important:

Do not rename packages you export, the rename function is comparable to moving a package, and when you import the renamed package, you lose any triggers, connections, and notifications associated with this package.

For details about importing and exporting packages, see the *webMethods Service Development Help*for your release.

Group Access Control

To control which groups have access to which adapter services, use access control lists (ACLs). For example, you can use ACLs to prevent one development group from inadvertently updating the work of another group, or to allow or deny access to services that are restricted to one group but not to others.

For information about assigning and managing ACLs, see the *webMethods Service Development Help* for your release.

Adapter for MongoDB in a Clustered Environment

Clustering is an advanced feature of the webMethods product suite that substantially extends the reliability, availability, and scalability of Integration Server. Clustering accomplishes this by providing the infrastructure and tools to deploy multiple Integration Servers as if they were a single virtual server and to deliver applications that leverage that architecture. Because this activity is transparent to the client, clustering makes multiple servers look and behave as one.

Integration Server 9.10 and higher versions support the caching and clustering functionality provided by Terracotta. Caching and clustering are configured at the Integration Server level and Adapter for MongoDB uses the caching mechanism that is enabled on Integration Server. Adapter for MongoDB does not explicitly implement any clustering or caching beyond what is already provided by Integration Server.

With clustering, you get the following benefits:

- **Load balancing.** This feature, provided automatically when you set up a clustered environment, allows you to spread the workload over several servers, thus improving performance and scalability.
- **Failover support.** Clustering enables you to avoid a single point of failure. If a server cannot handle a request, or becomes unavailable, the request is automatically redirected to another server in the cluster.

  Note:
  Integration Server clustering redirects HTTP and HTTPS requests, but does not redirect FTP or SMTP requests.
- **Scalability.** You can increase your capacity even further by adding new machines running Integration Server to the cluster.

For details on Integration Server clustering, see the *webMethods Integration Server Clustering Guide* for your release.

Adapter Service Support in Clusters

Adapter services are supported in a clustered environment. In order for a cluster to handle requests identically, you should be sure the identical service is in each server in the cluster so that if a given service is not available, the request can be redirected and handled by another server in the cluster.

For more details about adapter services in clusters, see "Clustering Considerations and Requirements" on page 26.

Replicating Packages to Integration Servers

Every Integration Server in the cluster should contain an identical set of packages that you define using Adapter for MongoDB; that is, you should replicate the Adapter for MongoDB services, the connections they use.

To ensure consistency, we recommend that you create all packages on one server, and replicate them to the other servers. If you allow different servers to contain different services, you might not derive the full benefits of clustering. For example, if a client requests a service that resides in only one server, and that server is unavailable, the request cannot be successfully redirected to another server.

For information about replicating packages, see the *webMethods Integration Server Administrator's Guide* for your release.

Clustering Considerations and Requirements

Note:
The following sections assume that you have already configured the Integration Server cluster. For details about webMethods clustering, see the *webMethods Integration Server Clustering Guide* for your release.

The following considerations and requirements apply to Adapter for MongoDB in a clustered environment.

Requirements for Each Integration Server in a Cluster

The following table describes the requirements of each Integration Server in a given cluster:

All Integration Servers in a given cluster must have identical...	For Example...
Integration Serverversions	All Integration Servers in the cluster must be the same version, with the same service packs and fixes applied.
Adapter packages	All adapter packages on one Integration Server should be replicated to all other Integration Servers in the cluster.

All Integration Servers in a given cluster must have identical...	For Example...
Adapter connections	If you configure a connection to the database, this connection must appear on all servers in the cluster so that any Integration Server in the cluster can handle a given request identically. If you plan to use connection pools in a clustered environment, see "Considerations When Configuring Connections with Connection Pooling Enabled" on page 27.
Adapter services	If you configure a specific Insert Adapter Service, this same adapter service must appear on all servers in the cluster so that any Integration Server in the cluster can handle the request identically. If you allow different Integration Servers to contain different services, you might not derive the full benefits of clustering. For example, if a client requests a service that resides on only one server, and that server is unavailable, the request cannot be successfully redirected to another server.

For information about replicating adapter packages, connections, and adapter services across multiple Integration Servers in a cluster, see "Replicating Packages to Integration Servers" on page 26.

Considerations When Installing Adapter for MongoDB Packages

For each Integration Server in the cluster, use the standard Adapter for MongoDB installation procedures for each machine, as described in "Overview of Installing, and Uninstalling Adapter for MongoDB" on page 18.

Considerations When Configuring Connections with Connection Pooling Enabled

When you configure a connection that uses connection pools in a clustered environment, be sure that you do not exceed the total number of connections that can be opened simultaneously for that database.

For example, if you have a cluster of two Integration Servers with a connection configured to a database that supports a maximum of 100 connections opened simultaneously, the total number of connections possible at one time must not exceed 100. This means that you cannot configure a connection with an initial pool size of 100 and replicate the connection to both servers, because there could be possibly a total of 200 connections opened simultaneously to this database.

In another example, consider a connection configured with an initial pool size of 10 and a maximum pool size of 100. If you replicate this connection across a cluster with two Integration Servers, it is possible for the connection pool size on both servers to exceed the maximum number of database connections that can be open at one time.

For more general information about connection pools, see the *webMethods Integration Server Administrator's Guide* for your release.

4 Adapter for MongoDB Connections

Overview of Adapter Connections

This chapter describes how to configure and manage Adapter for MongoDB connections. For more information about how adapter connections work, see “Adapter Connections” on page 11.

Before Configuring or Managing Adapter Connections

Perform the following steps before configuring or managing adapter connections.

To prepare to configure or manage adapter connections

1. Install webMethods Integration Server and Adapter for MongoDB on the same machine. For details, see “Overview of Installing, and Uninstalling Adapter for MongoDB” on page 18.

2. Install a compatible MongoDB driver. For instructions, see “Installing a MongoDB java driver on Integration Server ” on page 30. For a list of supported drivers, see *webMethods Adapters System Requirements*.

3. Make sure you have Integration Server Administrator privileges so that you can access Adapter for MongoDB's administrative screens. For information about setting user privileges, see "*WebMethods Integration Server Administrator's Guide*" for your release.

4. Check for a list of known driver limitations because it may affect how you configure your connections.

5. Start Integration Server and Integration Server Administrator, if they are not already running.

6. Enable the WmMongoDBAdapter package using Integration Server Administrator. For instructions, see "“Enabling Packages” on page 23".

7. Using Designer, create a user-defined package to contain the connection, if not already done. For more information about managing packages for the adapter, see “Adapter for MongoDB Package Management” on page 22.

Installing a MongoDB java driver on Integration Server

You must install a MongoDB java driver on Integration Server before you can specify connections. Integration Server requires access to the Java classes for each MongoDB driver that it will use to connect to a database.

For a list of supported drivers, see *webMethods Adapters System Requirements*.

To install a MongoDB java driver

1. Place the Java classes for the MongoDB driver in a location that Integration Server can access, typically the server's classpath.

To place the classes in the server's classpath, place the .zip or .jar file containing the classes in the *Integration Server_directory*\instances\ *instance_name*\packages \WmMongoDBAdapter\code\jars directory.

2. Restart Integration Server

   The server automatically adds the .zip or .jar libraries to its classpath after the restart.

Configuring Adapter for MongoDB Connections

When you configure Adapter for MongoDB connections, you specify information that Integration Server uses to connect to a MongoDB system. You can configure Adapter for MongoDB connections either manually using the Integration Server Administrator screen.

To configure an adapter connection

1. In the Adapters menu in Integration Server Administrator's navigation area. Click **webMethods Adapter for MongoDB**.

2. Click **Configure New Connection** on the **Connections** screen.

3. click **webMethods Adapter for MongoDB Connection** to display the Configure **Connection Type** screen.

4. In the webMethods Adapter for MongoDB section, use the following fields:

Field	Description/Action
Package Name	Name you want to give the package, that is used for creating connection in package. You must create the package using Designer before you can specify the package using this parameter. For general information about creating packages, see the *webMethods Service Development Help* for your release. **Note:** Configure the connection in a user-defined package rather than in the adapter's package.
Connection Type	Name you want to give the connection. Connection names cannot have spaces or use special characters reserved by Integration Server and Designer. For more information about the use of special characters in package, folder, and element names, see the *webMethods Service Development Help* for your release.

5. In the Connection Properties section, use the following fields:

Field	Description/Action
Connection URL	Connection URL used to connect to MongoDB as described in the MongoDB documentation. Connection URL syntax for MongoDB is `mongodb://host:port/`. Connection URL syntax for MongoDB Atlas is `mongodb+srv://host/`. For example: `mongodb+srv://cluster0.euv3f.mongodb.net/`.
Database Name	Name of the database on which the operations are performed.
Authentication Mechanism	User authentication mode. Select one of the following authentication modes: **Default**. You must provide values for the following fields: ■ **Authentication Database**. Database name which holds the user credentials. ■ **User**. User name that the connection uses to connect to MongoDB server. ■ **Password**. Password for the user name. ■ **Retype Password**. Retype the password you just entered. **X.509**. You must provide values for the following fields: ■ **User**. User name that the connection uses to connect to MongoDB server. **LDAP**. You must provide values for the following fields: ■ **User**. User name that the connection uses to connect to MongoDB server. ■ **Password**. Password for the user name. ■ **Retype Password**. Retype the password you just entered. **Kerberos**. You must provide values for the following fields: ■ **User**. User name that the connection uses to connect to MongoDB server. ■ **Other Authentication Mechanism Properties**. Other properties required for Kerberos authentication. **None**. No user authentication is required.
Read Preference	MongoDB supports the following read preferences: ■ **primary**. All operations reads data from the current replica set.

Field	Description/Action
	■ **primaryPreferred**. All the operation reads data from the primary replica set. The secondary members are preferred only if the primary member is unavailable. ■ **secondary**. All operations reads data from the secondary members of replica set. ■ **secondaryPreferred**. All operation reads data from the secondary replica set. The primary members are preferred only if secondary member is unavailable. ■ **nearest**. All operations read from the members of replica set with the least network latency.
Read Concern	MongoDB supports the following read concerns: ■ **local**. Returns without the guarantee of writing the data to a majority of member for a replica set. ■ **majority**. Returns data that has acknowledged by a majority of replica set members. ■ **available**. Returns without the guarantee of writing the data to a majority of member for a replica set. ■ **linearizable**. Returns data that reflects all the successful majority acknowledged writes.
Write Concern	Type of write concern the connection provides. Uses specification as `{ w: <value>, j: <boolean>, wtimeout: <number> }`.
Server Selection Timeout	Number of milliseconds the MongoDB driver waits for a server to raise an error if no server is selected.
Socket Connection Timeout(msec)	Time-out value in milliseconds to get the result for the query. **Note:** MongoDB driver throws an error if the query takes longer than the time-out value specified in the field.
Socket Read Timeout(msec)	Set the time-out limit for waiting to read data.

Enable SSL	Enables an SSL connection to the database. If SSL connection is enabled, then set the following fields:
	■ **Invalid Hostname Allowed**. Possible values are:
	  ■ `true`. Validates hostname certificates.
	  ■ `false`. Perform no validation of the certificates.

Field	Description/Action
	■ **Truststore Alias**. Required. Alias for the truststore file configured in Integration Server. ■ **Keystore Alias**. Required. Alias for keystore file configured in Integration Server for a two-way SSL authentication. For more information about using keystores and truststores with Integration Server, see *webMethods Integration Server Administrator's Guide*. **Note:** For Mongodb cloud connection, you must set **Enable SSL** to `true`.
Compression Set	Compresses the incoming and outgoing messages. **Note:** You can provide multiple values using a comma separator.
Required Replica Set	Specify the name of the replica set.

6. In the Connection Management Properties section, use the following fields:

Field	Description/Action
Enable Connection Pooling	Enables the connection to use connection pooling. For more information about connection pooling, see "Adapter Connections" on page 11. **Note:** If you are planning to enable the connection pooling in a clustered environment, then consider the size of the connection pool. For details, see "Considerations When Configuring Connections with Connection Pooling Enabled" on page 27.
Minimum Pool Size	If connection pooling is enabled, this field specifies the number of connections to create when the connection is enabled. The adapter will keep open the number of connections you configure here regardless of whether these connections become idle.
Maximum Pool Size	If connection pooling is enabled, this field specifies the maximum number of connections that can exist at one time in the connection pool.
Pool Increment Size	The password for the database user name specified in UserName. If connection pooling is enabled, this field specifies the number of connections by which the pool will be incremented if connections are needed, up to the maximum pool size.

Field	Description/Action
Block Timeout	If connection pooling is enabled, this field specifies the number of milliseconds that Integration Server will wait to obtain a connection with the database before it times out and returns an error. For example, you have a pool with **Maximum Pool Size** of 20. If you receive 30 simultaneous requests for a connection, 10 requests will be waiting for a connection from the pool. If you set the **Block Timeout** to 5000, the 10 requests will wait for a connection for 5 seconds before they time out and return an error. If the services using the connections require 10 seconds to complete and return connections to the pool, the pending requests will fail and return an error message stating that no connections are available. If you set the **Block Timeout** value too high, you may encounter problems during error conditions. If a request contains errors that delay the response, other requests will not be sent. This setting should be tuned in conjunction with the **Maximum Pool Size** to accommodate such bursts in processing.
Expire Timeout	If connection pooling is enabled, this field specifies the number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool. The connection pool will remove inactive connections until the number of connections in the pool is equal to the **Minimum Pool Size**. The inactivity timer for a connection is reset when the connection is used by the adapter. If you set the **Expire Timeout** value too high, you may have a number of unused inactive connections in the pool. This consumes local memory and a connection on your backend resource. This could have an adverse effect if your resource has a limited number of connections. If you set the **Expire Timeout** value too low, performance could degrade because of the increased activity of creating and closing connections. This setting should be tuned in conjunction with the **Minimum Pool Size** to avoid excessive opening/closing of connections during normal processing.
Startup Retry Count	The number of times that the system should attempt to initialize the connection pool at startup if the initial attempt fails. The default is 0.
Startup Backoff Timeout	The number of seconds that the system should wait between attempts to initialize the connection pool.

7. Click **Save Connection**.

The connection you created appears on the adapter's Connections screen and in Designer. You can enable a connection only if the parameters for the connection are valid.

Dynamically Changing a Service's Connection at Run Time

You can run a service using a connection other than the default connection that was associated with the service when the service was created.

Important:
At run time, you can change either the credentials (user name and password) or the connection name associated with a specific service, but not both at the same time. If you override both the credentials and the connection name, Adapter for MongoDB takes into account only the connection name override.

To override the default connection, you must code your flow to pass a value through the pipeline into a service's `$connectionName` field.

For example, you have a flow whose primary purpose is to update a production database. However, you want the flow to have the capability to update a test database, with the decision of which database to update to be made programmatically at runtime. The output signature of the flow's first service contains a field called Target. The flow could branch based on the value in Target. If Target contains the value Production, the second service in the flow would ignore `$connectionName`, thus using its default connection to connect to (and then update) the production database. However, if Target contains the value Test, the second service in the flow would use the value in the `$connectionName` from the pipeline and connect to (and then update) the test database.

Dynamically Changing the User Credentials of a Service's Connection at Run Time

In Adapter for MongoDB, you can dynamically provide the user name and password credentials associated with a specific adapter service at run time. This capability enables you to override the connection that is associated with the adapter service at design time. If you provide the user credentials in an adapter service at run time, Adapter for MongoDB connects to the database using the new credentials, along with the other connection parameters associated with the service's associated connection. If you do not provide any user credentials at run time, Adapter for MongoDB connects to the database using the user credentials provided at design time.

For more information, see "Changing the User Credentials of a Service's Associated Connection at Run Time" on page 14.

Viewing Adapter Connection Parameters

You can view a connection's parameters from Integration Server Administrator and Software AG Designer.

Using Integration Server Administrator to View Adapter Connection Parameters

Perform the following steps to view adapter connection parameters in Integration Server Administrator.

To view the parameters for a connection using Integration Server Administrator

1. In the **Adapters** menu in Integration Server Administrator's navigation area, click **webMethods Adapter for MongoDB**.

   When using the adapter with Integration Server 8.0 and later, you can sort and filter the list of connections that appears on the Connections screen.

   - To sort information on the **Connections** screen, click the **Up** and **Down** arrows at the top of the column you want to sort.
   - To filter the list of connections:

   a. On the **Connections** screen, click **Filter Connections**.

   b. Type the criterion by which you want to filter into the **Filter criteria** box. Filtering is based on the node name, not the connection alias. To locate all connections containing specific alphanumeric characters, use asterisks (*) as wildcards. For example, if you want to display all connections containing the string "abc", type `*abc*` in the **Filter criteria** box.

   c. Click **Submit**. The Connections screen displays the connections that match the filter criteria.

   d. To re-display all connections, click **Show All Connections**.

   The **Connections** screen appears, listing all the current connections. You can control the number of connections that are displayed on this screen. For more information, see "Controlling Pagination" on page 16.

2. On the **Connections** screen, click the icon for the connection you want to see.

   The View Connection screen displays the parameters for the connection. For descriptions of the connection parameters, see "Configuring Adapter for MongoDB Connections " on page 31.

3. Click **Return to webMethods Adapter for MongoDB Connections**to return to the main connections screen.

Using Designer to View Adapter Connection Parameters

Perform the following steps to view adapter connection parameters in Designer.

To view the parameters for a connection using Designer

1. From the Designer navigation area, open the package and folder in which the connection is located.

2. Double-click the connection you want to view.

The parameters for the connection appear on the **Connection Information** tab. For descriptions of the connection parameters, see "Configuring Adapter for MongoDB Connections " on page 31.

Editing Adapter Connections

If the login information for a database changes, or if you want to redefine parameters that a connection uses when connecting to a database, you can update a connection's parameters using Integration Server Administrator.

To edit a connection

1. In the **Adapters** menu in Integration Server Administrator's navigation area, click **webMethods Adapter for MongoDB**.

2. Make sure that the connection is disabled before editing it. For instructions, see "Disabling Adapter Connections" on page 40.

3. On the **Connections** screen, click the icon for the connection you want to edit.

   The Edit Connection screen displays the current parameters for the connection. Update the connection's parameters by typing or selecting the values you want to specify.

   For descriptions of the connection parameters, see "Configuring Adapter for MongoDB Connections " on page 31.

4. Click **Save Changes** to save the connection and return to the **Connections** screen.

Copying Adapter Connections

You can copy an existing Adapter for MongoDB connection to configure a new connection with the same or similar connection properties without having to re-type all of the properties for the connection. You copy adapter connections using Integration Server Administrator.

To copy a connection

1. In the **Adapters** menu in Integration Server Administrator's navigation area, click **webMethods Adapter for MongoDB**.

2. On the **Connections** screen, click the icon for the connection you want to copy.

   The Copy Connection screen displays the current parameters for the connection you want to copy. Name the new connection, specify a package name and folder name, and edit any connection parameters as needed by typing or selecting the values you want to specify.

   Note:

When you copy a connection, the new connection does not save the password of the original connection. You must enter and then retype the password before you can save the new connection.

For descriptions of the connection parameters, see "Configuring Adapter for MongoDB Connections " on page 31.

3. Click **Save Connection Copy** to save the connection and return to the **Connections** screen.

Deleting Adapter Connections

If you no longer want to use a particular Adapter for MongoDB connection, you can delete it. You delete adapter connections using Integration Server Administrator.

If you delete an Adapter for MongoDB connection, the adapter services or notifications that are defined to use the connection will no longer work. However, you can assign a different connection to an adapter service and re-use the service. To do this, use the setAdapterServiceNodeConnection built-in service. For more information, see "Changing the Connection Associated with an Adapter Service at Design Time" on page 13.

To delete a connection

1. In the **Adapters** menu in the Integration Server Administrator navigation area, click Adapter for MongoDB.

2. Make sure that the connection is disabled before deleting. To disable the connection, click **Yes** in the **Enabled** column and click **OK** to confirm. The **Enabled** column now shows **No** (Disabled) for the connection.

3. On the Connections screen, click ✕ for the connection you want to delete.

   Integration Server deletes the adapter connection.

Enabling Adapter Connections

An Adapter for MongoDB connection must be enabled before you can configure any adapter service using the connection, or before an adapter service can use the connection at run time. You enable adapter connections using Integration Server Administrator.

Note:
When you reload a package that contains enabled connections, the connections will automatically be enabled when the package reloads. If the package contains connections that are disabled, they will remain disabled when the package reloads.

To enable a connection

1. In the **Adapters** menu in the Integration Server Administrator navigation area, click **Adapter for MongoDB**.

2. On the **Connections** screen, click **No** in the **Enabled** column for the connection you want to enable.

   Integration Server Administrator enables the adapter connection and displays a ✔ and **Yes** in the **Enabled** column.

Disabling Adapter Connections

Adapter for MongoDB connections must be disabled before you can edit or delete them. You disable adapter connections using Integration Server Administrator.

To disable a connection

1. In the **Adapters** menu in the Integration Server Administrator navigation area, click **Adapter for MongoDB**.

2. On the **Connections** screen, click **Yes** in the **Enabled** column for the connection you want to disable.

   The adapter connection becomes disabled and you see a **No** in the **Enabled** column.

5 Adapter Services

Overview of Adapter Services

This chapter describes how to configure and manage Adapter for MongoDB services. For detailed descriptions of the available Adapter for MongoDB services, see "Adapter Services" on page 12.

Before Configuring or Managing Adapter Services

Perform the following steps before configuring or managing adapter services.

To prepare to configure or manage Adapter for MongoDB services

1. Start your Integration Server and Integration Server Administrator, if they are not already running.

2. Make sure you have Integration Server Administrator privileges so that you can access Adapter for MongoDB's administrative screens. For information about setting user privileges, see the *webMethods Integration Server Administrator's Guide* for your release.

3. If you have made changes to the table for a given adapter service, be sure to update the adapter service accordingly.

4. Using Integration Server Administrator, make sure the WmMongoDBAdapter package is enabled. For instructions, see "Enabling Packages" on page 23.

5. Using Integration Server Administrator, configure an adapter connection to use with the adapter service.

   Note:
   Integration Server provides a built-in service you can use at design time to change the connection associated with an adapter service. For more information, see "Changing the Connection Associated with an Adapter Service at Design Time" on page 13.

6. Start Software AG Designer if it is not already running.

7. Using Designer, create a user-defined package to contain the service, if you have not already done so. When you configure adapter services, you should always define them in user-defined packages rather than in the WmMongoDBAdapter package. For more information about managing packages for the adapter, see "Overview of Package Management" on page 22.

Configuring Select Document Service

A Select Document service retrieves specified information from a database collection. You can configure Adapter for MongoDB services using Designer. For more information about adapter services, see "Adapter Services" on page 12.

Refer the section ""Before Configuring or Managing Adapter Services" on page 42" before you configure adapter service.

To configure select service

1. Right-click the package in Designer, in which contains the service and select **New** > **Adapter Services**.
2. Select the parent namespace, type a name for adapter service and click **Next**.
3. Select **Adapter for MongoDB** as the adapter type and click **Next**.
4. Select the appropriate **Adapter Connection Name** and click **Next**.
5. Select the **Select Document** template from the list of available templates and click **Finish**.

   The adapter service editor appears for the adapter service. You can select the **Adapter Settings** tab anytime to confirm adapter service properties.
6. Use the **Collection** tab to configure the collections which the operation accesses.

Field	Description/Action
Database	Displays the database name that you provide in the **Connections**.
Collection	Lists the collection present in the **Database**.
Sampling Limit	Specify the number of sampling documents to be scanned to get the **Document Fields**.
Read Preference	The different types of read preference that MongoDB supports. For more information, see **Connection Properties** in "Configuring Adapter for MongoDB Connections " on page 31.
Read Concern	The different types of read concern that MongoDB supports. For more information, see **Connection Properties** in "Configuring Adapter for MongoDB Connections " on page 31.
Output Document Limit	Specifies the number of documents that can be displayed as output.

7. Use the **Select** tab to define the fields to be selected as follows:

   a. Use the  icon to create new rows as required. You can use  icon to fill in all the rows.

Field	Description/Action
Document Fields	Name of the fields from sampling documents that you want to use. To include a different field name, edit the field appropriately.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Output Field	Name of the output field based on the **Document Fields**.
Output Field Type	Specifies the Java data type.

8. Use **Filter** tab to specify the conditions for selecting information as follows:

   a. Select the icon to define new data fields.

   b. Select a logical operator from the AND/OR field, an Operator, and separators(left and right parenthesis), and specify values for the following fields:

Field	Description /Action
AND/OR	The logical operator.
Document Field	Name of the fields from sampling documents that you want to use. To include a different field name in the filter, edit the field appropriately. For more information, see " Examples of Designing Filters on an Array of Embedded Documents" on page 55.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Input Field	Provide a value for query input during the design time or runtime.
Input Field Type	Specifies Java data type.
Parameter	Displays the index of the output field.
Operator	The query filter operator.

   c. You can also use or icons to change the order of the data fields to ensure the parameters are parsed in the correct order.

   d. Repeat this procedure until you have specified all data fields.

9. From the File menu, select **Save**.

Configuring Insert Document Service

An Insert Document service inserts new document into database collections. You configure Adapter services using Designer. For more information about adapter services, see "Adapter Services" on page 12.

Refer the section "Before Configuring or Managing Adapter Services" on page 42 before you configure adapters services.

To configure Insert service

1. In Designer, right-click the package which contains the service and select **New > Adapter Service**.
2. Select the parent namespace, type a name for the adapter service, and click **Next** .
3. Select **Adapter for MongoDB** as the adapter type and click **Next**.
4. Select the appropriate **Adapter Connection Name** and click **Next**.
5. From the list of available templates, select the **Insert Document** template and click **Finish**.

   The adapter service editor for the adapter service appears. You can select the Adapter Settings tab at any time to confirm adapter service properties such as the **Adapter Name**, **Adapter Connection Name** , and **Adapter Service Template** .
6. Use the **Collection** tab to configure the collections which the operation accesses:

Field	Description/Action
Database	Displays the database name that you provide in the **Connections**.
Collection	Lists the collection present in the **Database**.
Sampling Limit	Specify the number of sampling documents to be scanned to get the **Document Fields**.
Acknowledgment	Requests the acknowledgment that the Write operation propagates to a specified number of `mongod` instances.
Journal	Select this option to request the **Acknowledgment** that the `mongod` instances writes to the on-disk journal.
TimeOut	Specify a time limit to avoid the Write operations from blocking indefinitely.
Bypass Document Validation	Select this option to enable the insert operation to bypass document validation.

7. Select the **Insert** tab to define the fields to be selected as follows:

   a. Use the icon to create new rows as needed. You can use the icon to fill in all rows to the table.

Field	Description/Action
Document Fields	Name of the fields from sampling documents that you want to use.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Input Field	Provide a value for query input during the design time or runtime.
Input Field Type	Specifies the Java data type.

   b. You can also use or icons to change the order of the fields or documents to ensure the parameters are parsed in the correct order.

   c. Repeat this procedure until you have specified all the parameters.

8. From the **File** menu, select **Save**.

Configuring Update Document Service

An Update Document service updates the existing information in a collection and includes a mapping for an output field count that stores the number of documents affected by the update operation.

Note:
Update service internally uses MongoDB `updateMany` operation. Refer to MongoDB documentation for more details on `updateMany`.

Refer the section "Before Configuring or Managing Adapter Services" on page 42 before configuring adapter services.

To configure Update service

1. In Designer, right-click the package which the service contains and select **New > Adapter Service**.

2. Select the parent namespace, type a name for the adapter service, and click **Next** .

3. Select **Adapter for MongoDB** as the adapter type and click **Next**.

4. Select the appropriate **Adapter Connection Name** and click **Next**.

5. From the list of available templates, select the **Update Document** template and click **Finish**.

   The adapter service editor for the adapter service appears. You can select the Adapter Settings tab at any time to confirm adapter service properties such as the **Adapter Name**, **Adapter Connection Name** , and **Adapter Service Template** .

6. Use the **Collection** tab to configure the collections which the operation accesses.

Field	Description/Action
Database	Displays the database name that you provide in the **Connections**.
Collection	Lists the collection present in the **Database**.
Sampling Limit	Specify the number of sampling documents to be scanned to get the **Document Fields**.
Acknowledgment	Requests the acknowledgment that the Write operation propagates to a specified number of `mongod` instances.
Journal	You select to request the acknowledgment that the `mongod` instances has written to the on-disk journal.
TimeOut	A time limit to avoid the Write operations from blocking indefinitely.
Upsert	You select to create a new document if the query is unable to retrieve any documents.
Bypass Document Validation	You select to enable the update operation to bypass document validation.

7. Select the **Update** tab to define the fields to be selected as follows:

   a. Use the icon to create new rows as needed. You can use the icon to fill in all rows to the table.

Field	Description/Action
Document Fields	Name of the fields from sampling documents that you want to use.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Input Field	Provide a value for query input during the design time or runtime.
Input Field Type	Specifies the Java data type.
Operator	Specifies the Update operator.

8. Use **Filter** tab to specify the conditions using the following fields for selecting information:

a. Select the icon to define new fields. You can use the icon to fill in all rows to the table.

b. Select a MongoDB Operator, and separators(left and right parenthesis), and specify values for the following fields:

Field	Description /Action
AND/OR	The logical operator.
Document Field	Name of the fields from sampling documents that you want to use. To include a different field name in the filter, edit the field appropriately. For more information, see " Examples of Designing Filters on an Array of Embedded Documents" on page 55.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Input Field	Provide a value for query input during the design time or runtime.
Input Field Type	Specifies Java data type.
Parameter	Displays the index of the output field.

c. You can also use or icons to change the order of the fields or documents to ensure the parameters are parsed in the correct order.

d. Repeat this procedure until you have specified all the parameters.

9. From the **File** menu, select **Save**.

Configuring Delete Document Service

A Delete Document service deletes the existing information in a collection and includes a mapping for an output field count that stores the number of documents affected by the delete operation.

Note:
Delete service internally uses MongoDB `deleteMany` operation. Refer to MongoDB documentation for more details on `deleteMany`.

Refer the section "Before Configuring or Managing Adapter Services" on page 42 before configuring adapter services.

To configure delete service

1. In Designer, right-click the package which the service contains and select **New > Adapter Service**.

2. Select the parent namespace, type a name for the adapter service, and click **Next** .

3. Select **Adapter for MongoDB** as the adapter type and click **Next**.

4. Select the appropriate **Adapter Connection Name** and click **Next**.

5. From the list of available templates, select the **Delete Document** template and click **Finish**.

   The adapter service editor for the adapter service appears. You can select the Adapter Settings tab at any time to confirm adapter service properties such as the **Adapter Name**, **Adapter Connection Name** , and **Adapter Service Template** .

6. Use the **Collection** tab to configure the collection using the following:

Field	Description/Action
Database	Displays the database name that you provide in the **Connections**.
Collection	Lists the collection present in the **Database**.
Sampling Limit	Specify the number of sampling documents to be scanned to get the **Document Fields**.
Acknowledgment	Requests the acknowledgment that the Write operation propagates to a specified number of `mongod` instances.
Journal	You select to request the acknowledgment that the `mongod` instances has written to the on-disk journal.
TimeOut	A time limit to avoid the Write operations from blocking indefinitely.
Bypass Document Validation	You select to enable the delete operation to bypass document validation.

7. Use **Filter** tab to specify the conditions using the following fields for selecting the documents that needs to be deleted:

   a. Select the icon to define new fields. You can use the icon to fill in all rows to the table.

   b. Select a MongoDB operator, and separators(left and right parenthesis), and specify values for the following fields:

Field	Description /Action
AND/OR	The logical operator.
Document Field	Name of the fields from sampling documents that you want to use. To include a different field name in the filter, edit the field appropriately. For more information, see " Examples of Designing Filters on an Array of Embedded Documents" on page 55.

Field	Description /Action
Field Type	Specifies the MongoDB data type for **Document Fields**.
Input Field	Provide a value for query input during the design time or runtime.
Input Field Type	Specifies Java data type.
Parameter	Displays the index of the input field.

c. You can also use or icons to change the order of the fields to ensure the parameters are parsed in the correct order.

d. Repeat this procedure until you have specified all the parameters.

8. From the File menu, select **Save**.

Configuring Aggregate Query Service

An Aggregate Query service runs aggregate operations on MongoDB collection. This service follows the concept of data processing pipelines.

Refer the section "Before Configuring or Managing Adapter Services" on page 42 before configuring adapter services.

To configure Aggregation service

1. In Designer, right-click the package which contains the service and select **New > Adapter Service**.

2. Select the parent namespace, type a name for the adapter service, and click **Next** .

3. Select **Adapter for MongoDB** as the adapter type and click **Next**.

4. Select an appropriate **Adapter Connection Name** and click **Next**.

5. From the list of available templates, select the **Aggregate Query** template and click **Finish**.

   The adapter service editor for the adapter service appears. You can select the Adapter Settings tab at any time to confirm adapter service properties such as the **Adapter Name**, **Adapter Connection Name** , and **Adapter Service Template** .

6. Use the **Aggregation** tab to configure the collection the operation accesses using the following fields:

   The **Collections** drop-down list the collections available in the specified **Database** and pick the appropriate collection from the list.

a. Use the icon to create new rows as needed. You can use the icon to fill in all rows to the top table:

Field	Description/Action
Stage	Lists various aggregation pipeline stages.
Query	Queries corresponding stages.

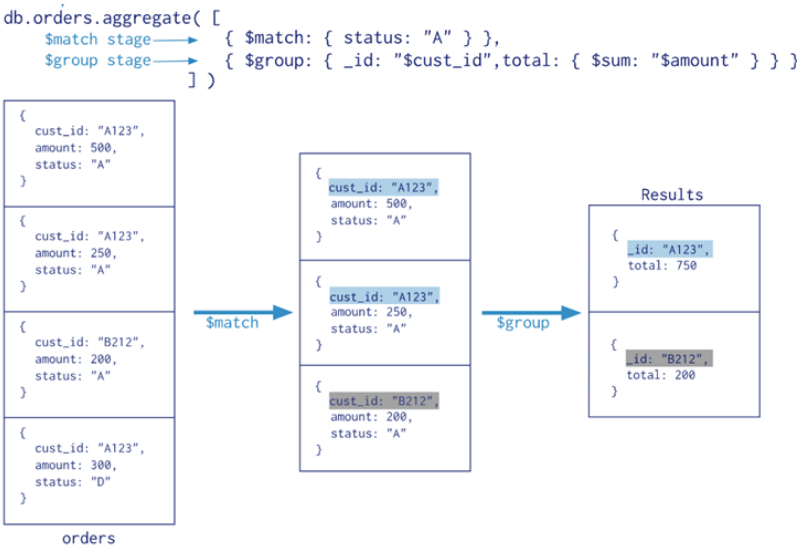
b. Use the icon to create new rows as needed. You can use the icon to fill in all rows to the following table:

Field	Description/Action
Variable	Considers any variable name provided in a **Query** as a variable. If the **Query** contains `${<var-name>}` then var-name will considered as variable.
Variable Type	MongoDB data types of the **Variable**. **Note:** If the variable type is BSON, the query input for the variable should be a JSON string.
Input Type	Specifies Java data types of the **Variable**.
Document Fields	Output document field name that you want to use.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Output Field	Name of the output field based on the **Document Fields**.
Output Field Type	Specifies the Java data type.

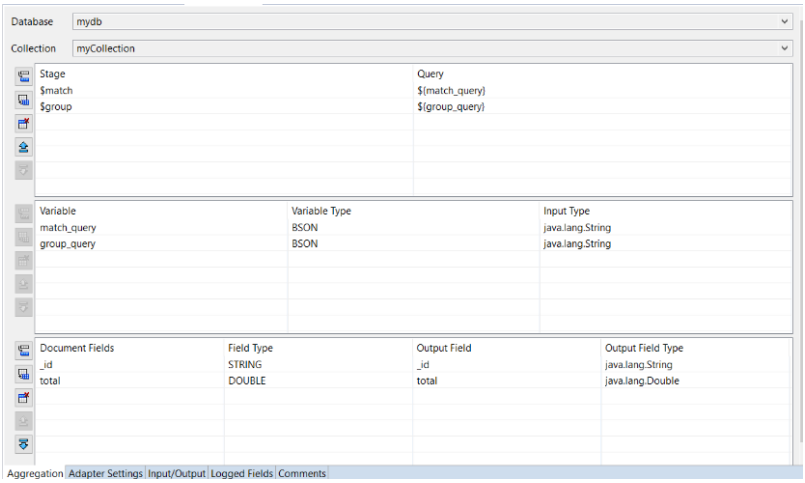
7. From the **File** menu, select **Save**.

Refer to the following example to understand the Aggregate Query service using the query:

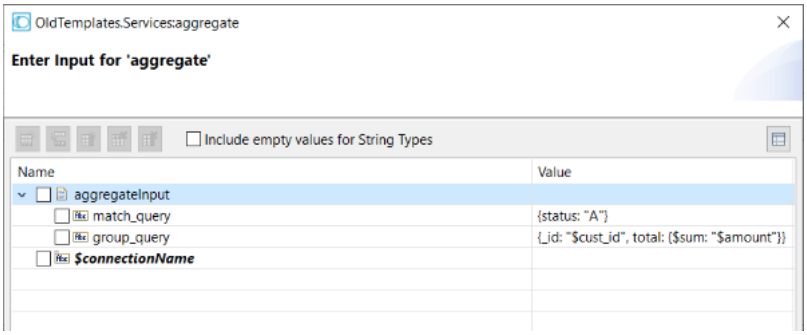
a. The following image shows the sample aggregate query used in MongoDB:



b. The following image shows the Aggregate Query service configuration with required stages:



c. The following image shows $match and $group related queries supplied at runtime:



Configuring Dynamic Query Executor Service

The Dynamic Query Executor service template is used for dynamic operations. Creating a Dynamic service allows you to execute MongoDB query directly to the adapter service as an input field. You configure Adapter services using Designer. Refer the section "Before Configuring or Managing Adapter Services" on page 42 before you configure adapter services.

To configure Dynamic Query Executor service

1. In Designer, right-click the package which contains the service and select **New > Adapter Service**.

2. Select the parent namespace, type a name for the adapter service, and click **Next** .

3. Select **Adapter for MongoDB** as the adapter type and click **Next**.

4. Select the appropriate **Adapter Connection Name** and click **Next**.

5. From the list of available templates, select the **Dynamic Query Executor** template and click **Finish**.

   The adapter service editor for the adapter service appears. You can select the Adapter Settings tab at any time to confirm adapter service properties such as the **Adapter Name**, **Adapter Connection Name** , and **Adapter Service Template** .

6. Use the **Dynamic Query** tab to configure the collections which the operation accesses.

   Select a name of **Database**, provide a JSON formatted query under **Query** at design time or runtime.

   Note:
   To understand the format for the input query, refer to the Database Commands section in *MongoDB* documentation.

   a. Use the  icon to create new rows as needed. You can use the  icon to fill in all rows to the following table.

Field	Description/Action
Variable	Considers the variable name provided in a **Query** as a *variable*. **Note:** If the **Query** contains `${<var-name>}`, then the `var-name` is considered as a *variable*.
Variable Type	MongoDB data types of the **Variable**.

Field	Description/Action
	Note: If the variable type is BSON, then the query input for that variable must be a JSON string.
Input Type	Java data types of the **Variable**.

b. Use the icon to create new rows as needed. You can use the icon to fill in all rows to the following table.

Field	Description/Action
Document Fields	Output document field name that you want to use.
Field Type	MongoDB data type for **Document Fields**.
Output Field	Name of the output field based on the **Document Fields**.
Output Field Type	Java data type.

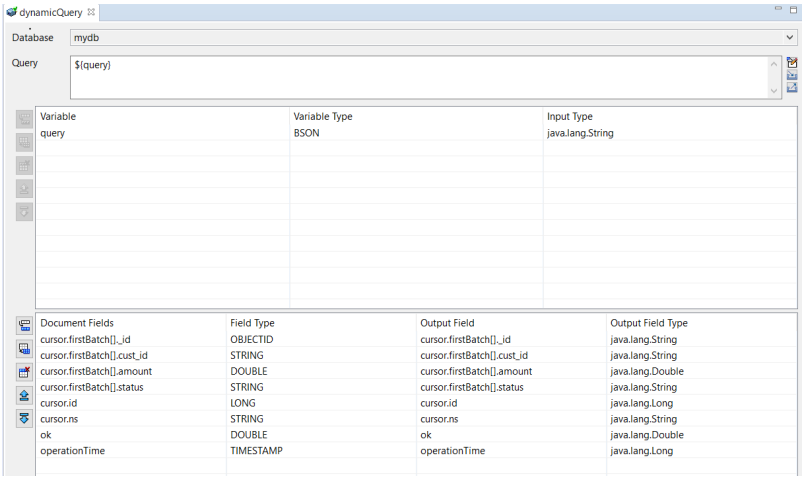
c. You can also use or icons to change the order of the fields or documents to ensure the parameters are parsed in the correct order.

d. Repeat this procedure until you have specified all the parameters.

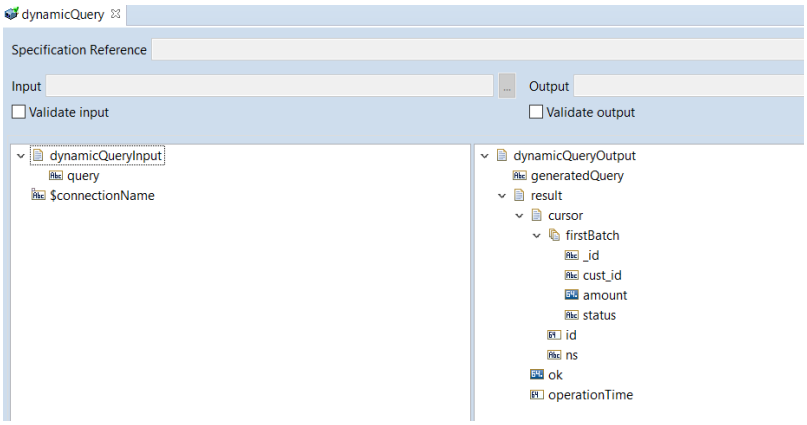
7. From the **File** menu, select **Save**.

Refer the following examples to understand the Dynamic Query Executor service using *query*:

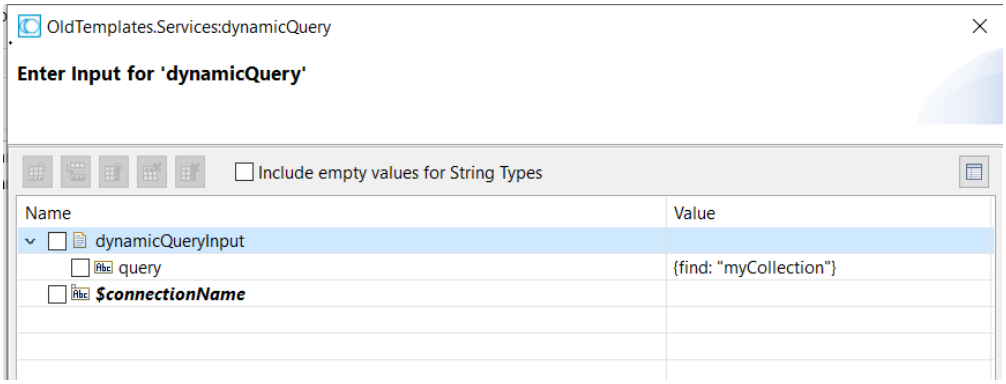
- The following image shows the query input variable and the output fields to generate **Input/Output** signature:



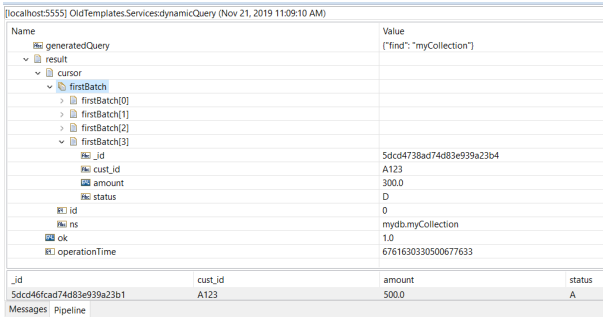
- The following image displays the generated input and output signatures:



- The following image displays the input query you provide at runtime for `find` command:



- The following image displays the results for the above input query:



Examples of Designing Filters on an Array of Embedded Documents

The examples in this section show the filters created in Adapter services or notifications to query data from an array of embedded documents.

The following image is a sample data in MongoDB.

```
_id: ObjectId("6007aa4baeebea418c72fc15")
customer: "A"
items: Array
  0: Object
     itemId: "001"
     qty: 5
  1: Object
     itemId: "003"
     qty: 15
```

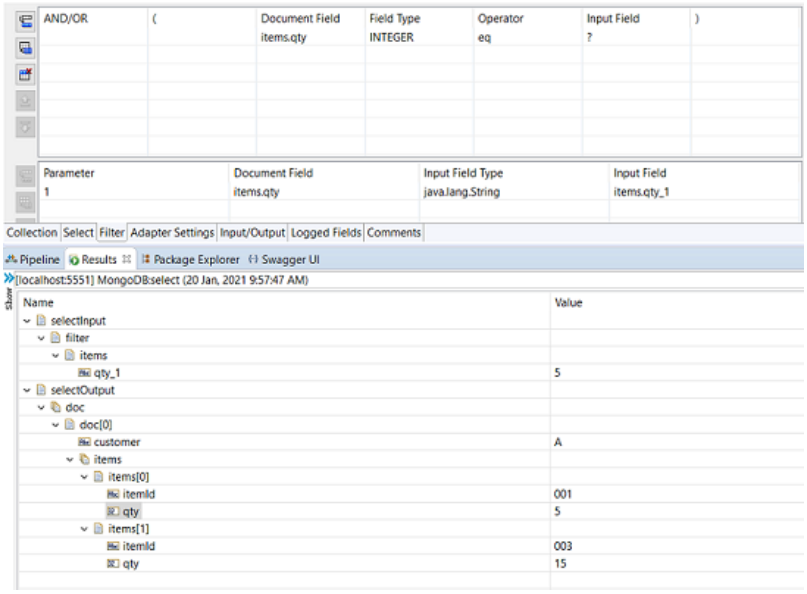
```
_id: ObjectId("6007aa89aeebea418c72fc16")
customer: "B"
items: Array
  0: Object
     itemId: "002"
     qty: 10
```

```
_id: ObjectId("6007aaa9aeebea418c72fc17")
customer: "C"
items: Array
  0: Object
     itemId: "002"
     qty: 10
  1: Object
     itemId: "004"
     qty: 15
```

Specifying a query condition on a nested field in the array of embedded documents

The following example shows a query condition that returns all the documents where the array has at least one embedded document that contains the field that matches the query condition.

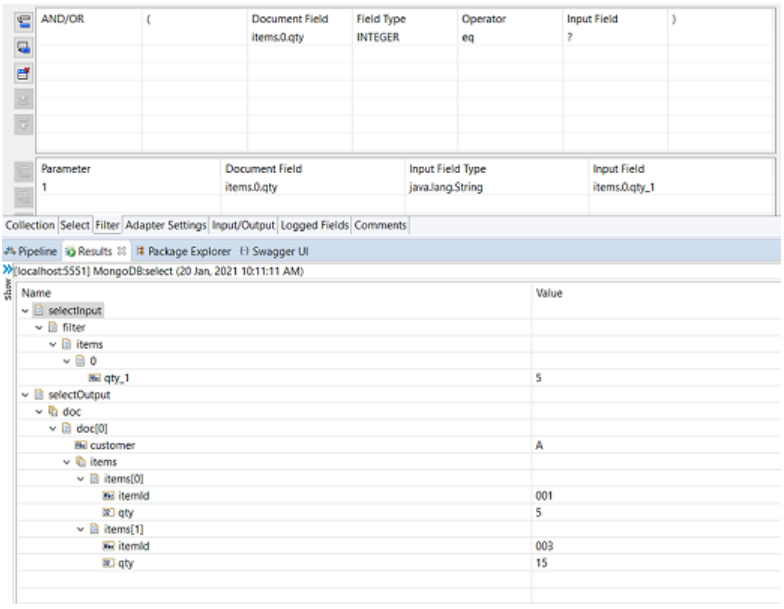
The following image shows a query condition that returns all the documents where the *items* array has at least one embedded document that contains the field *qty* whose value is less than or equal to 5.



Specifying a query condition on a nested field in the array of embedded documents using the array index

The following example shows a query condition that returns all the documents where the array has as its first element a document that contains the field that matches the query condition.

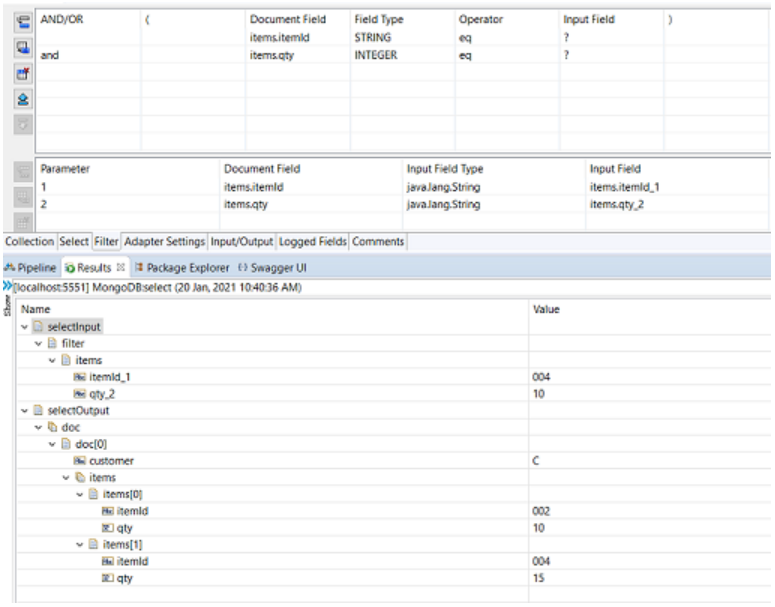
The following image shows a query condition that returns all the documents where the *items* array has as its first element a document that contains the field *qty* whose value is equal to `5`.



Specifying a query condition with a combination of nested fields in the array of embedded documents

The following example shows a query condition that returns all the documents where the array has at the least one embedded document(not necessarily the same embedded document but existing within the same array) that matches both the conditions.

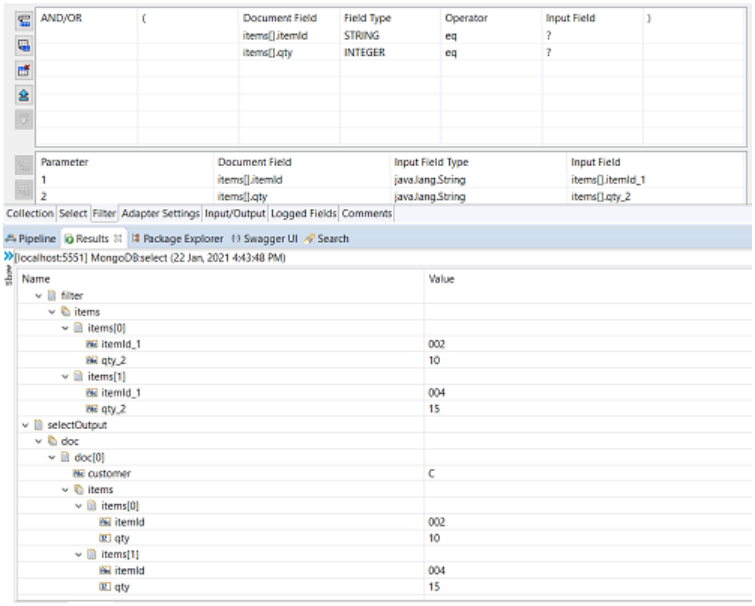
The following image shows a query condition that returns all the documents where the *items* array has one embedded document that contains the field *itemId* whose value is equal to `004` and another embedded document (within the same array) that contains the field *qty* whose value is equal to `10`.



Specifying an array of query conditions in the array of embedded documents

The following example shows an array of query conditions that returns all the documents within an array that exactly matches the query conditions.

The following image shows a query condition that returns all the documents where the *items* array has one embedded document that contains the field *itemId* whose value is equal to `002` and *qty* is equal to `10` and the second embedded document in the same array that contains the field *itemId* whose value is equal to `004` and *qty* is equal to `15`.



6 Adapter Listeners and Listener Notifications

Overview of Adapter Listeners and Listener Notifications

This chapter describes how to configure and manage Adapter for MongoDB listeners and listener notifications.

For detailed descriptions of the available Adapter for MongoDB notifications, see "Adapter Listeners and Listener Notifications" on page 14.

Preparing to Configure New Listeners

To prepare to configure a new listener

1. Ensure that Integration Server and Integration Server Administrator are started.

   Note:
   Make sure that you have webMethods administrator privileges so that you can access the adapter's administrative screens. For information about setting user privileges, see the *webMethods Integration Server Administrator's Guide*.

2. Using Integration Server Administrator, make sure that the WmMongoDBAdapter package is enabled. To verify the status of the WmMongoDBAdapter package, see "Package Management" on page 10.

3. Using Software AG Designer, create a user-defined package to contain the listener. For more information about managing packages, see "Overview of Package Management" on page 22.

Configuring Listener Notification

You can configure listener notification using Software AG Designer. For more information, see "Adapter Listeners and Listener Notifications" on page 14.

Configuring an Adapter Listener

1. In Integration Server Administrator, select **Adapters** > **Adapter For MongoDB**.

2. In the **Adapter for MongoDB** menu, select **Listeners**.

3. On the Listeners screen, select **Configure New Listener**.

4. On the Listener Types screen, select **MongoDB Database Listener** or **MongoDB Collection Listener**.

5. On the Configure Listener Type screen, configure the following fields:

Parameter	Description/Action
Package	The package in which to create the listener. Create the listener in a user-defined package rather than in the adapter's package. For other considerations when creating packages for the adapter, see "Overview of Package Management" on page 22. You must create the package using Software AG Designer before you can specify the package by using this parameter. For information about creating packages, see *webMethods Service Development Help*.
Folder Name	The folder in which to create the listener.
Listener Name	The name of the new listener.
Connection Name	The name of the connection.
Retry Limit	The number of times that the adapter tries to reconnect, if the adapter fails to connect to or loses connection with the MongoDB server.
Retry Backoff Timeout	The number of seconds that elapse between each of the retries specified in the retry limit.

Important:
The listener name is prefixed by the folder name and is separated by a colon. For example, if the folder name is "Folder1" and the listener name is "Listener1", the listener name in the Listeners screen will be "Folder1:Listener1".

6. In the **Listener Properties** section, configure the following fields:

Parameter	Description/Action
Collection Name	The name of collection for which change stream is created. **Note:** This field is applicable for MongoDB Collection Listener.

7. Click **Save.**

Enabling Listeners

Before you enable a listener, you need to configure one or more notifications to associate with the listener. If no notifications are configured when you enable the listener, Integration Server Administrator displays a warning message.

After you configure your notifications, you must enable the listener so that the associated notifications communicate appropriately with the listener at run time. You enable the listeners using Integration Server Administrator.

The **Status** column indicates the readiness of the listener. If the status is **Succeeded**, the listener is ready to be enabled. If the status is **Failed**, an error occurred during startup. If an error occurs during startup, the state will not change to **Enabled** when refreshing the page. Errors at this stage typically indicate a problem with either the listener configuration or the network. Review the listener settings and check the network.

For more information, see "Configuring an Adapter Listener" on page 60.

Note:
When you reload a package that contains enabled listeners, the listeners will automatically be enabled when the package reloads. If the package contains listeners that are disabled, they will remain disabled when the package reloads.

To enable a listener

1. In Integration Server Administrator, select **Adapters** > **webMethods Adapter for MongoDB**.
2. In the Adapter for MongoDB menu, select **Listeners**. The Listeners screen appears.
3. Select **Enabled** from the drop-down list in the **State** field. Integration Server Administrator enables the listener.

   The **Enable all suspended** link helps you change the state quickly for multiple listeners.

Configuring Insert Notification

Insert Notification publishes notification of insert operation in a database collection. You configure the notifications using Designer.

Be sure to review the sections "Adapter Listeners and Listener Notifications" on page 14 before you configure Insert Notifications.

To create an Insert Notification

1. In Designer, right-click the package in which the notification should be contained and select **New > Adapter Notification**.
2. Select the parent namespace, type a name for the adapter notification, and click **Next**.
3. Select **Adapter for MongoDB** as the adapter type and click **Next**.
4. Select the **Insert Notificaton** template and click **Next**.
5. Select the appropriate **Adapter Notification Listener** and click **Next**.

   The name of the publishable document associated with this notification is displayed.

6. Click **Finish**.

   For more information about listener notifications and publishable documents, see "Adapter Listeners and Listener Notifications" on page 14. For more details about the Integration Server publishable documents, see the *Publish-Subscribe Developer's Guide* for your release.

7. Select the **Collection** tab and use the following fields:

   a. The **Collection** field displays the collection name that you provide in the **MongoDB Collection Listener**.

   b. Use **Sampling Limit** to specify the number of sampling documents to be scanned to get the **Document Fields**.

8. Select the **Document Fields** tab and perform the following:

   a. Use the icon to create new rows and icon to fill in all rows to the table:

Field	Description/Action
Document Fields	Name of the fields from sampling documents that you want to use.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Output Field	Name of the output field based on the **Document Fields**.
Output Field Type	Specifies the Java data type.

9. Use **Filter** tab for defining the conditions using the following:

   a. Use the icon to create new rows and icon to fill in all rows to the table.

   b. Select a MongoDB operator, separator(left and right parenthesis), and specify values for the following fields:

Field	Description/Action
AND/OR	The logical operator.
Document Field	Name of the fields from sampling documents that you want to use. To include a different field name in the filter, edit the field appropriately. For more information, see " Examples of Designing Filters on an Array of Embedded Documents" on page 55.
Field Type	Specifies the MongoDB data Type for **Document Fields**.
Input Field	Provide a value for query input during the design time or runtime.

c. You can also use or icons to change the order of the fields or documents to ensure the parameters are parsed in the correct order.

d. Repeat this procedure until you have specified all the parameters.

10. From the **File** menu, select **Save**.

Configuring Update Notification

Update Notification publishes notification of update operations on a database collection. You configure the notifications using Designer.

Be sure to review the sections "Adapter Listeners and Listener Notifications" on page 14 before you configure Update Notifications.

To create a Update Notification

1. In Designer, right-click the package in which the notification should be contained and select **New > Adapter Notification**.

2. Select the parent namespace, type a name for the adapter notification, and click **Next**.

3. Select **Adapter for MongoDB** as the adapter type and click **Next**.

4. Select the **Update Notificaton** template and click **Next**.

5. Select the appropriate **Adapter Notification Listener** and click **Next**.

   The name of the publishable document associated with this notification is displayed.

6. Click **Finish**.

   For more information about listener notifications and publishable documents, see "Adapter Listeners and Listener Notifications" on page 14. For more details about the Integration Server publishable documents, see the *Publish-Subscribe Developer's Guide* for your release.

7. Select the **Collection** tab and use the following fields:

   a. The **Collection** field displays the collection name that you provide in the **MongoDB Collection Listener**.

   b. Use **Sampling Limit** to specify the number of sampling documents to be scanned to get the **Document Fields**.

8. Select the **Document Fields** tab and perform the following:

a. Use the icon to create new rows and icon to fill in all rows to the table:

Field	Description/Action
Document Fields	Name of the fields from sampling documents that you want to use.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Output Field	Name of the output field based on the **Document Fields**.
Output Field Type	Specifies the Java data type.

9. Use **Filter** tab for defining the conditions using the following:

a. Use the icon to create new rows and icon to fill in all rows to the table.

b. Select a MongoDB operator, separator(left and right parenthesis), and specify values for the following fields:

Field	Description/Action
AND/OR	Logical operator.
Document Field	Name of the fields from sampling document that you want to use. To include a different field name in the filter, edit the field appropriately. For more information, see " Examples of Designing Filters on an Array of Embedded Documents" on page 55.
Input Field	Provide a value for query input during the design time or runtime.
Field Type	MongoDB data types.

c. You can also use or icons to change the order of the fields or documents to ensure the parameters are parsed in the correct order.

d. Repeat this procedure until you have specified all the parameters.

10. From the **File** menu, select **Save**.

Configuring Delete Notification

A Delete Notification publishes notification of delete operation on a database collection. You configure the notifications using Designer.

Be sure to review the sections "Adapter Listeners and Listener Notifications" on page 14 before you configure Delete Notifications.

To create a Delete Notification

1. In Designer, right-click the package in which the notification should be contained and select **New > Adapter Notification**.
2. Select the parent namespace, type a name for the adapter notification, and click **Next**.
3. Select **Adapter for MongoDB** as the adapter type and click **Next**.
4. Select the **Delete Notificaton** template and click **Next**.
5. Select the appropriate **Adapter Notification Listener** and click **Next**.

   The name of the publishable document associated with this notification is displayed.
6. Click **Finish**.

   For more information about listener notifications and publishable documents, see "Adapter Listeners and Listener Notifications" on page 14. For more details about the Integration Server publishable documents, see the *Publish-Subscribe Developer's Guide* for your release.
7. In **Collection** tab, the **Collection** field displays the collection name that you provide in the **MongoDB Collection Listener**.
8. From the **File** menu, select **Save**.

Configuring Rename Notification

Rename Notification is generated when the name of an existing collection is renamed. You configure the notifications using Designer.

Be sure to review the sections "Adapter Listeners and Listener Notifications" on page 14 before you configure Rename Notification.

To create a Rename Notification

1. In Designer, right-click the package in which the notification should be contained and select **New > Adapter Notification**.
2. Select the parent namespace, type a name for the adapter notification, and click **Next**.
3. Select **Adapter for MongoDB** as the adapter type and click **Next**.
4. Select the **Rename Notificaton** template and click **Next**.
5. Select an appropriate **Adapter Notification Listener** and click **Next**.

The name of the publishable document associated with this notification is displayed.

6. Click **Finish**.

   For more information about listener notifications and publishable documents, see "Adapter Listeners and Listener Notifications" on page 14. For more details about the Integration Server publishable documents, see the *Publish-Subscribe Developer's Guide* for your release.

7. In the **Database** tab, the **Database** field displays the name of the database that you provide in the **Connections**.

   Note:
   The **Collection** field is empty, if the selected listener is **MongoDB Database Listener**.

8. Click **Save** from the **File** menu.

Configuring Replace Notification

A Replace Notification notifies notification of the replace operations on a collection. You configure the notifications using Designer.

Be sure to review the sections "Adapter Listeners and Listener Notifications" on page 14 before you configure Replace Notifications.

To create Replace Notification

1. In Designer, right-click the package in which the notification should be contained and select **New > Adapter Notification**.

2. Select the parent namespace, type a name for the adapter notification, and click **Next**.

3. Select **Adapter for MongoDB** as the adapter type and click **Next**.

4. Select the **Replace Notificaton** template and click **Next**.

5. Select the appropriate **Adapter Notification Listener** and click **Next**.

   The name of the publishable document associated with this notification is displayed.

6. Click **Finish**.

   For more information about listener notifications and publishable documents, see "Adapter Listeners and Listener Notifications" on page 14. For more details about the Integration Server publishable documents, see the *Publish-Subscribe Developer's Guide* for your release.

7. Select the **Collection** tab and use the following fields:

a. The **Collection** field displays the collection name that you provide in the **MongoDB Collection Listener**.

b. Use **Sampling Limit** to specify the number of sampling documents to be scanned to get the **Document Fields**.

8. Select the **Document Fields** tab and perform the following:

a. Use the icon to create new rows and icon to fill in all rows to the table:

Field	Description/Action
Document Fields	Name of the fields from sampling documents that you want to use.
Field Type	Specifies the MongoDB data type for **Document Fields**.
Output Field	Name of the output field based on the **Document Fields**.
Output Field Type	Specifies the Java data type.

9. Use **Filter** tab for defining the conditions using the following:

a. Use the icon to create new rows and icon to fill in all rows to the table.

b. Select a MongoDB operator, separator(left and right parenthesis), and specify values for the following fields:

Field	Description/Action
AND/OR	The logical operator.
Document Field	Name of the fields from sampling documents that you want to use. To include a different field name in the filter, edit the field appropriately. For more information, see " Examples of Designing Filters on an Array of Embedded Documents" on page 55.
Field Type	Specifies the MongoDB data Type for **Document Fields**.
Input Field	Provide a value for query input during the design time or runtime.

c. You can also use or icons to change the order of the fields or documents to ensure the parameters are parsed in the correct order.

d. Repeat this procedure until you have specified all the parameters.

10. From the **File** menu, select **Save**.

Configuring Drop Notification

Drop Notification publishes a notification of drop operation when an existing collection is dropped from the database collection. You configure the notifications using Designer.

Be sure to review the sections "Adapter Listeners and Listener Notifications" on page 14 before you configure Drop Notifications.

To create a Drop Notification

1. In Designer, right-click the package in which the notification should be contained and select **New > Adapter Notification**.
2. Select the parent namespace, type a name for the adapter notification, and click **Next**.
3. Select **Adapter for MongoDB** as the adapter type and click **Next**.
4. Select the **Drop Notificaton** template and click **Next**.
5. Select the appropriate **Adapter Notification Listener** and click **Next**.

   The name of the publishable document associated with this notification is displayed.
6. Click **Finish**.

   For more information about listener notifications and publishable documents, see "Adapter Listeners and Listener Notifications" on page 14. For more details about the Integration Server publishable documents, see the *Publish-Subscribe Developer's Guide* for your release.
7. In the **Database** tab, the **Database** field displays the database name that you provide in the **Connections**.

   Note:
   The **Collection** field is empty, if the selected listener is **MongoDB Database Listener**.
8. From the **File** menu, select **Save**.

   Note:
   This notification is not available for version 3.6.

Configuring Invalidate Notification

Invalidate Notification is generated, when a collection or database is dropped or a collection renamed. You configure the notifications using Designer.

Be sure to review the sections "Adapter Listeners and Listener Notifications" on page 14 before you configure Invalidate Notifications.

To create an Invalidate Notification

1. In Designer, right-click the package in which the notification should be contained and select **New > Adapter Notification**.

2. Select the parent namespace, type a name for the adapter notification, and click **Next**.

3. Select **Adapter for MongoDB** as the adapter type and click **Next**.

4. Select the **Invalidate Notificaton** template and click **Next**.

5. Select the appropriate **Adapter Notification Listener** and click **Next**.

   The name of the publishable document associated with this notification is displayed.

6. Click **Finish**.

   For more information about listener notifications and publishable documents, see "Adapter Listeners and Listener Notifications" on page 14. For more details about the Integration Server publishable documents, see the *Publish-Subscribe Developer's Guide* for your release.

7. In **Collection** tab, the **Collection** field displays the collection name that you provide in the **MongoDB Collection Listener.**

   Note:
   The **Collection** field is empty, if the selected listener is **MongoDB Database Listener**.

8. From the **File** menu, select **Save**.

7 Predefined Health Indicator

Predefined Health Indicator

Microservices Runtime includes predefined health indicators for some of its basic components. The health indicator captures the connection details for all the WmART based adapters at runtime. For more information, see *webMethods Adapter Runtime User's Guide*.

8 Administrator APIs

Administrator APIs

The Administrator APIs are available for Adapter for MongoDB. For more information about Administrator APIs and samples, see *webMethods Adapter Runtime User's Guide*.

9 Configuration Variables Templates for Adapter Assets in Microservices Runtime

Configuration Variables Templates for Adapter Assets in Microservices Runtime

The webMethods Adapter Runtime (ART) asset properties that can be configured from Integration Server Administrator are available in the configuration variables template (`application.properties` file) generated by Microservices Runtime. For more information, see *webMethods Adapter Runtime User's Guide* and *Developing Microservices with webMethods Microservices Runtime*.