

MongoDB Adapter for SAP Integration Suite

Version 1.0.1 – June 2026

THE BEST RUN SAP

Contents

1 Introduction 3
1.1 Objective 3
1.2 Coding Samples 3
1.3 Internet Hyperlinks 3
1.4 Overview 3
1.5 Features 4
2 Installation and Configuration 5
2.1 Adapter Installation on Cloud Foundry 5
2.1.1 Prerequisites 5
2.1.2 Procedure 5
2.2 Monitor the Deployment Status 7
3 Getting Started: MongoDB Adapter 8
3.1 Architecture Overview 8
3.2 Application Configuration 8
3.3 Authentication 9
3.3.1 Creation of User Credentials 9
4 MongoDB Adapter Configuration 11
4.1 Receiver Adapter 11
4.1.1 Connection 11
4.1.2 Processing 14
5 MongoDB Operations 18
5.1 Document: Insert 18
5.2 Document: Update 20
5.3 Document: Query (Multiple Records) 21
5.4 Document: Query (Single Record) 23
5.5 Databases: Run Command 24
5.6 Databases: Run Command (Bulk Write) 25
5.7 Collection: List 28
6 Support 29
6.1 Tips 29
6.1.1 Troubleshooting 29

1 Introduction

1.1 Objective

This is the official guide for the MongoDB Adapter for SAP Integration Suite. This guide covers all relevant information for integration developers to start working with the MongoDB adapter. Read this guide carefully before using the Adapter.

1.2 Coding Samples

Any software coding and/or code lines/strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended to better explain and visualize the syntax and phrasing rules of certain coding. The correctness and completeness of the Code given herein are not guaranteed.

1.3 Internet Hyperlinks

The documentation may contain hyperlinks to the Internet. These hyperlinks are intended to serve as a hint about where to find related information. The availability and the correctness of this related information, or the ability of this information to serve a particular purpose, are not warranted.

1.4 Overview

MongoDB is an open-source, document-oriented NoSQL database designed to store and manage data in a flexible, scalable, and high-performance way. Instead of using rows and tables like traditional relational databases, MongoDB organizes data into **collections** of **document**.

MongoDB adapter is a Cloud Integration capability available on SAP Integration Suite that allows you to access a MongoDB database and perform CRUD operations on data stored in MongoDB.

The adapter works with MongoDB's APIs, which allow developers to access and integrate its components into their applications, enabling the management of documents and collections.

1.5 Features

Adapter has the following features:

- Functions as a receiver adapter that allows you to work with MongoDB **Document**, **Collections** and **Databases** entities.
- Supports **LDAP (PLAIN)** via **User Credentials**.
- Provides **Query Options** to fetch data along with desired projection, limit, sorting, batch size, document count, etc.
- Supports **Multi Update** and **Multi Delete** capability for documents.
- Allows you to perform the **Run Command** operation to execute on the MongoDB.

2 Installation and Configuration

This section details the file(s) available as part of the installation package and the prerequisites to configure the MongoDB adapter.



The MongoDB adapter is available as part of your SAP Integration Suite license.

2.1 Adapter Installation on Cloud Foundry

Before the adapter can be used in the Cloud Foundry environment, it must be deployed to the SAP Integration Suite tenant.

2.1.1 Prerequisites

To deploy the MongoDB adapter, you must have access to the *SAP Integration Suite* license.

2.1.2 Procedure

You can deploy the adapter using the following methods:



The following installation procedure is compatible with Apache Camel 2, Apache Camel 3.

2.1.2.1 Adapter Installation by Creating a New Integration Flow

The MongoDB adapter is available for selection in the Receiver adapter list and can be deployed in the **Design** tab directly as you use it in an Integration flow.

Purpose

To install an adapter for use in your Integration flow.

Procedure

Go to the **Design** workspace and select the integration package where you want to create a new Integration flow.

1. Click **Edit** to make the package editable.
2. Go to the **Artifacts** tab. Click **Add** and select **Integration Flow**.

3. Enter **Name** and **ID** for your flow. Additionally, select **Runtime Profile** from the drop-down and choose **Sender** and **Receiver** systems from the list . Finally, click **Add** to create the integration flow.
4. Go to the newly created integration flow and click **Edit** to make it editable.
5. In the integration flow, click **End** to add a **Connector** between the **End** and the **Receiver** Box.
   A drop-down with the available adapters appears. The MongoDB adapter should show up in the list.
6. Select the MongoDB adapter from the list. The adapter is now imported, which *triggers* an adapter deployment. Once the adapter is deployed, a success message is displayed. After the above steps are done, the adapter is successfully deployed in your Design workspace of the SAP Integration Suite tenant.

2.1.2.2 Adapter Installation without Creating a New Integration Flow

> The following procedure explains how the adapter is migrated from the Discover workspace to the Design workspace of the SAP Integration tenant.

This method is useful for scenarios where integration flow packages are migrated from development to a higher environment, such as Production.

The adapter can be imported into the Design workspace without creating an integration flow. Use the Transport Management Service (TMS) to import/transport the adapter to a higher environment. Alternatively, if the TMS is not available in the landscape, the adapter package can be imported to the Design workspace by copying it from the Discover workspace.

Purpose

To copy the integration package from the Discover workspace and import the MongoDB adapter to the Design workspace, follow these steps:

Procedure

1. Go to the **Discover** workspace.
2. In the search box, search for the MongoDB **Adapter for SAP Integration Suite** package.
3. Select the package and click **Copy**. This copies the package from the Discover workspace to the Design workspace.
4. Go to the Design workspace and select the copied MongoDB **Adapter for SAP Integration Suite** package.
5. In the **Actions** tab of the selected package, click **Deploy**. This completes the adapter deployment to the Design workspace.

2.2 Monitor the Deployment Status

After the adapter deployment is complete, you can check the status in the **Monitor** section.

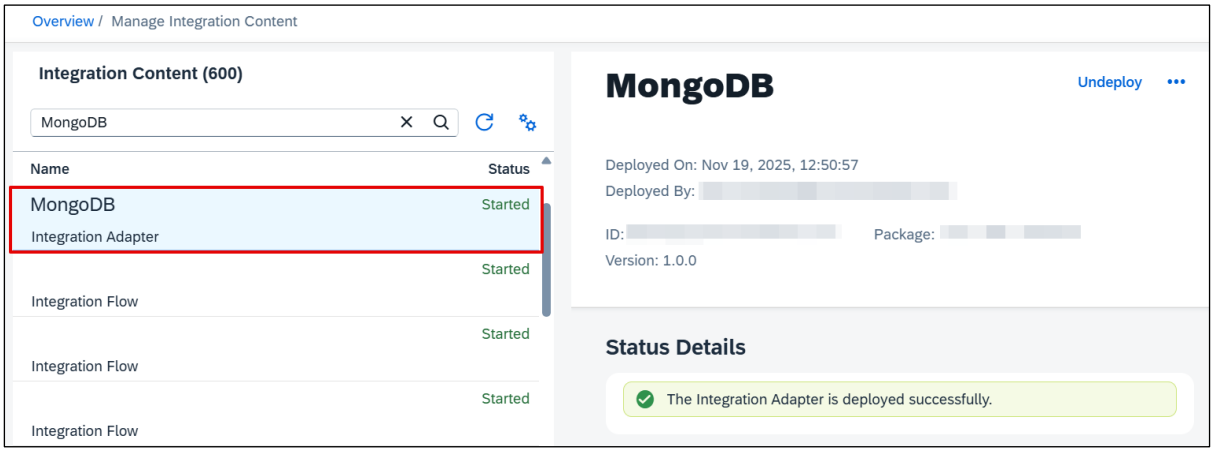
Purpose

To check the status of the deployed adapter.

Procedure

1. Under the **Monitor** tab, click **Integrations and APIs**. This opens the **Overview** page.
2. On the **Overview** page, go to the **Manage Integration Content** section and click **All**. This opens the **Integration Content** page with a list of all the deployed adapters.

Here, you can check and confirm the deployment status of your adapter.

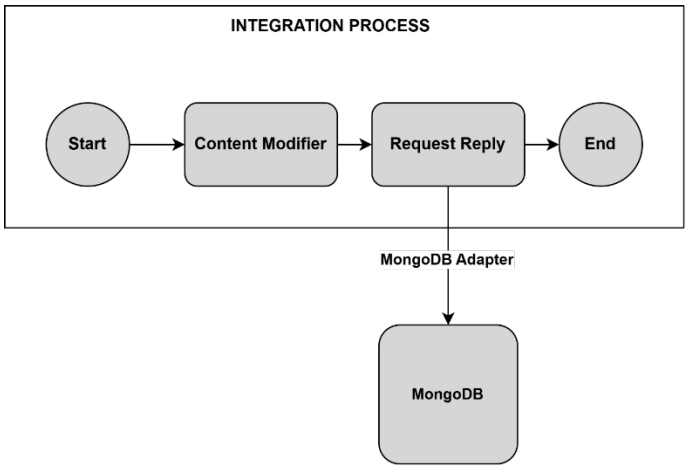


3 Getting Started: MongoDB Adapter

This section explains how to configure the MongoDB adapter for SAP Integration Suite. You can find information about adapter architecture, application configuration, and authentication for the MongoDB Adapter.

3.1 Architecture Overview

The MongoDB adapter is designed to function as a receiver adapter. In such a scenario, where the MongoDB adapter is used as a receiver adapter, SAP Integration Suite acts as the initiator of the calls. SAP Integration Suite tenant sends the operation request to MongoDB (this is a receiver system), MongoDB works on the request and sends the data back to the SAP Integration Suite tenant.



3.2 Application Configuration

This section contains information to configure MongoDB application before you use the adapter to connect to it.

Overview:

- For an overview of MongoDB, see MongoDB Documentation.

Authentication:

- For information regarding authentication, see Authentication and Authorization.
- For details regarding the Connection String, see Connection String.

Operations:

- For information on Databases and Collections, see Databases and Collections in MongoDB.
- For information related to operations, see CRUD Operations.
- For information related to the Run Command, see MongoDB Run Commands.

3.3 Authentication

This section details the authentication mechanism supported by the MongoDB Adapter in the SAP Integration Suite.

The adapter supports standard authentication methods, like **User Credentials**. You can securely store the security artifacts in SAP Security Material. This ensures that credentials can be securely provided to the Adapter.

Before setting up the authentication, you must create credentials in the Security Material of the SAP Integration Suite.

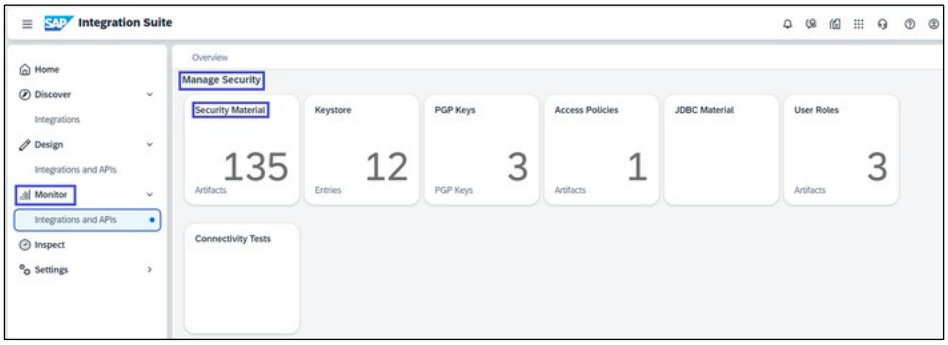
3.3.1 Creation of User Credentials

Purpose

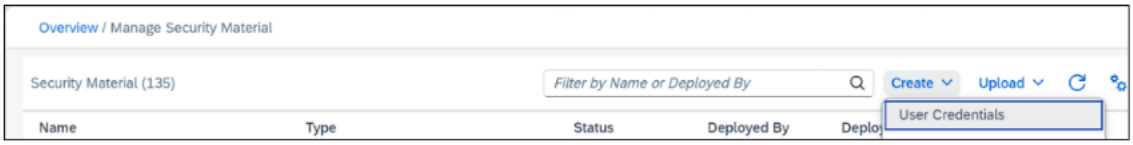
To create credentials in Security Material for **User Credentials**.

Procedure

1. Navigate to Monitor > Integrations and APIs in SAP Integration Suite. This opens the **Overview** page.
2. On the **Overview** page, go to the **Manage Security** section and click **Security Material**.



3. On the **Manage Security Material** page, click **Create** to select **User Credentials** from the dropdown.



4. In the **User Credentials** popup, provide the details below.

Parameter	Description
Name	Specify the name for the security artifact. The artifact name is used as an alias for the confidential data assigned by this parameter.
Description	Enter a description for the artifact.
Type	Select **User Credentials**. This allows you to configure a specific system to enable a connection with your integration flow artifact.
User	Specify the username used to invoke the receiver system.
Password	Specify the password against which the user must be authenticated.

5. Click **Deploy** to complete the process.

When you refresh the **Manage Security Material** page, the new artifact is displayed (with Type **User Credentials**) on the artifact table.

4 MongoDB Adapter Configuration

This section describes the parameters to be configured for your MongoDB adapter. You need to configure the **Connection** and **Processing** tabs. A description and example usage for every field have been added.

4.1 Receiver Adapter

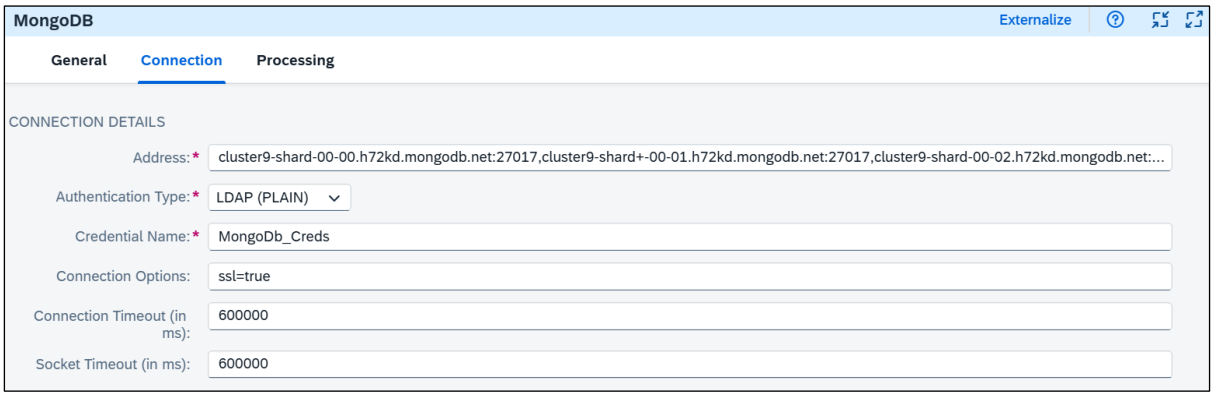
In this section, you will learn how to configure the MongoDB receiver adapter. On selecting the MongoDB adapter from the list of adapters, you must configure the **Connection** and **Processing** tabs.

4.1.1 Connection

The Connection tab contains connection and authentication parameters for the MongoDB adapter.

The Security artifact created in the previous section (Authentication) should be used in the **Connection tab** of the Adapter.

A sample screenshot for the Connection tab is given below.



The table below contains the complete list of fields that can be found in the **Connection** tab.

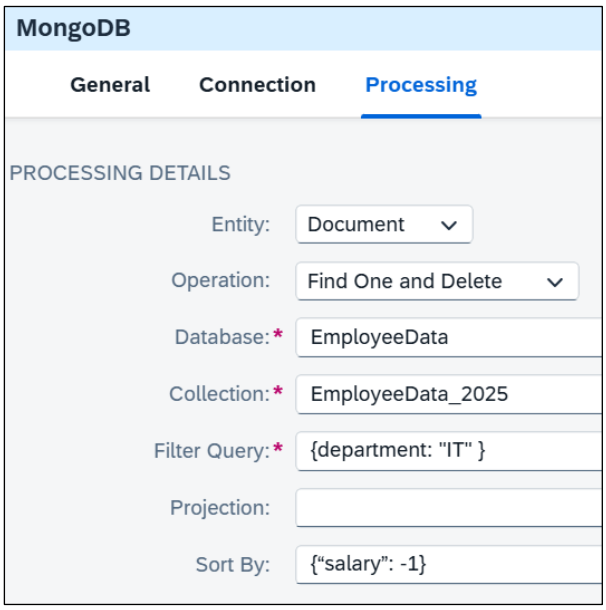
Parameter	Description
Address	Specify the Cluster address to connect to MongoDB Database. Syntax: host1:port1,host2:port2,hostn:portn Currently, only **Standard Connection String** is supported. You can find your Cluster address in MongoDB Atlas by navigating to **Connect > Compass**. Select **I have MongoDB Compass installed** and select version **1.11 or earlier**. Ensure that you copy the host address and port only as shown in below example. Example: `ac-utuhakc-shard-00-00.xqxxotp.mongodb.net:27017,ac-utuhakc-shard-00-01.xqxxotp.mongodb.net:27017,ac-utuhakc-shard-00-02.xqxxotp.mongodb.net:27017`
Authentication Type	Select the Authentication Type to connect to MongoDB server. Currently, only **LDAP (PLAIN)** is supported.
Credentials Name	Specify the User Credentials security artifact alias that stores the username-password pair. Example: `MongoDB_Creds`
Connection Options	Specify the optional parameters for the MongoDB connection string to customize connection preferences, such as authentication, timeouts, and read/write options, etc. Syntax: option1=value1&option2=value2. Example: `ssl=true&authSource=admin.` The `ssl=true` parameter must be included to establish the secure connection using SSL. For more information, refer to Connection String options.
Connection Timeout (in ms)	Specify the maximum waiting time (in milliseconds) for the connection to be established. Example: `60000`

Parameter	Description
Socket Timeout (in ms)	Specify the maximum waiting time (in milliseconds) to get a response. Example: `60000`

4.1.2 Processing

The **Processing** tab allows you to configure the operations that can be performed using the MongoDB Adapter.

A sample screenshot has been given below.



The table below contains the complete list of fields that can be found in the **Processing** tab.

Parameter	Description
Entity	Select the required Entity: • **Collections** • **Document** • **Databases**
Operation	Select the operation to be performed on the selected entity.
Database	Specify the name of the Database. Example: `EmployeeData` You will be able to find it inside your Cluster.

Parameter	Description
Collection	Specify the name of the MongoDB collection on which the operation is to be performed. Example: `EmployeeData_2025` ℹ You will be able to find it inside your Cluster.
Filter Query	Specify the expression to modify the matching documents from the collection. Syntax: {"field1":"value1", "field2":"value2" } Example: `{"department":"IT"}`
Projection	Specify the fields to be returned in the response. Example: `{"field1": 1, "field2": 0}`. Use 1 to include a field, 0 to remove implicitly included fields, and fields not mentioned are excluded from the response.
Sort By	Specify the fields based on which sorting is need to be performed. You can specify one or more fields and set the sort order for each field. Example: `{"field1": 1, "field2": -1}`, in this example, sorting is done first by the field1 in ascending order (1) and then by the field2 in descending order (-1).
Update Fields	Specify the operator for the fields to be modified along with their values. Syntax: {"operator":{"field1":"value"}} Example: `{"$set":{"Salary":"78000"}}`
Array Fields	Specify one or more array filter documents to identify which elements inside an array should be updated during the operation. Array filters are used with the $[<identifier>] syntax in your update statement and allow updates to only those array elements that match the specified conditions. Syntax: {"Identifier":{"operator":"value"}} Example: `{"element":{"$gte":100}}`
Enable Transaction	Enable to execute the operation as a transaction. If enabled, either all the operations in the transaction are successfully applied, or none.

Parameter	Description
Order Execution	Enable to stop execution on the first failure and return without processing the remaining documents. If disabled, continue processing despite failures and complete execution.
Multi Update	Enable to update all documents that match with the filter criteria. When disabled, only the first matching document will be updated.
Multi Delete	Enable to delete all documents that match with the filter criteria. When disabled, only the first matching document will be deleted.
Upsert	Enable to insert a new document if there is no document that matches the query. If a match is found, the document(s) will be updated.
Replace Document	Specify the document to be inserted or replaced. Example: `Order_Completed`
Return Document	Select to specify if the document must be returned before or after the modification: • **Before:** Before any modification is made to the document. • **After:** After any modification is made to the document.
Query Options	Enable to display more query filter options.
Document Count	Enable to return the number of documents matching the filter query.
Estimated Document Count	Enable to return the approximate count of documents in the collection.
Batch Size	Specify the maximum number of documents that must be processed in each batch of a query result. Example: `5`
Number of Documents to Skip	Specify the number of documents to be skipped. Example: `10`
Limit	Specify the maximum number of documents to be returned. Example: `4`
Max Time (in ms)	Specify the maximum amount of time (in milliseconds) for the query to be completed. Example: `30000`

Parameter	Description
Command	Specify the command to execute an operation. Syntax: {"operation to be performed": "name_of_collection","name_of_document": [{"field1": "value1","field2": "value2"}]} Example: `{"insert":"mycollection", "name_of_document":[{"field1":"value1", "field2": value2}]}`

5 MongoDB Operations

This section lists and describes some of the operations supported by the MongoDB adapter.

5.1 Document: Insert

The **Insert** operation is used to add one or more documents to a MongoDB collection. Each inserted document must be in **JSON** format and will automatically get a unique object ID field if not provided.

In the scenario below, we will insert the document into our collections.

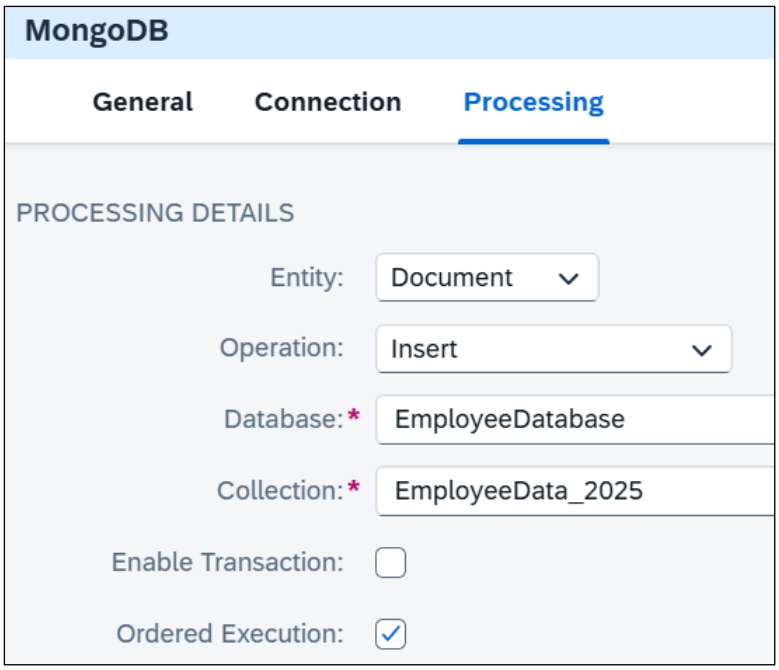
Sample Payload for **Single Insert**:

```
{
  "name": "user1",
  "email":
"user1@example.com"
}
```

Sample Payload for **Multiple Insert**:

```
[
  {
    "name": "user1",
    "email":
"user1@example.com"
  },
  {
    "name": "user2",
    "email":
"user2@example.com"
  }
]
```

A sample screenshot has been given below.



For the complete list of descriptions about the fields below, refer to the Processing tab.

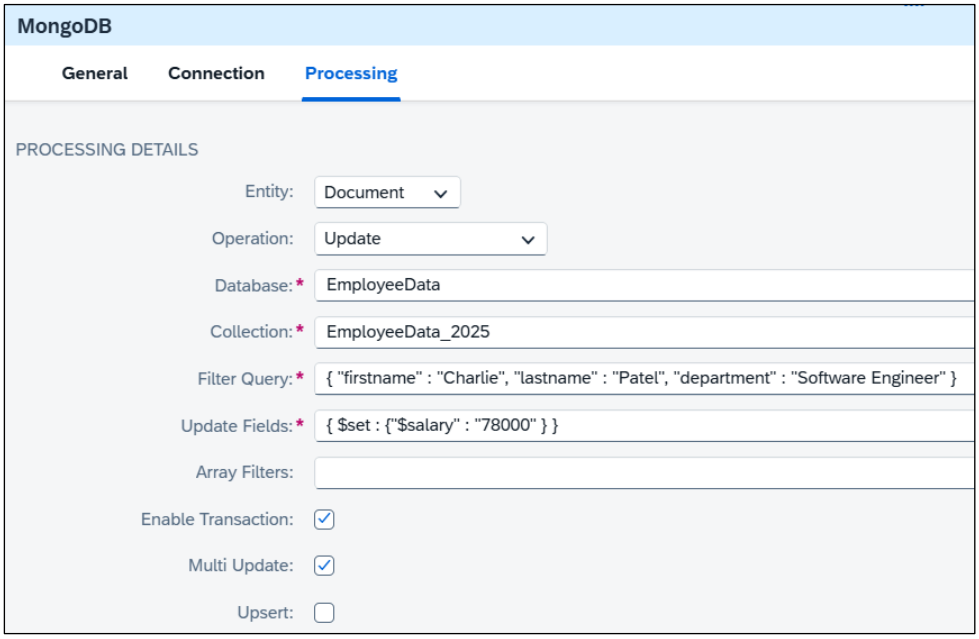
Parameter	Value
Entity	Select as **Document**
Operation	Select as **Insert**
Database	Specify the name of the Database. You will be able to find it inside your Cluster.
Collection	Specify the name of the Collection. You will be able to find it inside your Cluster.
Enable Transaction	If enabled, either all the operations in the transaction are successfully applied, or none.
Ordered Execution	Enable to stop execution on the first failure and return without processing the remaining documents. If disabled, continue processing despite failures and complete execution.

5.2 Document: Update

The **Update** operation in MongoDB is used to **modify existing documents** in a collection. It allows you to change the value of one or more fields, add new fields, or even replace the whole document **without deleting and reinserting** it.

In this scenario, we are updating an existing field **salary** for an employee.

A sample screenshot has been given below.



For the complete list of descriptions about the fields below, refer to the Processing tab.

Parameter	Value
Entity	Select as **Document**
Operation	Select as **Update**
Database	Specify the name of the Database. You will be able to find it inside your Cluster.
Collection	Specify your Collection name. You will be able to find it inside your Cluster.
Filter Query	Specify the filter expression for the query. Syntax: `{"field1":"value1", "field2" :"value2", "field3":"value3"}`

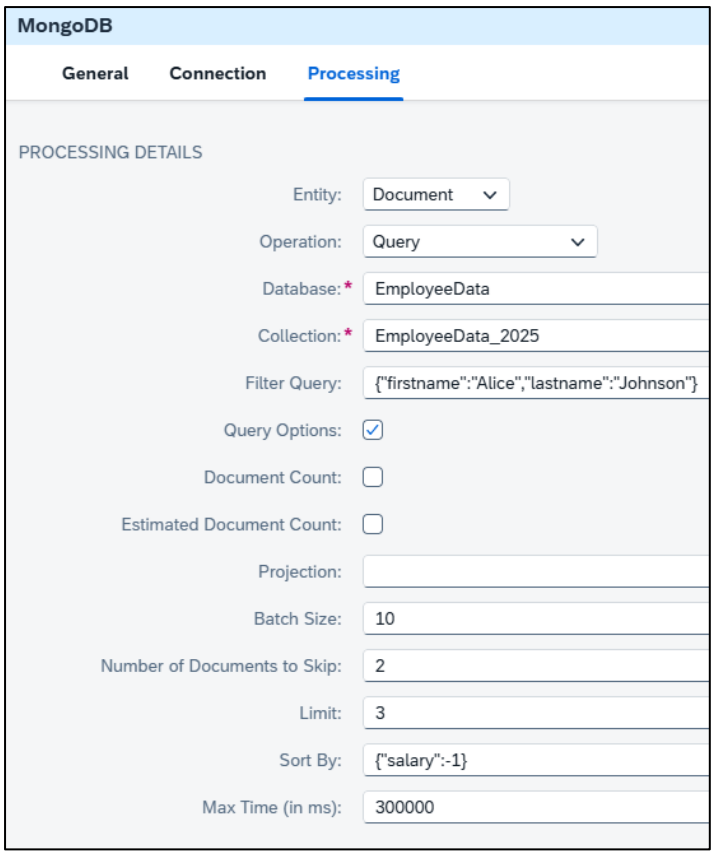
Parameter	Value
Update Field	Specify the criteria to update fields. Syntax: {"operator":{"field1":"value"}} Example: `{$set:{"$salary":"78000"}}`
Enable Transaction	If enabled, either all the operations in the transaction are successfully applied, or none.
Multi Update	Enable the checkbox to update all matching documents. If disabled, only the first matching document is updated.
Upsert	Disable the checkbox to stop inserting a new document when no match is found. If enabled, a new document will be inserted when no match is found.

5.3 Document: Query (Multiple Records)

A **Query** operation allows you to retrieve data from a collection by specifying a query filter that matches the documents you want. For more information regarding Query, see MongoDB Query.

In the Scenario below, we will make a query to filter the required data from the existing documents.

A sample screenshot has been given below.



For the complete list of values, refer to the Processing tab.

Parameter	Value
Entity	Select as **Document**
Operation	Select as **Query**
Database	Specify the name of the Database. ℹ You will be able to find it inside your Cluster.
Collection	Specify the name of the Collection. ℹ You will be able to find it inside your Cluster.
Filter Query	Specify the expression to modify the matching documents from the collection. Syntax: {"field1":"value1", "field2":"value2" } Specify as `{"firstName":"Alice", "lastname":"Johnson"}`
Query Options	Enable the checkbox to add additional query filters.

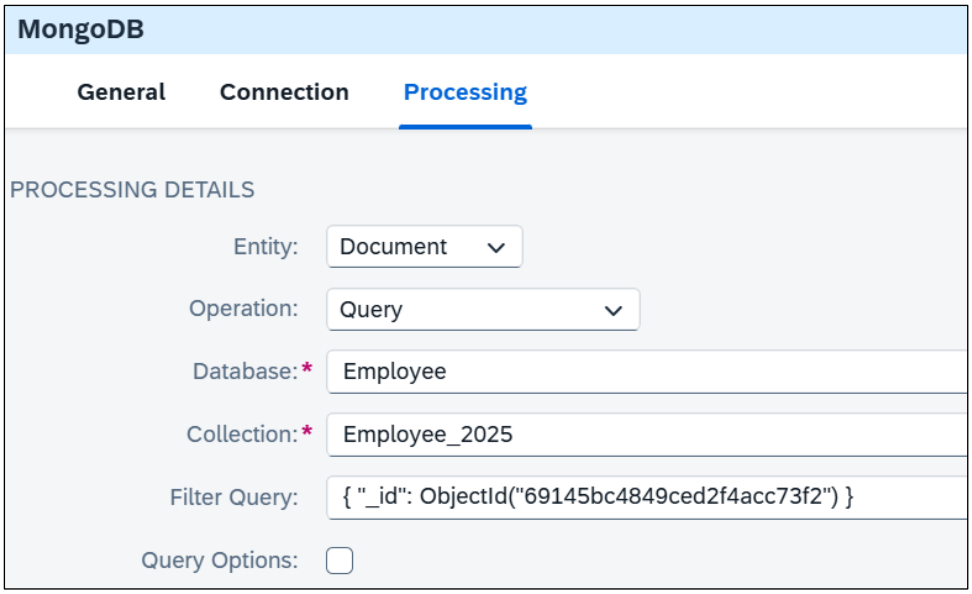
Parameter	Value
Document Count	Disable the checkbox so that the document count is not retrieved. If enabled, the documents count is retrieved.
Estimated Document Count	Disable the checkbox so that the approximate count of documents is not retrieved. If enabled, it will return an approximate document count.
Batch Size	Set value to `10` to receive the document in batches.
Number of documents to Skip	Set value to `2` to skip the documents in your collections.
Limit	Set the value to `3` to limit the number of received documents.
Sort by	Specify as `{"salary": -1}` to sort. Ascending order (1) or descending order (-1).
Max Time (in ms)	Set as `300000`

5.4 Document: Query (Single Record)

A **Query** operation allows you to retrieve data from a collection by specifying a query filter that matches the documents you want. For more information regarding Query, see MongoDB Query.

In the Scenario below, we will make a query to fetch data using the ID.

A sample screenshot has been given below.



For the complete list of values, refer to the Processing tab.

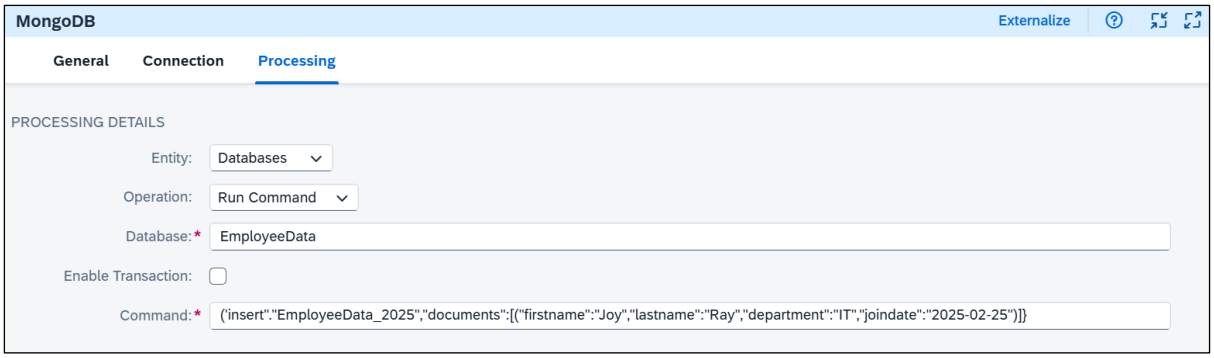
Parameter	Value
Entity	Select as **Document**
Operation	Select as **Query**
Database	Specify the name of the Database. You will be able to find it inside your Cluster.
Collection	Specify the name of the Collection. You will be able to find it inside your Cluster.
Filter Query	Specify as `{"firstName":"Alice", "lastname":"Johnson"}`

5.5 Databases: Run Command

The **Run Command** method enables you to execute raw database commands directly on the MongoDB server, offering enhanced control and flexibility over database operations. For more information, refer to the Run Commands.

In the scenario below, we will perform the Run Command operation to insert the document into our database.

A sample screenshot has been given below.



For the complete list of values, refer to the Processing tab.

Parameter	Value
Entity	Set as **Databases** to work on the Databases entity.
Operation	Set as **Run Command**, to interactively run database commands, switch databases, create new databases and collections, and manage data.
Database	Specify the name of the Database. You will be able to find it inside your Cluster.
Enable Transaction	If enabled, either all the operations in the transaction are successfully applied, or none.
Command	Specify as `{"insert":"EmployeeData_2025","documents":[{"firstname":"Joy","lastname":"Ray","department":"IT","joindate":"2025-02-25"}]}` You must enter a JSON value in the field.

5.6 Databases: Run Command (Bulk Write)

The **Run Command** method lets you execute raw database commands directly on the MongoDB server, providing greater control and flexibility over database operations. For more information, refer to Run Commands.

In the scenario below, we will perform the Run Command operation to bulk write the documents to our database.

A sample payload has been given below.

```
{
    "bulkWrite": 1,
```

```
    "nsInfo": [
        {
            "ns": "<enteryourdatabase>.<enteryourcollection>"
        }
    ],
    "ops": [
        {
            "insert": 0,
            "document": {
                "_id": 101,
                "name": "Liam",
                "department": "Engineering",
                "role": "Developer"
            }
        },
        {
            "insert": 0,
            "document": {
                "_id": 102,
                "name": "Sophia",
                "department": "Finance",
                "role": "Analyst"
            }
        },
        {
            "insert": 0,
            "document": {
                "_id": 103,
                "name": "Noah",
                "department": "Marketing",
                "role": "Coordinator"
            }
        },
        {
            "insert": 0,
            "document": {
                "_id": 104,
                "name": "Emma",
                "department": "Human Resources",
                "role": "HR Specialist"
            }
        }
    ]
}
```

A sample screenshot has been given below.



For the complete list of values, refer to the Processing tab.

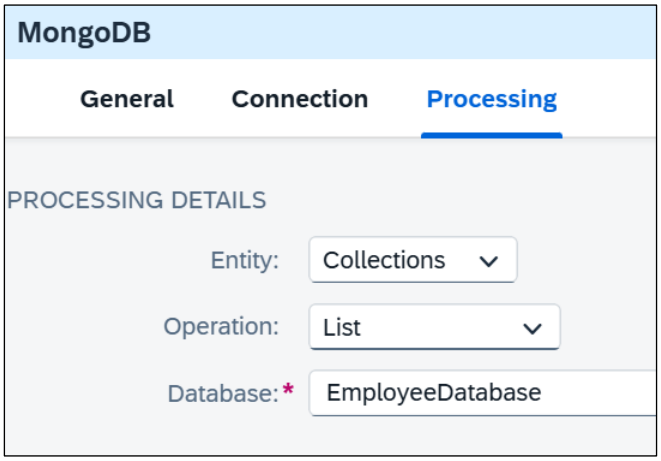
Parameter	Value
Entity	Set as **Databases** to work on the Databases entity.
Operation	Set as **Run Command**, to interactively run database commands, switch databases, create new databases and collections, and manage data.
Database	Specify the name of the Database. Bulk Write may only be run against the "**admin**" database.
Enable Transaction	If enabled in this scenario, on any failure, all the changes made during the transaction will not be reflected.
Command	Specify as `{"bulkWrite":1,"nsInfo":[{"ns":"<enteryourdatabase>.<enteryourcollection>"}],"ops":[{"insert":0,"document":{"_id":101,"name":"Liam","department":"Engineering","role":"Developer"}},{"insert":0,"document":{"_id":102,"name":"Sophia","department":"Finance","role":"Analyst"}},{"insert":0,"document":{"_id":103,"name":"Noah","department":"Marketing","role":"Coordinator"}},{"insert":0,"document":{"_id":104,"name":"Emma","department":"Human Resources","role":"HR Specialist"}}]}` You must enter a JSON value in the command field.

5.7 Collection: List

The **List** operation allows you to fetch documents from your Collections.

In the scenario below, we will fetch all the collections from the mentioned database.

A sample screenshot has been given below.



For the complete list of values, refer to the Processing tab.

Parameter	Value
Entity	Set as **Collections**.
Operation	Set as **List**.
Database	Specify the name of the Database. You will be able to find it inside your Cluster.

6 Support

6.1 Tips

You can monitor, debug, and analyze errors or issues by changing the Log level of your integration flow to Traces. For more information, see Tracing.

6.1.1 Troubleshooting

A few issues that you might encounter and possible solutions for them have been documented below.

Error Message	Error Reason	Possible Solution
`org. apache. camel.CamelExceptio n: Timed out while waiting for a server that matches WritableServerSelec tor. Client view of cluster state is {type=REPLICA_SET, servers=[{address=x x-xxxxxxx-shard-00- 01.xxxxxxx.mongodb. net:27017, type=UNKNOWN, state=CONNECTING, exception= com.mongodb.MongoSo cketWriteException: Exception sending message}, caused by {javax. net.ssl. SSLException: Received fatal alert: internal_error}}, {address=xx- xxxxxxx-shard-00- 02.xxxxxxx.mongodb. net: 27017, type=UNKNOWN, state=CONNECTING, exception={com. mongodb.MongoSocket WriteException: Exception sending message}, caused by {javax.net.ssl. SSLException:`	The client IP addresses are not whitelisted in MongoDB's network access settings, causing the SSL handshake to fail during connection.	To resolve this: • Log in to **MongoDB Atlas**. • Navigate to **Network Access**. • Open the **IP Access List** tab. • Add the **SAP tenant's IP Range** to the list of allowed IPs.

Error Message	Error Reason	Possible Solution
`Received fatal alert: internal_error}}, {address=xx-xxxxxxx-shard-00-02.xxxxxxx.mongodb.net: 27017, type=UNKNOWN, state=CONNECTING, exception= {com.mongodb.MongoSocketWriteException: Exception sending message}, caused by {javax.net.ssl.SSLException: Received fatal alert: internal_error}}]`		
`org. apache. camel. CamelException: Error message: Failed to parse input JSON, Reason: Unexpected character ('c' (code 99)): was expecting double-quote to start field name at [Source: REDACTED ('StreamReadFeature. INCLUDE_SOURCE_IN_LOCATION disabled)`	In the Run Command operation, the command is not provided in JSON format.	Ensure that the command is entered in a valid JSON format.
`org. apache. camel. CamelException: Error message: MongoDB Run Command Operation failed, Reason: Command failed with error 13 (Unauthorized): 'bulkWrite may only be run against the admin database.' on server cluster0-shard-00-02.890bl.mongodb.net:27017. The full response is {"ok": 0.0,`	The Run Command operation fails when the provided database is not `admin`.	Set the database name to `admin` to execute the command successfully.

Error Message	Error Reason	Possible Solution
<pre>"errmsg": "bulkWrite may only be run against the admin database.", "code": 13, "codeName": "Unauthorized", "$clusterTime": {"clusterTime": {"$timestamp": {"t": 1762436383, "i": 1}}, "signature": {"hash": {"$binary": {"base64": "ihWxg7GipwYabSfOcg/rFZUCf1U=", "subType": "00"}}, "keyId": 7512504026747895810 }}, "operationTime": {"$timestamp": {"t": 1762436383, "i": 1}}}</pre>		
<pre>{"failedCount":1,"successfulCount":0,"results":[{"id":"<ID>","message":"Timed out while waiting for a server that matches WritableServerSelector. Client view of cluster state is {type=UNKNOWN, servers=[{address=<server-address-1>, type=UNKNOWN, state=CONNECTING, exception={MongoSocketReadException: Prematurely reached end of stream}}, {address=<server-address-2>, type=UNKNOWN, state=CONNECTING, exception={MongoSocketReadException: Prematurely reached end of stream}}, {address=<server-</pre>	TLS/SSL settings are misconfigured. The client attempted a non-secure connection to a server that requires TLS/SSL.	• Specify the **Connection Option** as `ssl=true` • Verify host and port.

Error Message	Error Reason	Possible Solution
address-3>, type=UNKNOWN, state=CONNECTING, exception={MongoSocketReadException: Prematurely reached end of stream}}]}","status":"failed"}]}		