



**TGH**

Making Integrations Simpler



# INTEGRATION WITH ATOM IN BOOMI



## Contents

<b>INTEGRATION WITH ATOM IN BOOMI .....</b>	<b>2</b>
<b>What are Dead Letter Queues? .....</b>	<b>25</b>
<b>Pros and Cons of Atom Queue.....</b>	<b>30</b>

# INTEGRATION WITH ATOM IN BOOMI

In this Blog let us see

- How to integrate Atom Queue in Boomi.
- What are Dead Letter Queues
- Pros and Cons of Atom Queue

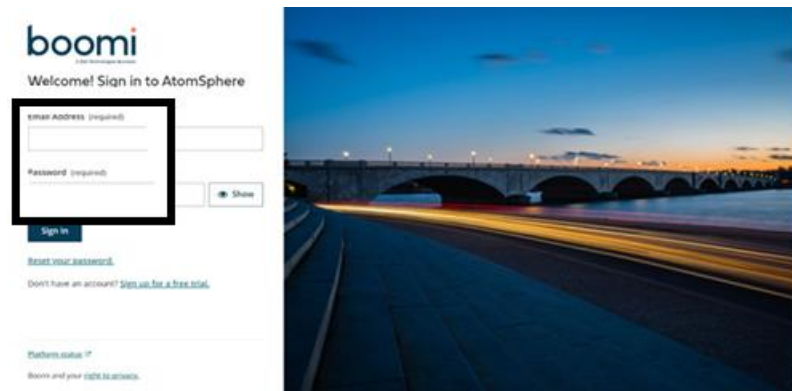
## What is Atom Queue?

Atom supports message queuing which means messages are managed by a shared server queue embedded in the atom. Each queue component specifies the configuration of a message queue, including its name and the messaging model with which the message queue can be used. Deploying an Atom Queue connector does not affect your license count.

In this Use Case, we have 2 processes where process 1 reads a file from Disk Connector and sends it to the Atom Queue. Process 2 will read a message from Atom Queue and send it to the ORACLE database.

## Let us begin with the steps.

**Step 1:** Log on to the Boomi platform (<https://platform.boomi.com/>) with the required credentials i.e. Email Address and Password.

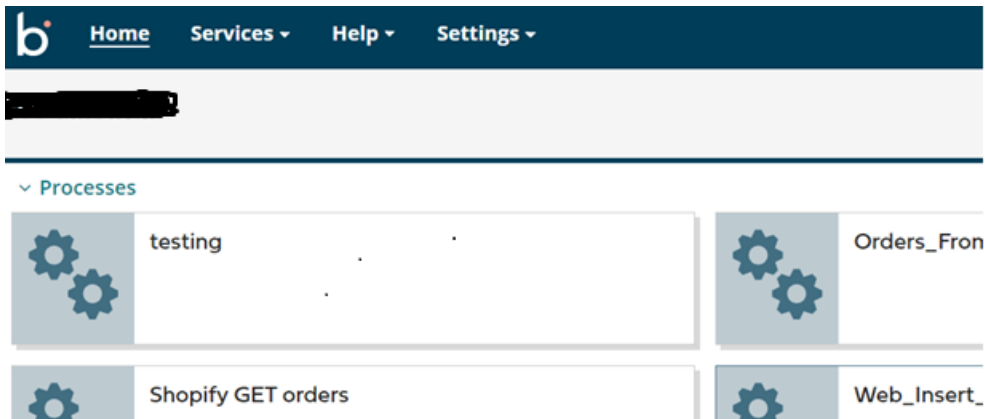


©[TGH Software Solutions Pvt. Ltd.](http://TGH Software Solutions Pvt. Ltd.)

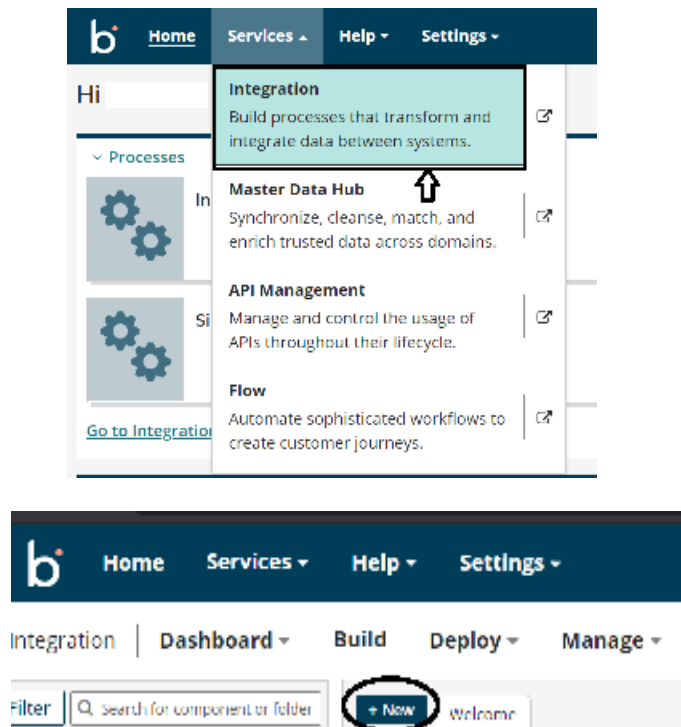
No part of this document may be copied, reproduced, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted or distributed in any way to any other computer, server, website or other medium for publication or distribution, without TGH's prior written consent



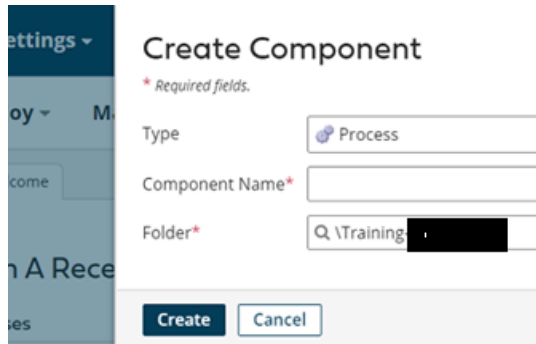
**Step 2:** Once logged into the Boomi platform, we will be able to view the Home page.



**Step 3:** Now, click on Services followed by Integration. We will see the Build page. Click on New.

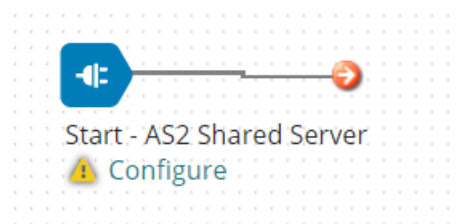


**Step 4:** Once, click on New, we will be able to see three fields i.e. Type, Component Name and Folder.

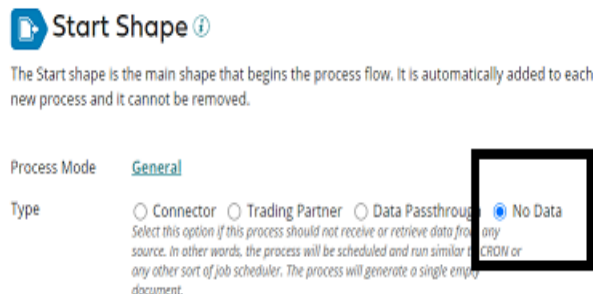


- Select Type as process as we are building a process. Component Name and Folder can be given based on your choice (i.e. which name to be given and where do we want to create the process). Click on Create.

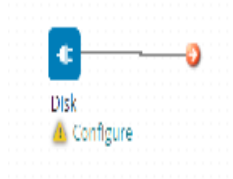
**Step 5:** We see that the process gets created with a start shape which is configured with AS2 Shared Server by default.



**Step 6:** Select the start shape and choose No Data. Click ok.



**Step 7:** Drag and drop the disk connector shape onto the process canvas to read a file from a specific directory.



- We have to configure 3 fields in connector i.e. Action, Connector and Operation.

**Connector Shape** ⓘ

Connector shapes are used to get data into and send data out of a process. Most processes have one "get" connector and one or more "send" connectors. The Connector shape uses a combination of predefined connection and operation components to establish where and how to get or send data.

---

**General** Parameters

Display Name

Connector ⓘ

Action

Connection ⓘ

Operation ⓘ

- We see 2 actions i.e. Get and Send in Actions.
  - **Get** – To get the data from a disk location.
  - **Send** – To send to the disk location.
- Here, we will choose action as **GET** as we are reading the file.

**General** Parameters

Display Name

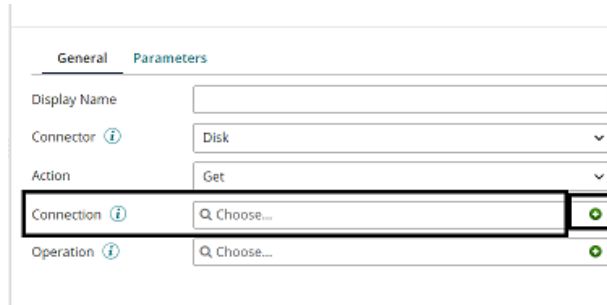
Connector ⓘ

Action

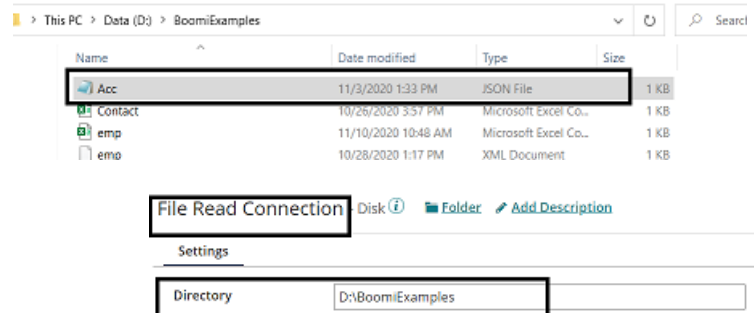
Connection ⓘ

Operation ⓘ

- Click + on connection to create a new one.



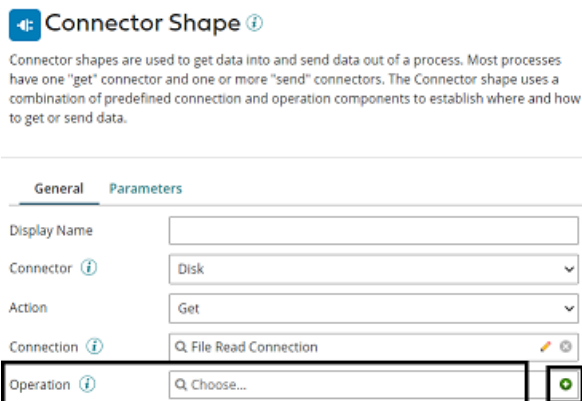
- Name the file and give the directory from where we want to read the file. Here, we are reading JSON files from D drive and BoomiExamples folder as shown in the screenshot.



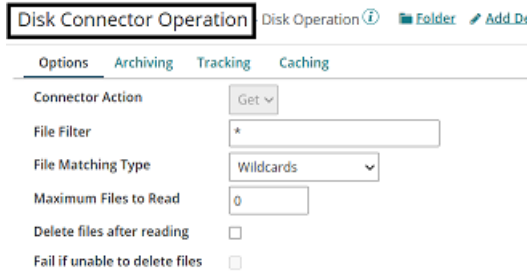
- Click save and close
- Now, we will configure the operation. Click + on operation to create a new one.

**Connector Shape**

Connector shapes are used to get data into and send data out of a process. Most processes have one "get" connector and one or more "send" connectors. The Connector shape uses a combination of predefined connection and operation components to establish where and how to get or send data.



- Name the operation and configure the following.



Disk Connector Operation Disk Operation ⓘ Folder Add Des

Options Archiving Tracking Caching

Connector Action Get ▾

File Filter \*

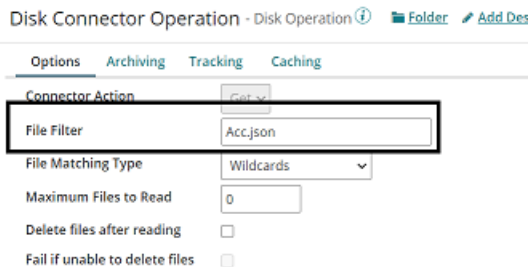
File Matching Type Wildcards ▾

Maximum Files to Read 0

Delete files after reading

Fail if unable to delete files

- **File Filter:** Read-only files with a file name that matches the file filter. Here, it will be **Acc.json**



Disk Connector Operation - Disk Operation ⓘ Folder Add Des

Options Archiving Tracking Caching

Connector Action Get ▾

File Filter Acc.json

File Matching Type Wildcards ▾

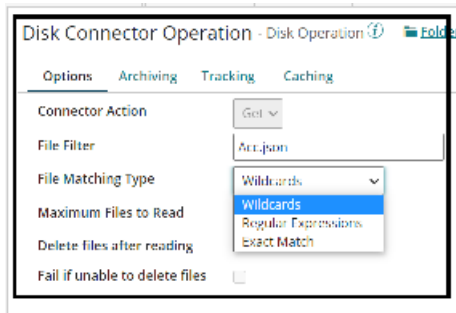
Maximum Files to Read 0

Delete files after reading

Fail if unable to delete files

### File Matching Type:

- **Wildcards** uses simple file filters like \* and. \* represent multiple characters and ? represents a single character.
- **Regular Expressions** can include complex regular expressions.
- **Exact Match** includes the filename that we are reading.



Disk Connector Operation - Disk Operation ⓘ Folder Add Des

Options Archiving Tracking Caching

Connector Action Get ▾

File Filter Acc.json

File Matching Type Wildcards ▾

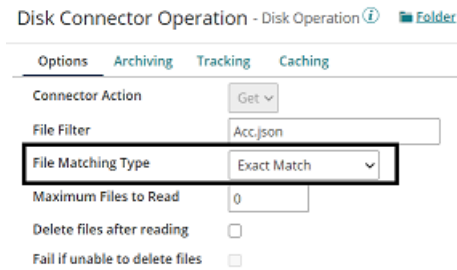
Maximum Files to Read 0

Delete files after reading

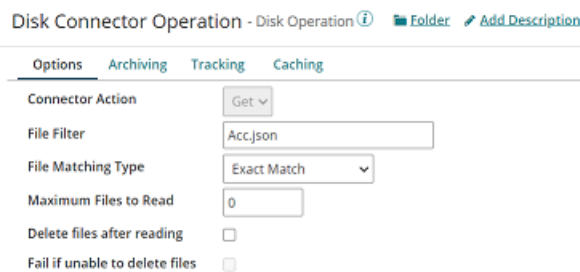
Fail if unable to delete files



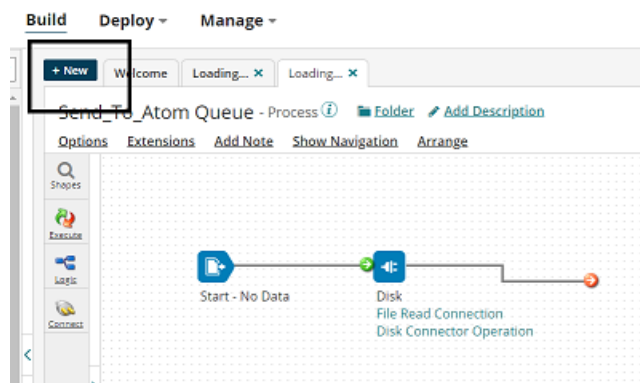
- Here, the file matching type would be **Exact Match** as we are giving the file name.



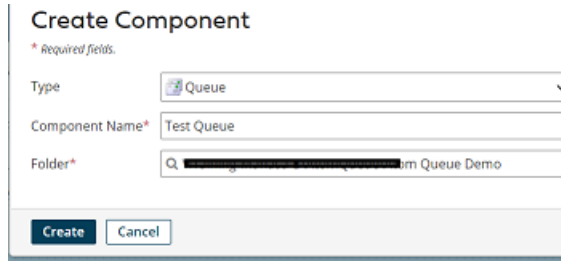
- **Maximum files to read:** It sets the maximum number of files to be read at one time. Let it be default i.e..
- **Delete files after reading:** If we want the file to be read and deleted, we can check this option. Here, we are leaving it to default. Click save and close
- The complete disk operation looks like this,



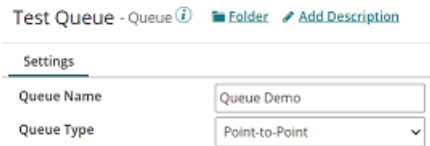
**Step 8:** First, we need to create a queue to store the message. Click on New as shown in the screenshot.



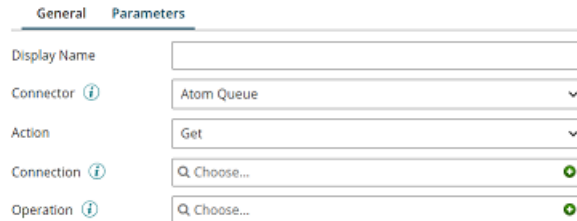
**Step 9:** Select Type as Queue as we are creating a queue to store the message. The component name will be Test Queue and save it in the Folder where you are creating the process. Click on Create.



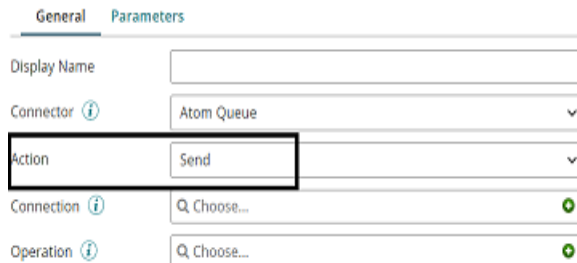
- Name the queue as Queue Demo. Click save and close.



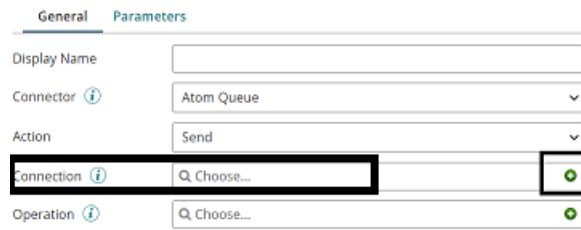
**Step 10:** Drag and drop Atom Queue connector onto the process canvas and we have to configure Action, Connection and Operation.



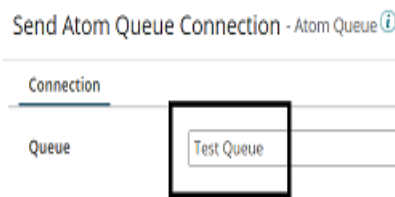
- We have 2 actions i.e. **GET** and **SEND**. Here, Action will be **Send** as we are sending message to the atom queue.



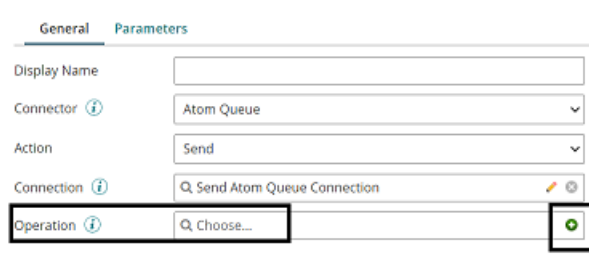
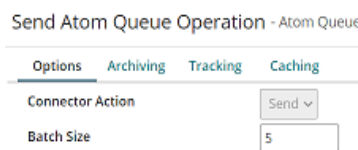
**Step 11:** Click + on connection and name it accordingly.



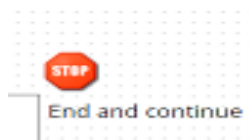
- Choose the atom queue which we have created. Here, it will be Test Queue. Click save and close.



**Step 12:** Click + on Operation and name it. Leave all the fields with default options. Click save and close.

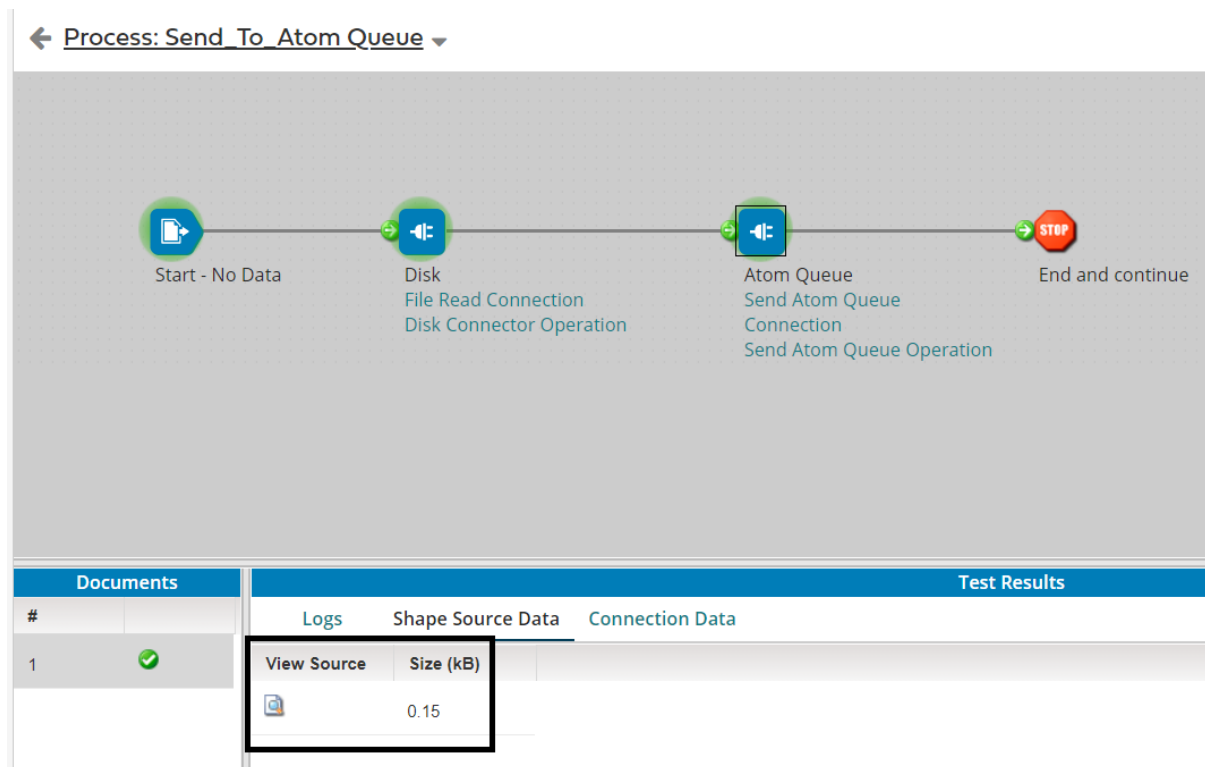
**Step 13:** Drag and drop stop shape on to the process canvas to indicate the end of the flow.



**Step 14:** Arrange all the shapes in an order and test it by configuring the local atom.



**Step 15:** We see that the process has been executed. Click on view source to see the output.



**Step 16:** Once, we open the document, this is how file looks like

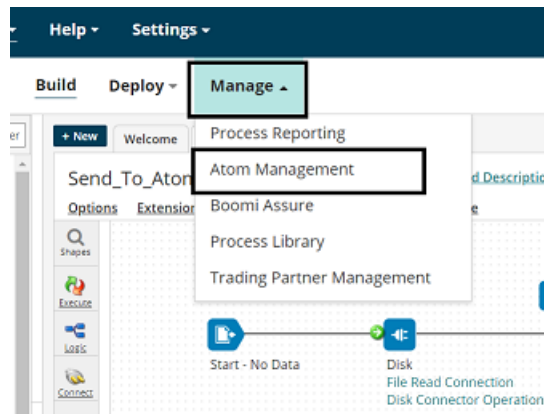
```

Document Viewer

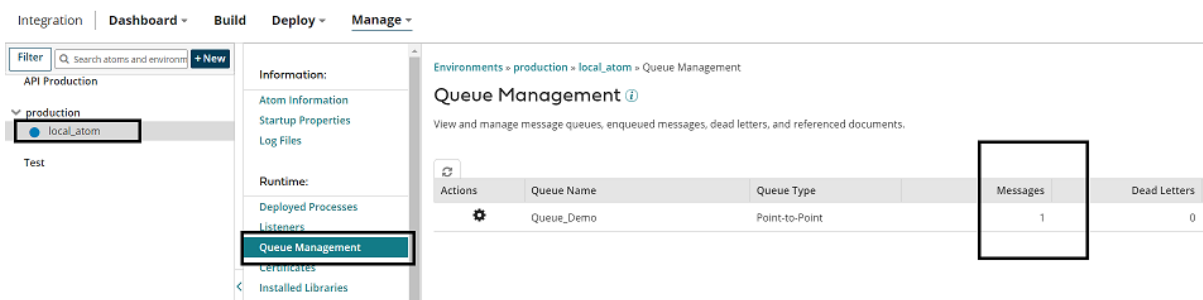
1 [
2 {
3   "Account_Id": 101,
4   "Account Name": "Mark",
5   "Branch": "Bangalore"
6 },
7 {
8   "Account_Id": 201,
9   "Account Name": "Sam",
10  "Branch": "Hyderabad"
11 }
12 ]

```

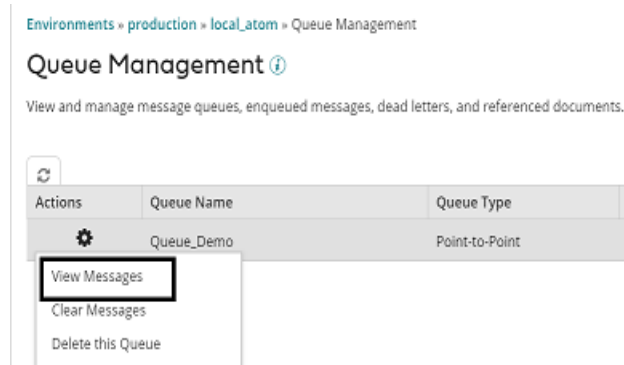
**Step 17:** To see the queue, Go to Manage > Atom Management and Click on the atom which we have configured.



**Step 18:** Next, go to Queue Management and we see a message existing in the queue.



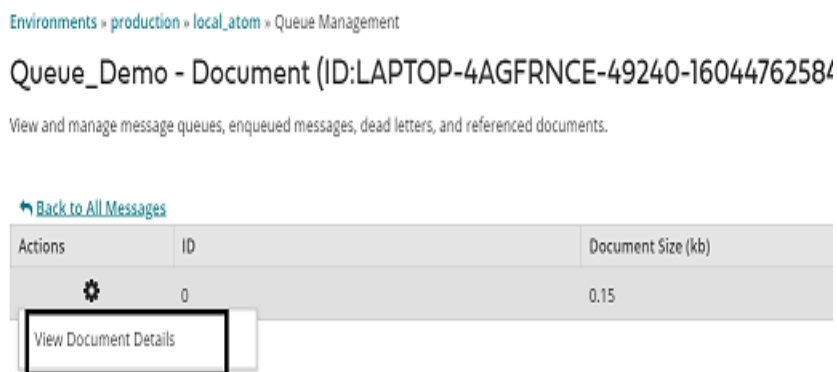
- Click on view messages as shown below.



- Click on Number of Documents (i.e. 1).



- We see the document size and click on view document details to see the message existing in Atom Queue.



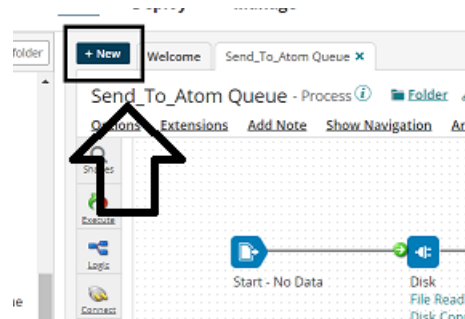
- We see that the Json file reading from disk has been sent to Atom queue.

```

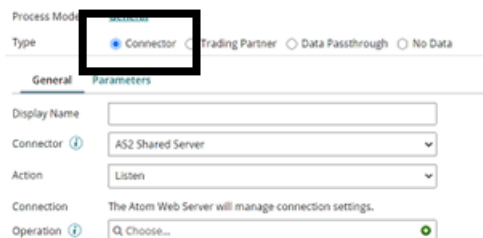
Document Viewer
Raw Formatted Document
[
  {
    "Account_Id": 101,
    "Account Name": "Mark",
    "Branch": "Banglore"
  },
  {
    "Account_Id": 201,
    "Account Name": "Sam",
    "Branch": "Hyderabad"
  }
]

```

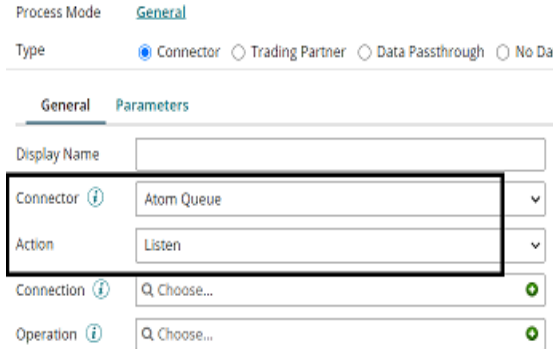
**Step 19:** Now, we will create other process which reads message from the atom queue and send it to the database. Click on New as shown and follow steps 4 and 5 which are mentioned above.



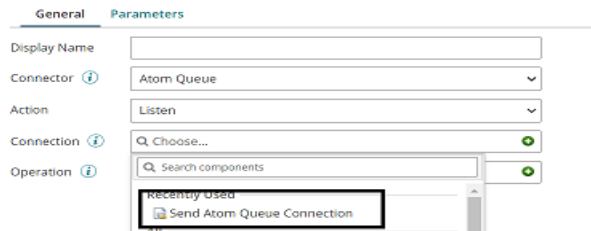
**Step 20:** Configure start shape with connector type as shown.



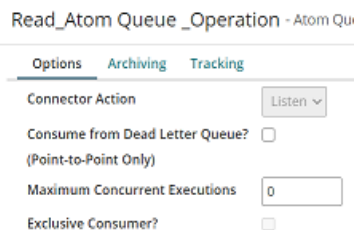
**Step 21:** Choose connector as Atom Queue. We have two Actions as GET and LISTEN. We will set it to listen as it will listen to the messages which comes into the queue and trigger the process automatically and reads the message.



**Step 22:** Do not click on + and choose the same connection which we have configured in 1<sup>st</sup> process (i.e. Send Atom Queue Connection)



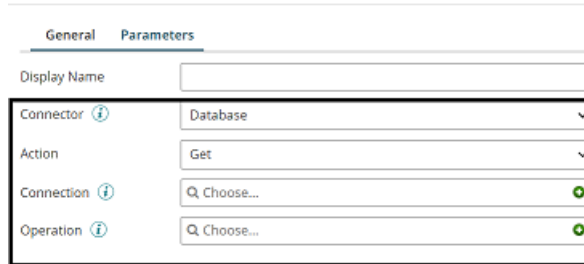
**Step 23:** Click + on Operation and name it as Read\_Atom Queue \_Operation. Leave all the options to default. Click save and close.





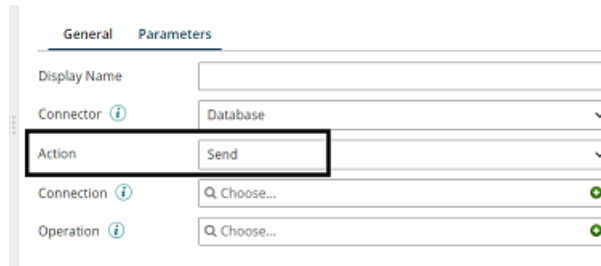
**Assuming that we have created a table named as Accounts in Oracle Database with 3 fields named as Account Id, Account Name and branch, will follow the below steps.**

**Step 24:** We will configure the database connector to place the message in database which we receive from atom queue. Here, we are using ORACLE database. Drag and drop database connector onto process canvas.



General Parameters	
Display Name	<input type="text"/>
Connector ⓘ	Database ▼
Action	Get ▼
Connection ⓘ	🔍 Choose... 🟢
Operation ⓘ	🔍 Choose... 🟢

- Here, we have 2 actions i.e. Get and Send.  
 GET—To get the data from database.  
 SEND—To send the data to database.
- Action will be send as we are sending the data to database.



General Parameters	
Display Name	<input type="text"/>
Connector ⓘ	Database ▼
Action	Send ▼
Connection ⓘ	🔍 Choose... 🟢
Operation ⓘ	🔍 Choose... 🟢

**Step 25:** Click + on connection and name it accordingly. Configure the following details as shown.

- **Driver Type:** Select the required Database from Dropdown. Choose Oracle as we are integrating with Oracle Database.
- **User Name and Password:** Give the Database user name and password.
- **Host:** Give the name or IP Address of the Database.
- **Port:** It is used to connect to Database Server. Default port for Oracle is 1521.
- **Database Name:** Give the Database name of the server.

Oracle\_DB\_Connection - Database ⓘ Folder Add Description

Connection Advanced Options Connection Pool

### Database Connection

The authentication details and location of the hosted Database.

Database URL ⓘ jdbc:oracle:thin:@localhost:1521:XE

Driver Type ⓘ Oracle

Class Name ⓘ oracle.jdbc.driver.OracleDriver

User Name ⓘ

Password ⓘ <Encrypted>

Host ⓘ localhost

Port ⓘ 1521

Database Name ⓘ XE

Additional Options ⓘ

- Click save and close.

**Step 26:** Click + on operation and name it. Click on profile as shown below.

Send\_Database\_Operation Database Operation ⓘ

Options Archiving Tracking Caching

### Database Options

The operation represents a specific action to be performed on the data; example, in the operation you define whether to get or send information; database inserts, etc.

Connector Action Send

Profile ⓘ Choose...

Commit Options

Commit Option ⓘ Commit By Profile

Batch Count ⓘ 0

JDBC Batching  Enable JDBC Batching ⓘ

**Step 27:** Give it a name and select statement. Choose Type as Dynamic Insert and click on import.

Send Database Profile Database Profile ⓘ Folder Add Description Import

Data Elements Options

Statement

Fields

### Statement Details

Define statements used in the profile. The available statement settings are dependent on the Execution Type (Read or Write) selected in the Options tab and the statement Type selected in this tab.

Display Name Statement

Type Standard Insert / Update / Delete

Position Dynamic Insert

SQL Script ⓘ

Stored Procedure Write

Dynamic Update

Dynamic Delete

**Step 28:** Chose the atom and connection. Add the table name as ACCOUNTS in Object Filter and click Next.

### Database Import Wizard

The Import Wizard auto-generates a SQL statement, profile fields, and columns that you select.

\* Required fields.

Browse In:  local\_atom

Connection\*:

Schema Filter:

Object Filter:

- Choose a table and select all the fields. Click Next.

### Choose a Table

### Choose Columns

For the table selected on the previous screen, select one or more fields.

Fields for:

- ACCOUNT\_ID
- ACCOUNT\_NAME
- BRANCH

- We see a success message that profile has been imported. Click on finish.

### Success

You successfully imported the fields into your profile.

- We see that the fields are imported from the table. Click save, close and ok.

Send Database Profile - Database Profile ⓘ [Folder](#) [Add Description](#)

Data Elements Options

- Statement
- Fields
  - ACCOUNT\_ID
  - ACCOUNT\_NAME
  - BRANCH

### Statement Details

Define statements used in the profile. The available statement settings are dependent on the tab and the statement Type selected in this tab.

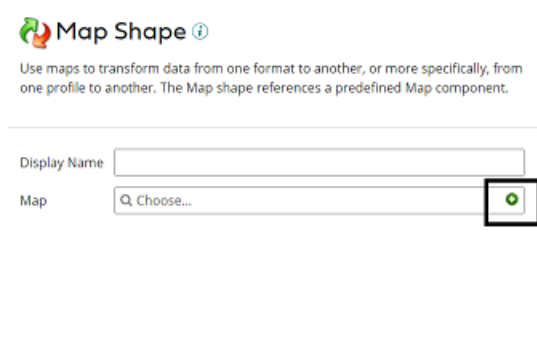
Display Name:

Type:

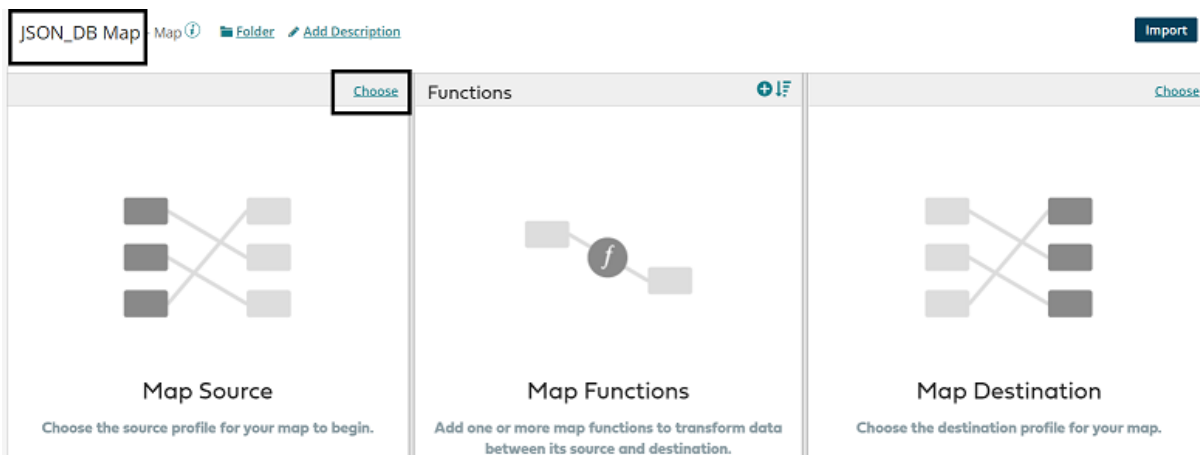
Position:

Table Name:

**Step 29:** Let us map Json profile to Database profile and send it to Database. Drag and drop Map shape onto the process canvas. Click + and Name the map.



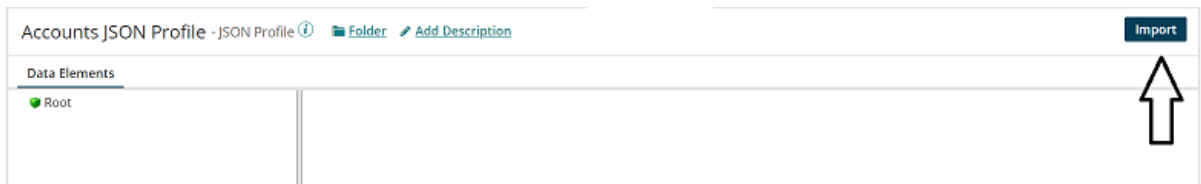
- Add source side and Destination side profiles to the map. Here, source side would be JSON profile and Destination side will be Database profile. First, we will add Source profile. Click choose and select the folder where we saved the profile.



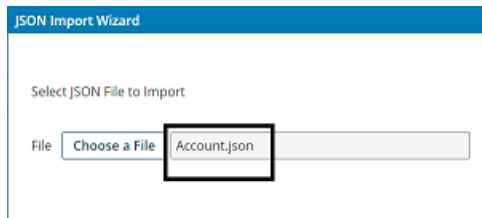
- Here, Profile Type will be JSON and click on Create New Profile.



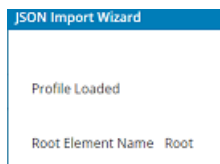
- Name the profile and click on import.



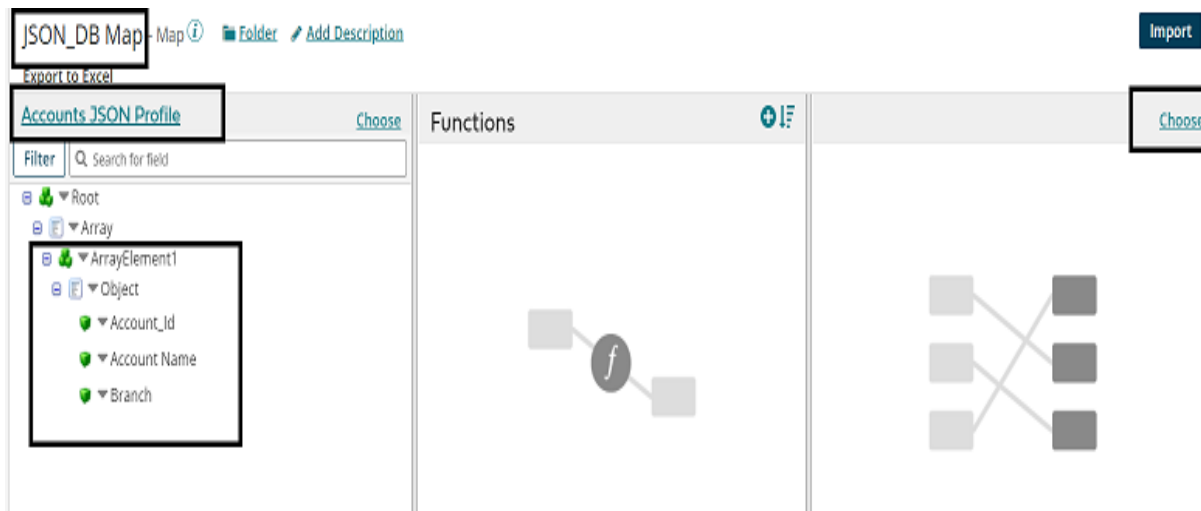
- Choose the file from the directory where we have saved the file. Click on Next.



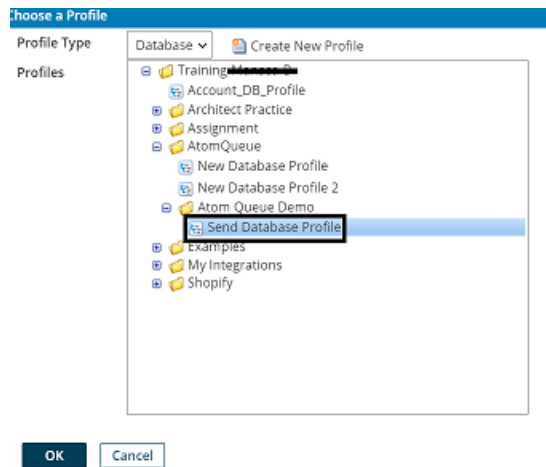
- We see that the profile is imported.



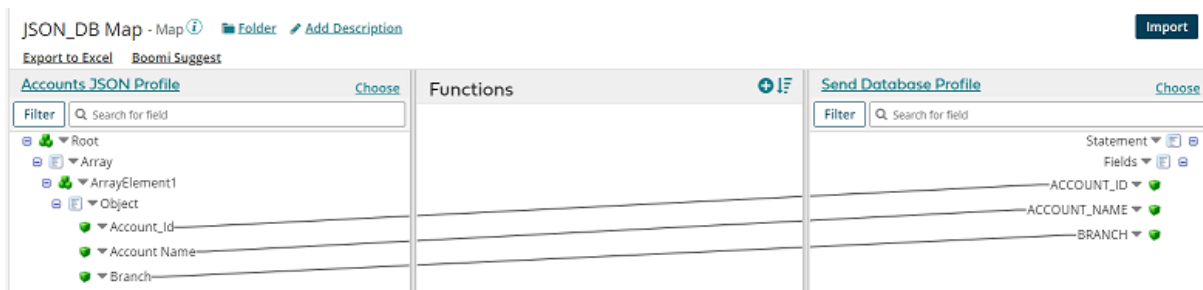
**Step 30:** Now, add the destination profile. Click choose on the right side as shown.



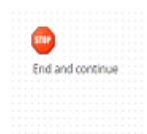
- Profile Type will be Database and select the Database profile which we have imported in the previous steps. Click ok.



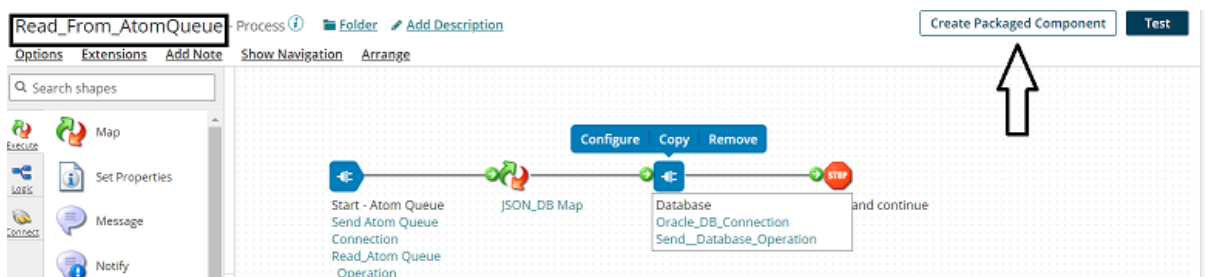
- We see that source and destination profiles have been imported. Provide one to one mapping from source to destination. Save and close.



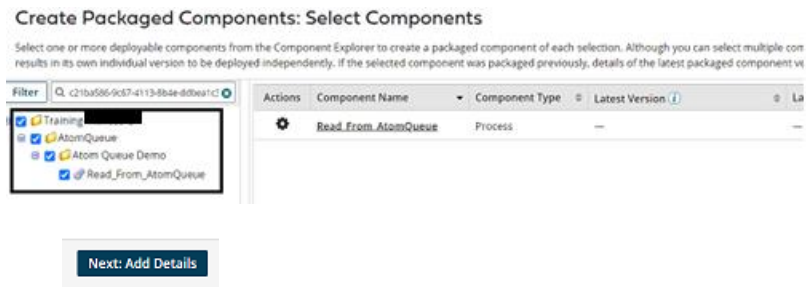
**Step 31:** Drag and drop stop shape onto the process canvas to indicate the end of the flow.



**Step 32:** Arrange all shapes in the order and deploy the process. Click on create packaged component as shown below.



- The process gets selected automatically. Click Add Details.



- Next, select the version and write notes if you have any. Click on create packaged component.

### Create Packaged Components: Add Details

Optionally apply details to the newest version of your packaged components. When you have multiple packaged components selected at one time, the details you specify for each selected component results in its own individual version.

Version for all

If you do not supply a name, a version number is automatically generated for each individual packaged component, and increments based on the latest version number.

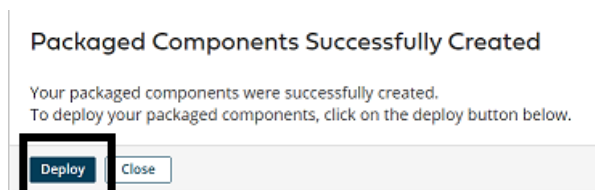
Package Notes for All

4000 characters remaining.

Sharing  Allow Processes and APIs to be publicly shared to subaccounts.  
This setting is ignored for components that cannot be shared.

### Create Packaged Component (1)

- Now, we see that the package has been created successfully and click on deploy.



- We will then have to select the environment. Choose production and click select version and review.

### Deploy: Select Environment

Select the environment in which to deploy your packaged component(s), and optionally add notes about the deployment.

Deployment Environment

Deployment Notes

4000 characters remaining.

### Deploy: Select Environment

Select the environment in which to deploy your packaged component(s), and optionally add notes about the deployment.

Deployment Environment

Deployment Notes  **choose production**

4000 characters remaining

[Next: Select Versions](#)

[Next: Review](#)

- We will be asked to cross check the environment which we have configured in deployment tab.

### Deploy: Review

You're almost done! Before deploying this version of your packaged component, confirm that the destination environment you have selected is correct.

Environment: production  
Deployment Notes:

Name	Type	Selected Version	Deployed Version
Read_From_AtomQueue	Process	1.0	N/A

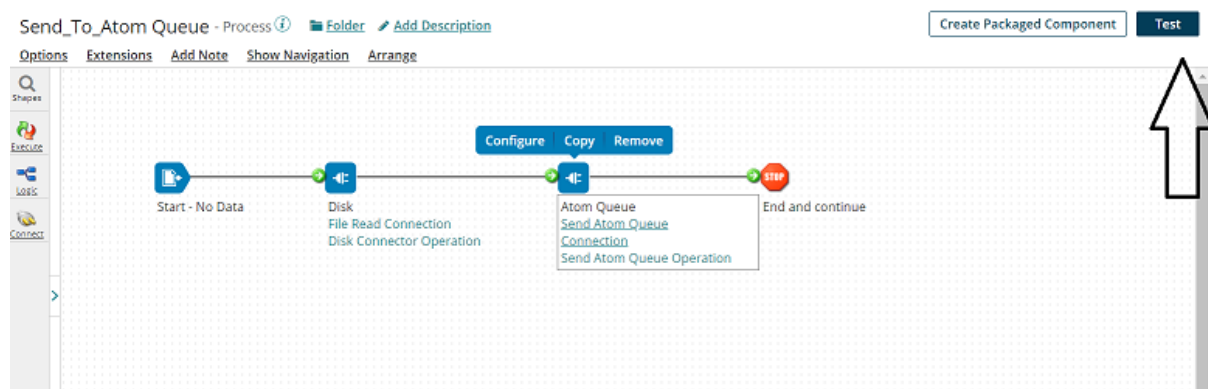
- Once, we click on deploy we will be able to see that deployment is done successfully.

### Deployment Successful

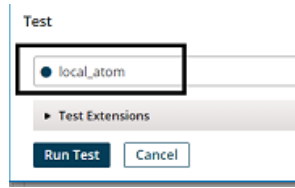
Your packaged components were successfully deployed.  
Click on the View Deployments button to see all deployments for this account.

[View Deployments](#) [Close](#)

**Step 33:** Now, we will test the first process and second process will automatically gets triggered and listens to the request.







**Step 34:** We see that the process has been executed. Click on view source to see the output.

← Process: Send\_To Atom Queue ▾

Documents		Test Results	
#		Logs	Shape Source Data
1	✓	View Source	Size (kB)
			0.15

**Step 35:** Go to Manage > Process Reporting and we see that the second process (i.e. Read\_From AtomQueue) has been executed automatically without errors.

Dashboard ▾ Build Deploy ▾ **Manage ▾**

- Process Reporting
- Atom Management
- Boomi Assure
- Process Library
- Trading Partner Management

---

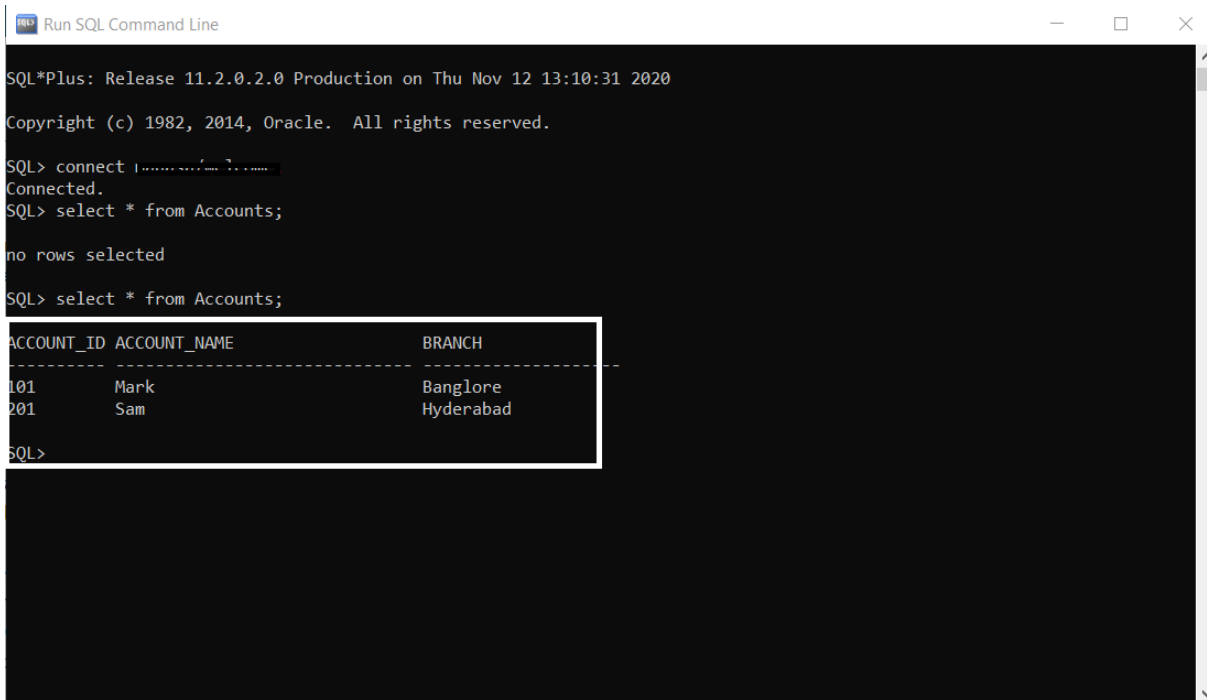
Integration | Dashboard ▾ Build Deploy ▾ **Manage ▾**

Executions ▾ Execute Process Auto Refresh is Off

Past Hour ▾ Add Filter Refresh Download All Errors Pending Successes

Time	Process	Actions	Atom	In	Out	Elapsed Time	Error Message
12 Nov 2020 13:06:35	Read_From AtomQueue		local_atom	1	1	0:00	

**Step 36:** We see that the message has been read from the queue and got inserted into database. If we look at the database table, we see 2 records have been inserted.



```
Run SQL Command Line
SQL*Plus: Release 11.2.0.2.0 Production on Thu Nov 12 13:10:31 2020
Copyright (c) 1982, 2014, Oracle. All rights reserved.
SQL> connect .....
Connected.
SQL> select * from Accounts;
no rows selected
SQL> select * from Accounts;
ACCOUNT_ID ACCOUNT_NAME          BRANCH
-----
101          Mark                          Bangalore
201          Sam                           Hyderabad
SQL>
```

**NOTE:** If something goes wrong in the above process, the failure messages will go into the Dead Letter Queue.

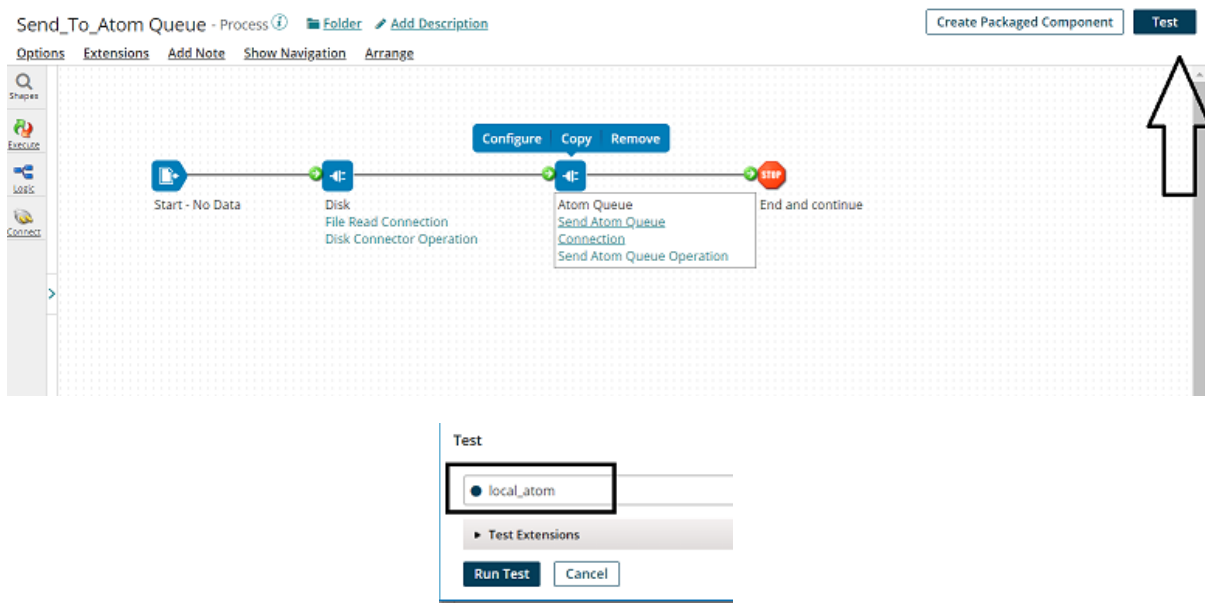
Now, let us see what are Dead Letter Queues.

## What are Dead Letter Queues?

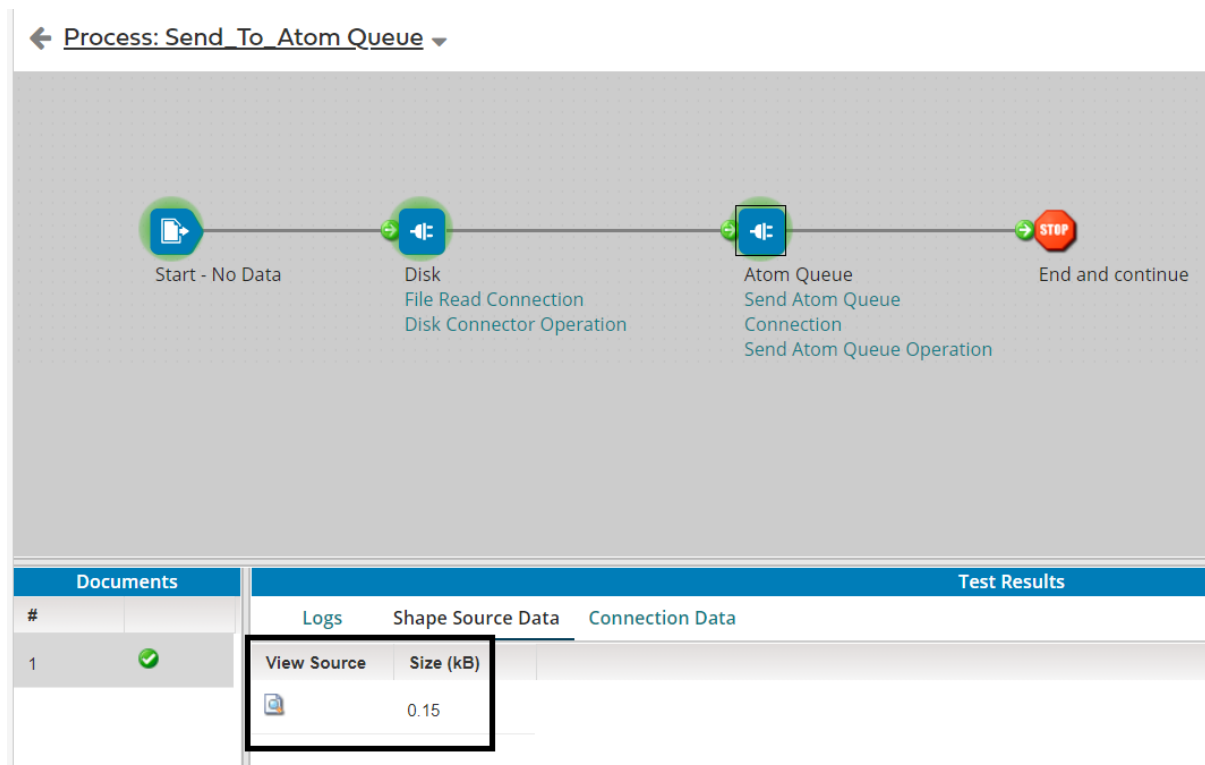
Dead Letter Queues contain failure of messages. When a failure occurs, the shared queue server will attempt to redeliver the message up to six more times. After the failures, the Dead Letter Queue will be created corresponding to the Destination Queue and the failed messages will be placed in the dead letter queue.

- We will see the concept of dead letter queues for the same Use Case. We have given wrong credentials for the data base so that the error pops out and the message gets inserted into Dead Letter Queue after seven attempts.

- We will run Send\_To\_Atom Queue process in Test Mode and monitor Read\_From\_AtomQueue in Process Reporting tab to see the retries and error message.












**Step 37:** We see that first process is executed and the message is pushed into the Atom Queue.



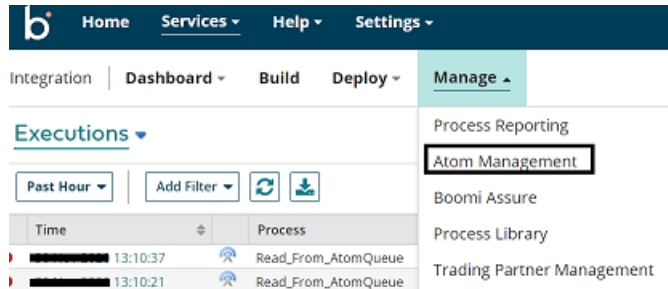
**Step 38:** Go to Manage > Process Reporting and monitor the second process. We see that process had failed to execute due to the wrong credentials and the shared queue server had attempted to redeliver the message up to six more times.

Executions ▾ Execute Process Auto Refresh is Off

Past Hour ▾ Add Filter ▾  

Time	Process	Actions	Atom	In	Out	Elapsed Time	Error Message
13:10:37	Read_From_AtomQueue		local_atom	1	0	0:02	Io exception: The Network Adapter could not establish t
13:10:21	Read_From_AtomQueue		local_atom	1	0	0:00	Io exception: The Network Adapter could not establish t
13:10:09	Read_From_AtomQueue		local_atom	1	0	0:02	Io exception: The Network Adapter could not establish t
13:10:03	Read_From_AtomQueue		local_atom	1	0	0:00	Io exception: The Network Adapter could not establish t
13:10:00	Read_From_AtomQueue		local_atom	1	0	0:00	Io exception: The Network Adapter could not establish t
13:09:58	Read_From_AtomQueue		local_atom	1	0	0:00	Io exception: The Network Adapter could not establish t
13:09:54	Read_From_AtomQueue		local_atom	1	0	0:02	Io exception: The Network Adapter could not establish t

**Step 39:** The failure message will now be sent to the Dead Letter Queue. Go to Manage > Atom Management

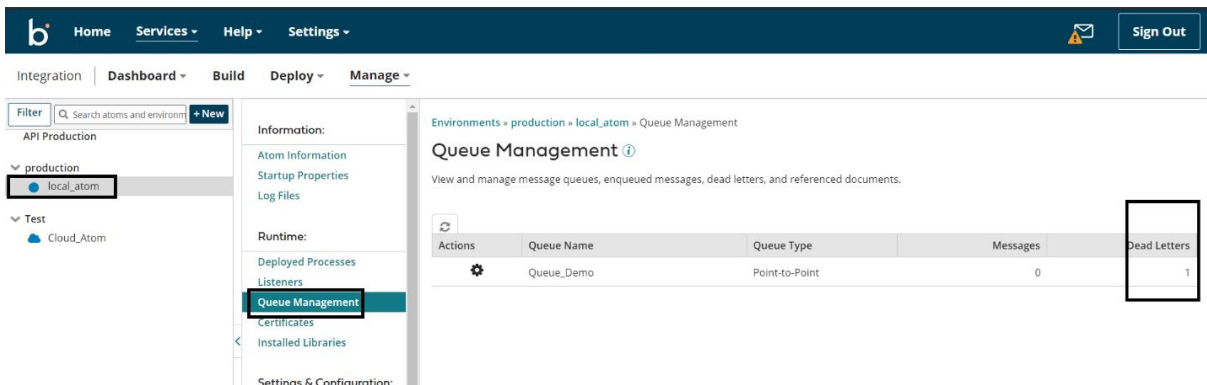


Home Services ▾ Help ▾ Settings ▾

Integration | Dashboard ▾ Build Deploy ▾ **Manage ▾**

- Process Reporting
- Atom Management**
- Boomi Assure
- Process Library
- Trading Partner Management

**Step 40:** Select the atom and click on Queue Management. We see that there is a message existing in Dead Letter Queue.



Home Services ▾ Help ▾ Settings ▾ Sign Out

Integration | Dashboard ▾ Build Deploy ▾ **Manage ▾**

Filter  + New

API Production


- production
  - local\_atom**
- Test
  - Cloud\_Atom

Information:

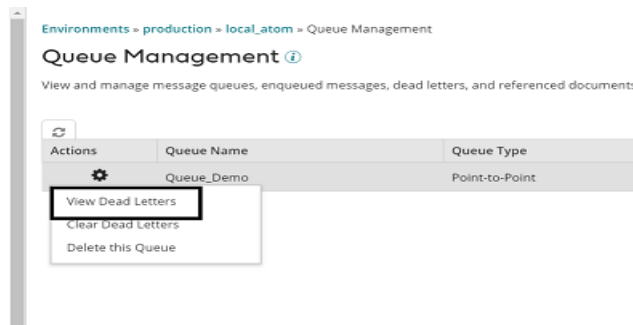
Environments » production » local\_atom » Queue Management

**Queue Management** ⓘ

View and manage message queues, enqueued messages, dead letters, and referenced documents.

Actions	Queue Name	Queue Type	Messages	Dead Letters
	Queue_Demo	Point-to-Point	0	1

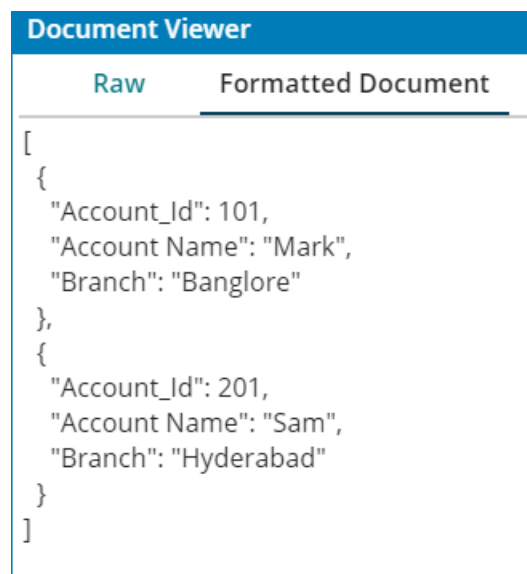
**Step 41:** To see the failure message, click on Actions and choose view Dead Letters.



**Step 42:** Click on Number of Documents followed by View Document Details.



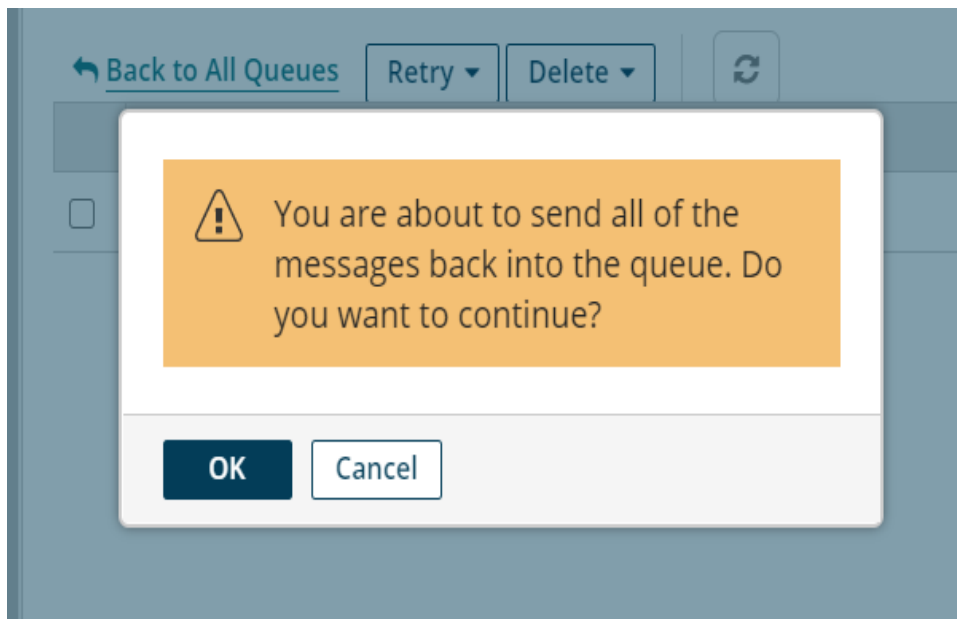
**Step 43:** We now see the document which has sent to the Dead Letter Queue.



**Step 44:** We have an option of Retrying the message and Delete the message in Dead Letter Queue. Here, we will retry and push the message from Dead Letter Queue to the Main Queue i.e. (Queue\_Demo) by selecting All Dead Letters in the Queue.





**Step 45:** We will be asked if we want to send messages from dead letter queue to Main Queue. Choose ok.



**Step 46:** We see that message had been pushed into Queue\_Demo which is the main Queue. Click on Back to All Messages > Back to All Queues and refresh it.

**Queue Management** ⓘ

View and manage message queues, enqueued messages, dead letters, and referenced documents.

Actions	Queue Name	Queue Type	Messages	Dead Letters
 	Queue_Demo	Point-to-Point	1	0

## Pros and Cons of Atom Queue

Advantages	Disadvantages
Atom Queues are simple, robust and reliable.	Messages cannot be sent between accounts and cannot be directly accessed from outside the Atom
Atom message queueing can be used to monitor Event based tracking and schedule based tracking	If the requirement is to set up a messaging service out of Boomi scope, then we don't prefer Atom Queues
Message queues can persist messages until they are fully processed	No guarantee of message delivery in atom queue
Atom message queueing allows you to scale your system when the number of messages grows by adding more listeners to a queue	If all the processes run on single atom, then a process can publish on Atom queue and other processes can subscribe to the message only if they are on same atom
Processes writing and reading data execute independently of each other in a real-time scenario in atom queues	Atom Queue is not designed to be an actual message service like JMS and compared to JMS based message queuing systems, it offers a limited set of features
If the scope of your messaging service is within a single atom, then we can use Atom Queue	Atom Queue life is within the atom which means messages published in atom cannot be consumed by any applications outside the atom
Dead letter queues helps in increasing accuracy about information transferred and shared with other servers	Boomi does not provide the ability to interact with dead letter queues during process execution





# TGH

Making Integrations Simpler



## TGH Software Solutions Pvt. Ltd.

[www.techygeekhub.com](http://www.techygeekhub.com)

At TGH, we specialize in driving digital transformation through seamless Integration Technologies.

Operating as an INTEGRATION FACTORY, we serve as a one-stop shop for all your integration needs. Our expert team is well-versed in enterprise software and legacy system integration, along with leading iPaaS technologies like Boomi, MuleSoft, Workato, OIC, and more.

We're committed to enhancing business processes and solving problems through our integration expertise.



### Email address

[connect@techygeekhub.com](mailto:connect@techygeekhub.com)



### Phone number

+ 011-40071137  
+ 91-8810610395



### Our offices

#### Noida Office

iThum  
Plot No -40, Tower A,  
Office No: 712,  
Sector-62, Noida,  
Uttar Pradesh, 201301

#### Hyderabad Office

Plot no: 6/3, 5th Floor,  
Techno Pearl Building,  
HUDA Techno Enclave,  
HITEC City, Hyderabad,  
Telangana 500081

