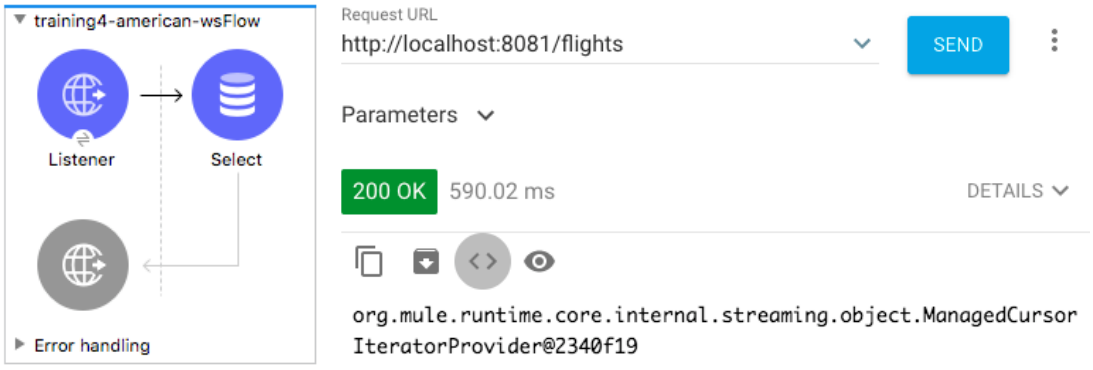
# Walkthrough 4-2: Connect to data (MySQL database)

In this walkthrough, you connect to a database and retrieve data from a table that contains flight information. You will:

- Add a Database Select operation.
- Configure a Database connector that connects to a MySQL database (or optionally an in-memory Derby database if you do not have access to port 3306).
- Configure the Database Select operation to use that Database connector.
- Write a query to select data from a table in the database.




## Locate database information

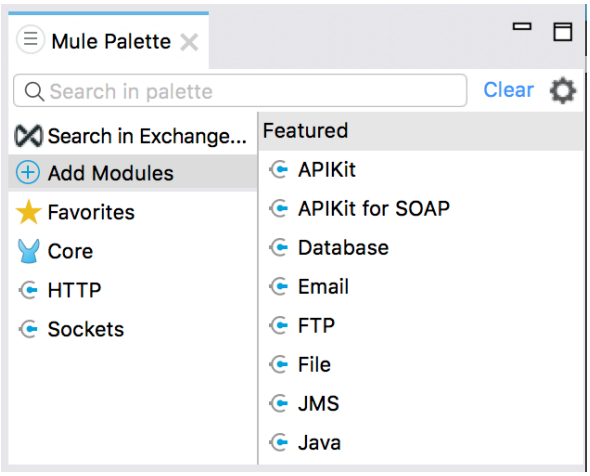
1. Return to the course snippets.txt file and locate the MySQL and Derby database information.

```
* MySQL database
db:
        host: "mudb.learn.mulesoft.com"
        port: "3306"
        user: "mule"
        password: "mule"
        database: "training"

American table: american
American table version2: flights
Account table: accounts
Account list URL:  http://mu.learn.mulesoft.com/accounts/show
or if using mulesoft-training-services.jar application:
http://localhost:9090/accounts/show.html

* MySQL database as URL and driver name
URL: jdbc:mysql://mudb.learn.mulesoft.com:3306/training?user=mule&password=mule
Driver class name: com.mysql.jdbc.Driver

* Derby database
URL: jdbc:derby://localhost:1527/memory:training
Driver class name: org.apache.derby.jdbc.ClientDriver
```

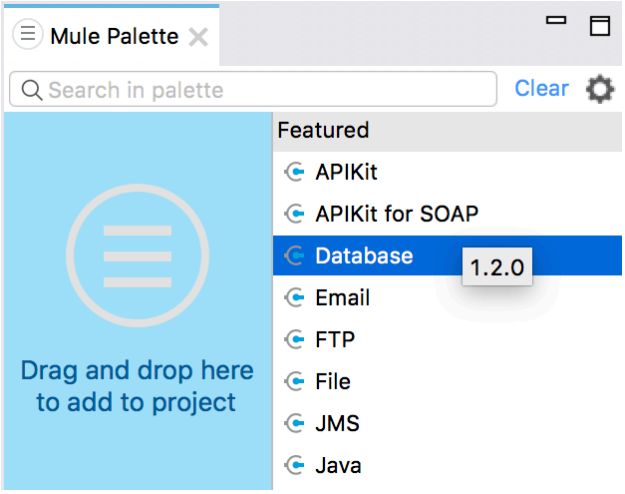
*Note: The database information you see may be different than what is shown here; the values in the snippets file differ for instructor-led and self-study training classes.*

## Add a Database connector endpoint

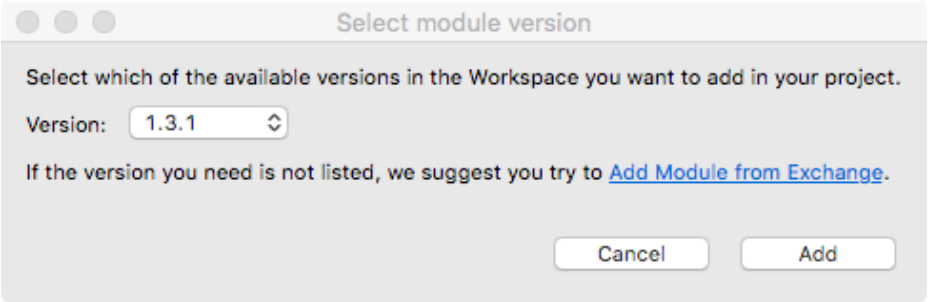
2. Return to Anypoint Studio.
3. Right-click the Set Payload message processor and select Delete.
4. In the Mule Palette, select Add Modules.



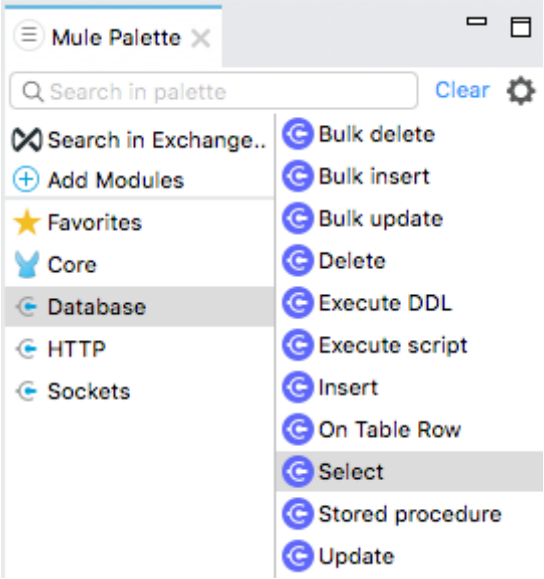

5. Select the Database connector in the right side of the Mule Palette and drag and drop it into the left side.



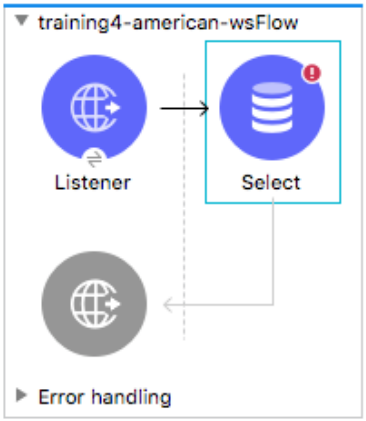

6. If you get a Select module version dialog box, select the latest version and click Add.



7. Locate the new Database connector in the Mule Palette.

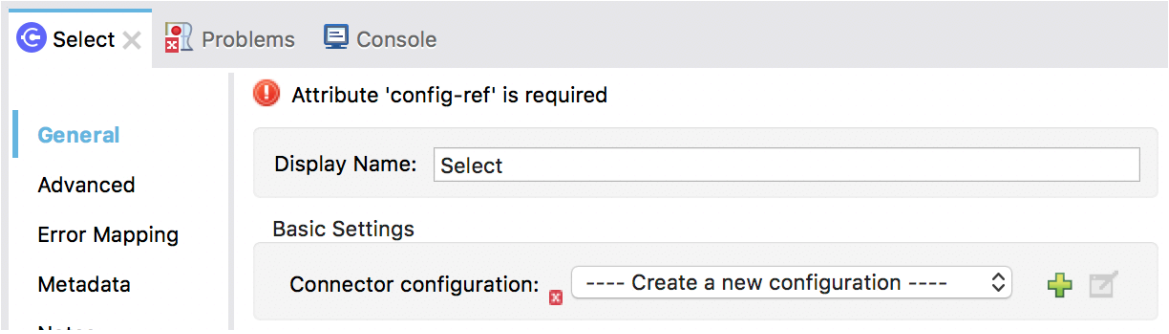



8. Drag and drop the Select operation in the process section of the flow.

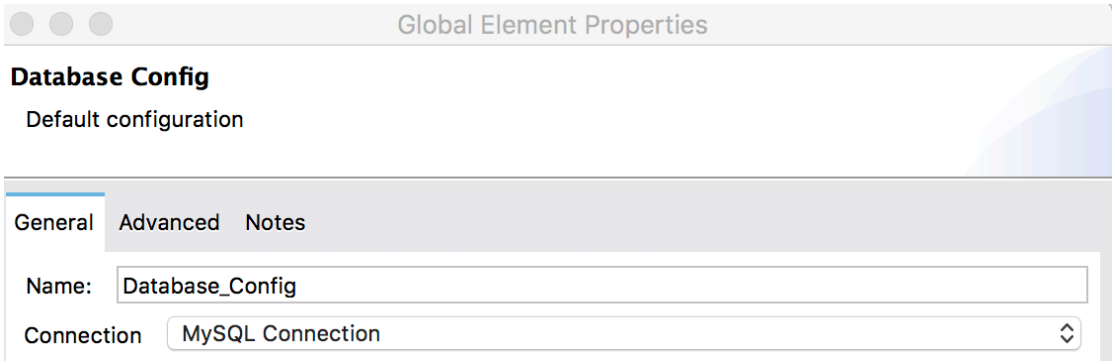



## Option 1: Configure a MySQL Database connector (if you have access to port 3306)

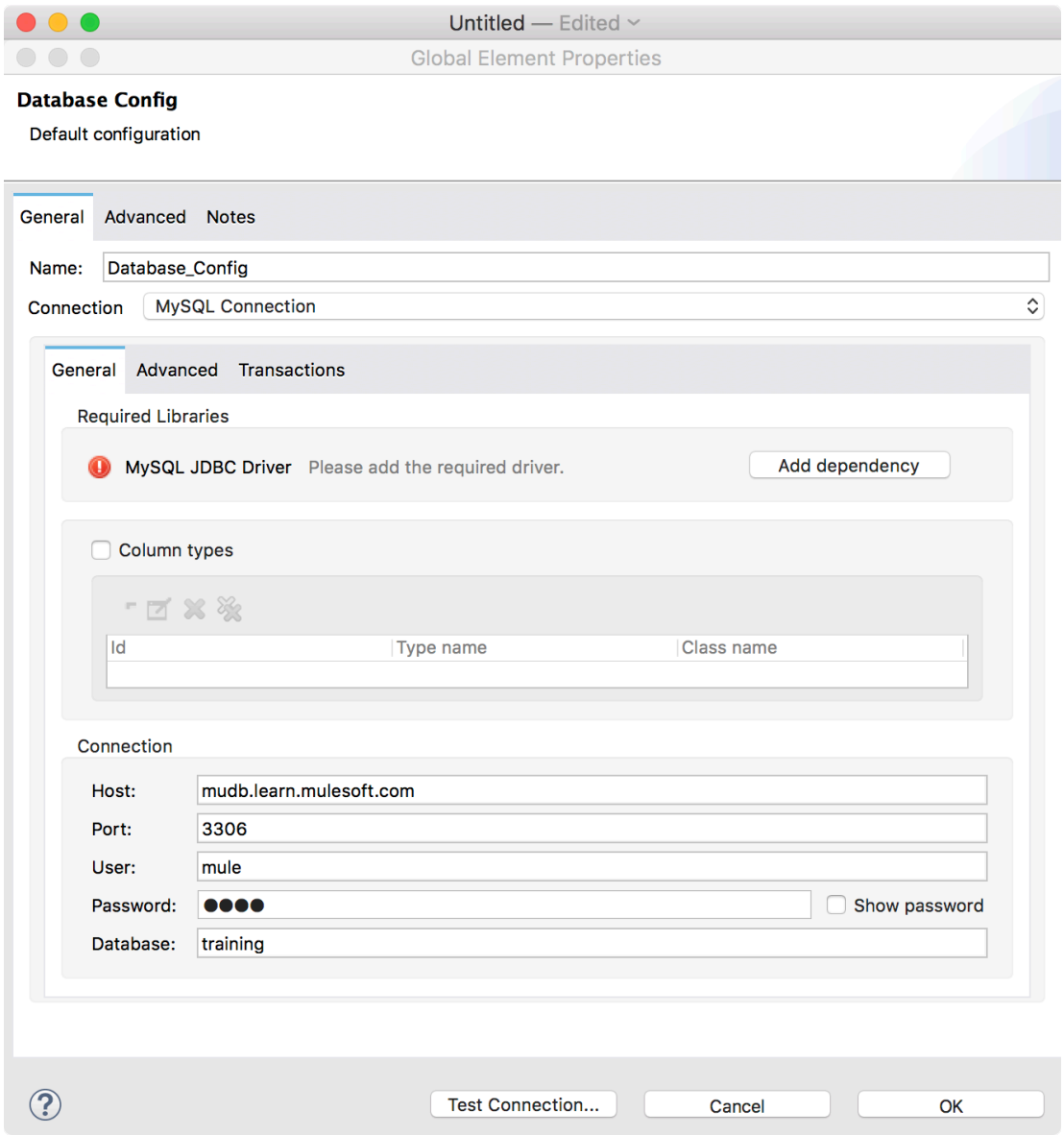
9. In the Select properties view, click the Add button next to connector configuration.



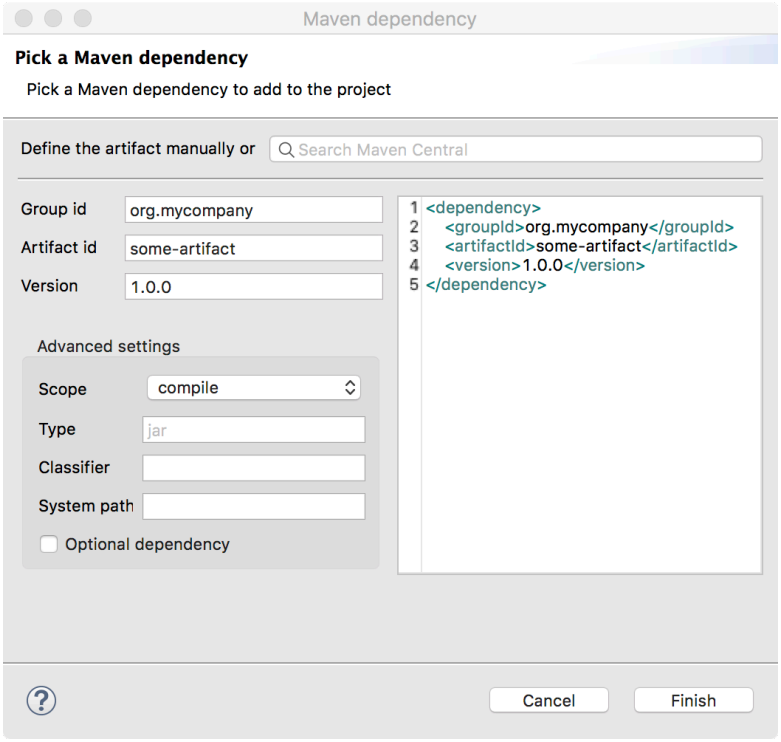
10. In the Global Element Properties dialog box, set the Connection to MySQL Connection.



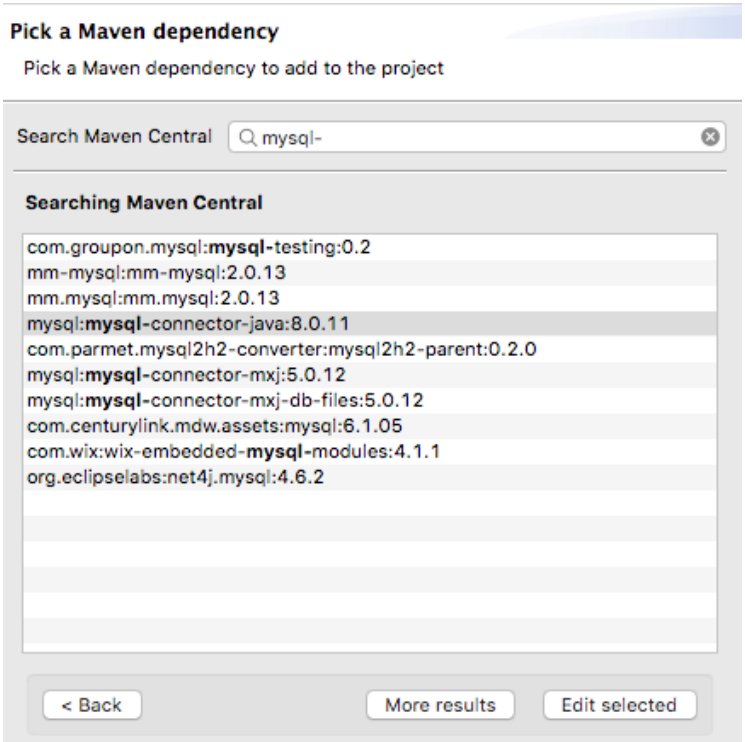

11. Set the host, port, user, password, and database values to the values listed in the course snippets.txt file.



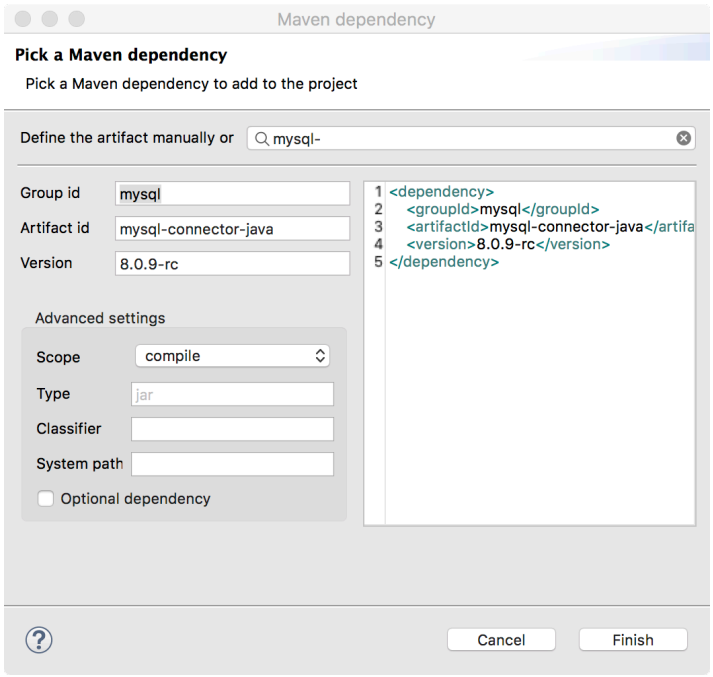
12. Click the Add dependency button next to MySQL JDBC Driver.
13. In the Maven dependency dialog box, locate the Search Maven Central text field.




14. Enter mysql- in the Search Maven Central text field.
15. Select mysql:mysql-connector-java in the results that are displayed.

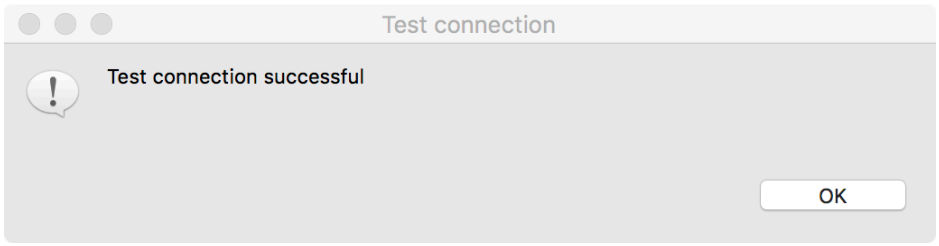


16. Click Edit selected.




17. Click Finish.
18. Back in the Global Element Properties dialog box, click the Test Connection button; you should get a successful test dialog box.

    *Note: Make sure the connection succeeds before proceeding.*




    *Note: If the connectivity test fails, make sure you are not behind a firewall restricting access to port 3306. If you cannot access port 3306, use the instructions in the next section for option 2.*

19. Click OK to close the dialog box.
20. Click OK to close the Global Element Properties dialog box.

## Option 2: Configure a Derby Database connector (if no access to port 3306)

21. In a command-line interface, use the cd command to navigate to the folder containing the jars folder of the student files.

22. Run the mulesoft-training-services.jar file.

```
java –jar mulesoft-training-services-X.X.X.jar
```

*Note: Replace X.X.X with the version of the JAR file, for example 1.6.2.*

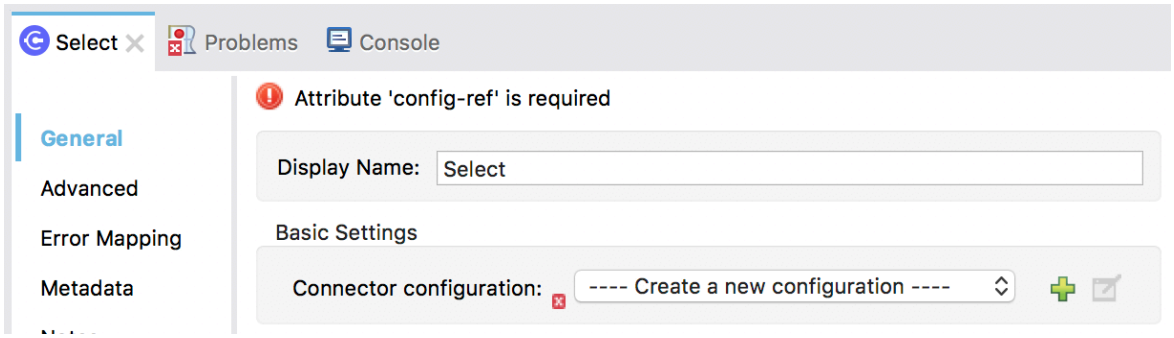
*Note: The application uses ports 1527, 9090, 9091, and 61616. If any of these ports are already in use, you can change them when you start the application as shown in the following code.*

```
java -jar mulesoft-training-services-X.X.X.jar --database.port=1530 --
ws.port=9092 --spring.activemq.broker-url=tcp://localhost:61617 --
server.port=9193
```

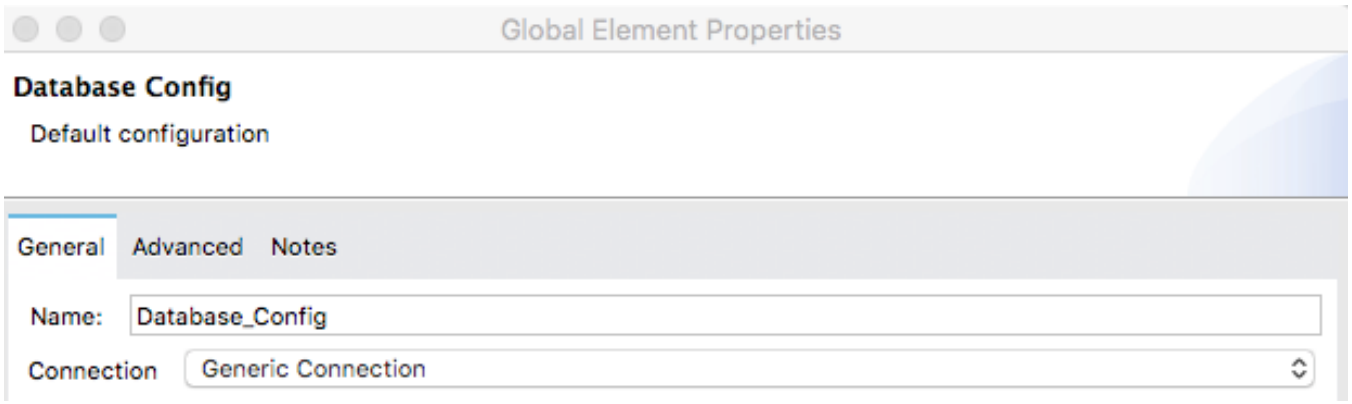
23. Look at the output and make sure all the services started.


*Note: When you want to stop the application, return to this window and press Ctrl+C.*

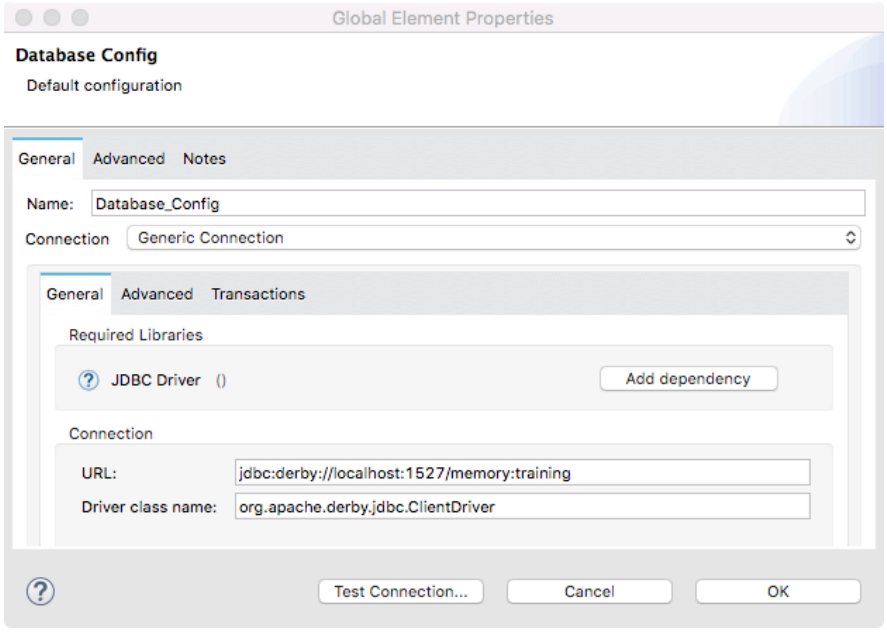
24. Return to Anypoint Studio.
25. In the Select properties view, click the Add button next to connector configuration.




26. In the Global Element Properties dialog box, set the Connection to Generic Connection.

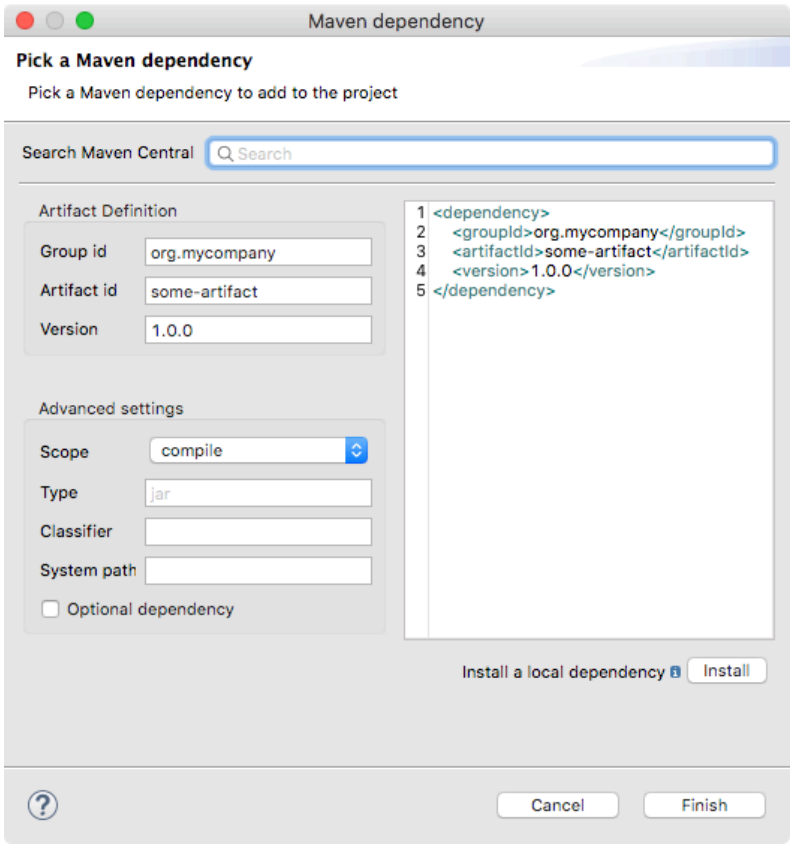



27. Set the URL and driver class name values to the values listed in the course snippets.txt file.

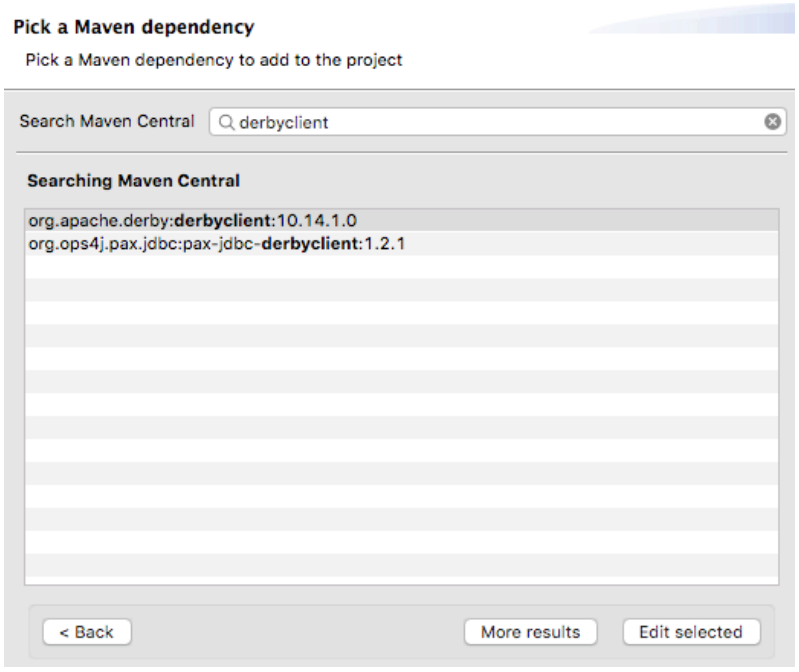



28. Click the Add dependency button next to JDBC Driver.

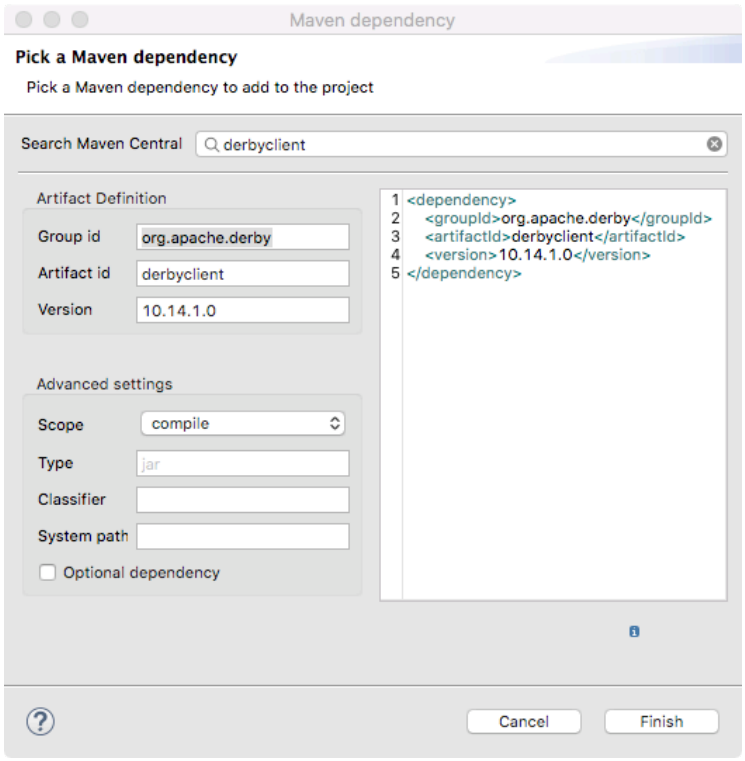
29. In the Maven dependency dialog box, locate the Search Maven Central text field.




30. Enter derbyclient in the Search Maven Central text field.
31. Select org.apache.derby:derbyclient in the results that are displayed.

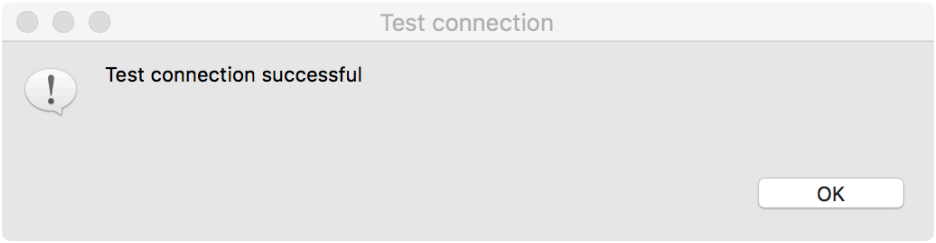


32. Click Edit selected.




33. Click Finish.
34. Back in the Global Element Properties dialog box, click the Test Connection button; you should get a successful test dialog box.

    *Note: Make sure the connection succeeds before proceeding.*

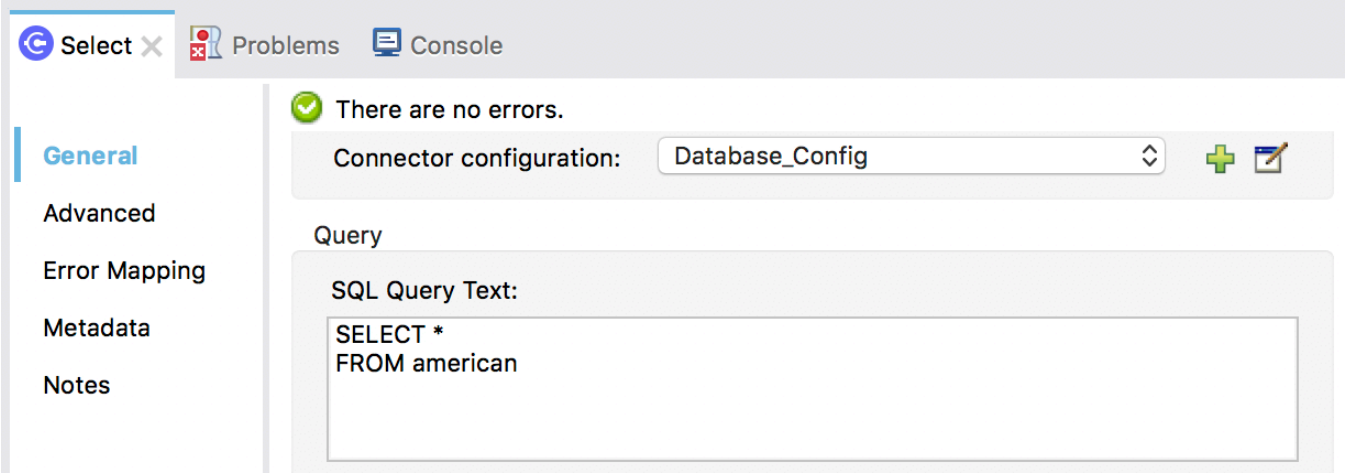



35. Click OK to close the dialog box.
36. Click OK to close the Global Element Properties dialog box.

## Write a query to return all flights

37. In the Select properties view, add a query to select all records from the american table.

```
SELECT *
FROM american
```




## Test the application

38. Run the project.
39. In the Save changes dialog box, select Yes.
40. Watch the console and wait for the application to start.
41. Once the application has started, return to Advanced REST Client.
42. In Advanced REST Client, make another request to http://localhost:8081/flights; you should get some type of Mule object.