Building Streaming Data Analytics Solutions on AWS -

Lab 4: MSK Streaming Pipeline and Application Deployment

Note: Do not include any personal, identifying, or confidential information into the lab environment. Information entered may be visible to others.

Corrections, feedback, or other questions? Contact us at *AWS Training and Certification*.

Lab overview

You are a data engineer at AnyCompany Online Shopping. AnyCompany Online Shopping sells a variety of consumer goods through e-commerce transactions in five categories: books, food, apparel, electronics, and home goods.

Your company wants a low-latency solution in real-time to analyze web clickstream events from its online customers who interact with its retail website. Your executive team wants to see top sales by category updated in near real time. It wants a solution with no setup costs and without having to manage servers. The team decides to build its solution on Amazon Web Services (AWS) using Amazon Managed Streaming for Apache Kafka (Amazon MSK) and Amazon Managed Service for Apache Flink services.

You are responsible to build a real-time streaming analytics pipeline in Managed Apache Flink Studio to meet the requirements. Managed Apache Flink Studio processes data with sub-second latencies from Amazon MSK and responds to events in real time. It can run Apache Flink applications continuously without having to manage servers. It can analyze streaming data interactively using managed Apache Zeppelin notebooks. Zeppelin notebooks provide a complete suite of analytics tools for data visualization, exporting data to files and controlling the output format for easier analysis.

In this lab, you build a stream processing pipeline with Amazon MSK by ingesting clickstream data and enriching the clickstream data with catalog data stored in Amazon Simple Storage Service (Amazon S3). You perform analysis on the enriched data to identify the sales per category in real time and visualize the output. You also build and deploy the Zeppelin notebook as a long-standing application storing data in Amazon S3 for fault tolerance and for further analysis.

OBJECTIVES

By the end of this lab, you will be able to do the following:

- Build a real-time Amazon MSK streaming analytics pipeline in Managed Apache Flink Studio using Apache Flink and Apache Zeppelin.
- Visualize the output.
- Build and deploy the Zeppelin notebook as a long-standing application with the ability to durably store data in Amazon S3.

ICON KEY

Various icons are used throughout this lab to call attention to different types of instructions and notes. The following list explains the purpose for each icon:

- **Command:** A command that you must run.
- **Expected output:** A sample output that you can use to verify the output of a command or edited file.
- **Note:** A hint, tip, or important guidance.
- **Additional information:** Where to find more information.
- **CAUTION:** Information of special interest or importance (not so important to cause problems with the equipment or data if you miss it, but it could result in the need to repeat certain steps).
- **WARNING:** An action that is irreversible and could potentially impact the failure of a command or process (including warnings about configurations that cannot be changed after they are made).
- **Consider:** A moment to pause to consider how you might apply a concept in your own environment or to initiate a conversation about the topic at hand.
- **Hint:** A hint to a question or challenge.
- An answer to a question or challenge.
- **Task complete:** A conclusion or summary point in the lab.

Start lab

1. To launch the lab, at the top of the page, choose Start lab.

Caution: You must wait for the provisioned AWS services to be ready before you can continue.

2. To open the lab, choose Open Console.

You are automatically signed in to the AWS Management Console in a new web browser tab.

WARNING: Do not change the Region unless instructed.

COMMON SIGN-IN ERRORS

Error: You must first sign out

If you see the message, **You must first log out before logging into a different AWS account:**

- Choose the **click here** link.
- Close your **Amazon Web Services Sign In** web browser tab and return to your initial lab page.
- Choose Open Console again.

Error: Choosing Start Lab has no effect

In some cases, certain pop-up or script blocker web browser extensions might prevent the **Start Lab** button from working as intended. If you experience an issue starting the lab:

- Add the lab domain name to your pop-up or script blocker's allow list or turn it off.
- Refresh the page and try again.

LAB ENVIRONMENT

The following diagram shows the basic architecture of the lab environment:

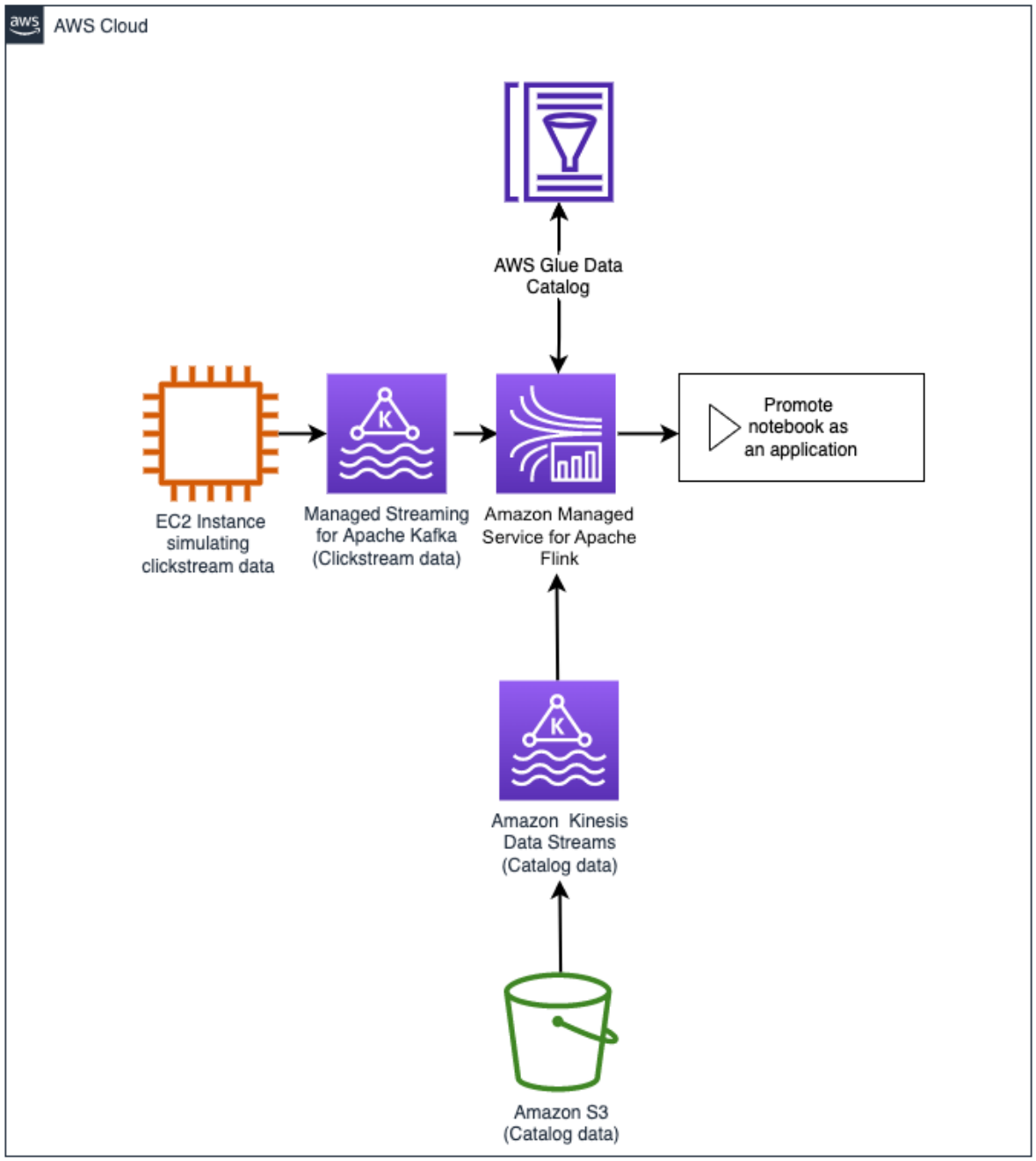

Image: The following list details the major resources in the diagram that have been per-provisioned for this lab:

- An *EC2 instance* that functions as a clickstream generator.
- An *Amazon MSK* cluster with TLS and AWS Identity and Access Management (IAM) authentication enabled on the cluster.
- An *Amazon Managed Apache Flink studio* for developing the analytics workload.
- An *S3 bucket* where catalog data (item_id, item_quantity, item_price, email and page) is stored and an *analytics* folder that contain the Zeppelin notebook files that are used in Managed Apache Flink Studio.

AWS SERVICES NOT USED IN THIS LAB

AWS service capabilities used in this lab are limited to what the lab requires. Expect errors when accessing other services or performing actions beyond those provided in this lab guide.

Task 1: Setting up the Zeppelin notebook environment

In this task, you set up a Managed Apache Flink Studio notebook for Amazon Managed Service for Apache Flink to interactively query data streams in real time.

TASK 1.1: INITIALIZE THE KINESIS ANALYTICS APPLICATION

In this step, you navigate to the created Kinesis Analytics application and initialize the application.

3. At the top of the console, in the search bar, search for and choose

   Kinesis.

Note: You navigate back to the console many times during this lab to launch multiple services. Open new services in a new tab by opening the context menu on the service and choosing *Open Link in New Tab*.

4. In the left navigation pane, choose **Managed Apache Flink**.
5. In the left navigation pane, choose **Studio notebooks**.
6. Choose the radio button next to the Studio notebook name starting with **KdaStudio-** and at the top of the page choose **Run**.
7. To confirm choose **Run**.

Caution: You can't perform any operations on the application while it is starting.

Note: It can take up to 5 minutes for the application to get to the *Running* state. Notice the Managed Apache Flink Studio notebook **Status** change from *Starting* to *Running*.

While waiting for the application to get to the *Running* state, complete the next task to download the two Apache Zeppelin Notebook files (*Lab4_MSK_Analytics.zpln* and *Lab4_MSK_Application.zpln*) from Amazon S3. You use these in the Zeppelin notebook environment.

TASK 1.2: DOWNLOAD THE ZEPPELIN FILES FROM AMAZON S3

Choose the links below to download each file.

8. Save the Lab4_MSK_Analytics.zpln file to your local machine.
9. Save the Lab4_MSK_Application.zpln file to your local machine.

Task complete: You have successfully set up the Zeppelin notebook environment.

Task 2: Create topics in the MSK cluster and simulate clickstream data generation

In this task, you create two topics in the MSK cluster (one for the streaming clickstream data and one for the reference user data stored in the S3 bucket). You also connect to an Amazon EC2 producer and follow the steps to start the clickstream generator.

TASK 2.1: ACCESSING MSK CLIENT INFORMATION

10. Navigate to the **AWS Management Console** browser tab.
11. At the top of the console, in the search bar, search for and choose

    MSK.

Note: You can close the notifications at the top of the page as they are just informative.

12. To access **Cluster summary**, choose the text link for the cluster name starting with **kafka-cluster-**.
13. Choose **View client Information**.

This page displays the connection strings for the bootstrap servers and the Apache Zookeeper.

Note: Keep this page open because you refer back to these connection strings while setting up various configurations in the course of this lab.

TASK 2.2: CREATE TOPICS

14. To access the Session Manager, in the list to the left of these instructions, copy the **CommandHostSessionManagementUrl** value. To display the EC2 producer terminal window, paste the **URL** into a new web browser tab and press **ENTER**.

Expected output:

```
******************************
**** This is OUTPUT ONLY. ****
******************************

Created topic clickstreamtopic.
Created topic catalog.
__amazon_msk_canary
__consumer_offsets
catalog
clickstreamtopic
```

Notice that the bootstrap script creates two topics (*clickstreamtopic* and *catalog*) in the MSK cluster, returns a confirmation that the topics have been created, and lists all the topics on the cluster including the ones created by default during provisioning of the cluster. You are returned to the shell prompt.

Note: If the command shell is unresponsive when you type a command, to return to the shell prompt, enter *Ctrl+C*.

TASK 2.3: SIMULATE CLICKSTREAM DATA GENERATION

Imagine owning a website or a web app. Users find their way to your site and perform various actions. Visiting a page, clicking a button, submitting information, and purchasing a product are examples of such activities. The users could also be using different devices when performing these operations. The simulator tries to imitate this payload structure across five sample segments (food, electronics, apparel, books, and home).

15. **Command:** To start the **clickmskgenerator_items.py** script,run the following command:

Note: This script starts the clickstream generator and writes to the *clickstreamtopic* MSK topic.

```
python3 clickmskgenerator_items.py $MSK_BOOTSTRAP clickstreamtopic 1
```

Expected output: Output is truncated.

```
*****************************
**** This is OUTPUT ONLY. ****
*****************************

 Number of arguments: 4 arguments

 Argument List: ['clickmskgenerator_items.py', 'b-
1.kafkacluster2134010371.647uhw.c5.kafka.us-west-2.amazonaws.com:9094,b-
2.kafkacluster2134010371.647uhw.c5.kafka.us-west-2.amazonaws.com:9094,b-
3.kafkacluster2134010371.647uhw.c5.kafka.us-west-2.amazonaws.com:9094',
'clickstreamtopic', '1']

 The program name is: clickmskgenerator_items.py

 The bootstrap server list : b-1.kafkacluster2134010371.647uhw.c5.kafka.us-
west-2.amazonaws.com:9094,b-2.kafkacluster2134010371.647uhw.c5.kafka.us-west-
2.amazonaws.com:9094,b-3.kafkacluster2134010371.647uhw.c5.kafka.us-west-
2.amazonaws.com:9094

 The msk topic name : clickstreamtopic

 Max interval in seconds between records : 1
{'event_id': '0341ab541f21029ae96f1af88f4d689d', 'event':
'entered_payment_method', 'user_id': 2, 'item_id': 12, 'item_quantity': 0,
'event_time': '2023-10-09 21:30:13.104087', 'os': 'web', 'page': 'apparel',
'url': 'www.example.com'}
sent event to Kafka! topic clickstreamtopic partition 2 offset 0

{'event_id': 'b77681ece5f7480cd112783a4e1b26ca', 'event': 'liked_item',
'user_id': 30, 'item_id': 53, 'item_quantity': 0, 'event_time': '2023-10-09
21:30:13.645905', 'os': 'ios', 'page': 'books', 'url': 'www.example.com'}
sent event to Kafka! topic clickstreamtopic partition 2 offset 1
```

The Amazon EC2 producer should now be producing clickstream data that contains *event_id*, *event*, *user_id*, *item_id*, *item_quantity*, *event_time*, *os*, *page*, and *url* information.

Note:

- Keep the web browser tab open for the duration of this lab.

- If the Session Manager window times out, repeat the steps outlined under *Simulate clickstream data generation*.

Task complete: You have successfully created topics in the MSK cluster and simulated clickstream data generation.

Task 3: Import the Zeppelin notebook

16. Navigate back to the **Managed Apache Flink Studio** page. Ensure that the Managed Apache Flink Studio notebook status is **Running**.

Note: You might need to refresh the page and navigate back to the **Studio notebooks** page to see the status update.

17. Choose the radio button next to the Studio notebook name starting with **KdaStudio-** and choose **Open in Apache Zeppelin** .

Note: You might need to turn off pop-up blockers on your browser.

This opens a web-based notebook that you can use for interactive analytics. All of the existing notes are listed on the left side of the page. You can also create a new note or import a note.

18. Choose **Import note**.
19. Choose **Select JSON File/IPYNB File** and select **Lab4_MSK_Analytics.zpln** from your local machine.
20. To open the notebook, choose the text link for **Lab4_MSK_Analytics**.

TASK 3.1: ZEPPELIN NOTEBOOK EXPLAINED

Here is a brief overview of the Apache Zeppelin notebook UI as it is used in this lab.

Experienced Zeppelin notebook users can skip this task and proceed to the next task.

Each Zeppelin note is composed of paragraphs. You can view the note as a paragraph container.

Each paragraph consists of the following two sections:

- *Code section*, where you put your source code. Locate the code section in your notebook UI.
- *Result section*, where you can see the result of the code run. Locate the result section in your notebook UI.
- To the right of these sections, the *Paragraph commands* run the paragraph code. Locate the paragraph commands in your notebook UI.

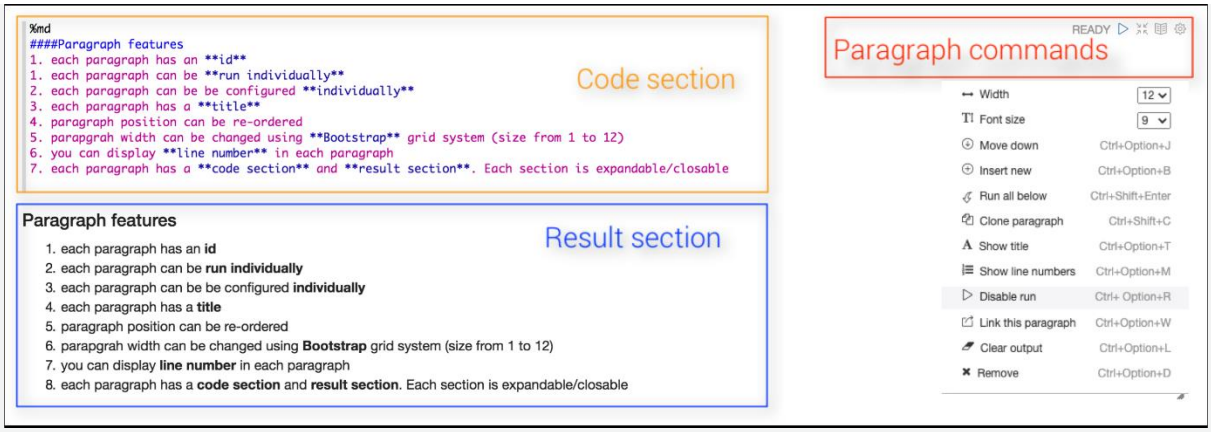

Image depicts layout for the Zeppelin note. One section for code, one for results, and one for paragraph commands.

The Managed Apache Flink Studio and the Zeppelin notebook environment is now set up for analysis.

Task complete: You have successfully imported the Zeppelin notebook.

Task 4: Analytics development in Managed Apache Flink Studio with Zeppelin notebook

Now that you are producing to the topic, you configure Managed Apache Flink as a consumer to query and analyze the streaming data as you build the streaming pipeline. You use a Managed Apache Flink Studio notebook to interactively query data streams in real time as you build the streaming pipeline. You use an Apache Flink interpreter to interact with your streaming data within the Zeppelin notebook environment.

Note:

- Managed Apache Flink Studio uses notebooks powered by Zeppelin and uses Apache Flink as the stream processing engine.
- Apache Zeppelin provides your Managed Apache Flink Studio notebook with a complete suite of analytics tools for data visualization, exporting data to files and controlling the output format for easier analysis.

21. Navigate to the **Lab4_MSK_Analytics** notebook.
22. Follow the instructions in the Zeppelin notebook environment and return for the next task where you build and deploy the streaming pipeline as an application that stores the data in Amazon S3.

Task complete: You successfully configured Managed Apache Flink as a consumer to query and analyze the streaming data as you built the streaming pipeline.

Task 5: Build and deploy the streaming pipeline as an application

Now that you have developed the streaming data pipeline in Zeppelin, you build and deploy the notebook into an application from Managed Apache Flink Studio.

There are two modes of running an Apache Flink application on Managed Apache Flink:

- With a Studio notebook, you have the ability to develop your code interactively, view results of your code in real time, and visualize it within your note. This was done in task 4 with the Lab4_MSK_Analytics notebook.
- You can also build your code and export it to Amazon S3. You can promote the code that you wrote in your note to a continuously running stream processing application. After you deploy a note to run in streaming mode, Managed Apache Flink creates an application for you. The application runs continuously, reads data from your sources, writes to your destinations, maintains long-running application state, and scales automatically based on the throughput of your source streams.

TASK 5.1 CONFIGURE AN AMAZON S3 DESTINATION

Check that the last code block in the notebook has been stopped.

23. Navigate back to the AWS Management Console tab.
24. At the top of the console, in the search bar, search for and choose

    Kinesis

    .
25. In the left navigation pane, choose **Managed Apache Flink**.
26. In the left navigation pane, choose **Studio notebooks**.
27. Choose the text link for the Managed Apache Flink Studio notebook starting with **KdaStudio-**.
28. Close to the middle of the page, **Monitoring** and **Configuration** options are available. Choose **Configuration**.
29. For **Deploy as application configuration - *optional***, choose **Edit** .
30. Choose **Browse** .
31. Choose the option button next to the bucket name starting with **databucket-** and **Choose**.
32. Choose **Save changes** which updates the notebook.

Note: The application can take 5-10 minutes for the status to change to *Running*.

You can follow the progress in the banner at the top of the page. In *Studio notebook details* section, you notice the **Status** change from *updating* to *running*.

TASK 5.2 IMPORT THE ZEPPELIN NOTEBOOK

33. In the left navigation pane, choose **Studio notebooks**.
34. Ensure that the notebook status is **Running** for the notebook name starting with **KdaStudio-**.
35. Choose the option button next to the Studio notebook name starting with **KdaStudio-** and choose **Open in Apache Zeppelin** .
36. Choose **Import note**.

37. Choose **Select JSON File/IPYNB File**, and select **Lab4_MSK_Application.zpln** from your local machine.
38. Choose **Lab4_MSK_Application** to open the notebook.

TASK 5.3 BUILD AND DEPLOY APPLICATION

In the task, you specify an Amazon S3 destination for the developed code. You then import a deployable notebook from Managed Apache Flink Studio, build your Zeppelin note and export it to Amazon S3 destination specified. After the build is complete, you deploy your code as a Kinesis streaming application with durable state and autoscaling. You also verify the output in the Amazon S3 destination specified.

You perform this step in the Zeppelin notebook environment (**Lab4_MSK_Application.zpln**) and return for the next step.

TASK 5.4 TEST APPLICATION DEPLOYMENT

You now *Force-Stop* the Managed Apache Flink Studio notebook, enable the streaming application and verify that streaming data is being written to the *Data* folder specified for output.

39. Navigate back to the **Managed Apache Flink** console.
40. In the left navigation pane, choose **Studio notebooks**.
41. Choose the option button next to the Studio notebook name starting with **KdaStudio-** and from **Actions** , choose **Force-stop**.
42. To confirm force-stop, in the text input field, enter the Managed Apache Flink Studio notebook name and choose **Force-stop**.

Expected output:

Notice a message in the top of the page that **Studio notebook KdaStudio-** has been successfully force-stopped.

43. In the left navigation pane, choose **Apache Flink applications**.
44. Choose the option button next to the streaming application name starting with **KdaStudio-** and at the top of the page, choose **Run** .
45. On the next screen, choose **Run without snapshot** and choose **Run**.

Note: The application can take 3 - 5 minutes for the status to change to **Running**.

Expected output:

You can follow the progress in the banner at the top of the page.

46. In the left navigation pane, choose **Apache Flink applications**.

For *streaming application*, notice the *Status* changes from *starting* to *running*.

Task complete: You successfully built and deployed the streaming pipeline as an application.

Task 6: Verify your output in Amazon S3

In this task, you verify that your output in Amazon S3 is available in accordance with the configured dynamic partitioning.

TASK 6.1: LOCATE THE OUTPUT IN AMAZON S3

47. Navigate back to the AWS Management Console tab.
48. At the top of the AWS Management Console, in the search bar, search for and choose

    S3

    .
49. Choose the Amazon S3 bucket with the name that starts with **databucket-**.
50. Navigate to the **data** folder.

Note: Choose the refresh icon until you see the folders with hive style partitioning. It can take up to 5 minutes for the output to be available in the Amazon S3 bucket.

Notice that the incoming data stream has been dynamically partitioned into **page/** and **event/**.

TASK 6.2: QUERY DATA WITH S3 SELECT

You can now query the data with S3 Select.

51. Choose any of the **page** and then choose an **event** by checking the box next to the object.

Note: Choose a file that does not start with prefix _ as these objects are still being used in writing.

52. In the **Actions** menu, choose **Query with S3 Select**.
53. In the **Input settings**, for **Format**, choose **JSON**.
54. In the **Output settings**, for **Format**, choose **JSON**.
55. In **SQL query**, choose **Run SQL query**. This returns a JSON output limited to five results.

The query results return a json output of the *event_id*, *event*, *user_id*, *event_time*, *os*, *page*, and *url*.

Task complete: You have built and deployed a resilient and fault tolerant application that can ingest the streaming data and store in Amazon S3 with the data partitioned into categories for further processing. Your solution requires no servers to manage and meets all the requirements set forth by your company.

Conclusion

You have done the following:

- Built a stream processing pipeline with Amazon MSK in Managed Apache Flink Studio using Apache Flink and Apache Zeppelin by ingesting clickstream data and enriching the clickstream data with catalog data stored in Amazon S3. You performed analysis on the enriched data to identify the sales per category in real time.
- Visualized the output.
- Built and deployed the Zeppelin notebook as a long-standing application with the ability to durably store data in Amazon S3.

End lab

Follow these steps to close the console and end your lab.

56. Return to the **AWS Management Console**.
57. At the upper-right corner of the page, choose **AWSLabsUser**, and then choose **Sign out**.
58. Choose End lab and then confirm that you want to end your lab.