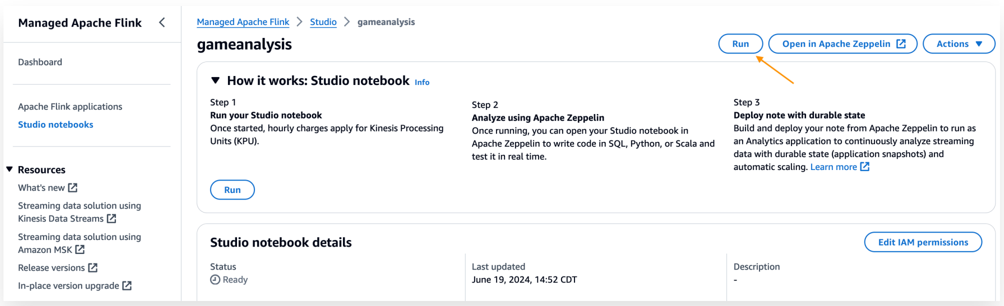
Run Studio Notebook

1. Navigate to the Amazon Managed Apache Flink and select Studio notebooks tab. Select **gameanalysis** notebook. Select Run. **A studio notebook can take a few minutes to start.**


Managed Apache Flink
Dashboard
Apache Flink applications
Studio notebooks
Resources
What's new
Streaming data solution using Kinesis Data Streams
Streaming data solution using Amazon MSK
Release versions
In-place version upgrade

Managed Apache Flink > Studio > gameanalysis
gameanalysis
Run | Open in Apache Zeppelin | Actions

How it works: Studio notebook Info

Step 1
Run your Studio notebook
Once started, hourly charges apply for Kinesis Processing Units (KPU).
Run

Step 2
Analyze using Apache Zeppelin
Once running, you can open your Studio notebook in Apache Zeppelin to write code in SQL, Python, or Scala and test it in real time.

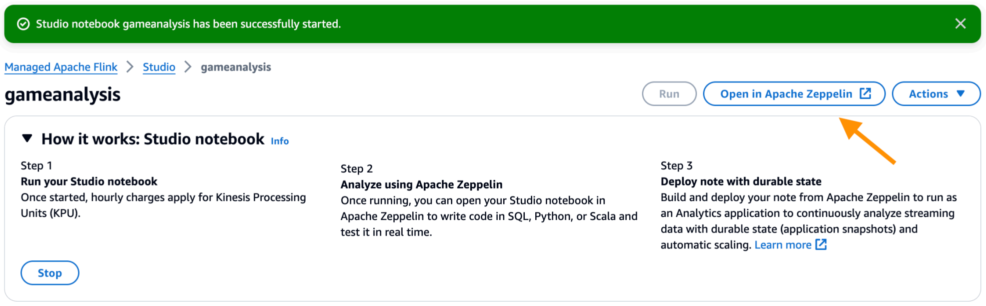
Step 3
Deploy note with durable state
Build and deploy your note from Apache Zeppelin to run as an Analytics application to continuously analyze streaming data with durable state (application snapshots) and automatic scaling. Learn more

Studio notebook details
Edit IAM permissions

Status	Last updated	Description
Ready	June 19, 2024, 14:52 CDT	-

Important
You will get a notification for hourly charge to run the Studio notebook. You can ignore that if you are executing the workshop in an AWS event and click **Run**. If you are executing the workshop in your own AWS environment, it may incur costs.

2. Choose **Open in Apache Zeppelin**.


Studio notebook gameanalysis has been successfully started.

Managed Apache Flink > Studio > gameanalysis
gameanalysis
Run | Open in Apache Zeppelin | Actions

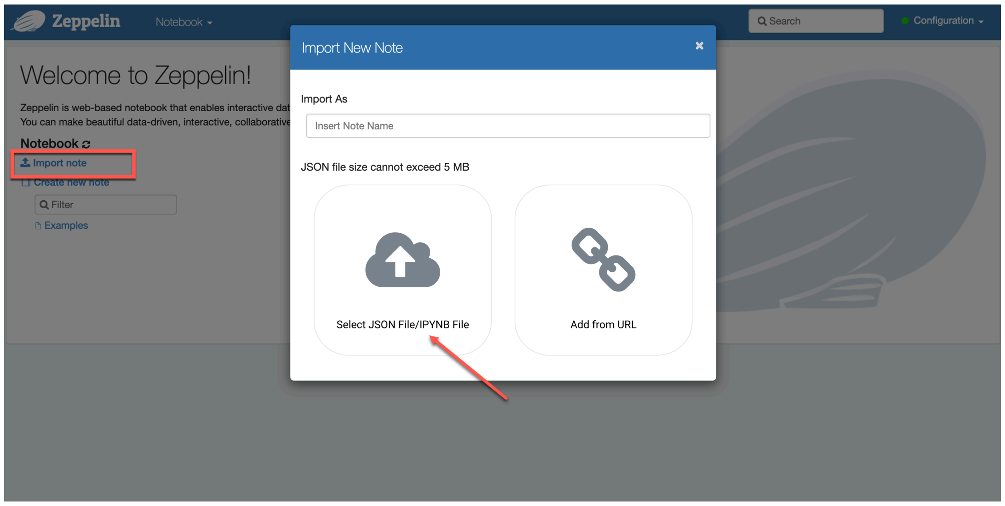
How it works: Studio notebook Info

Step 1
Run your Studio notebook
Once started, hourly charges apply for Kinesis Processing Units (KPU).
Stop

Step 2
Analyze using Apache Zeppelin
Once running, you can open your Studio notebook in Apache Zeppelin to write code in SQL, Python, or Scala and test it in real time.

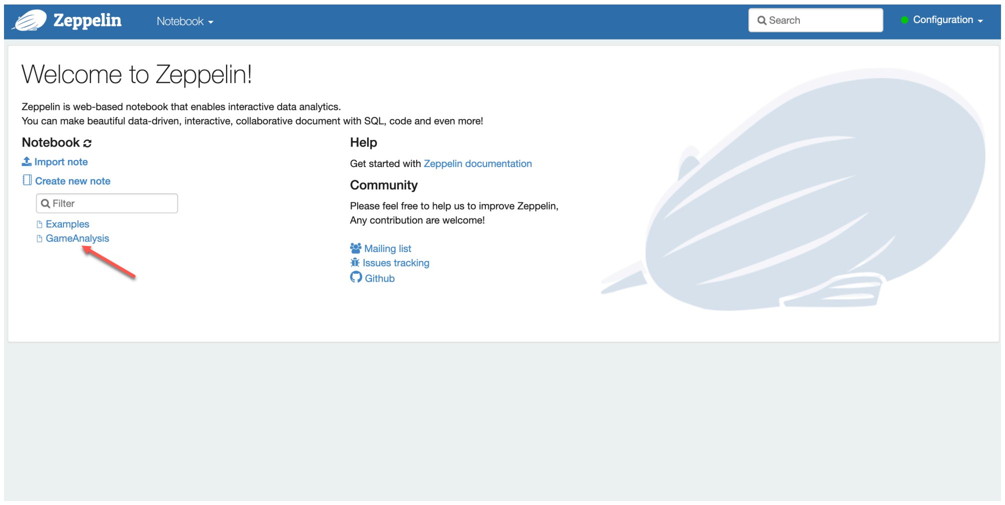
Step 3
Deploy note with durable state
Build and deploy your note from Apache Zeppelin to run as an Analytics application to continuously analyze streaming data with durable state (application snapshots) and automatic scaling. Learn more

3. Let's download the Zeppelin Notebook File which we'll use to read and load data into Kinesis Stream.
4. Import the Zeppelin Notebook previously downloaded.


Zeppelin Notebook Search Configuration
Welcome to Zeppelin!
Zeppelin is web-based notebook that enables interactive dat
You can make beautiful data-driven, interactive, collaborative
Notebook
Import note
Create new note
Filter
Examples

Import New Note
Import As
Insert Note Name
JSON file size cannot exceed 5 MB
Select JSON File/IPYNB File
Add from URL

5. Open the Notebook.



Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

6. Run the table creation script for the incoming stream. This SQL statement is creating a table named **gamestream** in a Flink stream processing application to consume data from an Amazon Kinesis stream named **data2dashboard** in the "us-east-1" region, with the data being in JSON format.

#1 Create the incoming streaming table

```
%flink.ssql(type=update)

CREATE TABLE gamestream (
    gameId BIGINT,
    creationTime TIMESTAMP(3),
    t1_playerId INT,
    t2_playerId INT,
    t1_kills INT,
    t1_death INT,
    t2_kills INT,
    t2_death INT,
    t1_towerKills INT,
    t1_inhibitorKills INT,
    t1_baronKills INT,
    t1_dragonKills INT,
    t1_riftHeraldKills INT,
    t1_ban INT,
    t2_towerKills INT,
    t2_inhibitorKills INT,
    t2_baronKills INT,
    t2_dragonKills INT,
    t2_riftHeraldKills INT,
    t2_ban INT
) WITH (
  'connector' = 'kinesis',
  'stream' = 'data2dashboard',
  'aws.region' = 'us-east-1',
  'scan.stream.initpos' = 'LATEST',
  'format' = 'json',
  'json.timestamp-format.standard' = 'ISO-8601'
);
```

7. View the streaming data in real time.

#2 View the Game Stream data in real time

```
%flink.ssql(type=update)
SELECT * FROM gamestream;
```

8. Analyze the total kills of each team.

#3 Total kills of each team

```
%flink.ssql
SELECT gameId,
SUM( t1_towerKills + t1_inhibitorKills + t1_baronKills + t1_dragonKills + t1_riftHeraldKills ) as t1_total_kills,
SUM( t2_towerKills + t2_inhibitorKills + t2_baronKills + t2_dragonKills + t2_riftHeraldKills ) as t2_total_kills
FROM gamestream
group by gameId;
```

9. You can view total bans of Team 1 by running the query in Step #4.

#4 Total bans of Team 1

```
%flink.ssql
SELECT
t1_playerId as playerId,
SUM(t1_ban) as no_of_ban
FROM gamestream
group by t1_playerId;
```

10. Create player kill ratio schema as a target by running the query in Step #5.

#5 Create the player kill ratio stream

```
%flink.ssql(type=update)

CREATE TABLE player_kill_ratio(
player_id INT,
kill_ratio DOUBLE,
window_end TIMESTAMP(3)
) WITH (
  'connector' = 'kinesis',
  'stream' = 'playerkillratio',
  'aws.region' = 'us-east-1',
  'format' = 'json'
);
```

11. Populate playerkillratio stream by executing Step #6.

#6 Generate the kill ratio realtime and populate target stream

```
%flink.ssql(type=update)

INSERT INTO player_kill_ratio
SELECT
  t1_playerId AS player_id,
  CAST(sum(t1_kills) AS DOUBLE) / NULLIF(sum(t1_death), 0) AS kill_ratio,
  TUMBLE_END(PROCTIME(), INTERVAL '30' SECOND) AS window_end
FROM gamestream
GROUP BY
  t1_playerId,
  TUMBLE(PROCTIME(), INTERVAL '30' SECOND);
```

12. Validate the player kill ratio stream.

#7 Validate the player kill ratio stream

```
%flink.ssql(type=update)
SELECT * FROM player_kill_ratio;
```

Conclusion

In this module, you gained hands-on experience with Amazon Managed Apache Flink and its Studio Notebook feature abd explored creating in-memory tables to ingest, analyze, and insert data between Kinesis Data Streams. With this, let's move on to next module.

Previous Next