GUIDE NO. 3

AWS IoT Analytics

Data Stores and Data Sets

AWS IoT Analytics Mini-User Guide

Introduction

Now that your data is cleansed, filtered, transformed, and enriched, you're ready to store that data and build data sets.

Data stores offer reliable and flexible storage for processed data and make it available for queries and analysis on large scale. Data stores do not require data schemas to be defined.

After processing the data in the Pipeline, AWS IoT Analytics stores it in an IoT-optimized data store for analysis for a set amount of time of your choosing.

At this point, you can query the data using the built-in IoT Analytics SQL query engine to answer specific business questions or create a data set. Data sets are a powerful and extensible query and analytics feature that are kept in the data store. Once created, data sets provide triggers, which allow queries and analysis to run on a regularly scheduled basis (configurable with cron-like syntax schedule).

In this guide, we will go through the steps of sending your data to the data store and then explain how you can create data sets. We will then cover how to create a container data set. The goal here is for you to understand where the data goes once processed, feel confident that your data is secure, and then show you how you can build your data sets and container data sets which are also securely stored.

Adding Value to Your Business

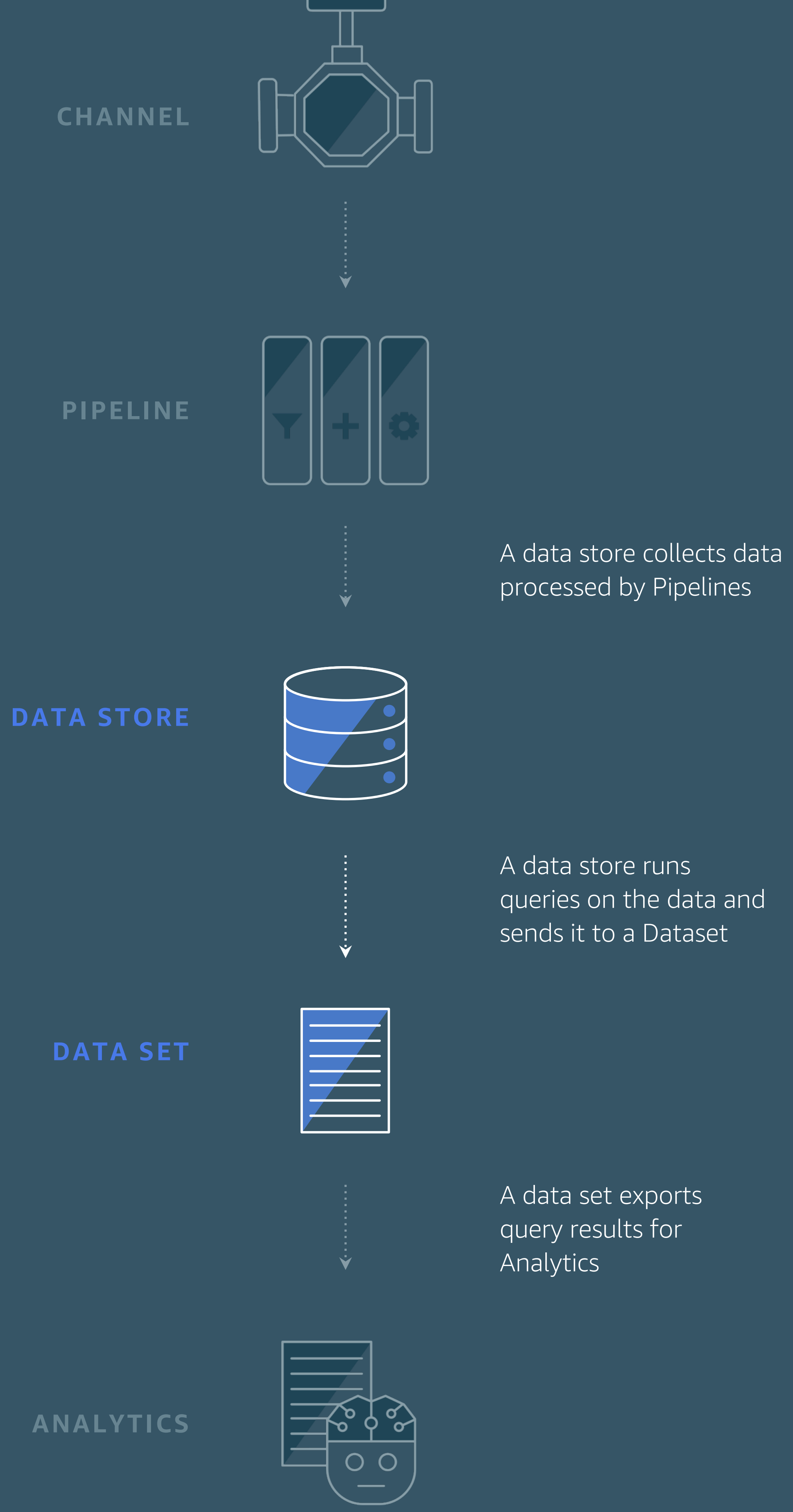
Data stores are optimized for processed IoT data to provide you with greater data efficiency. You can store the processed and the raw data for easy reprocessing using different logic as your needs change.

Data stores are partitioned by time which provides faster query responses on time series data. This is critical when data driven decisions need to be made on the fly.

Data sets can be created running ad hoc or scheduled queries through the Console Query Editor. Automating this process frees up time to focus on more pressing analytics needs.

Container data sets can automatically run your analysis tools and generate results by bringing together a SQL data set as input, a Docker container with your analysis tools and needed library files, input and output variables, and an optional schedule trigger. Container data sets allow you to automate your custom analysis.

What is a Data Store, a Data Set, and a Container Data Set?



A data store collects data processed by Pipelines

A data store runs queries on the data and sends it to a Dataset

A data set exports query results for Analytics

A data store is IoT-optimized data storage for data processed by Pipelines.

A data store also makes it available to run queries or analysis on the data.

A data set contains the result of query or analysis against a data store and a definition of how and when the data is generated.

Data sets support running built-in SQL queries, exporting query results to AWS Quicksight for visualization, and running custom analysis using Jupyter Notebook.

A container data set allows you to automatically run your analysis tools and generate results.

You can bring your own custom analysis container to IoT Analytics and schedule its execution through a container data set.

AWS IoT Analytics stores the processed device data in a time-series data store that is optimized to deliver fast response times on IoT queries that typically include time as a criteria. The raw data is also automatically stored for later processing or to reprocess it for another use case.

Step by Step

Create and Use a Data Store and SQL Data Set

After data is processed in the Pipeline, it will be stored into the data store. There is no need to provide schemas. Assume that there is already a Pipeline with data store activity at its last stage, and the data store activity specifies “demo_datastore” as output data store.

Step 1: Create a data store and a SQL data set

```
> aws iotanalytics create-datastore --datastore-name demo_datastore
```

Once a message arrives at data store, it can be queried or analyzed by a data set. Create a json file “demo_Dataset.json” with the following content:

```
    {
        "datasetname": "demo_dataset",
        "actions": [
            {
                "actionName":"myaction",
                "queryAction": {
                    "sqlQuery": "select * from demo_datastore limit 50"
                }
            }
        ],
        "triggers": [
            {
                "schedule": {
                    "expression": "cron(0 0 1 * ? *)"
                }
            }
        ]
    }
```

Create “demo_dataset” with “demo_dataset.json” using AWS CLI

```
> aws iotanalytics create-dataset --cli-input-json file://demo_dataset.json
```

As shown in demo_dataset.json, the data set will run a SQL query that simply pulls everything from demo_datastore. The “triggers” part specified that the query will run monthly (defined in cron like schedule)

Step 2: Run data set query and get result using AWS CLI

```
> aws iotanalytics describe-dataset --Dataset-name demo_dataset
```

The result may be like this:

```
    {
        "Dataset": {
            "contentStatus": {
                "state": "CREATING"
            },
            "actions": [
                {
                    "actionName": "myaction",
                    "queryAction": {
                        "sqlQuery": "select * from demo_datastore limit 100"
                    }
                }
            ],
            "name": "demo_dataset",
            "arn": "arn:aws:iotanalytics:us-west-2:xxxx:Dataset/demo_dataset",
            "triggers": [
                {
                    "schedule": {
                        "expression": "cron(0 0 1 * ? *)"
                    }
                }
            ]
        }
    }
```

The result in "contentStatus"."state" is "CREATING", which means the query is running. If it becomes "SUCCEEDED", the query has finished. (You may need to check the status multiple times to wait until status becomes "SUCCEEDED"). Fetch query result using AWS CLI:

```
> aws iotanalytics get-Dataset-content --dataset-name demo_dataset
```

The result will look like this:

```
    {
        "entries": [
            {
                "dataURI": "https://aws-iotanalytics-Dataset-xxx-xxx-xxx-xxx-
xxx.s3.us-west-2.amazonaws.com/results/xxx-xxx-xxx-xxx-xxx.csv?X-Amz-Security-
Token=xxx&X-Amz-Algorithm=xxx&X-Amz-Date=xxx&X-Amz-SignedHeaders=host&X-Amz-
Expires=7200&X-Amz-Credential=xxx&X-Amz-Signature=xxx"
            }
        ],
        "timestamp": 1521588047.388
    }
```

Download the result file by copying the “dataURI” into a browser. **Note:** Published messages can take minutes to be able to get queried.

Create a Container Data Set

You can create a container data set in the IoT Analytics console or API by specifying your SQL dataset, Docker image, input and output variables and an optional schedule trigger as a parameter, where a trigger can be an event of SQL data set content creation or a schedule expression. The container data set feeds the content of SQL data set to the analytical model available in the Docker image and automatically executes the model based on the trigger specified by the customers to generate KPIs, metrics, and notifications.

One option to perform advanced analytical functions is to use a Jupyter Notebook. Jupyter Notebooks provide powerful data science tools that can perform machine learning and a range of statistical analyses. You can package your Jupyter Notebooks and libraries into a container that periodically runs on a new batch of data as it is received by AWS IoT Analytics during a DeltaTime window you define. See the steps to create a container from your Jupyter Notebooks in AWS IoT Analytics User Guide. You can schedule an analysis job that uses the container. The new, segmented data captured within the specified time window then stores the job’s output for future scheduled analytics.

Example

Once you have setup a Docker image, uploaded it to an ECR repository, upload your sample data to S3, and. created a customer execution role. You are ready to create a simple container data set.

Step 1: Set up a data set to run a container action.

Download the following to a file named "cli-input.json" and replace all instances of "<your-account-id>" and "<region>" with the appropriate values:

```
{
    "datasetName": ”demo_dataset",
    "actions": [
        {
            "actionName": ”demo",
            "containerAction": {
                "image": "<your-account-id>.dkr.ecr.<region>.amazonaws.com/demo-
moment",
                "executionRoleArn": "arn:aws:iam::<your-account-
id>:role/container-execution-role",
                "resourceConfiguration": {
                    "computeType": "ACU_1",
                    "volumeSizeInGB": 1
                },
                "variables": [
                    {
                        "name": ”demoResultS3URI",
                        "outputFileUriValue": {
                            "fileName": "output.mat"
                        }
                    },
                    {
                        "name": "inputDataS3BucketName",
                        "stringValue": ”demo-sample-data-<your-account-id>"
                    },
                    {
                        "name": "inputDataS3Key",
                        "stringValue": "input.txt"
                    },
                    {
                        "name": "order",
                        "stringValue": "3"
                    }
                ]
            }
        }
    ]
}
```

.

Step 2: Create a data set using the file "cli-input.json" you just downloaded and edited:

```
aws iotanalytics create-dataset –cli-input-json file://cli-input.json
```

After you create your container action, you can invoke data set content generation, get the data set content, and print.

For more details on how to schedule a container data set using custom analysis code from external tools, see the AWS IoT Analytics User Guide.

Next Steps

Once the data store and data set are built, analysis using Jupyter Notebook and visualization using AWS Quicksight could be linked to the data set for further analysis or visualization. To automate your workflow using data sets, follow steps above, then use the trigger from the SQL data set to create a container data set. After you have created an initial container data set, you can see the embedded results in the IoT Analytics console itself.

Easily analyze data for deeper insights to make better, more accurate decisions for IoT applications and machine learning use cases. With AWS IoT Analytics, you can collect, pre-process, enrich, store and analyze your IoT data.

Start using Data Stores and Data Sets with AWS IoT Analytics in minutes:

aws.amazon.com/iot-analytics