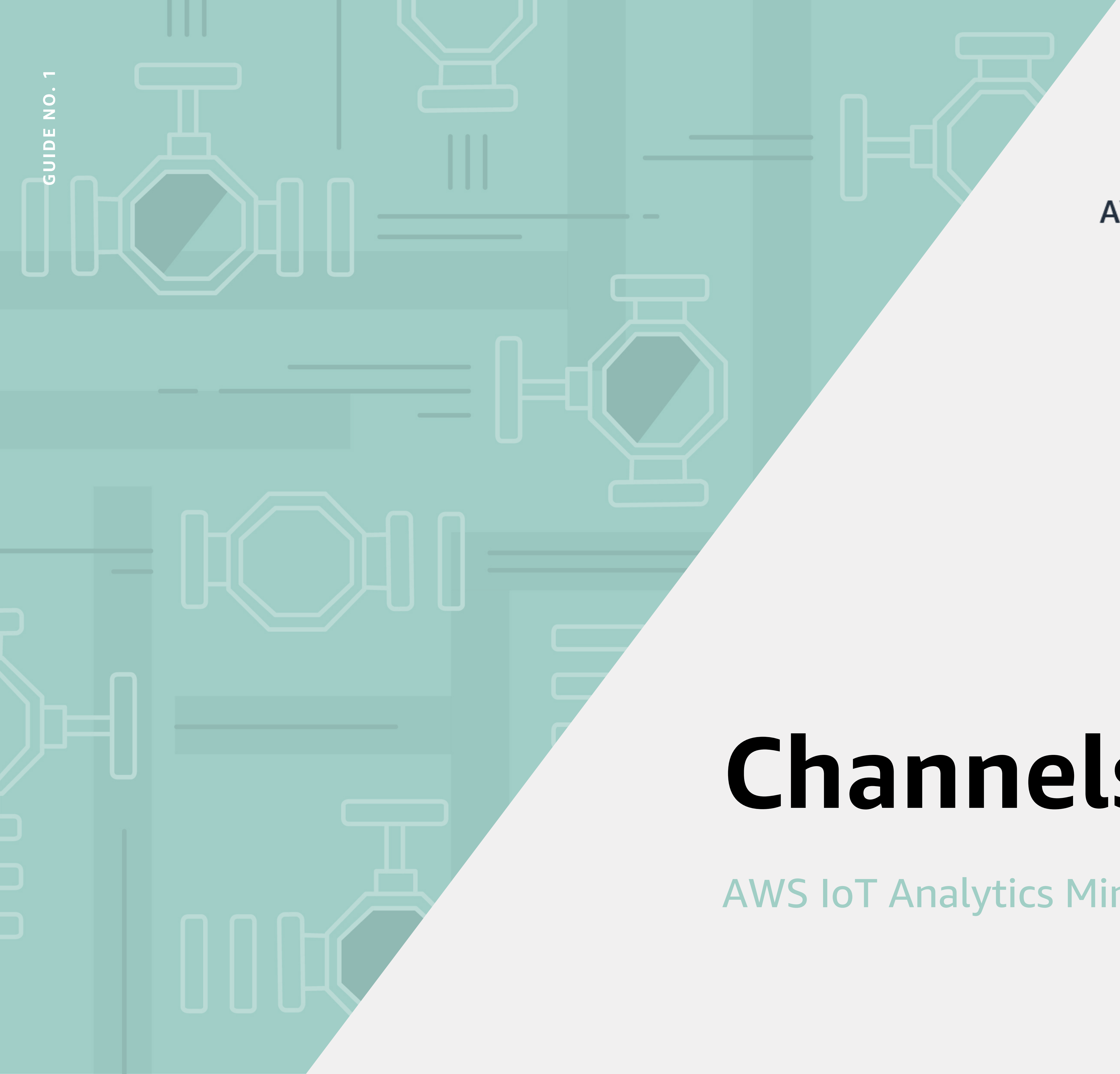
GUIDE NO. 1

AWS IoT Analytics

Channels

AWS IoT Analytics Mini-User Guide



Introduction

In this first user guide from in our series, we aim to provide you with the information you need to understand what a Channel is, why it's important to your data flows and how you can create one.

IoT data is growing exponentially as customers invest in more connected devices and seek to gain intelligence from the data they generate. But sorting through billions of messages from millions of devices is a huge challenge.

With AWS IoT Analytics—a fully-managed IoT analytics service—you can collect, pre-process, enrich, store and analyze IoT device data at scale.

AWS IoT Analytics can perform simple ad hoc queries as well as complex analysis. Understand the performance of devices. Predict device failures and machine learning. Designed specifically for IoT, it automatically captures and stores the message timestamp as well as the device ID making it easy to perform time-series analytics. AWS IoT Analytics can also enrich the data with IoT-specific metadata such as device type and location using the AWS IoT Device Registry. AWS IoT Analytics stores data in an IoT-optimized data store so you can run queries on large data sets.

AWS IoT Analytics can accept data from any source like Amazon Kinesis, S3 or other sources, using ingestion API and is fully integrated with AWS IoT Core so it is easy to get started. First, you define an IoT Analytics Channel and select the data you want to collect so you only store and analyze the data of interest, such as sensor temperature. Once the Channel is set up, you configure IoT Analytics Pipelines to process your data. IoT Analytics Pipelines support transformations like Celsius to Fahrenheit conversion, conditional statements, message filtering and message enrichment using external data sources and AWS Lambda functions.

After processing the data in the Pipeline, AWS IoT Analytics stores it in an IoT-optimized data store for analysis for the amount of time you set. You can query the data using the built-in IoT Analytics SQL query engine to answer specific business questions. For example, you may want to know how many monthly active users there are for each device in your fleet. Through integration with Amazon SageMaker, IoT Analytics supports more sophisticated analytics with machine learning. Easy to build visualizations and dashboards. Get business insights quickly from your IoT Analytics data since it is integrated with Amazon QuickSight.

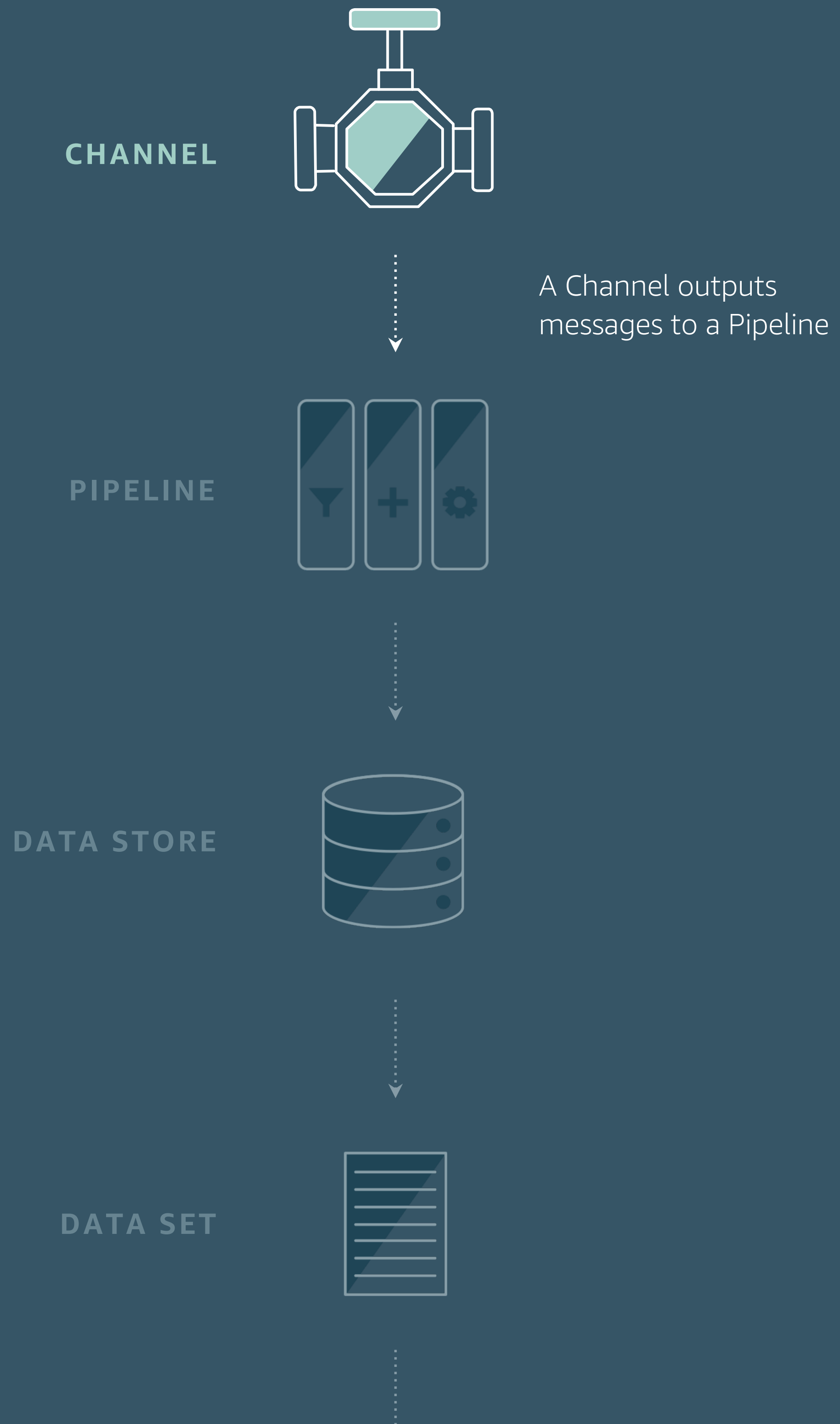
Adding value to your business

With Channels, customers are liberated from continually connecting to the same data source repeatedly. And, with raw data storage initiated as soon as customer connects a Channel, there's always a record of what IoT Analytics has received, allowing for reprocessing or backfilling Pipelines and data stores at a later date.

Fully integrated with AWS IoT Core for easy, fast data ingest into AWS IoT Analytics.

Greater flexibility with our ingest API that allows you to send your data to IoT Analytics from S3, Kinesis or any other source.

Collect all your raw data or only the data you want to store and analyze. The AWS IoT Analytics console is designed to receive messages that can be filtered in various formats and frequencies.



What is a Channel?

A Channel is used to ingest data and feed a Pipeline(s) while keeping a copy of the raw messages for a period of time. You can create single or multiple Channels to store all raw unprocessed JSON messages For sending data to the Channel, we would have to create an IoT rule in the AWS IoT. The following shell command can be used to create the rule.

```
aws iotanalytics create-topic-rule --rule-name
analytics_aus_weather --topic-rule-payload file://rule.json
```

A Channel collects data from an MQTT topic and archives the raw, unprocessed messages before publishing the data to a Pipeline.

Channels offer greater flexibility because devices can send data to multiple Channels which allows customers to partition data by type, device or another attribute.

Step by step

How to get started

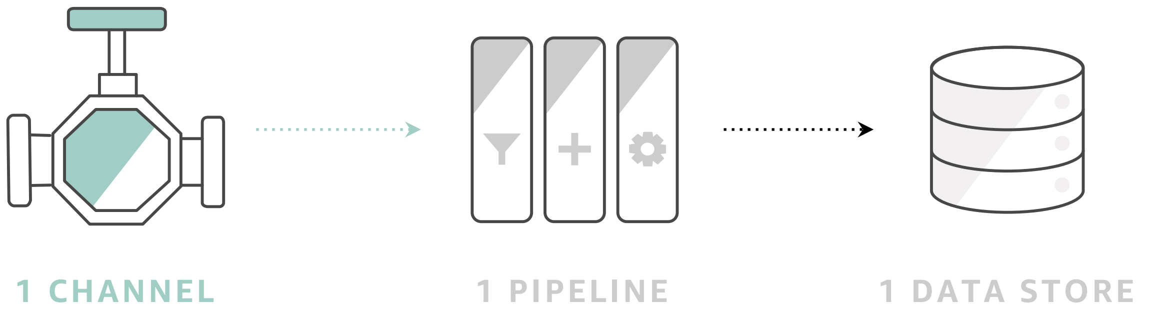
It all starts with data:

When devices generate data, the data is sent to IoT Analytics either from AWS IoT Core or through an API accessed from any other source like, S3, Kinesis or AWS Greengrass. It's a challenging problem to design data flows properly as we need to clean, process, enrich and transform incoming IoT messages quickly and at a lower cost. Without knowing about the available types of data flows, it's difficult to leverage the use of the AWS IoT Analytics platform.

At first, we will walk through the available types of data flows and then we will describe a use case and try to find the best design solution for it.

Let's look at what the AWS IoT Analytics service has to offer for designing different data flows. Based on above-mentioned inferences, we conclude that 3 different types of data flows can be designed.

2.1 Type 1 Data flow: This is a simple yet very powerful configuration. From a Channel, a Pipeline sources raw messages and stores the normalized messages in data store.



For the simplest way to leverage channels and pipelines, it's best for all messages to follow the same structure, JSON. This helps us keep our pipeline operations simple. For multi-schema messages, this requires a more complex pipeline architecture or include a lambda function to ensure data is processed correctly.

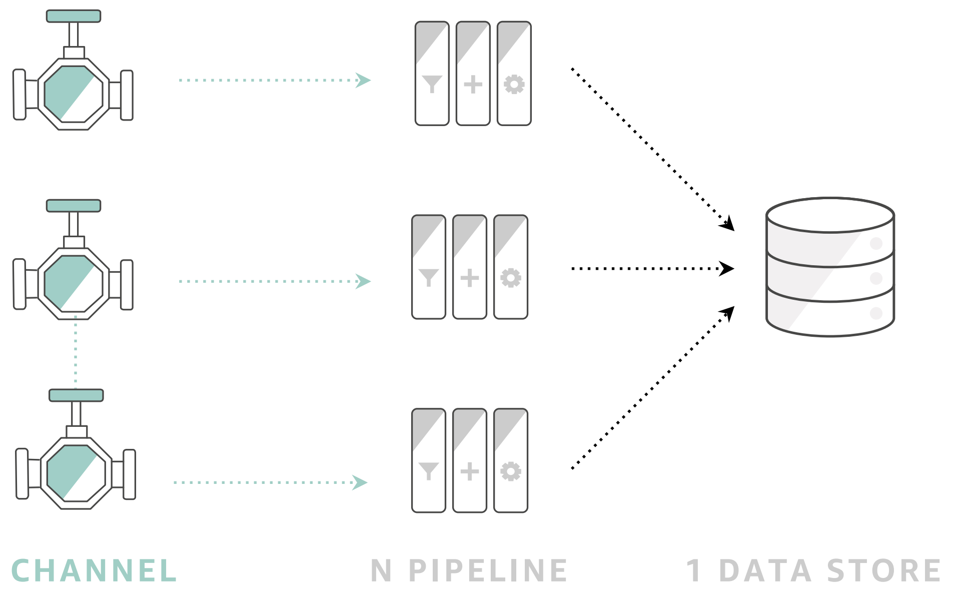
2.1.1 Pros

1. Better suitable for fixed or uni-schema message processing or when message schema varies in a predictable manner.
2. All the messages are in one data store, makes it easier to generate the data sets as we don't depend on other data stores.
3. Increase the flexibility for analyzing the data as you will all the attributes being store in a normalized manner in the data store.
4. Cost effective as a message is ingested, processed and stored only once (actually twice, one in the Channel and one in the data store).

2.1.2 Cons

1. Cannot handle messages with different schemas.
2. Activities in the Pipeline can become very complex.

2.2 Type 2 Data flow: Here, we see multiple Pipelines, each sourcing data from their respective Channel and then one data store for storing the transformed data.



For the type of use case, where we want to perform analysis on different attributes originating from different sources and in different schemas, we can design dedicated pipelines for transformation of multischema messages. This kind of design allows us to store relevant data in one data store.

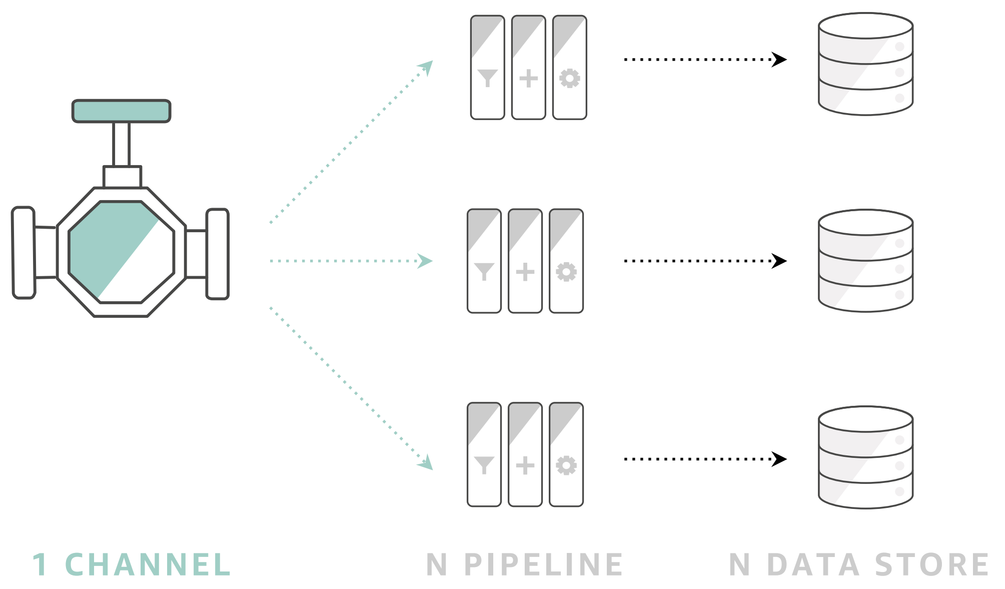
2.2.1 Pros

1. Makes processing of multi schema messages a lot easier.
2. Easy to limit functionalities of the Pipelines for specific types of transformation.
3. Easy to generate data sets from one data store, as we don’t have ate worry about joining multiple data stores.
4. Each message goes through the Pipeline only once, thus saving us the cost of replication at the Channel level.

2.1.2 Cons

1. Data store may grow quickly as multiple Pipelines are feeding data into it.
2. Requires queries to scan lots of data for creating data sets. This might increase the cost of creating data sets as the data store grows.

2.3 Type 3 Data flow: Here, we see multiple Pipelines, each sourcing data from a single Channel and then multiple data stores for storing the transformed data.



This type of configuration replicates the messages at the channel output. Each pipeline is presented with the same raw message as received by the channel. Thus, a channel over here acts as a broadcaster device.

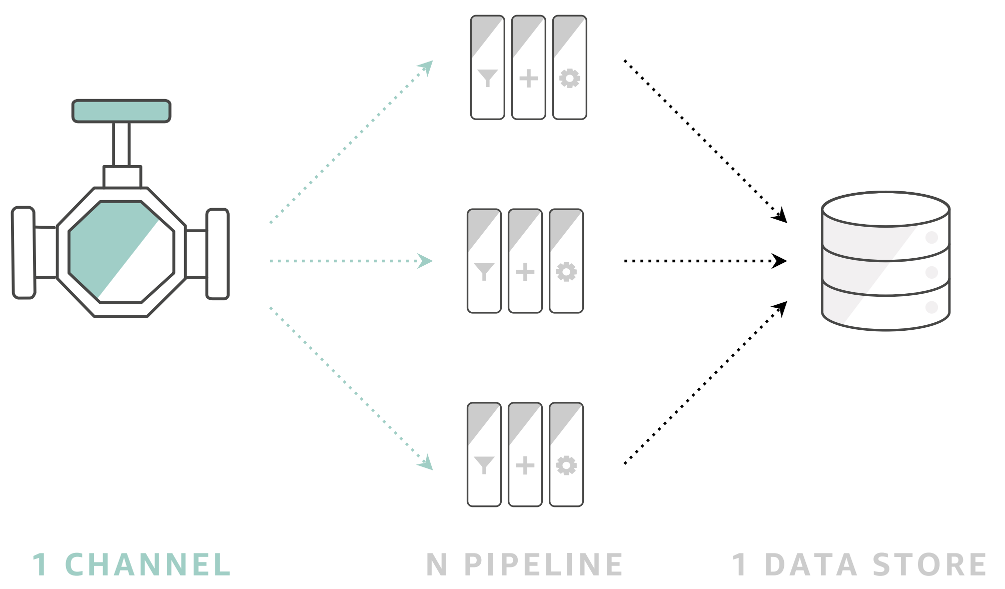
2.3.1 Pros

1. Can perform the multiple and independent transformation on the same message.
2. Individual data store size doesn't grow as quickly when compared to other data flows.
3. Query for generating a data set doesn't have to scan lots of data.

2.3.2 Cons

1. It's important to consider cost as this kind of configuration can become expensive because the devices are emitting billions of messages as they will get replicated among all the pipelines. An alternative solution for such a data flow, would be to create multiple channels with each channel configured to receive specific types of messages from IoT Rule engine. In this case, the solution would just look like repeating Type 1 solution multiple times.

2.4 Type 4 Data flow: Here, we see multiple Pipelines, each sourcing data from a single Channel and then one data store for storing the transformed data.



Like in previous data flow types, the messages are replicated at a channel level. Each pipeline is presented with the same raw message as received by the channel.

2.4.1 Pros

1. Can perform the multiple and independent transformation on the same message.
2. The transformed messages ends up in one data store, makes it easier for generating multiple data sets.

2.4.2 Cons

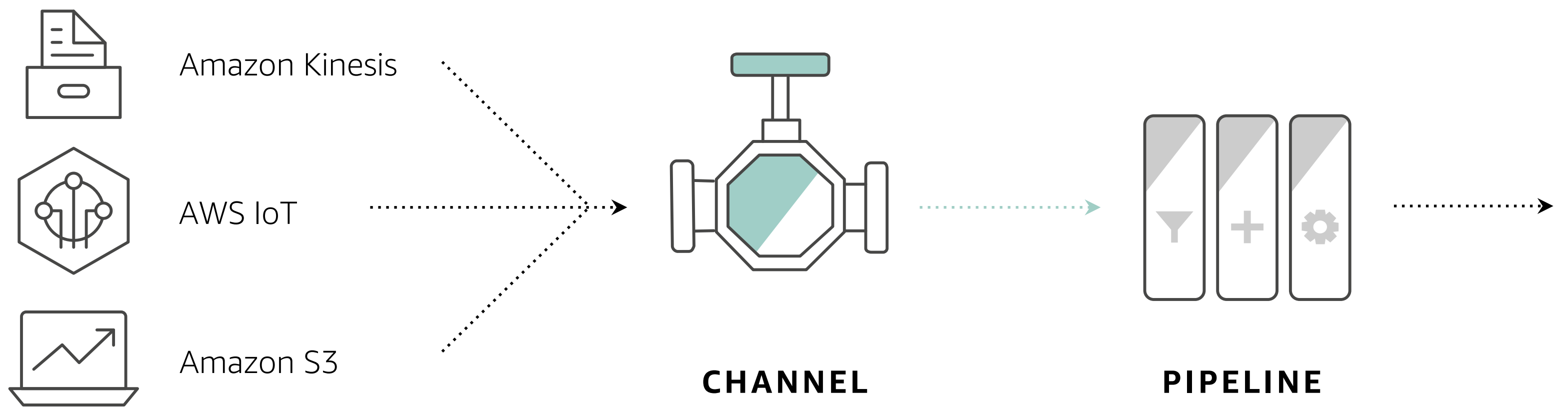
1. One must keep the cost in the mind as such kind of configuration can become costly, if the devices are emitting billions of messages as they will get replicated among all the pipelines. An alternative solution for such data flow would be to create multiple channels and each channel can be configured to receive specific types of messages from IoT Rule engine. In such a case, the solution would look like repeating Type 1 data flow multiple times.

Once you've decided on the optimal data flow type, you can define an AWS IoT Analytics Channel and select the data you want to collect so you only store and analyze the data of interest.

Once the Channel is set up, you configure AWS IoT Analytics Pipelines to process your data. Pipelines support transformations like Celsius to Fahrenheit conversion, conditional statements, message filtering, and message enrichment using external data sources and AWS Lambda functions. After processing the data in the Pipeline, AWS IoT Analytics stores it in an IoT-optimized data store for analysis. You can query the data using the built-in IoT Analytics SQL query engine to answer specific business questions. AWS IoT Analytics also provides endpoints for collecting message data stored in other sources like Amazon Kinesis and S3.

Collect only the data you want to store and analyze:

+ Use the AWS IoT Analytics console to configure AWS IoT Analytics . Receive messages from devices through MQTT topic filters in various formats and frequencies.
+ IoT Analytics validates that the data is within specific parameters you define and then creates Channels.
+ The service routes the Channels to appropriate Pipelines for message processing, transformation and enrichment.



How to build a simple Channel

A Channel is the bridge between a downstream ingestion service, like AWS IoT core, and the rest of AWS IoT Analytics. To get started with a Channel, you will create a Channel and then will create an AWS IoT rule that uses the Channel as the rule action.

Step 1: Create an AWS IoT Analytics Channel using the AWS CLI

```
> aws iot create-Channel --Channel-name myfirstChannel
```

Step 2: Copy the ChannelArn that is returned from the CLI

```
{
    "ChannelArn":"arn:aws:iot:us-east-1:0123456789:Channel/myfirstChannel"
    "ChannelName":"myfirstChannel"
}
```

Step 3: Create a json file containing the following json payload and replace the <Channel_ARN> value with the ChannelArn that you copied in the previous step

```
{
    "sql": "SELECT * FROM 'hello/world'",
     "ruleDisabled":false,
     "awsIotSqlVersion":"2016-03-23",
     "actions": [ {
          "iotAnalytics": {
               "ChannelArn":"<Channel_ARN"
          }
     }]
}
```

Step 4: Use the CLI to create an AWS IoT Rule using your rule.json file

```
> aws iot create-topic-rule --rule-name firstChannelRule --topic-rule-payload
file://rule.json
```

How to use Channels

Now that you've configured your Channel and associated it to an AWS IoT rule, the Channel is ready to start receiving messages from AWS IoT. In order to leverage the full value of an AWS IoT Channel, you will need to create a data store and sample Pipeline to complete the end to end experience. The Channel uses these other features of AWS IoT Analytics for message processing and visualization.

Channels are an important part of AWS IoT Analytics. The first feature required when building an IoT application is the streaming layer that separates the connectivity aspects of IoT from the analytics aspects.
The Channel allows you to easily configure the start of your analytics Pipeline.

In addition, the ability to configure a specific IoT rule to a Channel gives you the flexibility to choose which types of messages should be processed through AWS IoT Analytics.

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 Channels with AWS IoT Analytics in minutes:

aws.amazon.com/
iot-analytics