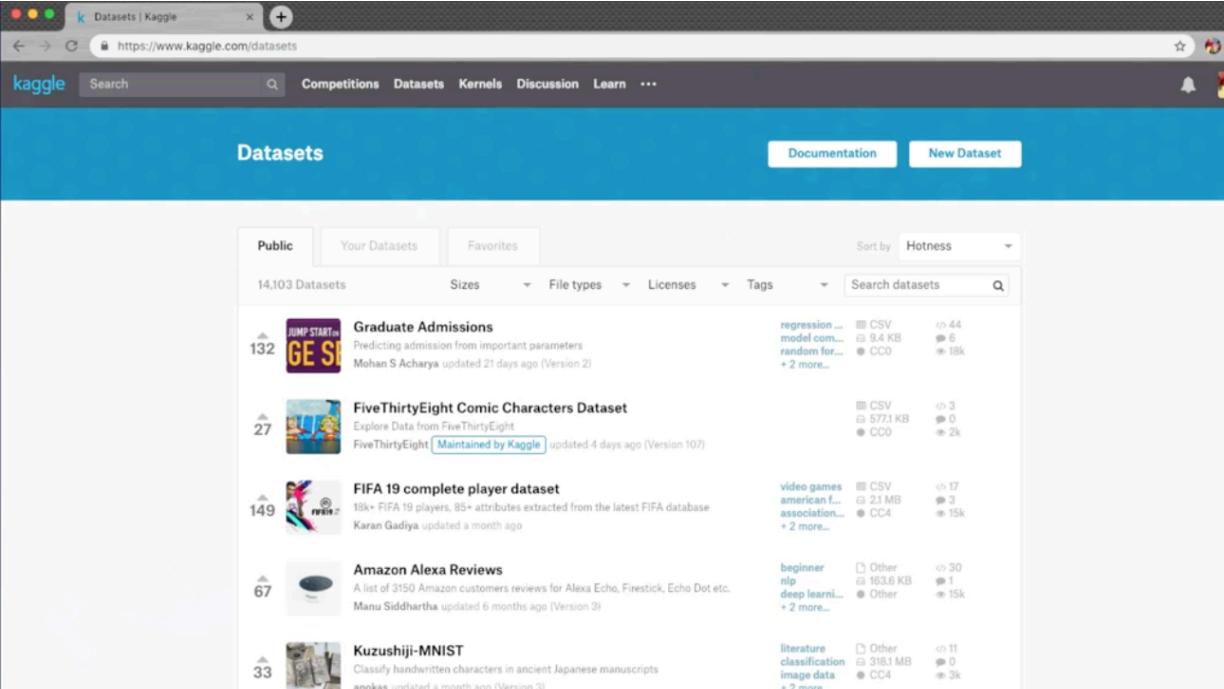


Cloud Guru - 2 - Streaming Data Collection

https://acloud.guru/course/aws-certified-machine-learning-specialty/learn/c756bf92-fc24-f757-1b0d-5072064ea51a/ae135d9e-fd47-7028-b748-677d20da6b23/watch?_ga=2.65864555.490643139.1596727443-133961477.1588770985

Datasets we can use for ML

Kaggle - online platform that offers datasets and set competitions



The screenshot shows the Kaggle Datasets page. At the top, there are tabs for 'Public', 'Your Datasets', and 'Favorites'. Below this, a search bar and filter options for 'Sizes', 'File types', 'Licenses', and 'Tags' are visible. The main area displays a list of datasets:

- Graduate Admissions** (132 datasets): Predicting admission from important parameters. Mohan S Acharya updated 21 days ago (Version 2). Tags: regression, CSV, model com..., random for..., CC0. 44 rows, 6 columns, 18k rows.
- FiveThirtyEight Comic Characters Dataset** (27 datasets): Explore Data from FiveThirtyEight. FiveThirtyEight Maintained by Kaggle updated 4 days ago (Version 107). Tags: CSV, 573.1 KB, CC0. 3 rows, 2 columns, 2k rows.
- FIFA 19 complete player dataset** (149 datasets): 18k+ FIFA 19 players, 85+ attributes extracted from the latest FIFA database. Karan Gadiya updated a month ago. Tags: video games, american f..., association..., CC4. 17 rows, 3 columns, 15k rows.
- Amazon Alexa Reviews** (67 datasets): A list of 3150 Amazon customers reviews for Alexa Echo, Firestick, Echo Dot etc. Manu Siddhartha updated 6 months ago (Version 3). Tags: beginner, nlp, deep learni..., Other. 30 rows, 1 column, 15k rows.
- Kuzushiji-MNIST** (33 datasets): Classify handwritten characters in ancient Japanese manuscripts. anokas updated a month ago (Version 3). Tags: literature classification, image data, CC4. 11 rows, 0 columns, 3k rows.

Other website with open source datasets: UCI

UCI Machine Learning Repository mlr.cs.umass.edu/ml/datasets.html

Machine Learning Repository
Center for Machine Learning and Intelligent Systems

Browse Through: 22 Data Sets

Table View List View

Name	Data Types	Default Task	Attribute Types	# Instances	# Attributes	Year
Abalone	Multivariate	Classification	Categorical, Integer, Real	4177	8	1995
Adult	Multivariate	Classification	Categorical, Integer	48842	14	1996
Annealing	Multivariate	Classification	Categorical, Integer, Real	798	38	
Anonymous Microsoft Web Data		Recommender-Systems	Categorical	37711	294	1998
Arrhythmia	Multivariate	Classification	Categorical, Integer, Real	452	279	1998

Default Task
 Classification (19)
 Regression (3)
 Clustering (0)
 Other (1)

Attribute Type
 Categorical (8)
 Numerical (3)
 Mixed (10)

Data Type
 Multivariate (20)
 Univariate (1)
 Sequential (0)
 Time-Series (0)
 Text (1)
 Domain-Theory (0)
 Other (2)

Area
 Life Sciences (8)
 Physical Sciences (1)

AWS also allows for Open Data

Registry of Open Data on AWS <https://registry.opendata.aws>

Registry of Open Data on AWS

About
 This registry exists to help people discover and share datasets that are available via AWS resources. [Learn more about sharing data on AWS](#). See all usage examples for datasets listed in this registry.

Search datasets (currently 88 matching datasets)

Add to this registry
 If you want to add a dataset or example of how to use a dataset to this registry, please follow the instructions on the [Registry of Open Data on AWS GitHub repository](#).

Unless specifically stated in the applicable dataset documentation, datasets available through the Registry of Open Data on AWS are not provided and maintained by AWS. Datasets are provided and maintained by a variety of third parties under a variety of licenses. Please check dataset licenses and related documentation to determine if a dataset may be used for your application.

Sentinel-2
[earth observation](#) [satellite imagery](#) [glue](#) [natural resource](#) [sustainability](#) [disaster response](#)
 The Sentinel-2 mission is a land monitoring constellation of two satellites that provide high resolution optical imagery and provide continuity for the current SPOT and Landsat missions. The mission provides a global coverage of the Earth's land surface every 5 days, making the data of great use in on-going studies. L1C data are available from June 2015 globally. L2A data are available from April 2017 over wider Europe region and globally since December 2018.
[Details →](#)

Usage examples

- Sterling Geo Using Sentinel-2 on Amazon Web Services to Create NDVI by Sterling Geo
- Satellite Search by Remote Pixel by Remote Pixel
- Using Vector tiles and AWS Lambda, we can build a really simple API to get Landsat and Sentinel Images by Remote Pixel
- Tutorial for using Sentinel-2 data by Antti Lipponen
- Spectator - tracking Sentinel 2, accessing the data and quick preview by Spectator

[See 16 usage examples →](#)

Landsat 8
[earth observation](#) [satellite imagery](#) [glue](#) [natural resource](#) [sustainability](#) [disaster response](#)
 An ongoing collection of satellite imagery of all land on Earth produced by the Landsat 8 satellite.
[Details →](#)

Usage examples

Same with Google Big Queries

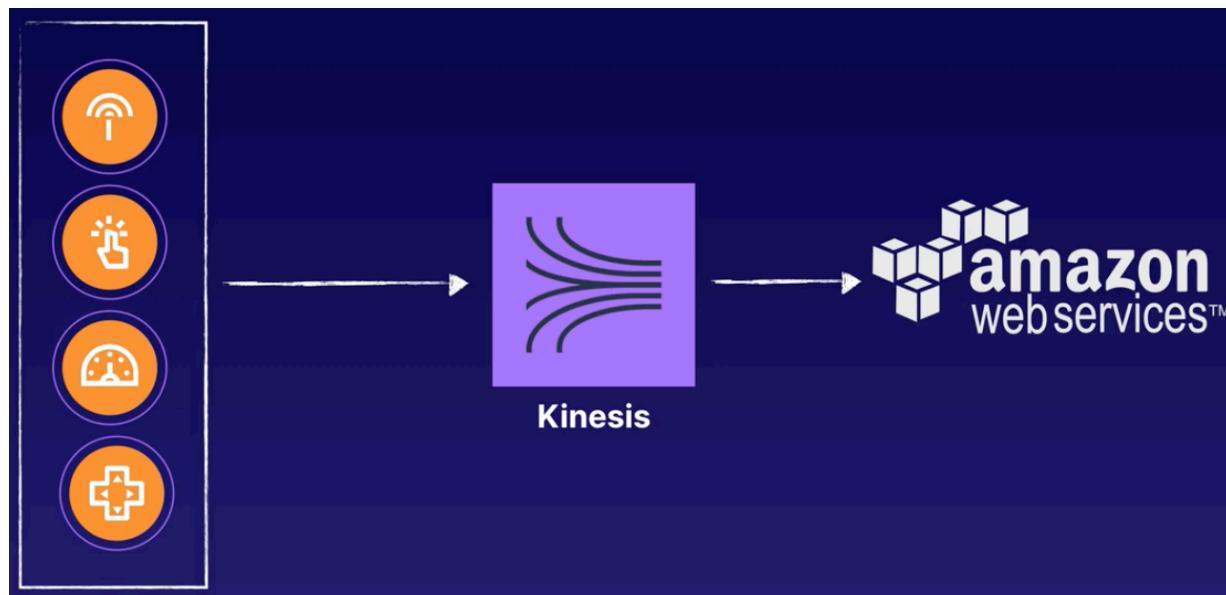
The screenshot shows the Google Cloud Platform BigQuery interface. On the left, there's a sidebar with navigation links like 'Query history', 'Saved queries', 'Job history', 'Transfers', 'Resources', and a search bar. The main area has tabs for 'Unsaved query' and 'Edited'. An unsaved query is shown in the editor:

```
1 SELECT * FROM `fh-bigquery.reddit_posts.2017_09` WHERE subreddit IN ('aws', 'googlecloud', 'AZURE');
```

Below the editor are buttons for 'Run', 'Save query', 'Save view', and 'More'. A message indicates the query will process 4.72 GB when run. The 'Query history' panel shows a single entry for 11/21/18, which was 'Query canceled' at 7:48 PM. The query and job details are identical to the one in the editor.

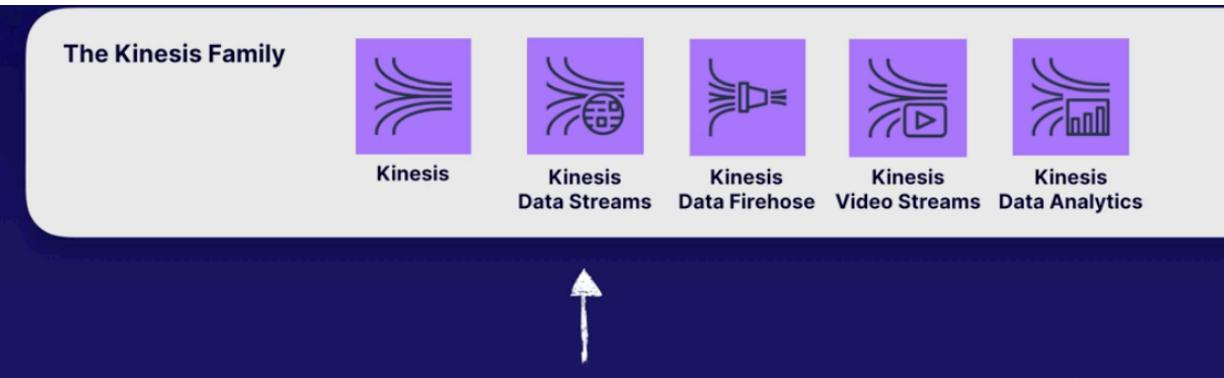
Example of Streaming data: sensor, industrial equipments, financial stock market, real estate websites to make real time recommendations, click stream data for recommendation, online gaming company about game player interaction and feedback back to platform...

How to handle streaming data? Via a family of tools: Kinesis





Kinesis Data Streams



Kinesis Data Streams get data from data producers: typically have JSON data as payload, or any type of data into a data blob

Data producers can be application log files, social media streams, real time user interaction from video games, IoT devices, click stream, user interaction for online website/app,...

Once data is produced, we use Kinesis Data Streams to transfer/load/stream that data into AWS

It leverages Shards - a container that holds the data that we want to send to AWS

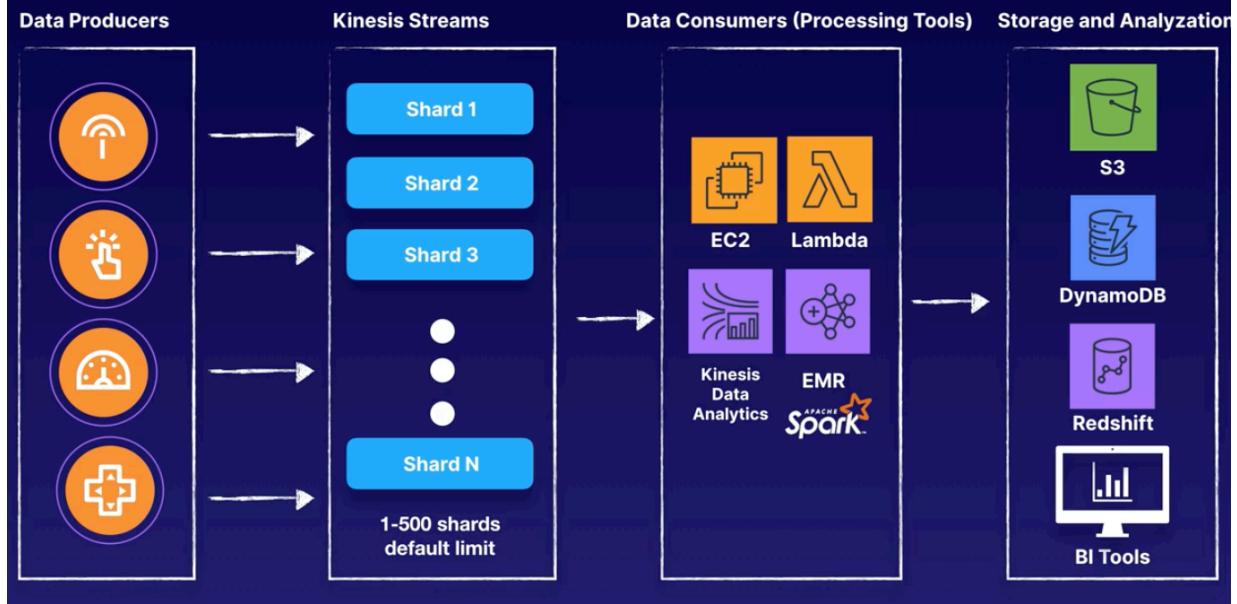
Once data is contained within a shard, we use data consumers to process or analyze that data => EC2 instances, lambda function with real time ETL process, We can also use Kinesis Data Analytics to run real time SQL queries on our streaming data.

Or we can use streaming data to an EMR cluster and process it with Spark.

To note we can have multiple consumers consume these shards and do multiple processes/analysis/...

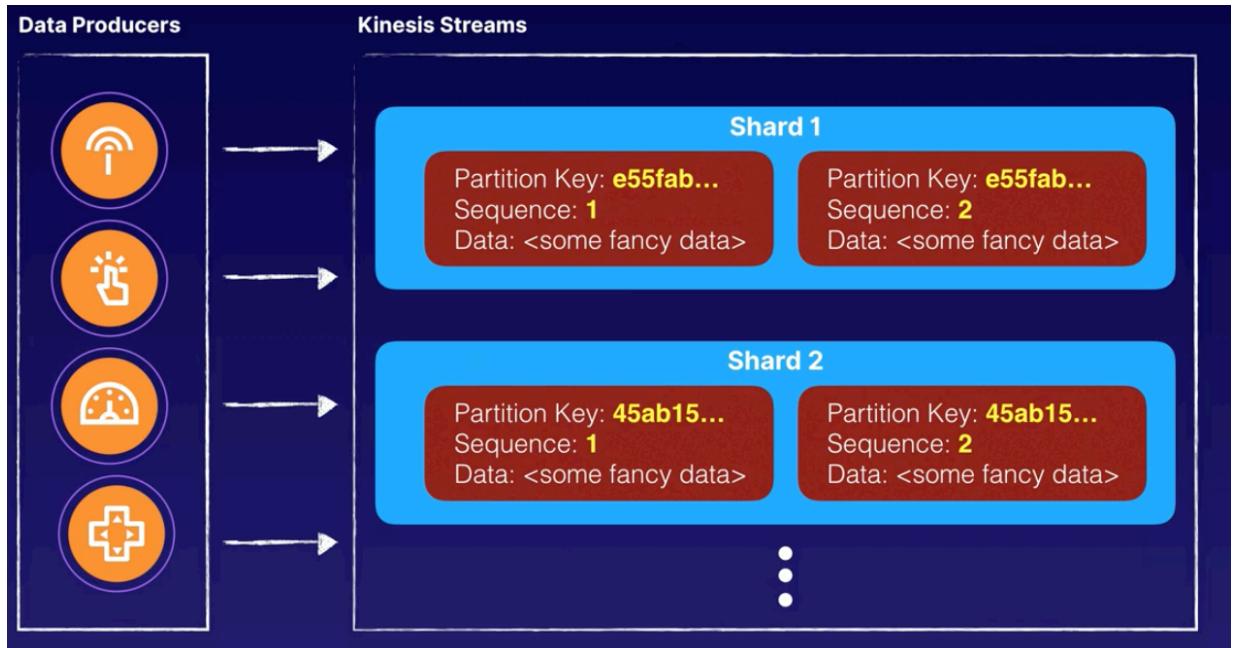
Kinesis Data Streams

A CLOUD GURU



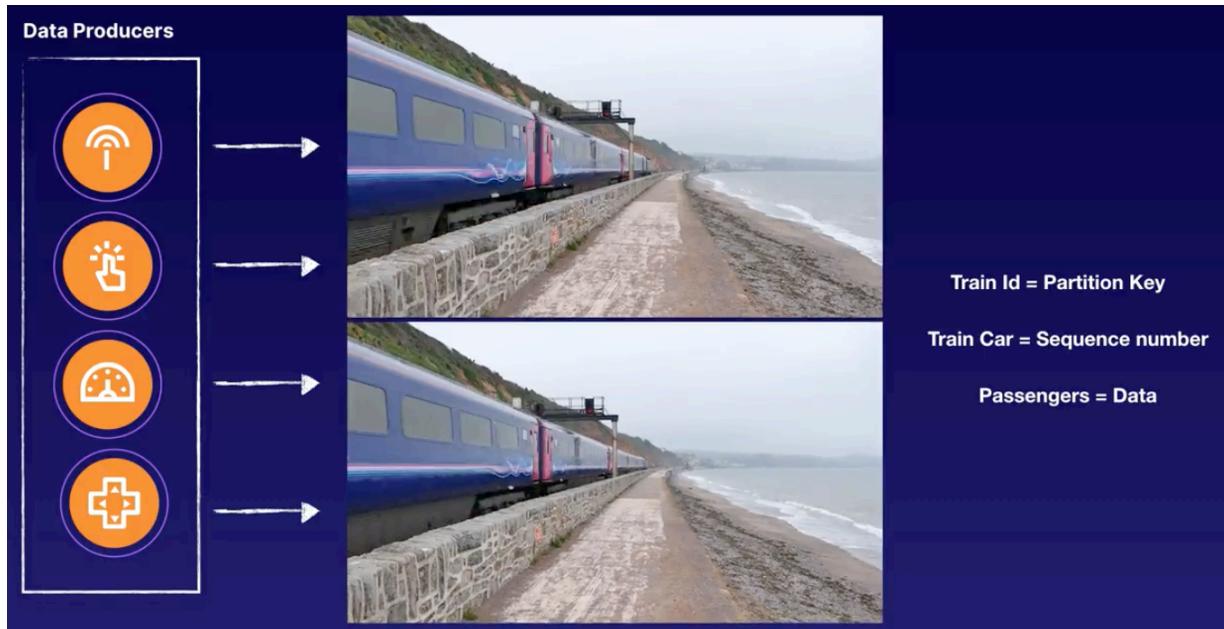
A **shard** is made up of a couple of parts:

- unique **partition key** per shard
- **sequence** / associated with a shard.



The more requests we make and the larger the payload will determine the # of shards

Analogy for Kinesis Data Stream shard/partition key => train



The train ID is the **partition key**

Each car on the train is the **sequence #**

The passengers in each car is the **data**

Each shard consists of data records. Default limit of shards is 500 but can go beyond,. Data can be ingested at 1000 records / second

Transient data store - data held for 24 hours by default but can go up to 7 days

- Each shard consists of a sequence of data records. These can be ingested at 1000 records per second.
- Default limit of 500 shards, but you can request increase to unlimited shards.
- A data record is the unit of data captured.
 - sequence number
 - partition key
 - data blob (your payload, up to 1 MB)
- Transient Data Store - retention period for the data records are 24 hours to 7 days.

Interacting with Kinesis Data Streams via:

- KPL: Kinesis Producer Library to write. Can install it onto EC2 instance or in our

Java app. Handles automatically retries, aggregate of records for optimization

- KCL: Kinesis Client Library: to interact with KPL to consume and process the data
- Kinesis API (SDK): more for low level interactions

1 Kinesis Producer Library (KPL) Easy to use library that allows you to write to a Kinesis Data Stream.	2 Kinesis Client Library (KCL) Integrated directly with KPL for consumer applications to consume and process data from Kinesis Data Stream.	3 Kinesis API (AWS SDK) Used for low level API operations to send records to a Kinesis Data Stream.
Kinesis Producer Library (KPL)		<ul style="list-style-type: none">• Provides a layer of abstraction specifically for ingesting data• Automatic and configurable retry mechanism• Additional processing delay can occur for higher packing efficiencies and better performance• Java wrapper
Kinesis API		<ul style="list-style-type: none">• Low-level API calls (<code>PutRecords</code> and <code>GetRecords</code>)• Stream creations, resharding, and putting and getting records are manually handled• No delays in processing• Any AWS SDK

If need data stream immediately avail within ms => use Kinesis API

If app is written in other language than Java => use Kinesis API

Console

Amazon Kinesis

Create Kinesis stream

Kinesis stream name*

Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.

Shards

A shard is a unit of throughput capacity. Each shard ingests up to 1MB/sec and 1000 records/sec, and emits up to 2MB/sec. To accommodate for higher or lower throughput, the number of shards can be modified after the Kinesis stream is created using the API. [Learn more](#)

▶ Estimate the number of shards you'll need

Number of shards*

You can provision up to 498 more shards before hitting your account limit of 500.
[Learn more or request a shard limit increase for this account](#)

Total stream capacity Values are calculated based on the number of shards entered above.

Write MB per second

Records per second

Read MB per second

A shard is a unit of throughput capacity. Each shard ingests up to 1 MB/sec and 1000 records/sec, and emits up to 2 MB / sec.

To accommodate for higher/lower throughput, adjust shard #

When to use Kinesis Data Streams?

When should you use Kinesis Data Streams?

- Needs to be processed by consumers.
- Real time analytics.
- Feed into other services in real time.
- Some action needs to occur on your data.
- Storing data is optional.
- Data retention is important.

If data is important, data retention is built in Kinesis and we can hold onto it for 24 hours (and up to 7 days). Can allow us to reprocess data in a stream

Good options for Kinesis Data Streams:

- **Process and evaluate logs immediately**

Example: Analyze system and application logs continuously and process within seconds.

- **Real-time data analytics**

Example: Run real-time analytics on click stream data and process it within seconds.

Kinesis Firehose:

Easily streams data



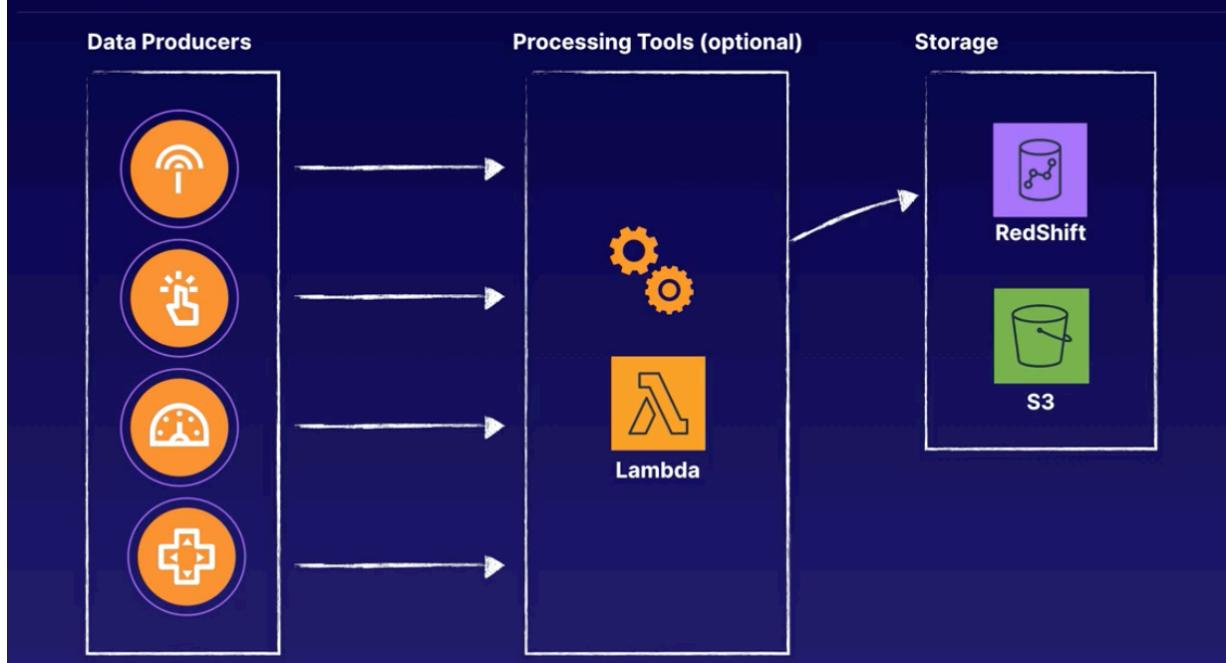
No longer have to worry about shards (as with Kinesis Data Streams)

We can preprocess the data with Lambda, optionally, before we land the data in S3, Redshift, splunk and Amazon Elastic Serach

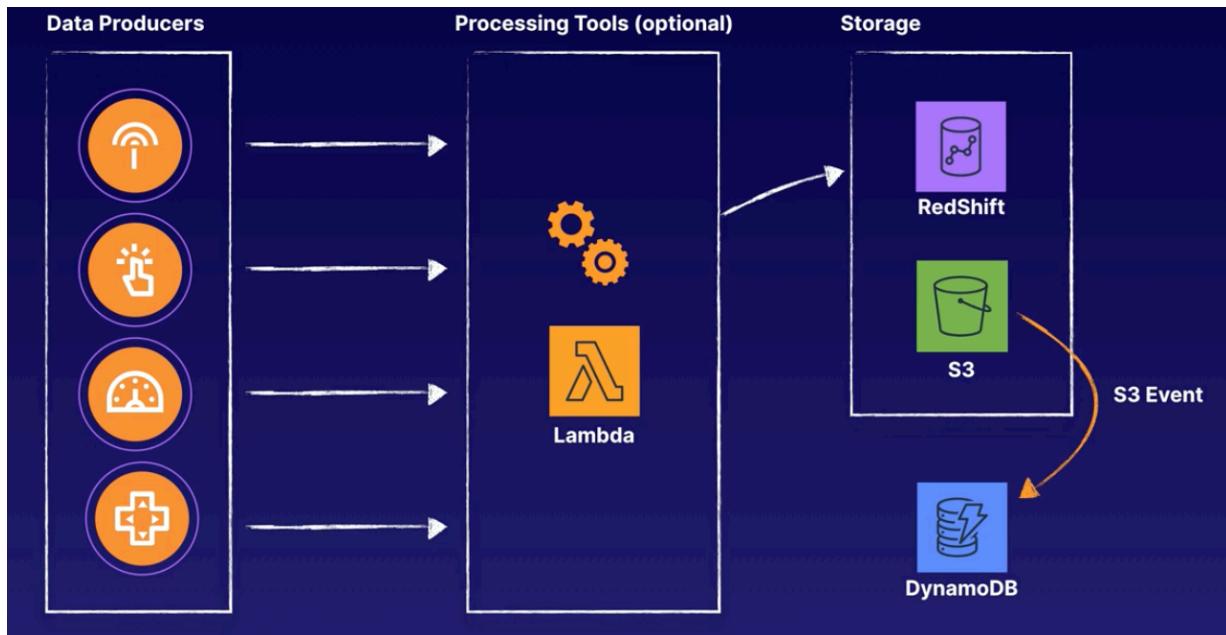
We could also stream from Data Firehose to storage directly

Kinesis Data Firehose

AC



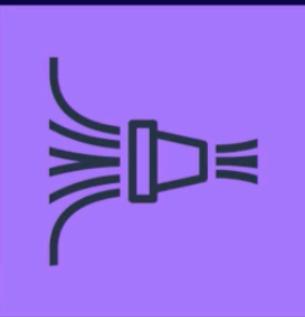
We can also use S3 event to push data to Dynamo DB



Difference between Kinesis Data Streams and Data Firehose:

- Streams needs shards and has data retention (24 hours default, up to 7 days)
- Firehose: no need to worry about shards. Used for streaming straight the data to repository like S3 (or with light processing with lambda). However no data retention. If we lose some data, not the end of the world

When to use Kinesis Data Firehose?



Kinesis Data Firehose

When should you use Kinesis Data Firehose?

- Easily collect streaming data.
- Processing is optional.
- Final destination is S3 (or other data store).
- Data retention is not important.

Use cases for Data Firehose:

- **Stream and store data from devices**

Example: Capturing important data from IoT devices, embedded systems, consumer applications and storing it into a data lake.

- **Create ETL jobs on streaming data**

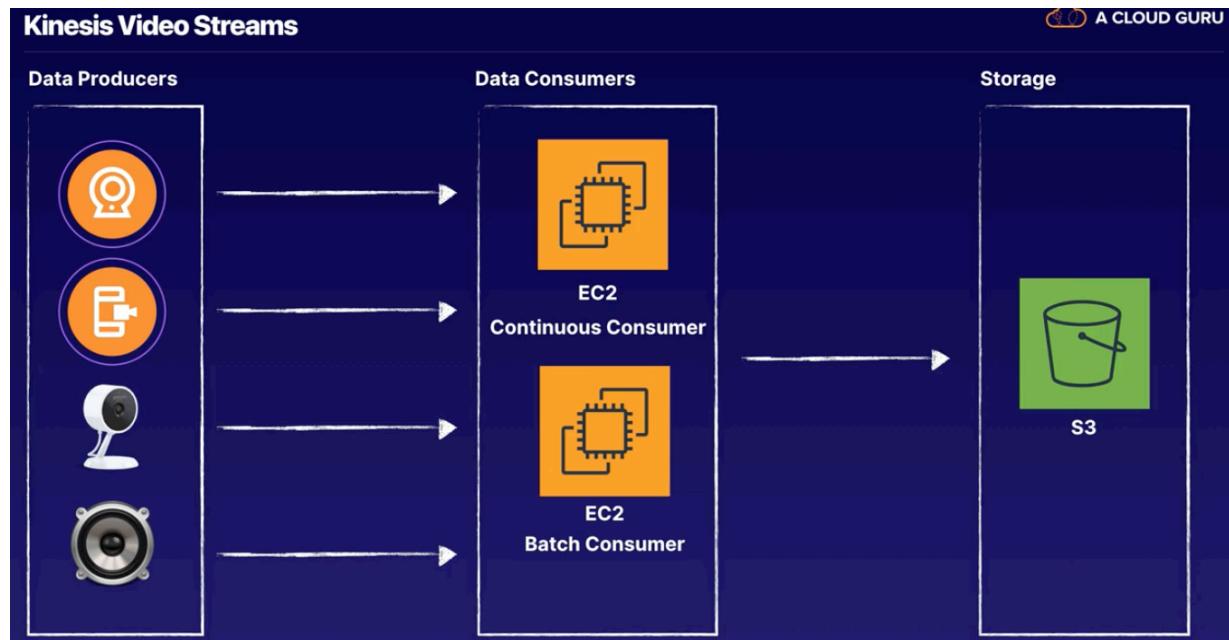
Example: Running ETL jobs on streaming data before data is stored into a data warehousing solution.

Kinesis Video Stream



To build real time processing application on Video data or Audio files
Data producers: web cams, security cameras, audio feeds, radar data,...

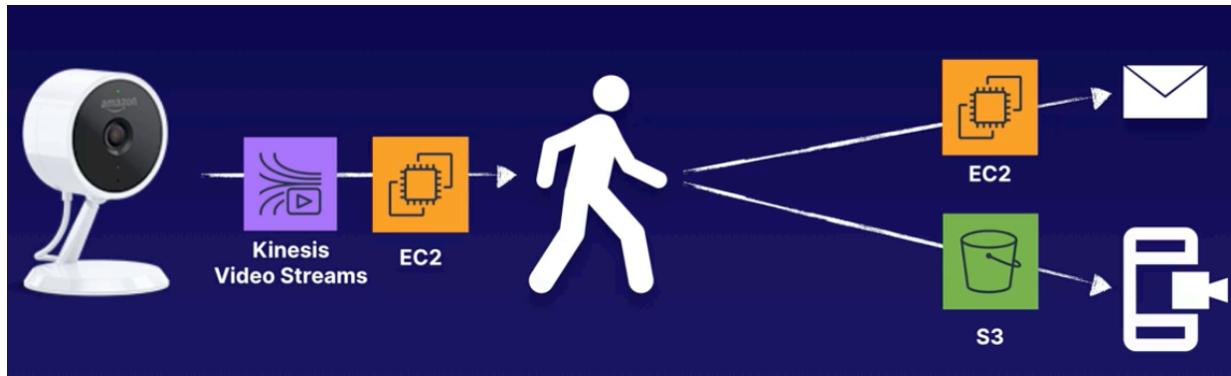
Can consume that data in real time or in batch



When to use Kinesis Video Streams?



Amazon Cloud Cam example:
Can detect people, dogs barking,...



Kinesis Data Analytics

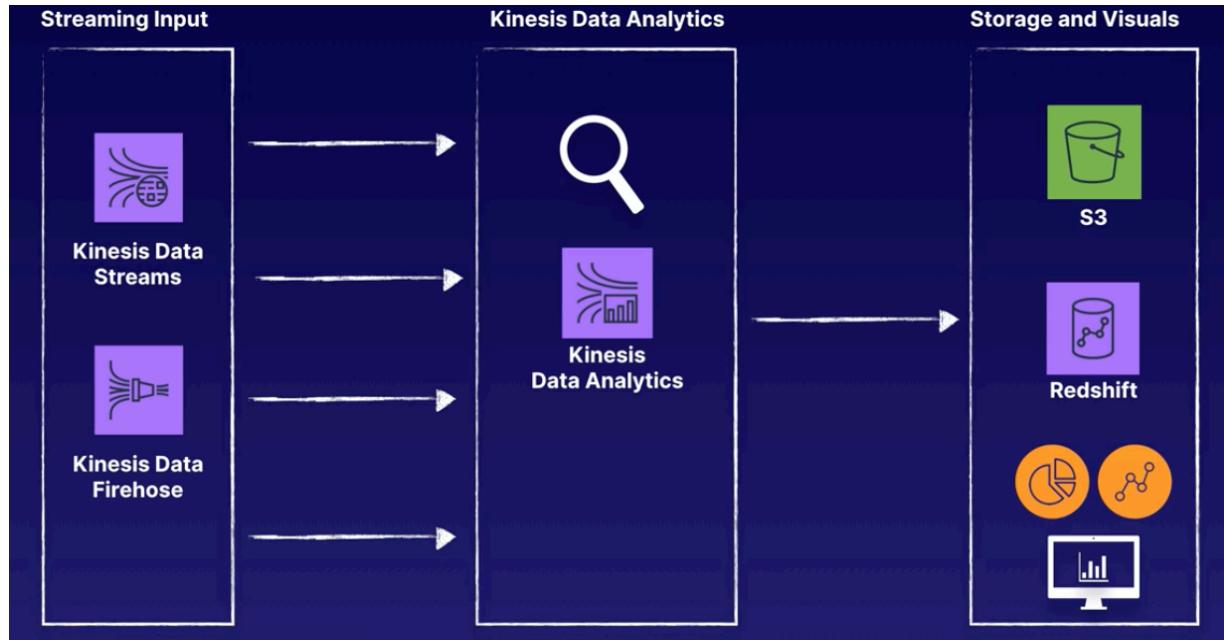


Last service in Kinesis family: Kinesis Data analytics

Pretty powerful tool to continuously read and process streaming data in real time, using SQL queries to process incoming dat and prod output data from that.

It gets streaming data from Kinesis Data streams or Firehose, then run real time SQL queries and output to S3 or Redshift or other visual/BI tools

=> can run complex queries (change data type/column names, complex joins, aggregate,...) in real time and store in S3 for ML process down the road
Allows us to do pre-processing.



The screenshot shows the Amazon Kinesis Data Analytics SQL editor interface. The left sidebar shows the navigation path: Kinesis Analytics applications > website-kinesis-analytics > SQL editor. The main area is titled "Real-time analytics".

Key elements in the interface include:

- A toolbar with buttons for "Save and run SQL", "Add SQL from templates", "Download SQL", "SQL reference guide", and "Kinesis data generator tool".
- A code editor window containing the SQL query:

```
1 SELECT STREAM ROWTIME FROM SOURCE_SQL_STREAM_001;
```
- A status bar at the bottom right indicating "Application status: RUNNING".
- A tab bar with "Source data", "Real-time analytics", and "Destination".
- A "Streaming data" section showing "SOURCE_SQL_STREAM_001" selected.
- A "Reference data (optional)" section with a "Connect reference data" button.
- An "Actions" dropdown menu.
- A table view showing sample data from the stream. The columns are: ROWTIME (TIMESTAMP), type (VARCHAR(16)), x (INTEGER), y (INTEGER), url (VARCHAR(8)), and APPROXIMATE_ARRIVAL_TIMESTAMP. The data rows are:

ROWTIME (TIMESTAMP)	type (VARCHAR(16))	x (INTEGER)	y (INTEGER)	url (VARCHAR(8))	APPROXIMATE_ARRIVAL_TIMESTAMP
2019-04-01 10:38:45.768	mousemove	806	8	/about	2019-04-01 10:38:44.117
2019-04-01 10:38:45.768	mousemove	806	8	/about	2019-04-01 10:38:44.127
2019-04-01 10:38:45.768	mousemove	790	31	/about	2019-04-01 10:38:44.144
2019-04-01 10:38:45.768	mousemove	747	90	/about	2019-04-01 10:38:44.152
2019-04-01 10:38:45.768	mousemove	733	115	/about	2019-04-01 10:38:44.153

When to use Kinesis Data Analytics?

When we need to run SQL queries on real time data

Can be useful for monitoring too



Kinesis Data Analytics

When should you use Kinesis Data Analytics?

- Run SQL queries on streaming data.
- Construct applications that provide insight on your data.
- Create metrics, dashboards, monitoring, notifications, and alarms.
- Output query results into S3 (other AWS datasources).

Use Cases for Kinesis Data Analytics:

- **Responsive real-time analytics**

Example: Send real-time alarms or notifications when certain metrics reach predefined threshold.

- **Stream ETL jobs**

Example: Stream raw sensor data then, clean, enrich, organize, and transform it before it lands into data warehouse or data lake.

Kinesis Family - Use Cases

At exam, we may be asked what Kinesis service to use for a streaming data. Make sure to remember Kinesis Firehose does not have data retention and does not have shards as opposed to Kinesis Data Streams, that uses shards and offers data retention (default 24h, up to 7 days). Kinesis Firehose mainly used to directly output streaming data into some datastore like S3

- 1/ Since there is no preprocessing on this data, we can use Kinesis Firehose to easily stream it to final destination. Once data is in S3, we can copy it over to Redshift.
- 2/ Since we get video data, we will use Kinesis Video Streams
- 3/ Since we need some real time transform, we will use Kinesis Data Streams. It allows us to stream huge amount of data, transform it and feed it to other services

(lambda-> S3, custom apps,...)

4/ To create metric graphs, we will run live SQL queries on the streaming data and hence use Kinesis Data Analytics

The Kinesis Family - Use Cases		
Task at hand	Which Kinesis service to use?	Why?
Need to stream Apache log files directly from (100) EC2 instances and store them into Redshift.	Kinesis Firehose	Firehose is for easily streaming data directly to a final destination. First the data is loaded into S3, then copied into Redshift.
Need to stream live video coverage of a sporting event to distribute to customers in near real-time.	Kinesis Video Streams	Kinesis Video Streams processes real-time streaming video data (audio, images, radar) and can be fed into other AWS services.
Need to transform real-time streaming data and immediately feed into a custom ML application.	Kinesis Streams	Kinesis Streams allows for streaming huge amounts of data, process/transform it, and then store it or feed into custom applications or other AWS services.
Need to query real-time data, create metric graphs, and store output into S3.	Kinesis Analytics	Kinesis Analytics gives you the ability to run SQL queries on streaming data, then store or feed the output into other AWS services.

Exam Tips with Streaming Data

Streaming Data Collection



Loading Data into AWS

- Understand how to get data from public or in house data sets and load it into AWS.
- Know the different ways to upload into S3 by using the console, the S3 API, or the AWS cli.

The Kinesis Family

- Know what each service is and how it processes/handles streaming data.
- Know what shards are, what a data record is, and the retention period for a shard.
- Know the difference in the KPL, KCL, and Kinesis API.
- For a given scenario, know which streaming Kinesis service to use.

Additional contents:

1 - Real Time Machine Learning using Amazon Kinesis (AWS Summit 2018):

<https://www.youtube.com/watch?v=M8jVTI0wHFM>

2 - White Paper- Streaming Data Solutions on AWS with Amazon Kinesis (July 2017): <https://d0.awsstatic.com/whitepapers/whitepaper-streaming-data-solutions-on-aws-with-amazon-kinesis.pdf>

Data engineers, data analysts, and big data developers are looking to evolve their analytics from batch to real-time so their companies can learn about what their customers, applications, and products are doing right now and react promptly. This whitepaper discusses the evolution of analytics from batch to real-time. It describes how services such as Amazon Kinesis Streams, Amazon Kinesis Firehose, and Amazon Kinesis Analytics can be used to implement realtime applications, and provides common design patterns using these services.

Toll Use Case

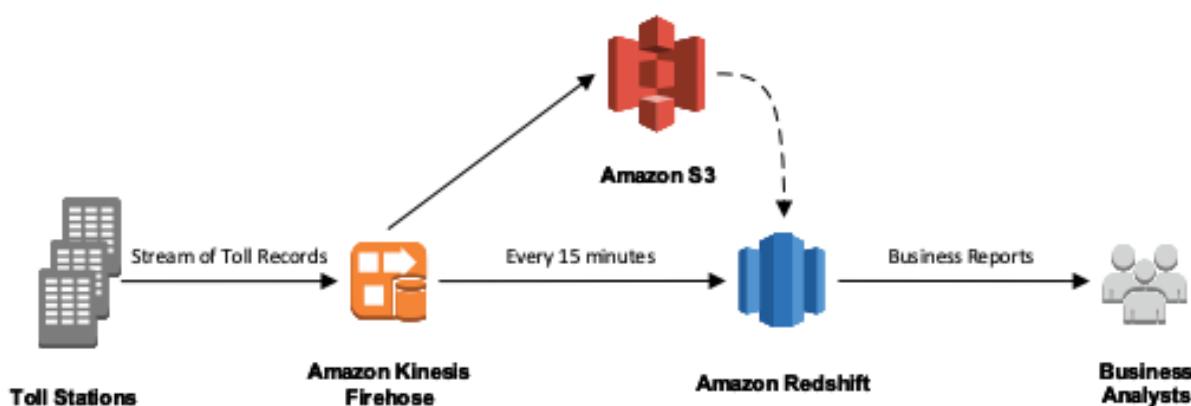


Figure 1: New solution using Amazon Kinesis Firehose

Amazon Kinesis Firehose Amazon Kinesis Firehose is the easiest way to load streaming data into AWS. It can capture, transform, and load streaming data into Amazon Kinesis Analytics, Amazon S3, Amazon Redshift, and Amazon Elasticsearch Service, enabling near real-time analytics with existing business intelligence tools and dashboards that you're already using today. It's a fully managed service that automatically scales to match the throughput of your data and requires no ongoing administration. It can also batch, compress, and encrypt the data before loading it, minimizing the amount of storage used at the destination and increasing security.

Sending Data to an Amazon Kinesis Firehose Delivery Stream To send data to your delivery stream, there are several options. AWS offers SDKs for many popular programming languages, each of which provides APIs for Kinesis Firehose. AWS has also created a utility to help send data to your delivery stream.

Using the API

The Kinesis Firehose API offers two operations for sending data to your delivery stream. PutRecord sends one data record within one call. PutRecordBatch can send multiple data records within one call.

More info at: <https://docs.aws.amazon.com/firehose/latest/dev/writing-with-sdk.html>

Using the Amazon Kinesis Agent

The Amazon Kinesis Agent is a stand-alone **Java** software application that offers an easy way to collect and send data to Kinesis Streams and Kinesis Firehose. The agent continuously monitors a set of files and sends new data to your stream. The agent handles file rotation, checkpointing, and retry upon failures. It delivers all of your data in a reliable, timely, and simple manner. It also emits Amazon CloudWatch metrics to help you better monitor and troubleshoot the streaming process.

Data Transformation

In some scenarios, you might want to transform or enhance your streaming data before it is delivered to its destination.

For example, data producers might send unstructured text in each data record, and you need to transform it to JSON before delivering it to Amazon Elasticsearch Service. To enable streaming data transformations, Kinesis Firehose uses an AWS **Lambda** function that you create to transform your data.

Data Transformation Flow When you enable Kinesis Firehose data transformation, Kinesis Firehose buffers incoming data up to 3 MB or the buffering size you specified for the delivery stream, whichever is smaller. Kinesis Firehose then invokes the specified Lambda function with each buffered batch asynchronously. The transformed data is sent from Lambda to Kinesis Firehose for buffering. Transformed data is delivered to the destination when the specified buffering size or buffering interval is reached, whichever happens first

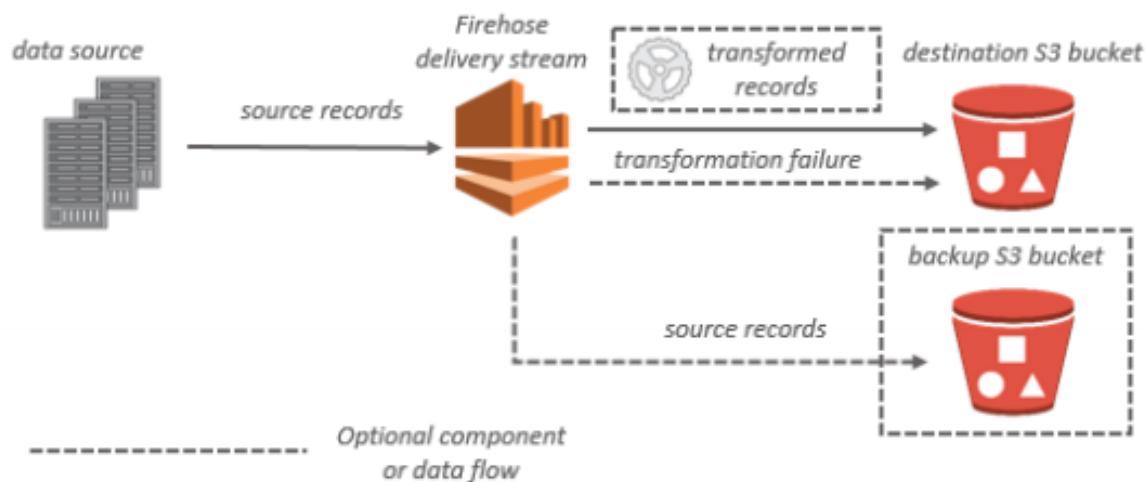
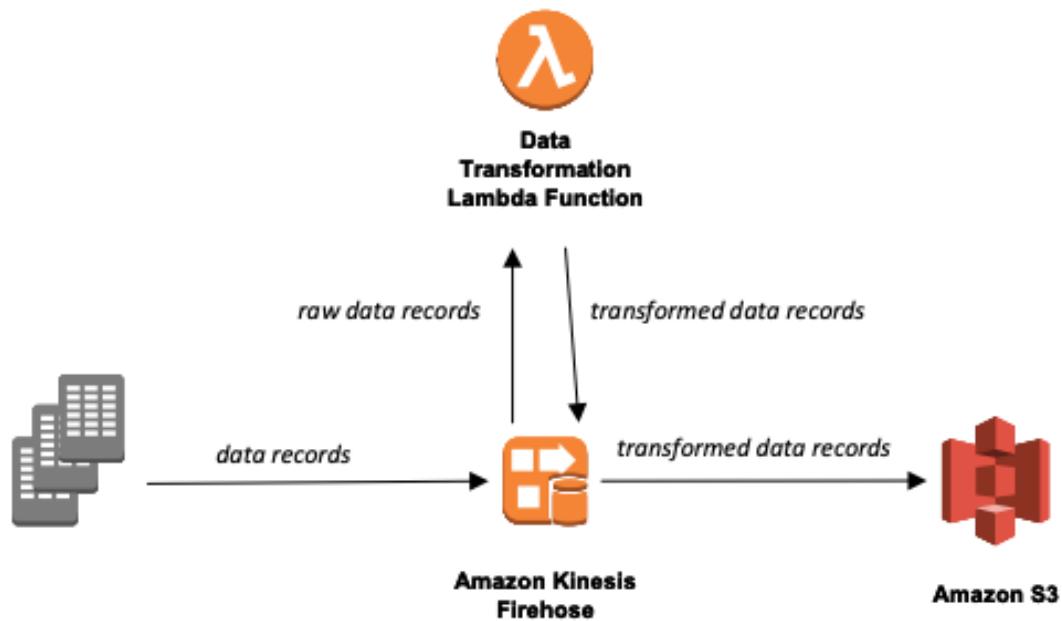


Figure 3: Data flow from Kinesis Firehose to S3 buckets

Amazon Redshift

Amazon Redshift is a fast, fully managed data warehouse that makes it simple and cost-effective to analyze all your data using standard SQL and your existing business intelligence (BI) tools.⁷ It allows you to run complex analytic queries against petabytes of structured data using sophisticated query optimization, columnar storage on high-performance local disks, and massively parallel query execution. Most results come back in seconds.

For data delivery to Amazon Redshift, Kinesis Firehose first delivers incoming data

to your S3 bucket in the format described earlier. Kinesis Firehose then issues an Amazon Redshift COPY command to load the data from your S3 bucket to your Amazon Redshift cluster

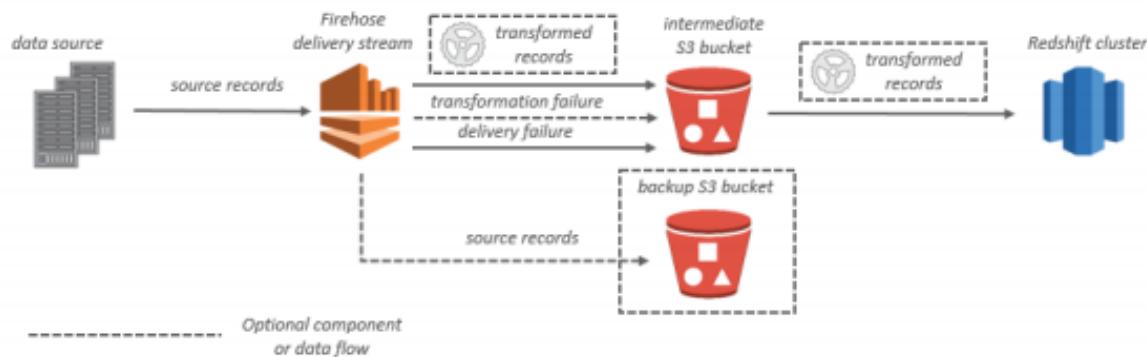


Figure 4: Data flow from Kinesis Firehose to Amazon Redshift

Amazon Elasticsearch Service

Amazon Elasticsearch Service (Amazon ES) is a fully managed service that delivers the Elasticsearch easy-to-use APIs and real-time capabilities along with the availability, scalability, and security required by production workloads.⁸ Amazon ES makes it easy to deploy, operate, and scale Elasticsearch for log analytics, full text search, application monitoring, and more.

For data delivery to Amazon ES, Kinesis Firehose buffers incoming records based on the buffering configuration of your delivery stream and then generates an Elasticsearch bulk request to index multiple records to your Elasticsearch cluster. You need to make sure that your record is UTF-8 encoded and flattened to a single-line JSON object before you send it to Kinesis Firehose.

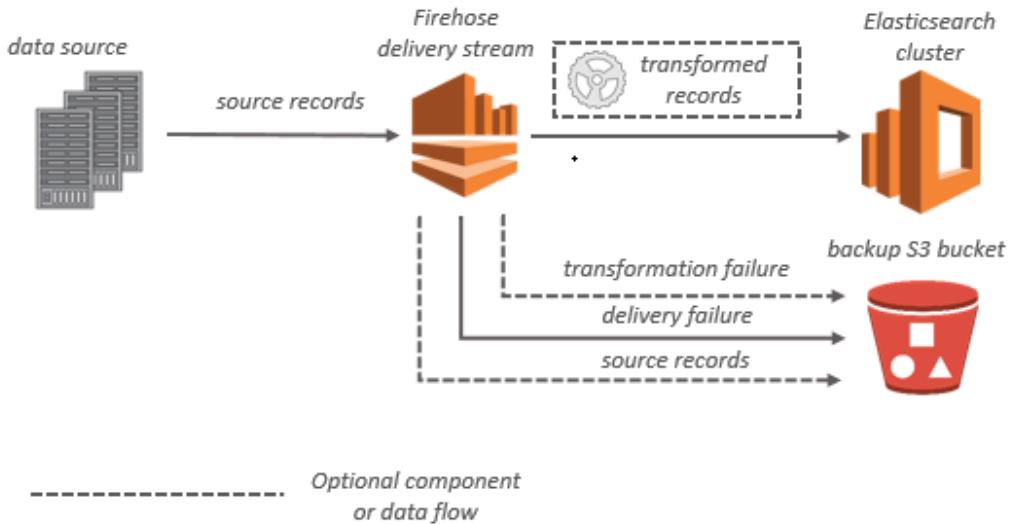


Figure 5: Data delivery from Kinesis Firehose to Amazon ES cluster

Running Analytics

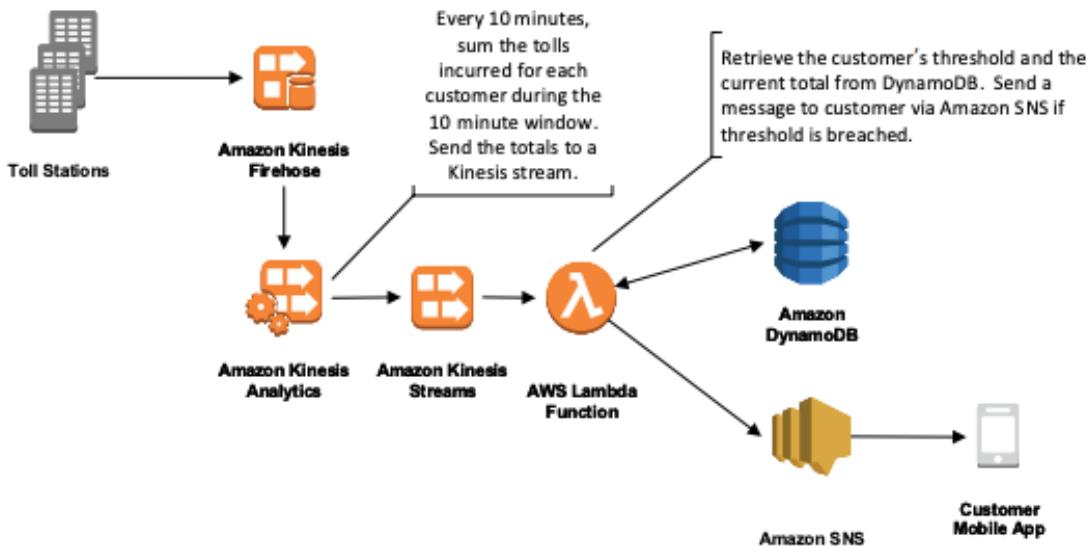


Figure 6: Architecture for billing threshold alerts and notifications

Amazon Kinesis Analytics

With Kinesis Analytics, you can process and analyze streaming data using SQL. The service enables you to quickly author and run powerful SQL code against streaming sources to perform time series analytics, feed real-time dashboards, and create real-time metrics.

Amazon Kinesis Streams

Amazon Kinesis Streams enables you to build custom, real-time applications using popular stream processing frameworks and load streaming data into any data store. You can configure hundreds of thousands of data producers to continuously put data into a Kinesis stream, for example, data from website clickstreams, application logs, IoT sensors, and social media feeds. Within less than a second, the data will be available for your application to read and process from the stream

Sending Data to Amazon Kinesis Streams

There are several mechanisms to send data to your stream. AWS offers SDKs for many popular programming languages, each of which provides APIs for Kinesis Streams. AWS has also created several utilities to help send data to your stream. Let's review each of the approaches you can use and why you might choose each.

Amazon Kinesis Agent The Amazon Kinesis Agent was discussed earlier as a tool that can be used to send data to Kinesis Firehose. The same tool can be used to send data to Kinesis Streams. For details on installing and configuring the Kinesis agent, see Writing to Amazon Kinesis Firehose Using Amazon Kinesis Agent:

<https://docs.aws.amazon.com/firehose/latest/dev/writing-with-agents.html>

Amazon Kinesis Producer Library (KPL)

The KPL simplifies producer application development, allowing developers to achieve high write throughput to one or more Kinesis streams. The KPL is an easy-to-use, highly configurable library that you install on your hosts that generate the data that you wish to stream to Kinesis Streams. It acts as an intermediary between your producer application code and the Kinesis Streams API actions. Because the KPL buffers your records before they're sent to a Kinesis stream, the KPL can incur an additional processing delay

Amazon Kinesis API

After a stream is created, you can add your data records to it. A record is a data structure that contains the data to be processed in the form of a data blob. After you store the data in the record, Kinesis Streams does not inspect, interpret, or change the data in any way. There are two different operations in the Kinesis Streams API that add data to a stream: PutRecords and PutRecord. The PutRecords operation sends multiple records to your stream per HTTP request, and the singular PutRecord operation sends records to your stream one at a time (a separate HTTP request is required for each record). You should prefer using PutRecords for most applications because it will achieve higher throughput per data producer. Because the APIs are exposed in all AWS SDKs, using the API to write records provides the most flexible solution to send data to a Kinesis stream.

Processing Data in Amazon Kinesis Streams

Using Amazon Kinesis Analytics

Earlier, we discussed how Kinesis Analytics can be used to analyze streaming data using standard SQL. Kinesis Analytics can read the data from your Kinesis stream, and process it using the SQL you provide.

Using the Amazon Kinesis Client Library (KCL)

You can develop a consumer application for Kinesis Streams using the KCL.

Although you can use the Kinesis Streams API to get data from an Amazon Kinesis stream, we recommend using the design patterns and code for consumer applications provided by the KCL. The KCL helps you consume and process data from a Kinesis stream. This type of application is also referred to as a consumer. The KCL takes care of many of the complex tasks associated with distributed computing, such as load balancing across multiple instances, responding to instance failures, checkpointing processed records, and reacting to resharding. The KCL enables you to focus on writing record-processing logic. The KCL is a Java library; support for languages other than Java is provided using a multi-language interface.

Using AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers.¹⁶ AWS Lambda executes your code only when needed and scales automatically. With AWS Lambda, you can run code with zero administration. AWS Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, and code monitoring and logging. All you need to do is supply your code in one of the languages that AWS Lambda supports. You can subscribe Lambda functions to automatically read batches of records off your Kinesis stream and process them if records are detected on the stream. Amazon Web Services – Streaming Data Solutions on AWS with Amazon Kinesis Page 21 AWS Lambda then polls the stream periodically (once per second) for new records. When it detects new records, it invokes your Lambda function by passing the new records as a parameter. If no new records are detected, your Lambda function is not invoked.

Using the API

For most use cases, you should use the KCL or AWS Lambda to retrieve and process data from a stream. However, if you prefer to write your own consumer application from scratch, there are several methods that enable this. The Kinesis Streams API provides the GetShardIterator and GetRecords methods to retrieve data from a stream.

Choosing the Best Consumer Model for Your Application

How do you know which consumer model is best for your use case? Each approach has its own set of tradeoffs and you'll need to decide what's important to you. Here is some general guidance to help you choose the correct consumer model.

In most cases, consider starting with **AWS Lambda**. Its ease of use and the simple deployment model will enable you to quickly build a data consumer. The tradeoff to using AWS Lambda is that each invocation of your Lambda function should be considered stateless. That is, you can't easily use results from previous invocations of your function (e.g., earlier batches of records from your stream). Also consider that the maximum execution time for a single Lambda function is 5 minutes. If a single batch of records takes longer than 5 minutes to process, AWS Lambda might not be the best consumer for your use case.

If you decide that you can't use AWS Lambda, consider building your own processing application with the **KCL**. Because you deploy KCL applications to EC2 instances within your AWS account, you have a lot of flexibility and control in the local data persistence and state requirements for your data.

Your third option is to build your own application using the **APIs** directly. This gives you the most control and flexibility, but you also need to build your own logic to handle common consumer application features like checkpointing, scaling, and failover.

Here is more info from AWS console:

Kinesis Data Streams



Collect and store data streams
Collect gigabytes of data per second and make it available for processing and analyzing in real time.

[Create data stream](#)

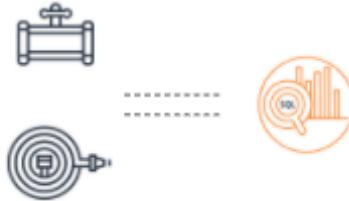
Kinesis Data Firehose



Process and deliver data streams
Prepare and load real-time data streams into data stores and analytics tools.

[Create delivery stream](#)

Kinesis Data Analytics



Analyze streaming data
Get actionable insights from streaming data in real time.

[Create application](#)

Benefits and features

Kinesis Data Streams

Real-time data capture

Ingest and store data streams from hundreds of thousands of data sources:

- Log and event data collection
- IoT device data capture
- Mobile data collection
- Gaming data feed

Kinesis Data Firehose

Load real-time data

Load streaming data into data lakes, data stores, and analytics tools for:

- Log and event analytics
- IoT data analytics
- Clickstream analytics
- Security monitoring

Kinesis Data Analytics

Get insights in real time

Analyze streaming data and gain actionable insights in real time:

- Real-time streaming ETL
- Real-time log analytics
- Ad tech and digital marketing analytics
- Real-time IoT device monitoring