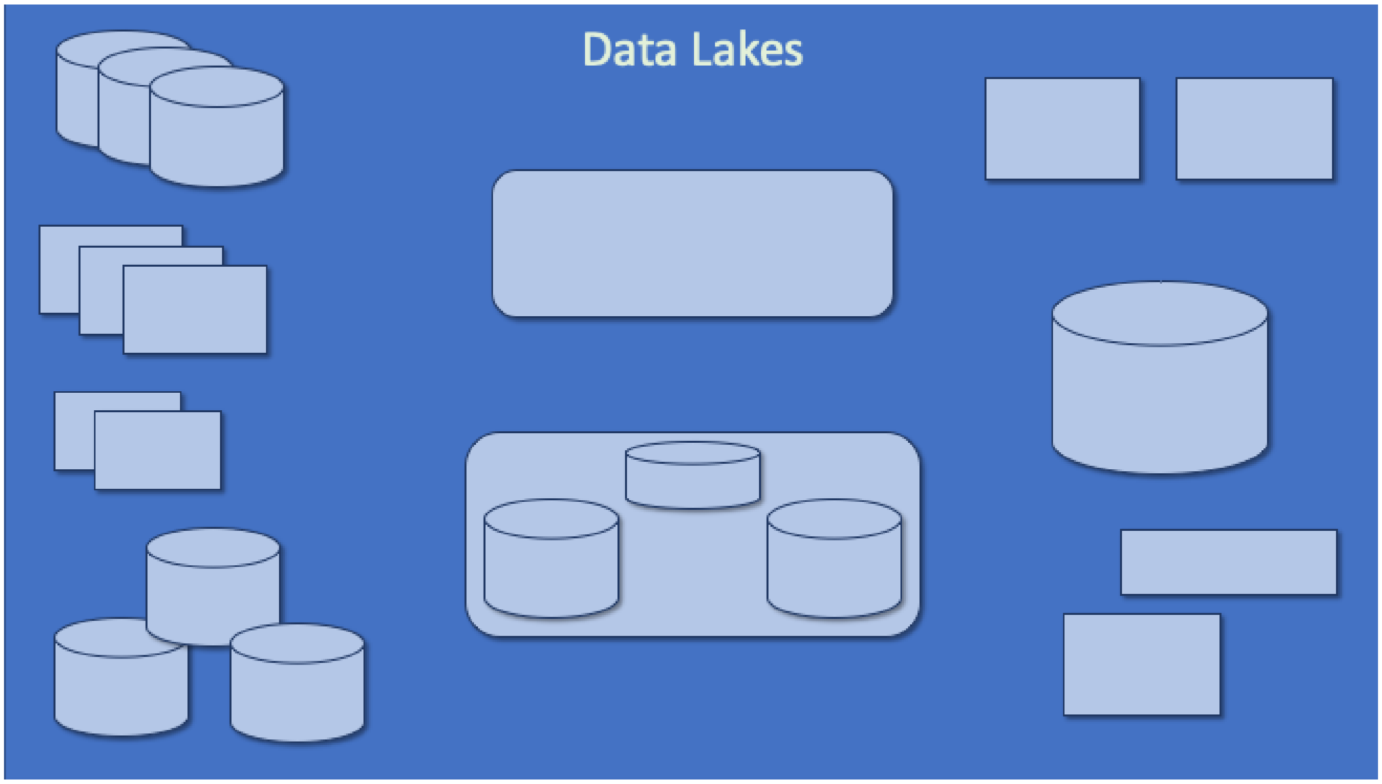
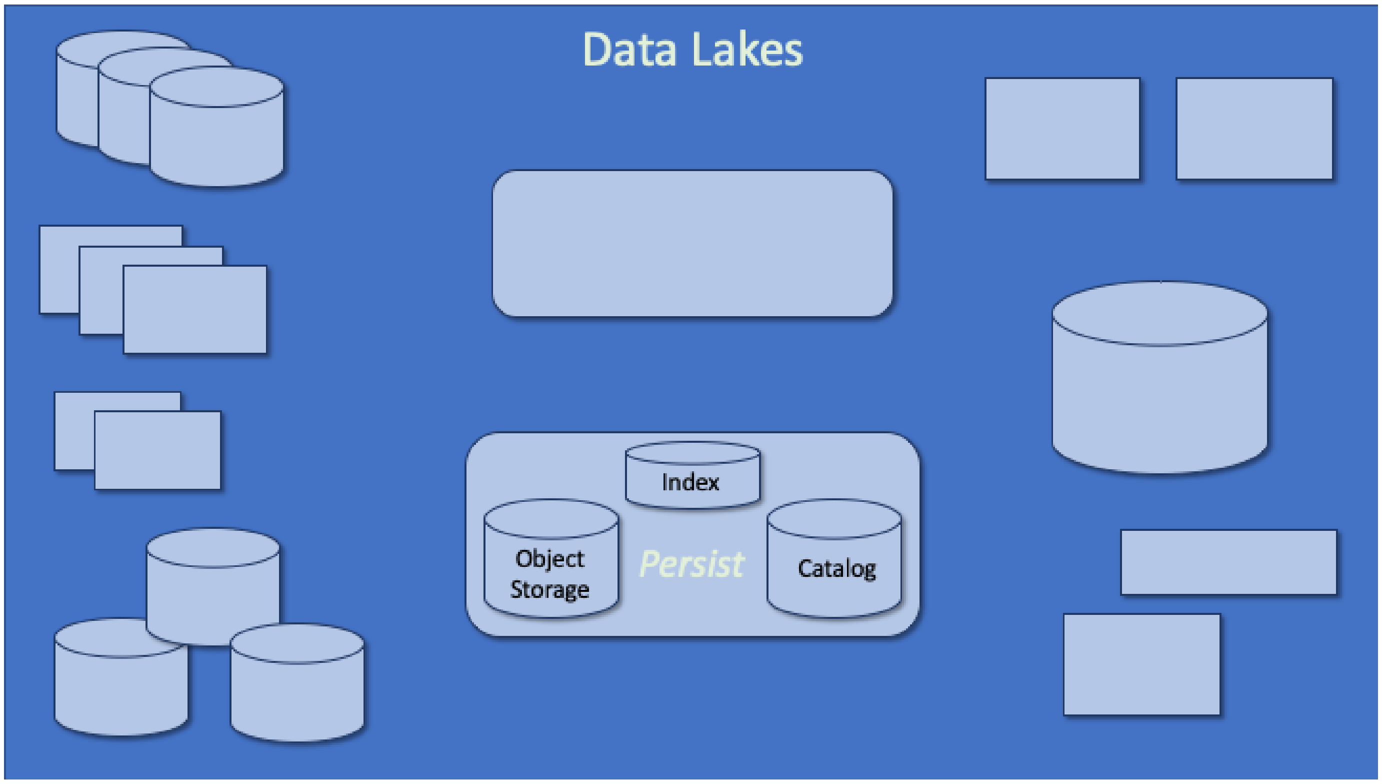
How a Data Lake in the Cloud Works



A data lake in the cloud can be a very useful tool when it comes to storing, transforming, detailing and analyzing various forms of data.

It is a centralized area to perform many operations, but first we'll start with an empty data lake and build it up to a fully functioning, eventually automated data lake.

For a video version of this walk through from the original source you can visit Data Lakes in the Cloud by IBM.



The center of a date a lake in the cloud is the data **persistency** itself.

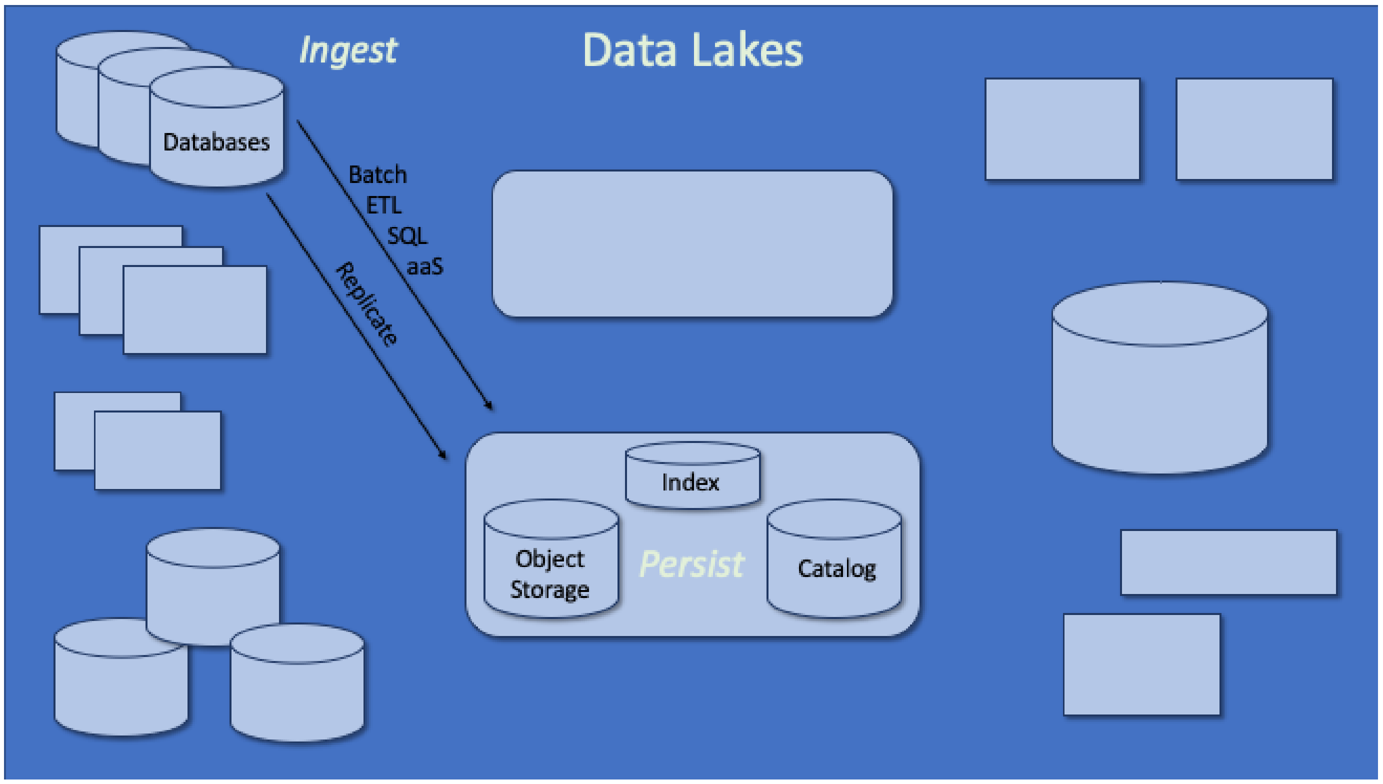
- ***Persistent*** *data in the field of data processing denotes information that is infrequently accessed and not likely to be modified.*
- ***Static*** *data is information, for example a record, that does not change and may be intended to be permanent.*
- ***Dynamic*** *(transactional) data is information that is asynchronously updated as new information becomes available.*

To start with, the persistency of data, and the data itself within the data lake is located in the object storage. But we don't just persist the data itself, we also persist information about the data.

This includes the indexing of data so that we can make use of the data lake efficiently. This also includes storing metadata about the data in a catalog.

- *Metadata is also known as data that provides information about other data* (i.e. a map of certain key codes for a data set, any type of descriptive information about a resource, etc.).

So, this is our persistency of the data lake.

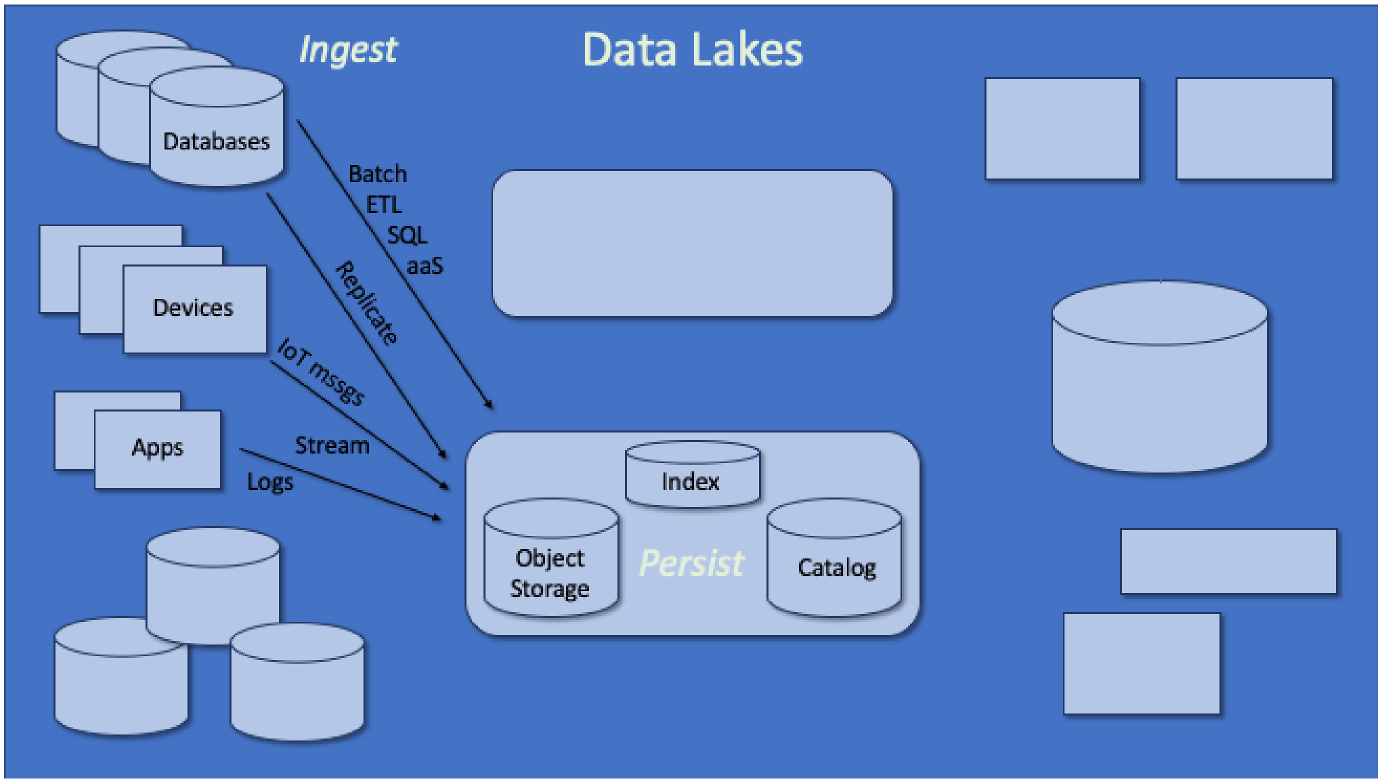


Now we need to get this data into the data lake. There are different types of data that we can ingest into the data lake.

There are situations where some of our data is already persistent in databases. These may be relational databases, other operational databases, noSQL databases, etc .

To get this data into a data lake, there are two fundamental mechanisms. One is basically an ETL (Extract-Transform-Load) and is done in a batch fashion (scheduled time once sufficient data has accumulated). This is typically done using SQL, but since we're talking about cloud data lakes, this is SQL-as-a-service.

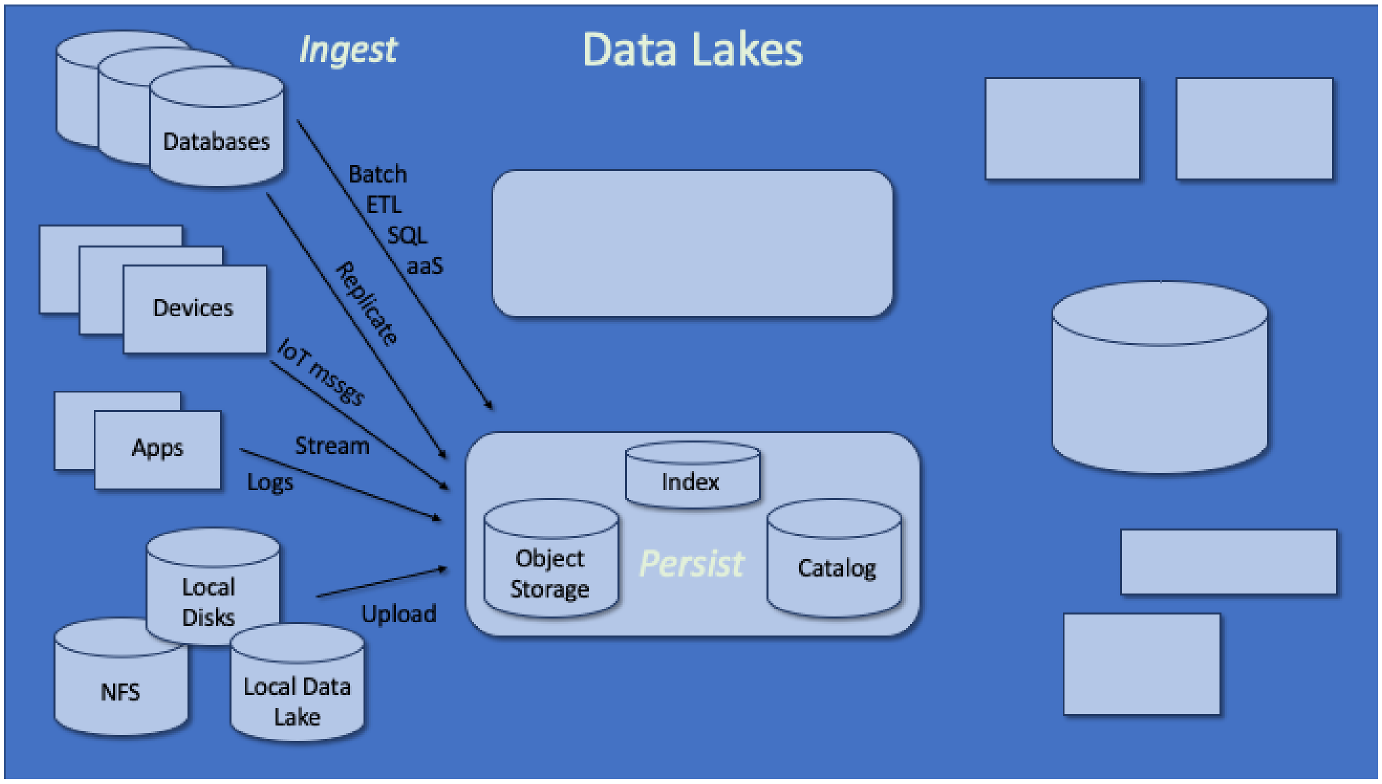
There’s also the mechanism of replication which is basically more of the change feeds. So after a batch ETL is done on the initial data set, we look at how to replicate the initial batch ETL on new, incoming information.



Next, there is data that has not been persisted at all which can be generated from devices. So, we may have things like IoT devices (internet of things), driving cars, etc.

They are producing a lot of IoT messages all the time, continuously, and they also need to basically stream into the date lake. We need a streaming mechanism for this in order to store in the object storage.

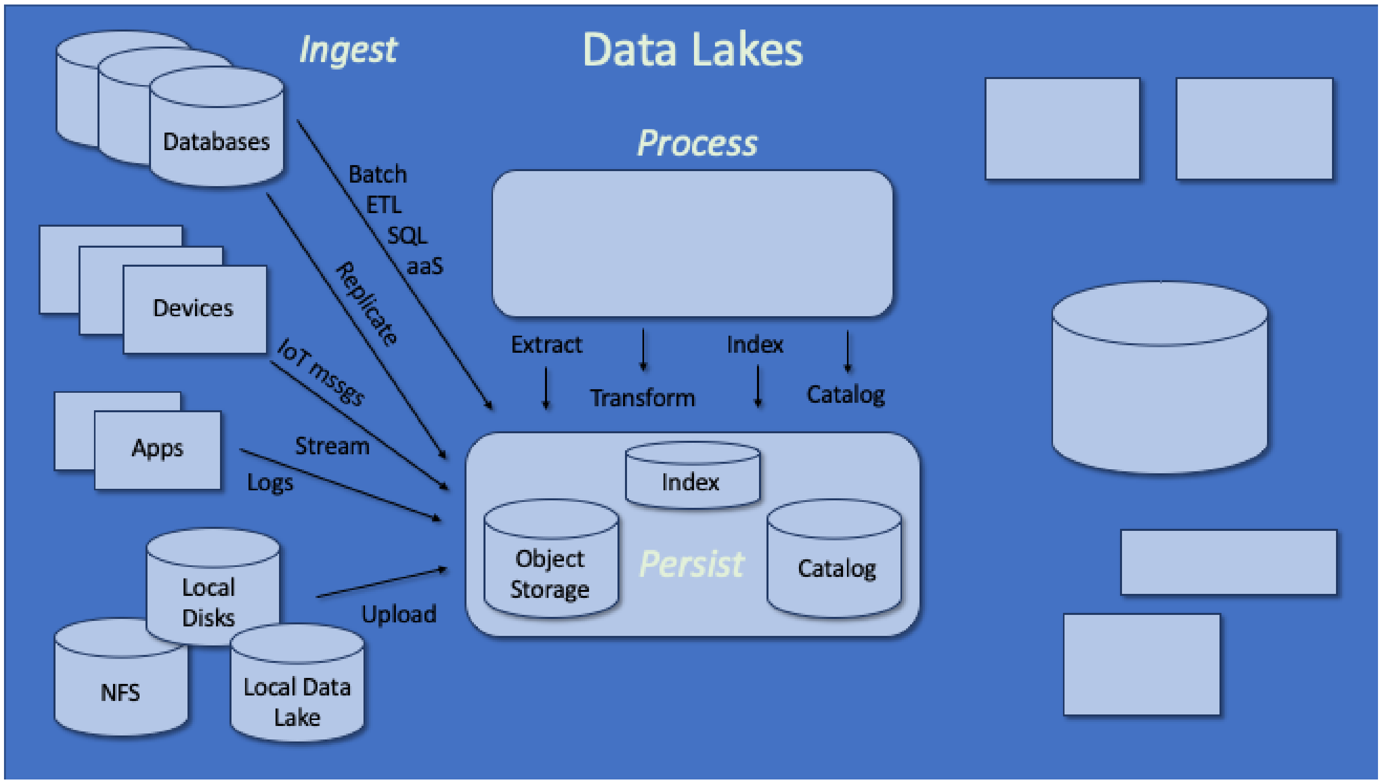
In addition, we have data that is originated from applications, or services that are used by applications, that are running in the cloud. They are producing logs, which is valuable information, especially for operational optimizations and getting business insights from user behavior. We also need a streaming mechanism for this in order to store in the object storage.



Lastly, we may have a situation where we already have data sitting around in local discs or our own machines. This may include a local (classical) data lake not in a cloud, which is typically Hadoop clusters.

We may have NFS (network file system) shares that are used in teams to store certain data.

To get this information into a data lake, we need an upload mechanism, which is provided efficiently in a data lake.

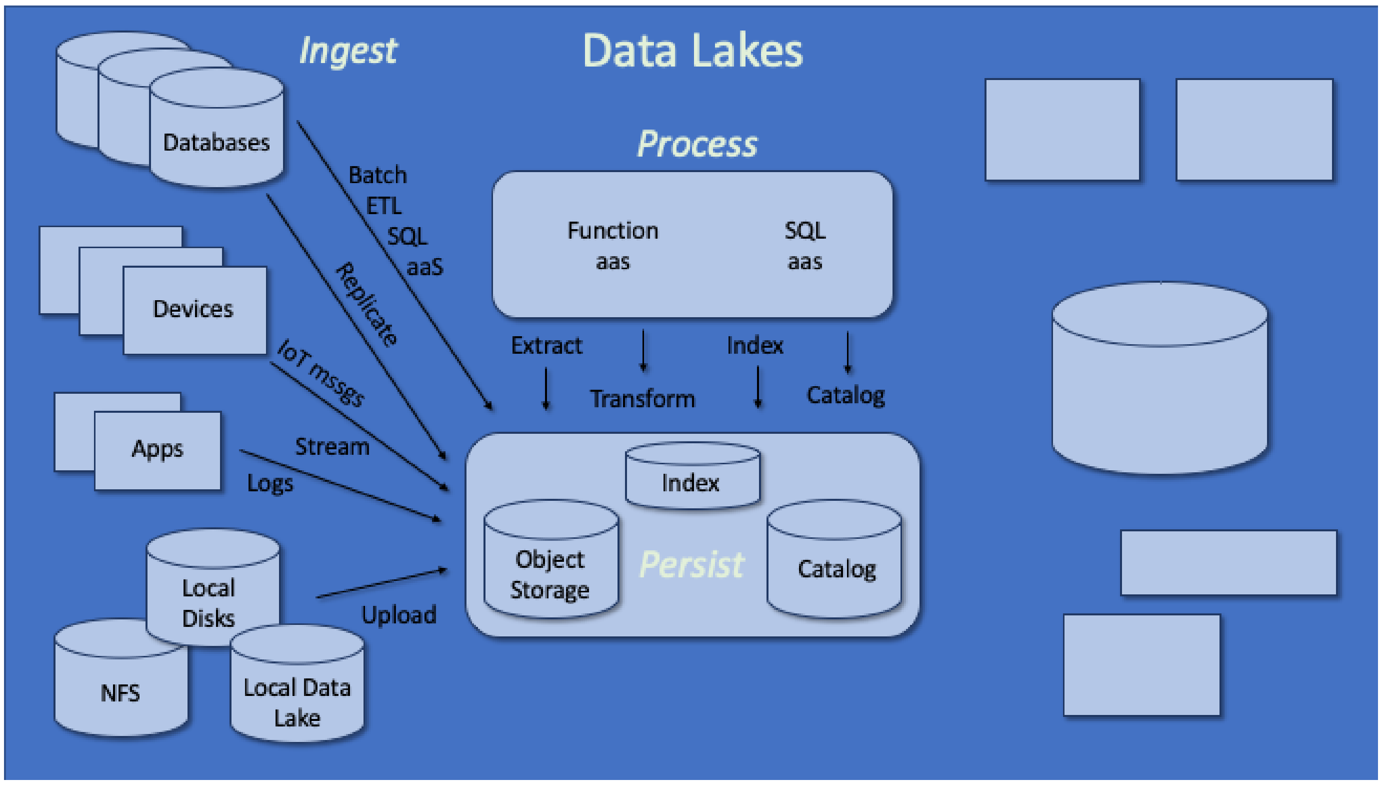


Now that we have the data, the next thing we need to do is process it. Processing data is especially important if that data hasn't gone through an initial processing.

For example, device data and application data are pretty raw and has a very raw format, that is very volatile (different structures, changing schema). It may also have no structure, like binary data. Even images from a camera need features extracted.

After this, data may still need a lot of cleansing, such as normalizing certain units, rounding up to certain time boundaries, getting rid of null values, assigning certain features, etc. So, basically there’s still much to do about certain types of transformations before we can move the data onto analytics.

Additionally, we need to create new indexes for the processed data (if applicable) in order to know more about the data and catalog it in order to let the data lake know more about the data.

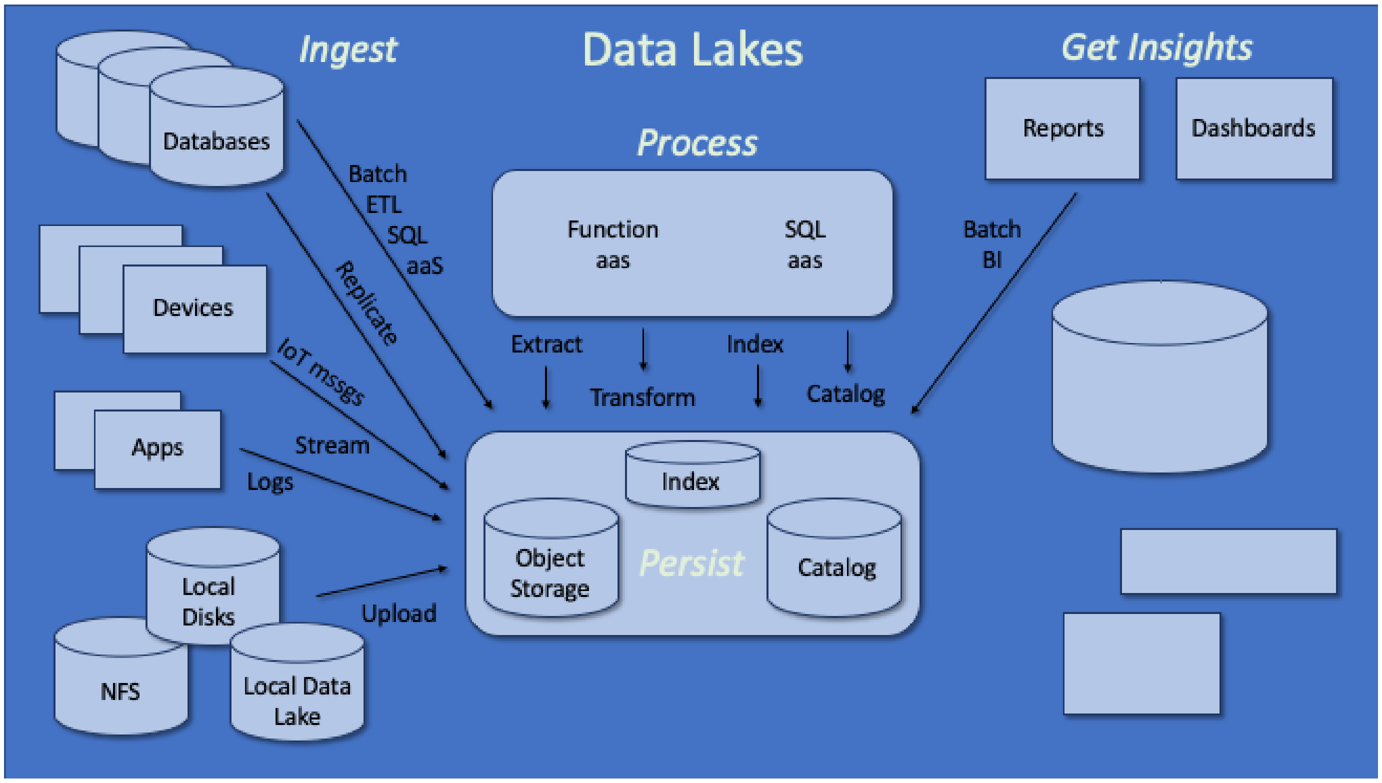


There are multiple steps in data transformations and we often need a pipeline. There are two processes or types of services that are suited for this type of processing.

1. Functions-as-a-service (FaaS)
2. SQL-as-a-service (SaaS)

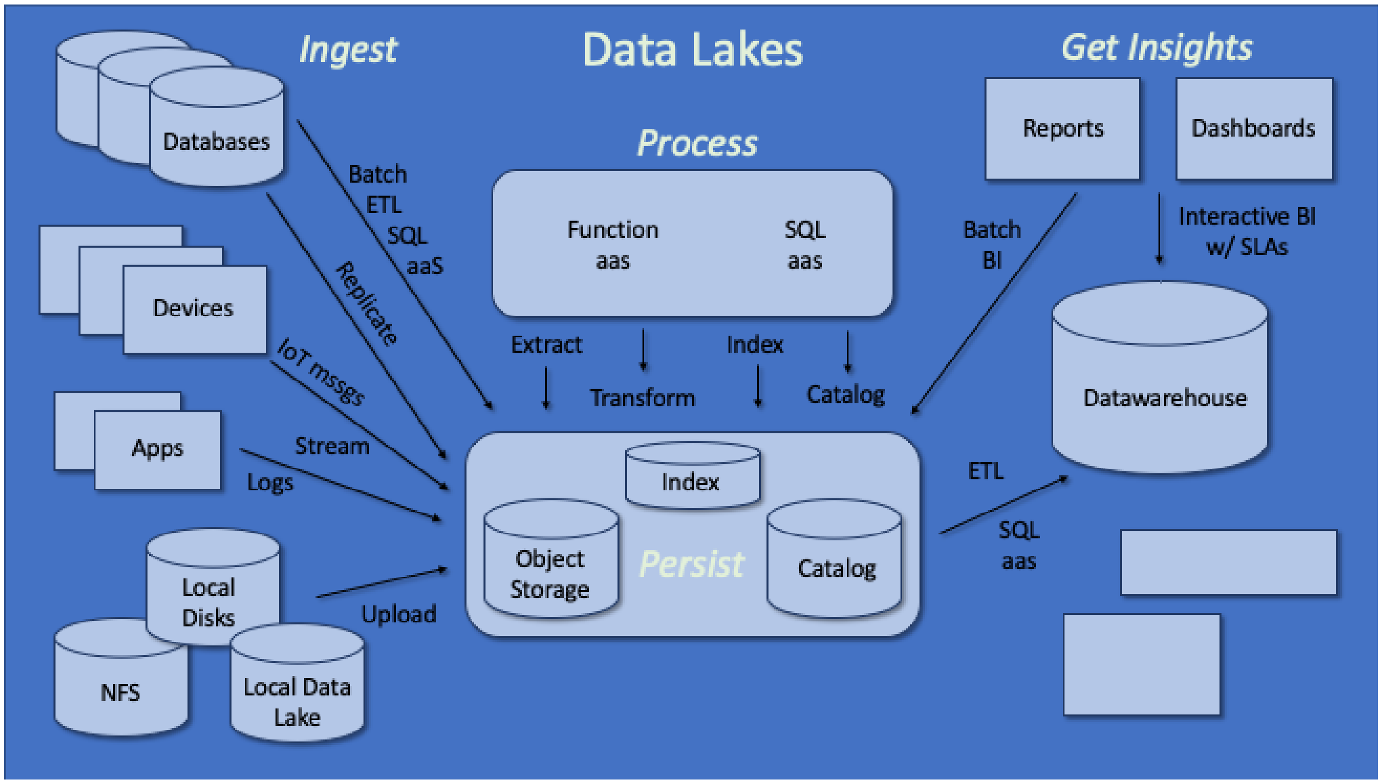
With SQL and functions as a service we can do a whole range of things. We can create indexes and tables through SQL DDLs (data description language). We can use functions with custom libraries and custom code to do future extractions and transformations of future data with the same format.

So, now we have have gone through this pipeline and ingested, prepared, processed and cataloged all of this data, and we now know what kind of data we have.



Now we are ready to start generating insights using business intelligence. This consists of things like creating reports or dashboards.

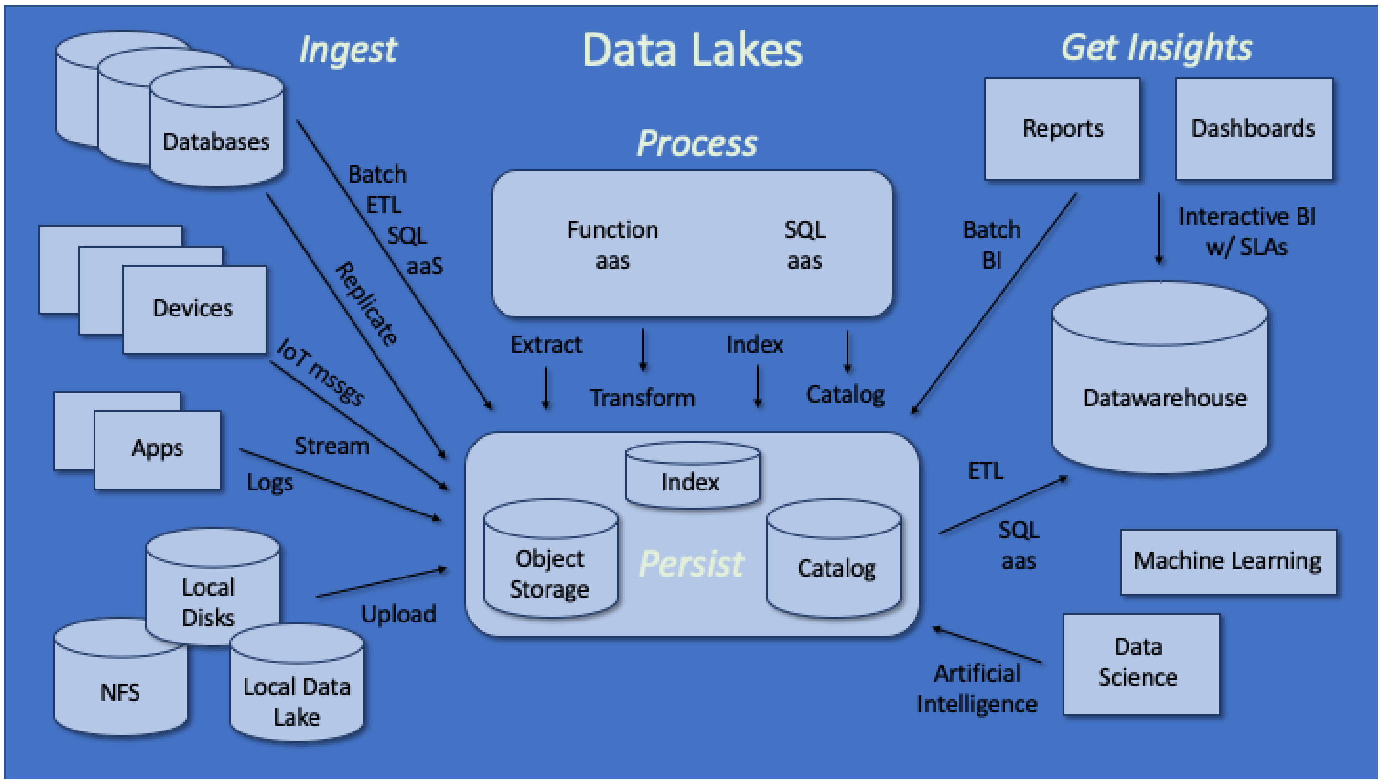
One option that is possible now because of the data lake in the cloud is to generate insights directly against this data in the data lake. We can create reports and dashboards in a batch function within the batch ETL options (batch business intelligence).



For more interactive requirements (sitting in front of the screen) and the need to refresh, for example a dashboard, we have another mechanism in place within the data lake ecosystem. This is a data warehouse or database.

After data has gone through all of the data preparation in the data lake, we can move it over to a data warehouse through another ETL. SQL-as-a-service (SaaS) will be useful again because it is in the cloud and already being used.

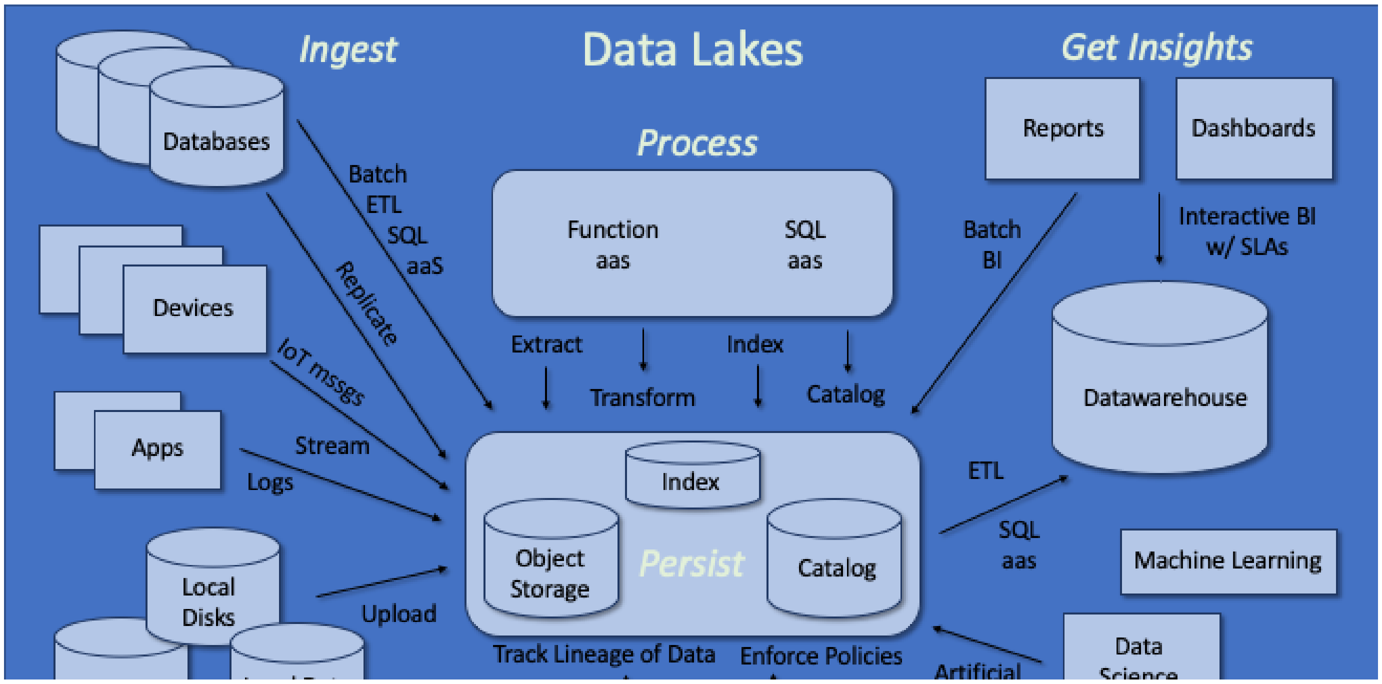
The data warehouse creates a more established way of doing traditional business intelligence that can be used by our BI tools, reporting tools and dashboarding tools in order to do interactive BI with performance and response time SLAs (service level agreements).



For more advanced types of analytics such as machine learning, we have a whole set of tools that can be used to accomplish this need. We can also perform AI against the data we've prepared in a catalog as well as data science.

These tools or types of analytics have very strong support for accessing data in an object storage. These tools can connect directly to the data lake.

This is the end-to-end process of using a data lake.



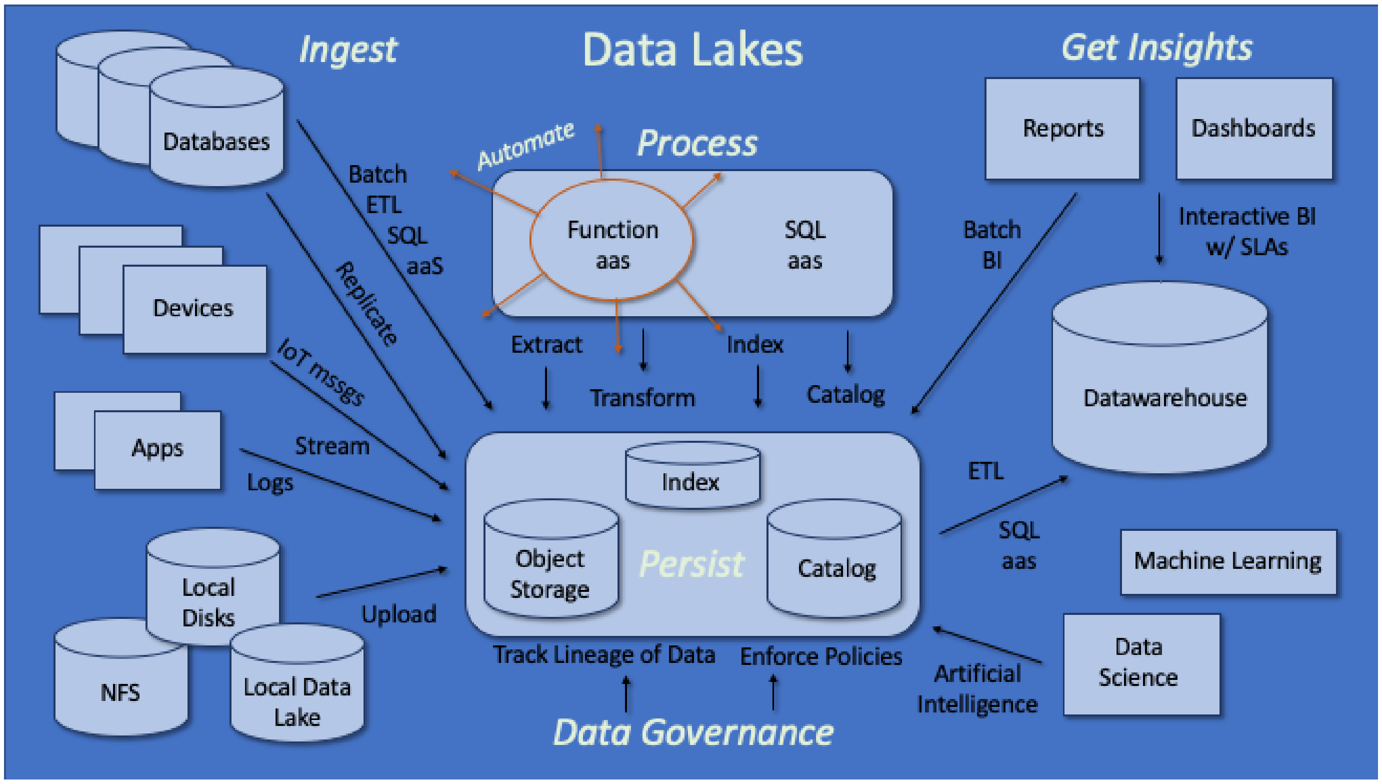
With data lakes, there are some big problems that need to be addressed.

- Can we prove and explain how we got to this insight?
- Can we trust this insight?
- Can we reproduce this type of insight?

So a way of addressing these problems is through data governance, which has two main things that we need to take care of.

1. We need to be able to track the lineage of the data because it is coming from different sources, through preparation, into some insights in the form of reports. We always need to be able to back track and see:
    a. Where did this report come from?
    b. Why is it looking like this?
    c. What's the data that basically produced it?
        i. We want to record the process that the data goes through to get from beginning to end
        ii. Usually through a use case diagram for each set of processes that we use -
2. We need to be able to enforce policies within our data lake.
    a. Who is able to access what?
    b. Who is able to see personal information?
    c. Can we access it directly, or only in an anonymized or masked form?

These are all governance rules and there are governance services available in the cloud that basically a data lake needs to apply and use in order to track all of this.



Lastly, because we are in the cloud, we would like to deploy our entire pipeline of data traveling through this whole infrastructure using automation.

Function-as-a-service (FaaS) plays a special role because FaaS has a lot of mechanisms that can be used to schedule and automate things like batch ETL, report generation, batch or real-time data ingestion and more.

Eventually we would like our data lake to perform all of these operations automatically once we have completely built it up and this would be the final phase.