2018

Understanding Data Lakes and Data Lake Platforms

White Paper

Intro

If you're working with data in any capacity, you should be familiar with Data Lakes. Even if you don't need one today, the rapid growth of data and demand for increasingly versatile analytic use cases (such as reporting, machine learning, and predictive analytics) could result in your organization outgrowing its data infrastructure much sooner than you currently foresee.

We've prepared this guide to help get you ready for that day, and keep you from asking embarrassing questions such as "what kind of data lake do you think we should buy?" To understand what's wrong with that line of reasoning, let's jump right into:

WHO SHOULD READ THIS?

- BI and analytics leaders looking to future-proof their organization's data architecture
- Executives who want to understand the latest developments in the Big Data landscape
- DBA / DevOps teams who are currently unfamiliar with data lakes

Data Lakes: Challenges and Opportunities

What is a Data Lake?

The very first thing to understand, and which often confuses people who come from a database background, is that the term “data lake” is most commonly used to describe a certain type of big data architecture, rather than a specific product or component.

The specific architectural paradigm people refer to when discussing Data Lakes is: the idea that when working with massive amounts of data, we would opt to store raw data in its original form in one centralized repository (the ‘lake’), and use this repository to later support a broad range of use cases.

The term ‘Data Lake’ refers to an architectural paradigm, not a specific type of product.

Store Now, Analyze Later

The core tenet of the data lake approach is to separate storage from analysis. Data lakes ingest streams of structured, semi-structured and unstructured data sources, and store the data as-is and schema-less.

This is a sharp deviation from traditional analytics, where we would build our database in a way that was best suited to support a particular use case - transactional, reporting, ad-hoc analytics, etc, and then structure the data accordingly.

After all data has been stored successfully, and we've decided what we would like to do with it, data can then be outputted to other systems that will make it workable for various consumer applications: Data warehouse, machine learning, BI tools, NoSQL databases and dozens of other platforms that manage, integrate and structure the data for analysis.

A Data Lake is built on the premise of storing raw data in its original form, while transforming and processing it later down the road.

Typical Use Cases for Data Lakes

Data lakes are mostly helpful when working with big data - either high-volume, high-velocity or both. This could typically include:

- **Streaming data** - event-based data streams generated continuously, such as by IoT devices, clickstream tracking, or product/server logs. Typically these are small records in very large quantities, in semi-structured format (often JSON).
- **Real-time applications** - recommender systems or predictive decision engines need to be able to pull in relevant data points from a vast number of records in sub-second timeframes.
- **Archiving and historical storage** - leveraging the inexpensive storage of data lakes to store historical data that is infrequently analyzed.
- **Staging area for data warehousing** - organizations can use data lakes as arenas for manipulating and transforming data before moving it into a data warehouse for further analysis.

Data Lakes are often used to store data that is either high-volume, high-velocity or both.

Data Lakes vs Databases

When working with databases, data needs to be structured according to a predetermined schema when it is stored. This makes it easier to later access and analyze the data using SQL-based tools, but makes it more difficult and expensive to store.

With Data Lakes storage is simple and cheap, but structuring the data for analysis can be challenging.

	Database	Data Lake
Storage	Expensive	Cheap
Analysis	Simple	Complex
Scaling	Difficult	Easy
Adding sources	Difficult	Easy

Data Lake Platforms, covered in the next chapters, help bridge the gap between database and data lake by simplifying the analytic layer of the lake architecture.

Data Lakes: Advantages and Challenges

Advantages

- **Resource optimization**: by decoupling (cheap) storage from (expensive) compute resources, data lakes can be tremendously cheaper than databases when working with high scales.
- **Less ongoing maintenance**: the fact that data is ingested without any kind of transformation or structuring means it's easy to add new sources or modify existing ones without having to build custom pipelines.
- **Broader range of use cases**: data lakes give organizations more flexibility in how they eventually choose to work with the data, and can support a broader range of use cases, since you are not limited by the way you chose to structure your data upon its ingestion.

Challenges

- **Technical complexity**: Not only are most data lake architectures not self-service for business users, they are not self-service even for experienced developers - instead requiring a team of dedicated data engineers to maintain the multitude of building blocks, pipelines and moving parts that compose a data lake architecture. As we've previously written, this process kind of sucks.
- **Slow time-to-value:** Data lake projects can drag on for months or even years, creating a resource drain that is increasingly difficult to justify.
- **Data swamps**: Storing raw data provides a high level of flexibility, but foregoing all data governance and management principles can lead organizations to hoard massive amounts of data that they will likely never use, while making it more difficult to access the data that could actually be useful.

Data Lake Deployments

While data lakes are often associated with Hadoop, the terms are far from interchangeable. In fact, it would be safe to say that most new data lakes are being built in the Cloud (with Amazon Web Services leading the pack, following by Microsoft Azure and Google Cloud Platform). For these reasons Gartner has recently predicted that Hadoop implementations are approaching obsoletion.

However, our experience indicates that organizations which work with truly massive data (in the tens to hundreds of petabytes) might be deterred by the costs of storing and processing all of that data in the Cloud, and hence we are still seeing large data lake implementations built on HDFS on-premises - so we are not so quick to proclaim the death of Hadoop.

Data Lakes can be built both on Hadoop on-premises, as well as in the Cloud using products from Amazon, Microsoft or Google.

Data Lake Platforms

What is a Data Lake Platform?

A Data Lake Platform (DLP) is a software that is meant to address the data lake challenges which we've covered in the previous section - complexity, sluggish time to value, and overall messiness. The DLP addresses these challenges by:

- **Unifying data lake operations** and combining building blocks to create simpler data lake architectures - instead of a patchwork of glued-together systems, a DLP offers a single platform that takes care of everything from data management and storage to processing, ETL jobs and outputs.
- **Enforcing best practices** and optimizing data flows, and thus replacing months of manual coding in Apache Spark or Cassandra with automated actions managed through a GUI.
- **Improved performance and resource utilization** throughout storage, processing and serving layers

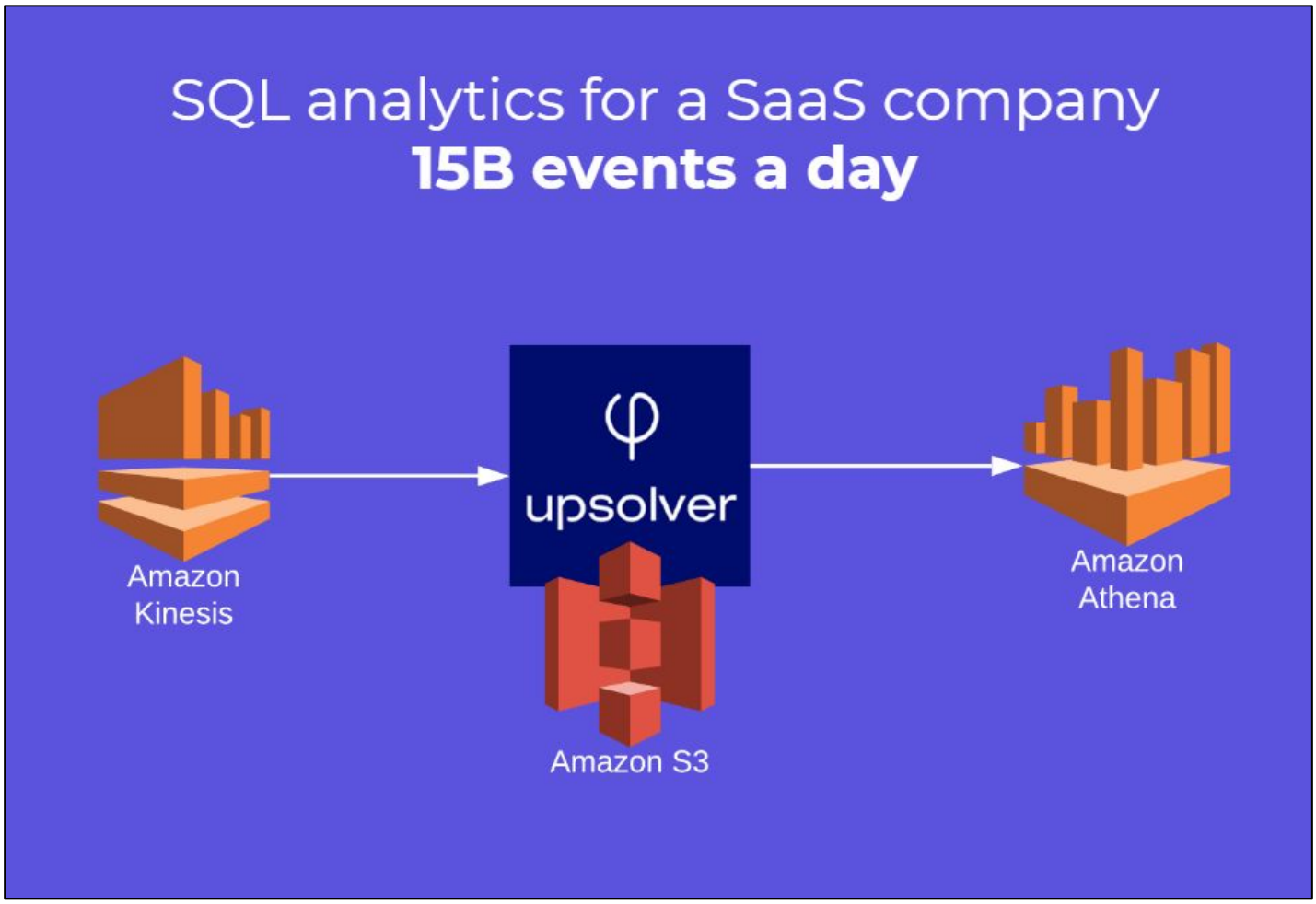
Data Lake Platforms enable developers without an extensive big data background to create a complete pipeline from incoming data streams to structured data, that can be queried using SQL or other analytic tools.

By doing so, DLPs enable organizations to generate more business value from their data lakes, at a faster pace.

Components of a Data Lake Platform

- **Data management**: a DLP should be a safeguard against your data lake becoming a data swamp by giving you visibility into the data being ingested, including a data catalog and metadata store.
- **In-memory data preparation**: the need to perform stateful ETL on streaming data can be a major obstacle to effective data lake implementations, but should be easy to handle with a data lake platform.
- **Support for diverse outputs**: keeping in line with the data lake approach, DLPs should be able to support a broad range of use cases by sending data outputs to a wide variety of systems (data warehouses, SQL engines, BI tools) and formats (CSV, Parquet, etc.)
- **Real-time serving**: since real-time applications are a very common use case in the data lake world, your DLP should be able to keep up and provide this functionality without having to spin up a separate NoSQL database.

Using a DLP for Ad-hoc Analytics



In this example, the Upsolver Data Lake Platform is storing streaming data from Amazon Kinesis on the S3 data lake, and sending structured tables for ad-hoc analytics to Amazon Athena

Using a DLP for Real-time Decisioning

In this example, data is being sent from Apache Kafka, processed by the Upsolver DLP, and then specific user-level records are accessed via REST API to power a real-time predictive algorithm.

upsolver

The **Data Lake Platform**

Gartner estimates that 85% of big data projects fail, often due to lack of internal resources and knowledge, and the difficulty of managing complex architectures and data pipelines.

Upsolver is here to change this reality with a complete Data Lake Platform that's powerful, agile and simple enough for any developer to launch and maintain.

Upsolver's Data Lake Platform takes the complexity out of streaming data integration, management and preparation on any data lake - whether it's HDFS on-premise or on AWS, Azure or Google Cloud.

› Schedule a Demo

› Start Free Trial

www.upsolver.com