“Data Architecture Basics: An Illustrated Guide for Non-Technical Readers” Notes

Pierre Parel

April 12, 2023

Contents

1 Introduction to Data Architecture

- Data architecture is the blueprint for how data is organized, stored, integrated, and used within an organization
- The goal of data structure is to ensure that an organization's data is structured in a way that facilitates efficient and effective use.
- Data architecture allows businesses to organize their data in a way that makes it easier to analyze and use for decision-making.
- Data architecture can be complex, especially for large organizations with a lot of data and it can be difficult to ensure that the data is accurate and up-to-date.

2 Scalability in Data Architecture

- **Distributed Systems** - A network of computers working together to achieve a common goal. Building blocks of distributed systems are *Nodes.* A *Cluster* is a collection of multiple nodes.
- **High Availability** - The ability of a system to function continually without failure for a long period of time.
- **Fault Tolerant** - The ability of a system to maintain usage during the failure of a component.
- **Hadoop** - A popular cluster-based open-source framework for distributed storage.
- **Storage Resources** - The data storage capabilities of an architecture and a main requirement for all data science operations.
- **Synchronous Replication** - Process by which data is written to the primary storage location and a replica or backup location simultaneously.

2.1 CAP Theorem

The CAP Theorem helps set expectations when it comes to distirbuted systems. It states that no distributed data storage system can have more than two of the following:

- **Consistency** - every read request returns the most recently written information (or error).
- **Availability** - every read request receives a non-error response, but the information returned may be old.
- **Partition Tolerance** - even if messages between nodes fail, the system will sustain operations.

2.2 The Cloud

Cloud computing is definitely not some next-step distributed computing; it's just a different way to provision and use infrastructure and managed services.

Cloud providers are only as good as their security, so this remains a top priority. While no system can be completely fault-tolerant and secure in all cases, the security measures and access permission restraints are severe. However, the CAP Theorem still holds, so no system will be able to solve every data concern with absolute certainty.

- The Pros of Cloud Architecture
  - Rapid scalability means the architecture can keep up with the business
  - Global accessibility enables comparisons between different markets
  - Fewer upfront IT charges
  - No hardware requirements
  - Cloud service is responsible for fault tolerance
- The Cons of Cloud Architecture
  - Forfeit of data custody means businesses must trust their cloud provider to protect their data

2.3 Scalability and the End-User

One critical part to scaling data architecture is making sure that the end users can interact successfully with the data pipeline.

2.4 What is the data pipeline?

- Data pipelines are paths data takes through the architecture once it's been created
- Data pipeline follows the flow of processes and systems that enable the data to be used in the conext of a data project.
- Everything from data collection, storage, acess, cleaning, analysis, and presentation are part of the data pipeline
- Building data pipelines is not a one-off task; continued maintenance and opitimization are a necessary part of the infrastructure.

3 Security in Data Architecture

Security includes technologies, techniques, processes, and best practices that guarantee the integrity, availability, and the enforcement of data and information system governance. It encompasses technical protocols and user behavior and usually centers on what behavior to normalize.
The three main questions to ask about security are:

- **AUTHENTICATION**: Is the user who he/she says he/she is?

- **AUTHORIZATION**: To what data does the user have access to?
- **AUDITIBILITY**: Can we see later who accessed what, when?

3.1 Authentication

The purpose is to ensure that the user is who (s)he is declaring to be. In data architecture, authentication happens through something the user knows (i.e., a password) plus sometimes for even more security, something (s)he has.

Using *Multi-Factor Authentication*, users must provide at least two proofs that they are who they claim to be when logging in. While this takes longer to sign in, the proof that users are who they claim to be is much stronger.

Single Sign-On (SSO) is a system that allows a user to authenticate once and then access a variety of systems based on the authentication. It proves that the user does not need to show ID for each purchase since (s)he has already been verified.

3.1.1 Pros and Cons

- Pros
  - Helps evalutate authorizations and can be used to trim permissions wherever possible
  - Useful safeguard against malign internal actors or for tracking system errors
- Cons
  - Requires unique IDs for each user
  - Logs can take up exorbitatnt space depending on the frequency of access and number of users

3.2 Authorization

Authorization is distinctly different from authentication.

From an architectural standpoint, this can become expensive, as the additional computing power required for a sandbox environment is never enough (according to developers). However, this sort of system mirroring is both beneficial to services and critical for security so that developers cannot inadvertently modify user data.

3.2.1 Pros and Cons

- Pros
  - Detemine what data to put in archival storage and what to keep in memory
  - Can help target compression
- Cons
  - May result in confirmation bias if users always turn to the same source of information when others are available.
  - Not particularly useful without other audit metrics.

3.3 Auditibility

Aside from maintaining development environments to ensure data doesn't get modified unintentionally, knowing who modified what data is also critical when it comes to collaboration on data projects

3.3.1 Pros and Cons

- Pros
  - Accentuates deviations and unsanctioned behavior
- Cons
  - Not particularly useful without other audit metrics