devfest

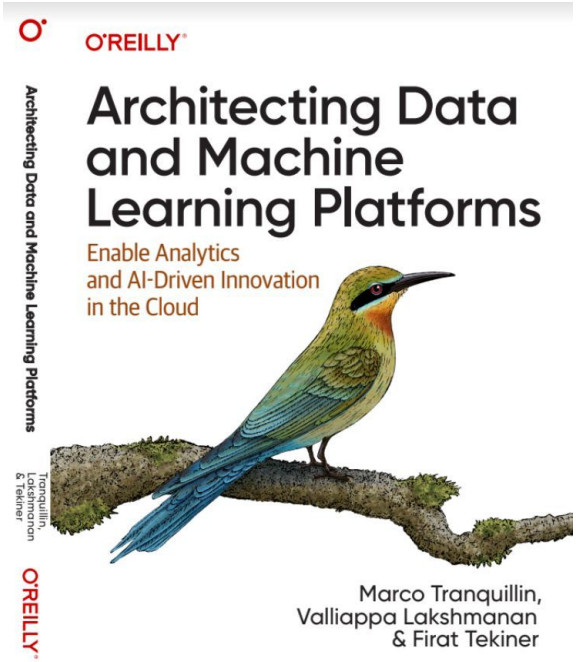
Architecting Data and ML Platforms

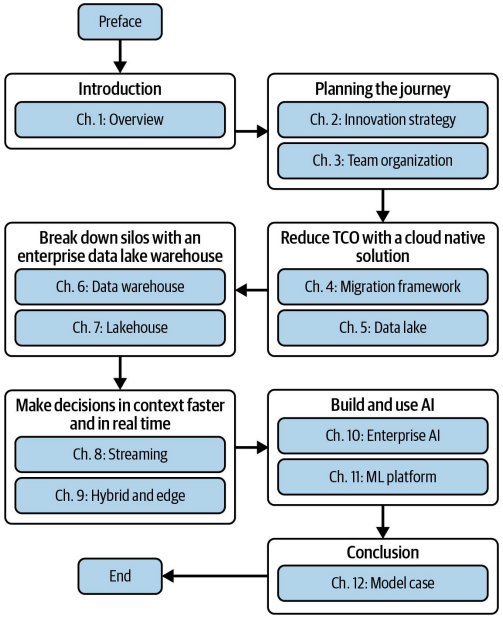
Lak Lakshmanan, Seattle, Oct 2023

Google Developer Groups

SLIDES LINK

These are excerpts from our new book



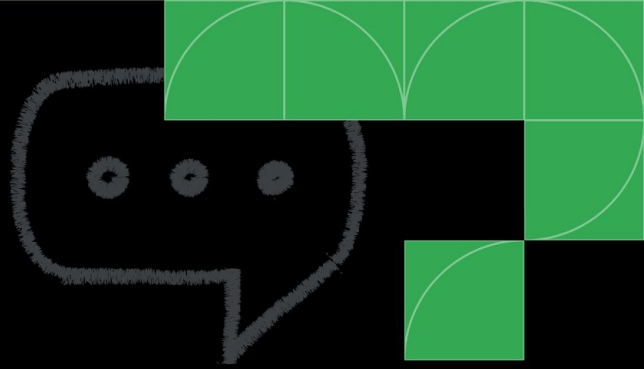


Google Developer Groups

```
Text(
  'Simple Statement or URL',
  style: TextStyle(
    color: Colors.green[200],
  ),
),
```

devfest

```
s.star,
r: Colors.green[500],

Text('23'),
```



0. Why?

Your enterprise needs a Data and ML platform to:

Collect data from a **variety of sources** such as operational databases, customer clickstream, Internet of Things (IoT) devices, software as a service (SaaS) applications, etc.

Break down **silos** between different parts of the organization

Process data while ingesting it or after loading it while guaranteeing proper processes for **data quality and governance**

Analyze the data routinely or ad hoc

Enrich the data with prebuilt AI models

Build ML models to carry out **predictive analytics**

Act on the data routinely or in response to triggering events or thresholds

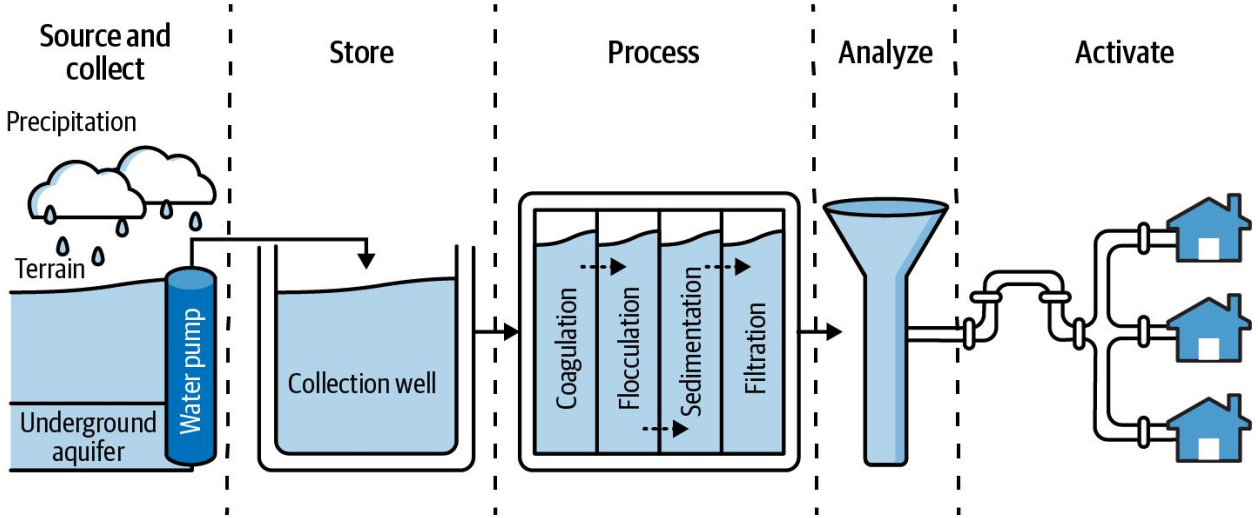
Disseminate insights and embed analytics

Reduce the effort involved in getting value out of data


'Simple Statement or URL',
style: TextStyle(
color: Colors.green[200],
),
),
devfest
s.star,
r: Colors.green[500],
Text('23'),
1. Anti-patterns

A data platform doesn't just happen because you do these steps in your application

one-offs don't reduce effort



A platform has to support *specialization*, "*horizontals*", and *future* use cases

Volume, velocity, variety

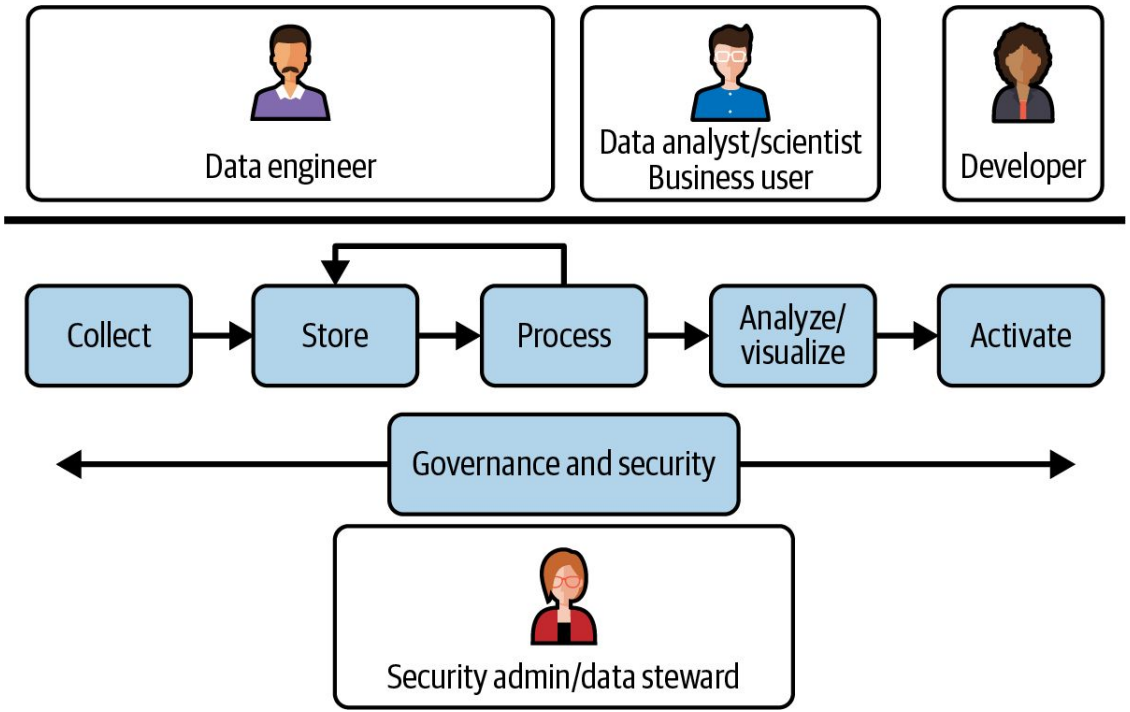
Vertical, horizontal scale

Performance vs. cost

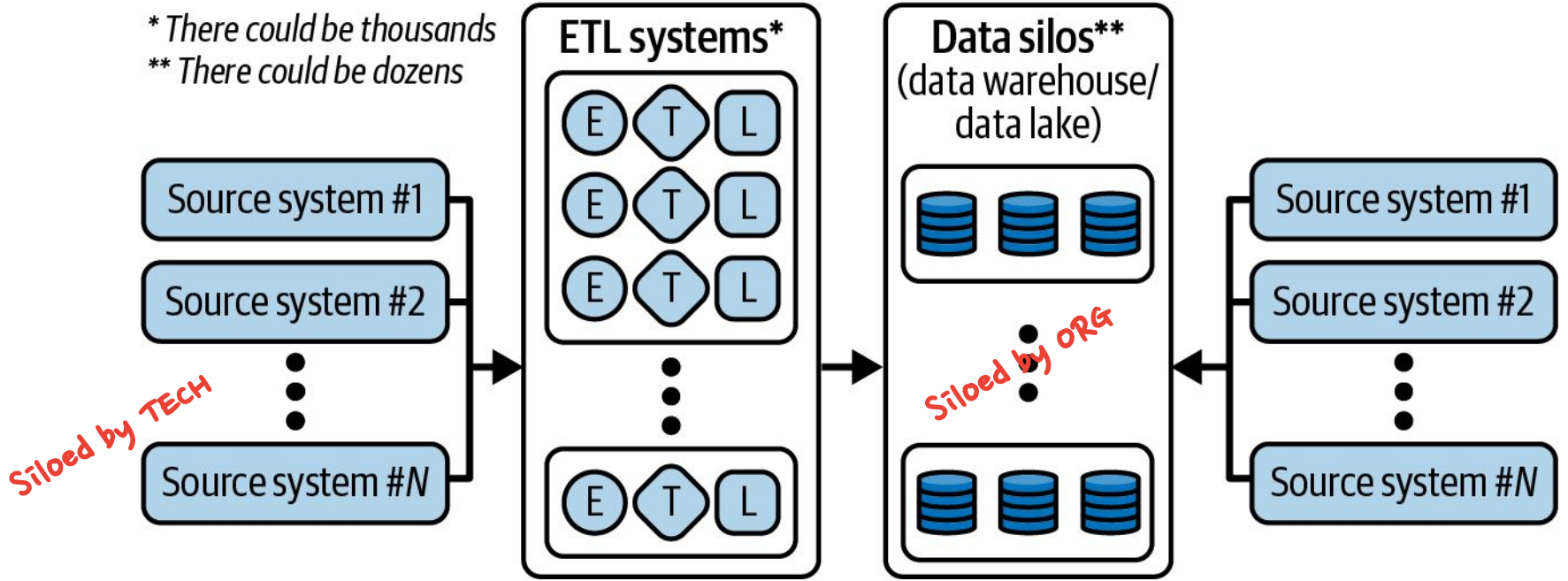
High availability

Durability

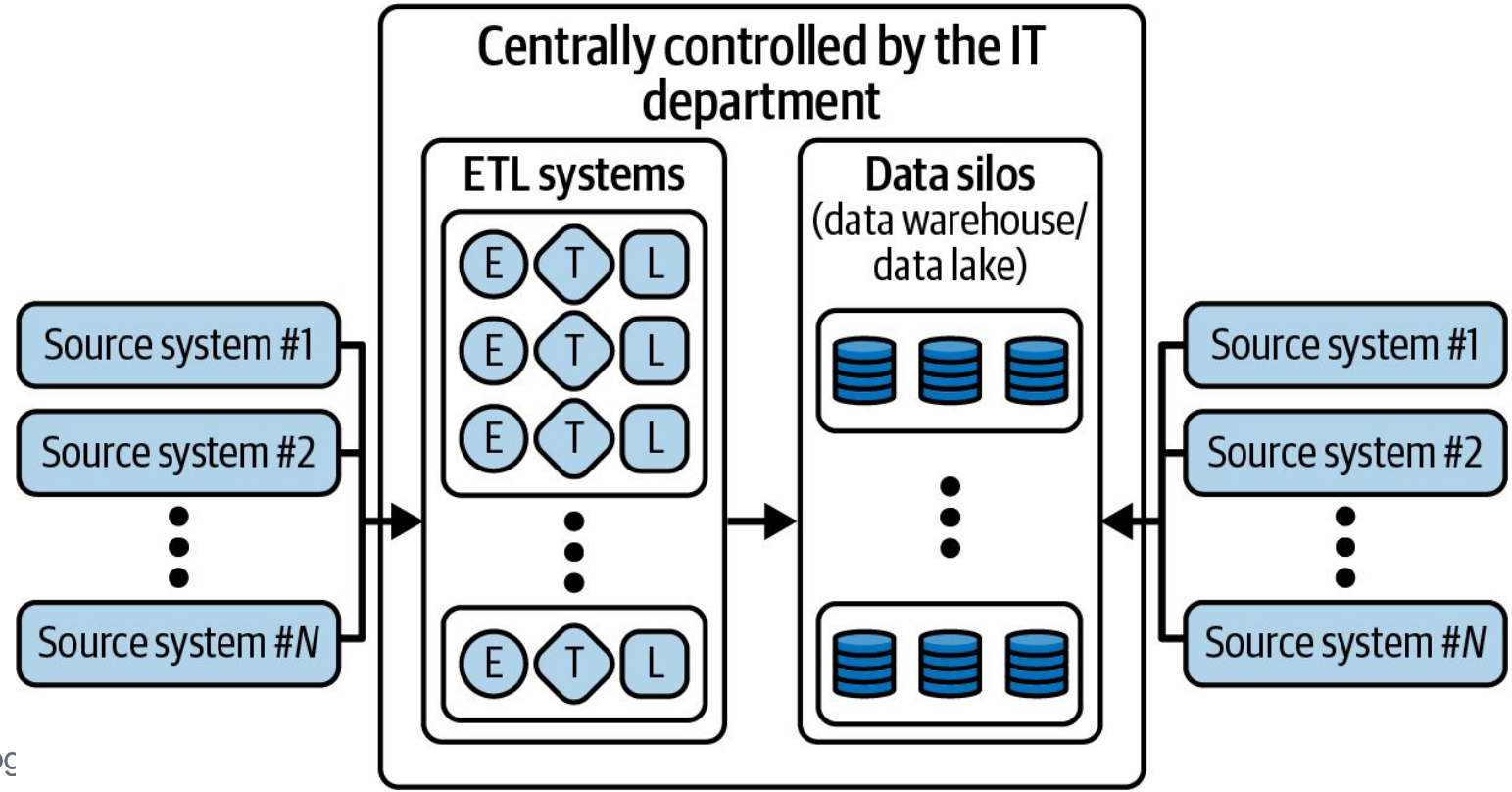
Openness



The problem with using ETL to break data silos is that every application requires bespoke transformation → data silos



Anti-pattern: Centralization of Control



Anti-pattern: Data Marts, Hadoop Data Lake

Data Marts: subset of enterprise data suited to specific workloads

Scalability

Difficult to use

Cost of infrastructure

Negative ROI

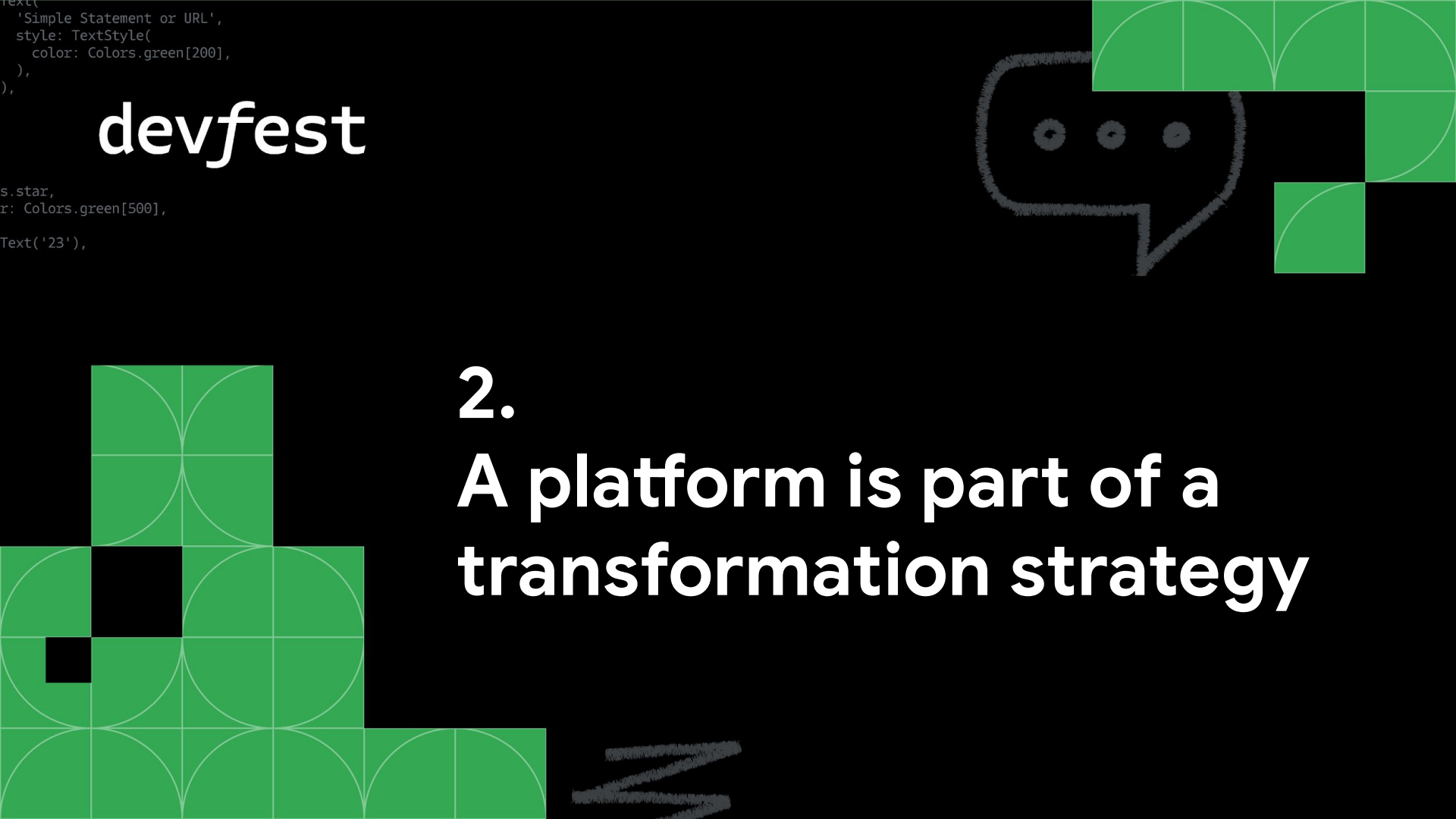
Hadoop: Distributed data processing using low-cost commodity servers

Ungoverned dumping ground

Skills gap

Compute unavailable during peak

Negative ROI

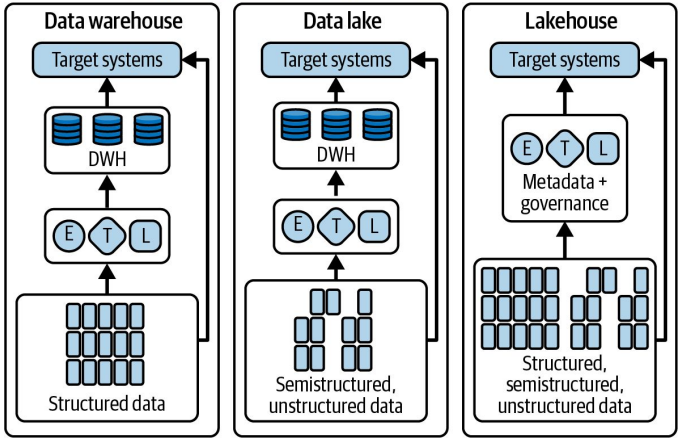


2.
A platform is part of a transformation strategy

Cloud, Hybrid, or Edge?

Cloud data platforms promise:

- Centralized governance and access management
- Increased productivity and reduced operational costs
- Greater data sharing across the organization
- Extended access by different personas
- Reduced latency of accessing data



Google Developer Groups

Here are some key business reasons for choosing hybrid and/or multicloud:

Data residency regulations
Some may never fully migrate to the public cloud, perhaps because they are in finance or healthcare and need to follow strict industry regulations on where data is stored. This is also the case with workloads in countries without a public cloud presence and a data residency requirement.

Legacy investments
Some customers want to protect their legacy workloads like SAP, Oracle, or Informatica on prem but want to take advantage of public cloud innovations like, for example, Databricks and Snowflake.

Transition
Large enterprises often require a multiyear journey to modernize into cloud native applications and architectures. They will have to embrace hybrid architectures as an intermediate state for years.

Burst to cloud
There are customers who are primarily on premises and have no desire to migrate to the public cloud. However, they have challenges of meeting business service-level agreements (SLAs) due to ad hoc large batch jobs, spiky traffic during busy periods, or large-scale ML training jobs. They want to take advantage of scalable capacity or custom hardware in public clouds and avoid the cost to scale up on-premises infrastructure. Solutions like MotherDuck, which adopt a "local-first" computing approach, are becoming popular.

Best of breed
Some organizations choose different public cloud providers for different tasks in an intentional strategy to choose the technologies that best serve their needs. For example, Uber uses AWS to serve their web applications, but it uses Cloud Spanner on Google Cloud for its fulfillment platform. Twitter runs its news feed on AWS, but it runs its data platform on Google Cloud.

Now that you understand the reasons why you might choose a hybrid solution, let's have a look at the main challenges you will face when using this pattern; these challenges are why hybrid ought to be treated as an exception, and the goal should be to be cloud native.

Challenges of Hybrid Cloud

Edge Computing

Another incarnation of the hybrid pattern is when you may want to have computational power spanning outside the usual data platform perimeter, maybe to interact directly with some connected devices. In this case we are talking about *edge computing*. Edge computing brings computation and data storage closer to the system where data is generated and needs to be processed. The aim in edge computing is to improve response times and save bandwidth. Edge computing can unlock many use cases and accelerate digital transformation. It has many application areas, such as security, robotics, predictive maintenance, smart vehicles, etc.

As edge computing is adopted and goes mainstream, there are many potential advantages for a wide range of industries:

Faster response time
In edge computing, the power of data storage and computation is distributed and made available at the point where the decision needs to be made. Not requiring a round trip to the cloud reduces latency and empowers faster responses.

In preventive maintenance, it will help stop critical machine operations from breaking down or hazardous incidents from taking place. In active games, edge computing can provide the millisecond response times that are required. In fraud prevention and security scenarios, it can protect against privacy breaches and denial-of-service attacks.

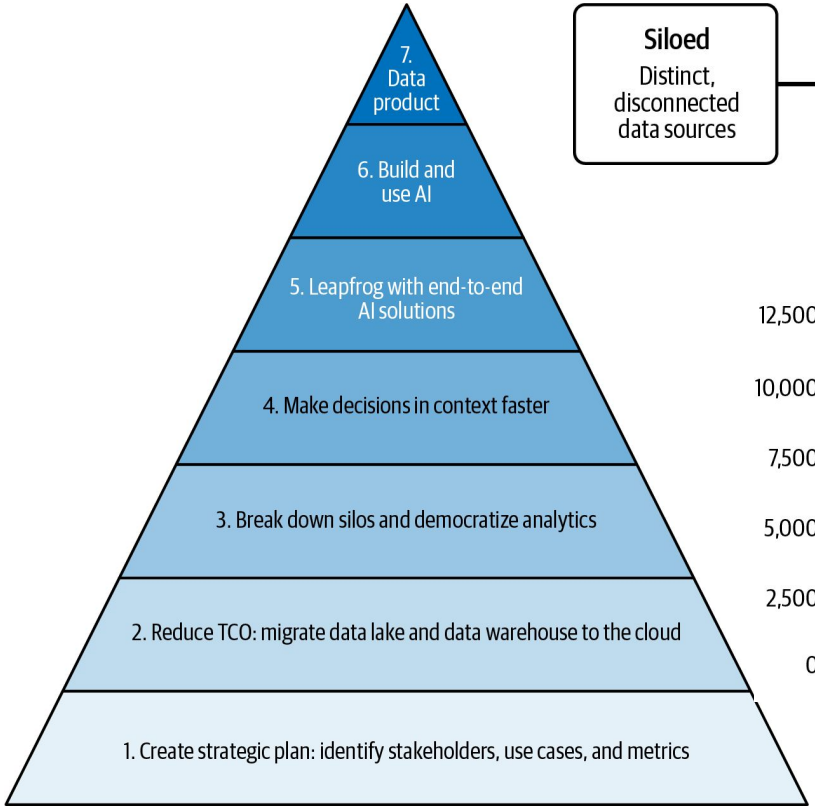
Intermittent connectivity
Unreliable internet connectivity at remote assets such as oil wells, farm pumps, solar farms, or windmills can make monitoring those assets difficult. Edge devices' ability to locally store and process data ensures no data loss or operational failure in the event of limited internet connectivity.

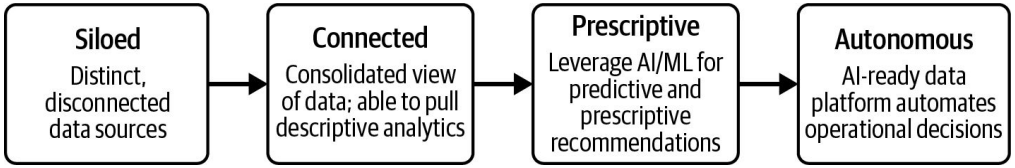
Security and compliance
Edge computing can eliminate a lot of data transfer between devices and the cloud. It's possible to filter sensitive information locally and only transmit critical data model building information to the cloud. For example, with smart devices, watch-word processing such as listening for "OK Google" or "Alexa" can happen on the device itself. Potentially private data does not need to be collected or sent to the cloud. This allows users to build an appropriate security and compliance framework that is essential for enterprise security and audits.

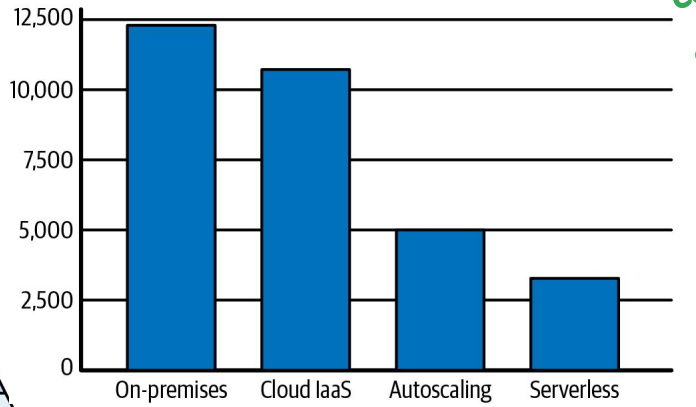
Cost-effective solutions
One of the practical concerns around IoT adoption is the up-front cost due to network bandwidth, data storage, and computational power. Edge computing can locally perform a lot of data computations, which allows businesses to decide which services to run locally and which ones to send to the cloud, which reduces the final costs of an overall IoT solution. This is where low-memory binary deployment of embedded models in a format like Open Neural Network Exchange (ONNX), built from a modern compiled language like Rust or Go, can excel.

Interoperability
Edge devices can act as a communication liaison between legacy and modern machines. This allows legacy industrial machines to connect to modern machines or IoT solutions and provides immediate benefits of capturing insights from legacy or modern machines.

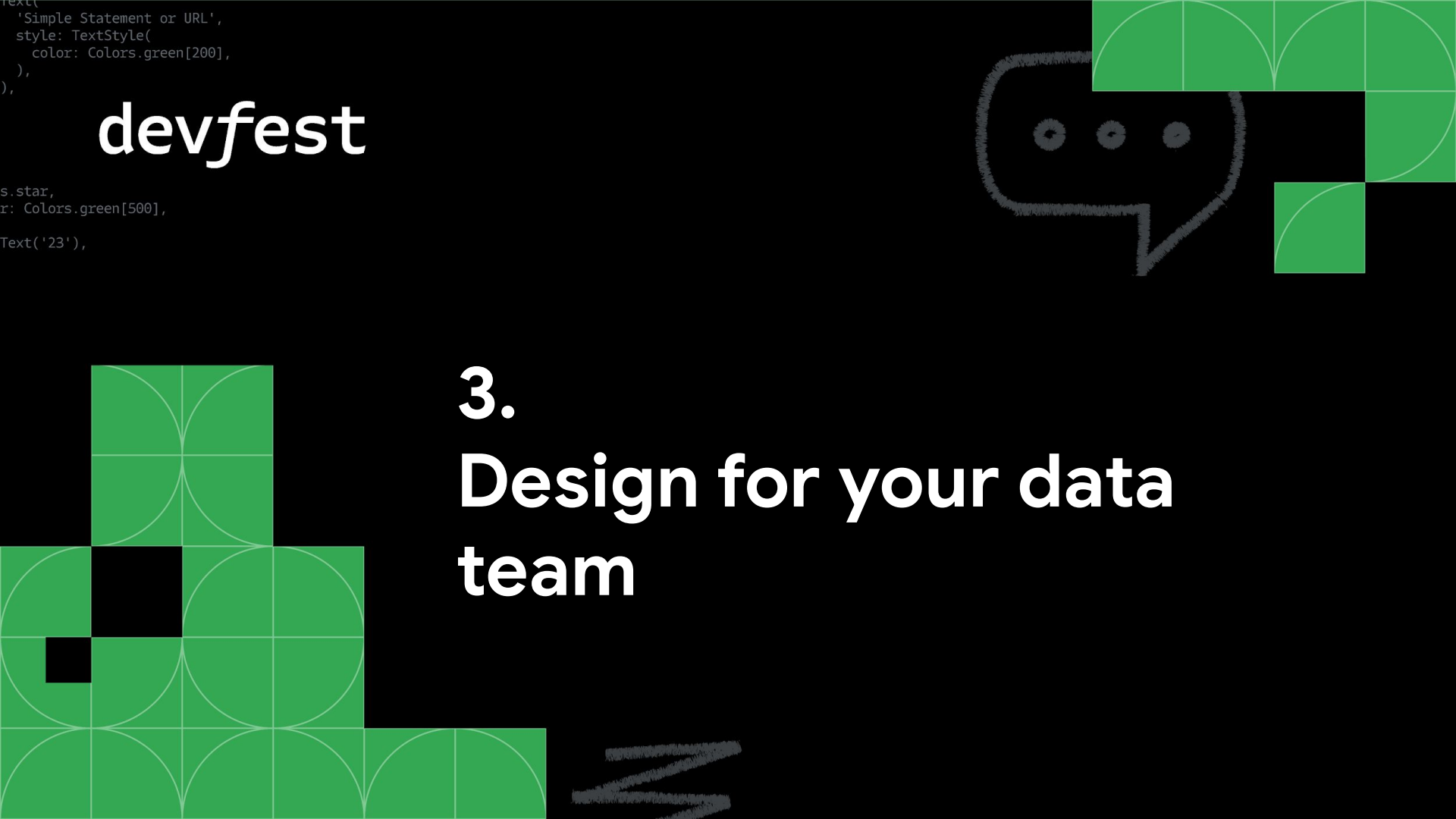
Recommended strategic journey to become awesome at Data & AI



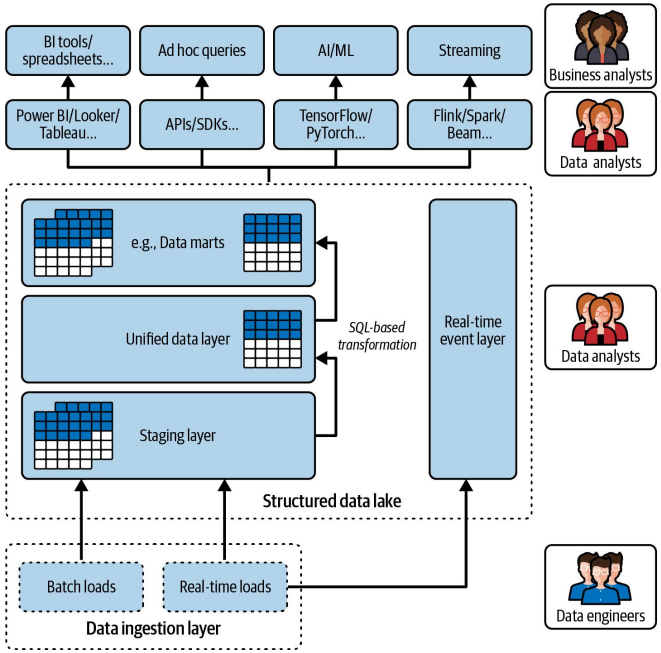




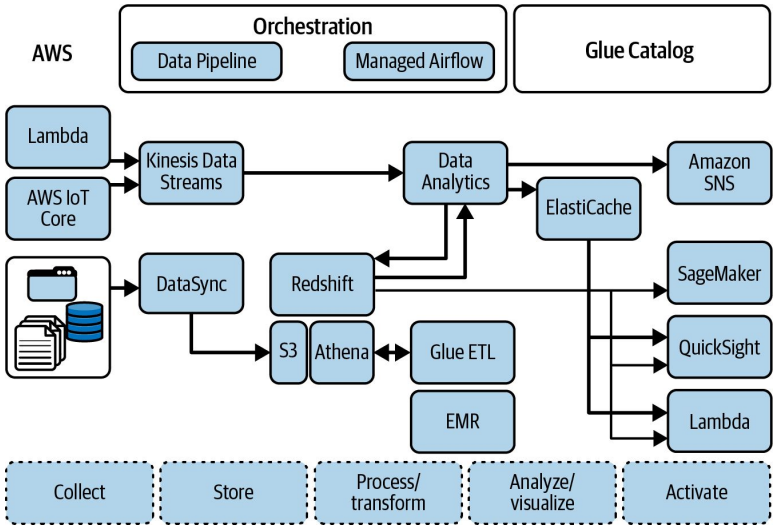
cost goes down even as capability goes up


'Simple Statement or URL',
style: TextStyle(
color: Colors.green[200],
),
),
devfest
s.star,
r: Colors.green[500],
Text('23'),
3.
Design for your data team

What type of organization are you?



For analyst-driven organization

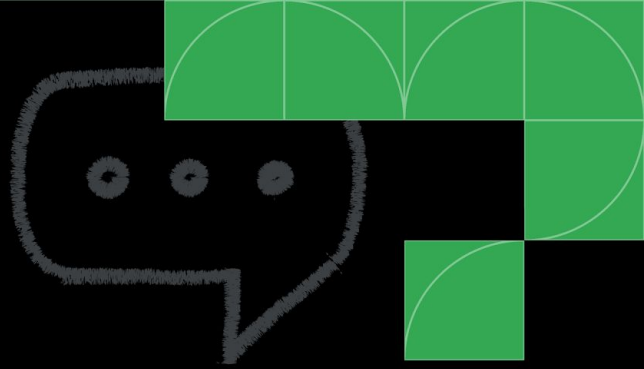


For eng-driven organization (AWS)

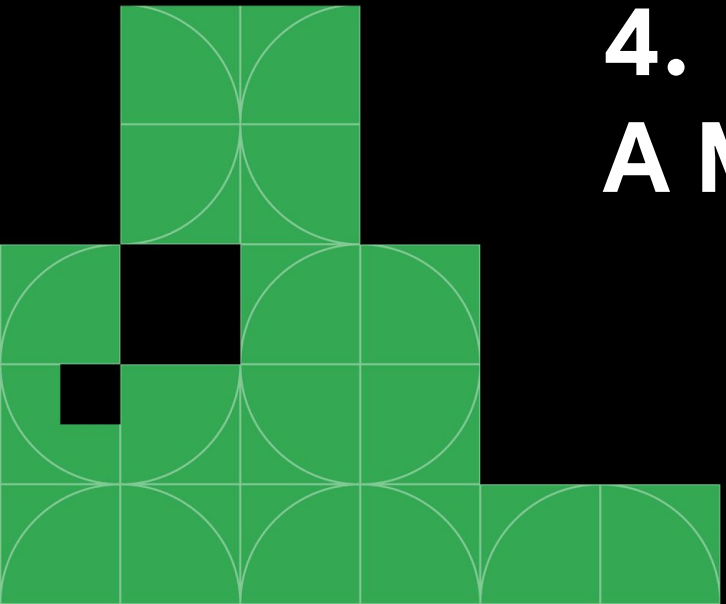
```
Text(
  'Simple Statement or URL',
  style: TextStyle(
    color: Colors.green[200],
  ),
),
```

devfest

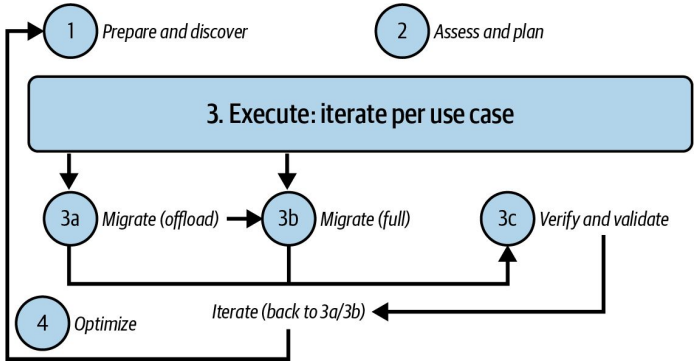
```
s.star,
r: Colors.green[500],

Text('23'),
```

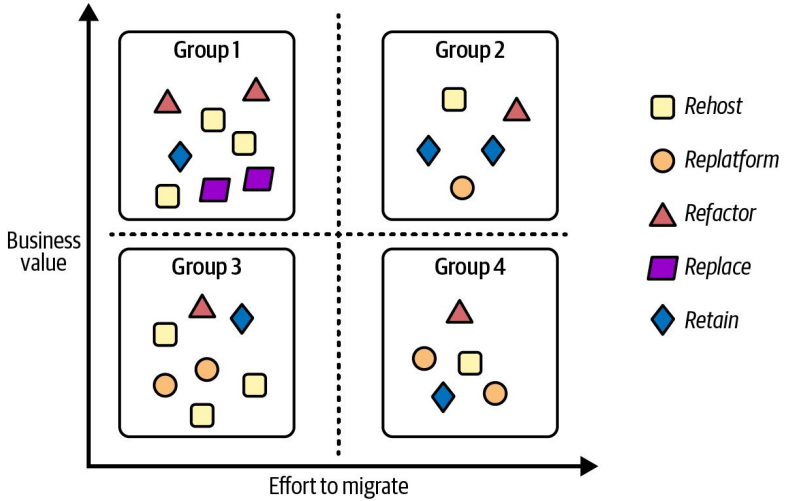


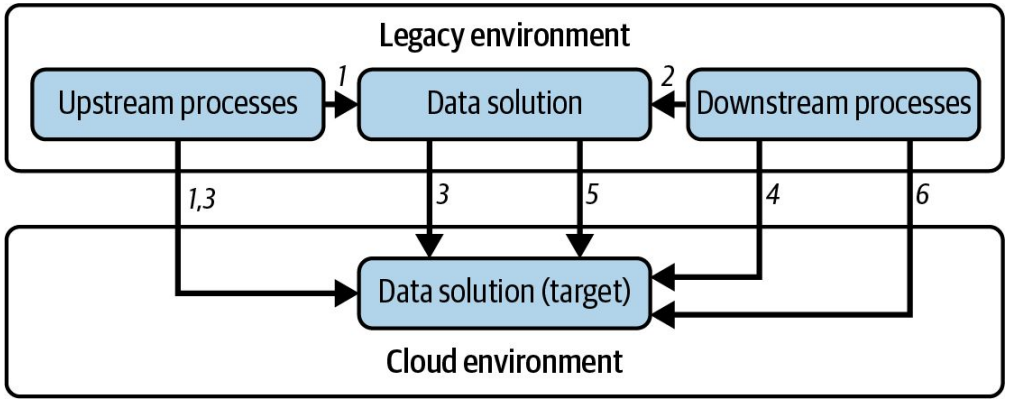
4.
A Migration Framework



Don’t bite off too much





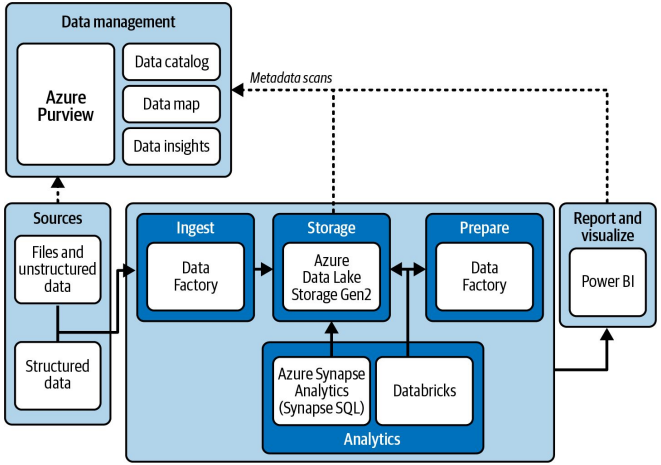


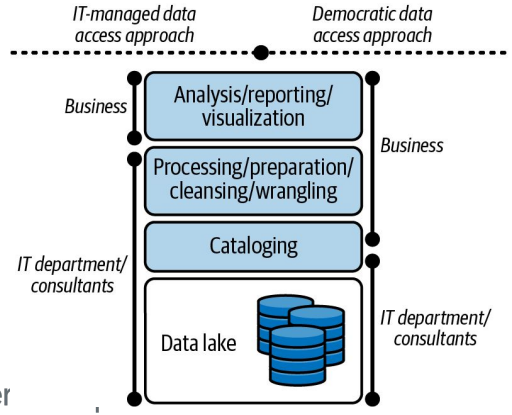
5.
Architecting a Data Lake

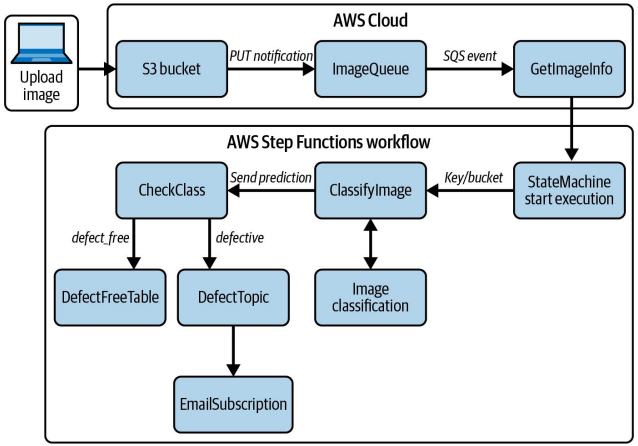
Building and using a data lake

Table 5-1. Hadoop solutions by environment

Use case	On premises, Databricks	AWS	Azure	Google Cloud Platform
Workflows	Airflow, Oozie	Data Pipeline, Airflow on EC2, EMR	HDInsight, Data Factory	Cloud Composer, Cloud Dataproc
Streaming ingest	Apache Kafka, MapR Streams	Kinesis, Kinesis Data Streams, Managed Kafka	Event Hubs	Cloud Pub/Sub, Confluent Apache Kafka
Streaming computation	Beam, Storm	Beam on Flink, Kinesis Data Streams	Beam on HDInsight, Stream Analytics	Cloud Dataflow
SQL	Drill, Hive, Impala	Athena, Redshift	Synapse, HDInsight	BigQuery
NoSQL	HBase, Cassandra	DynamoDB	Cosmos DB	Cloud Bigtable
Filesystem	HDFS, Iceberg, Delta Lake	EMR	HDInsight, Data Lake Storage	Cloud Dataproc
Security	Sentry, Ranger, Knox	AWS IAM	Azure IAM	Cloud IAM, Dataplex
Batch computation	Spark	EMR	HDInsight, Databricks	Cloud Dataproc, Serverless Spark



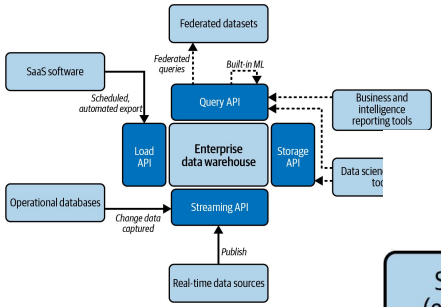




Google Developer

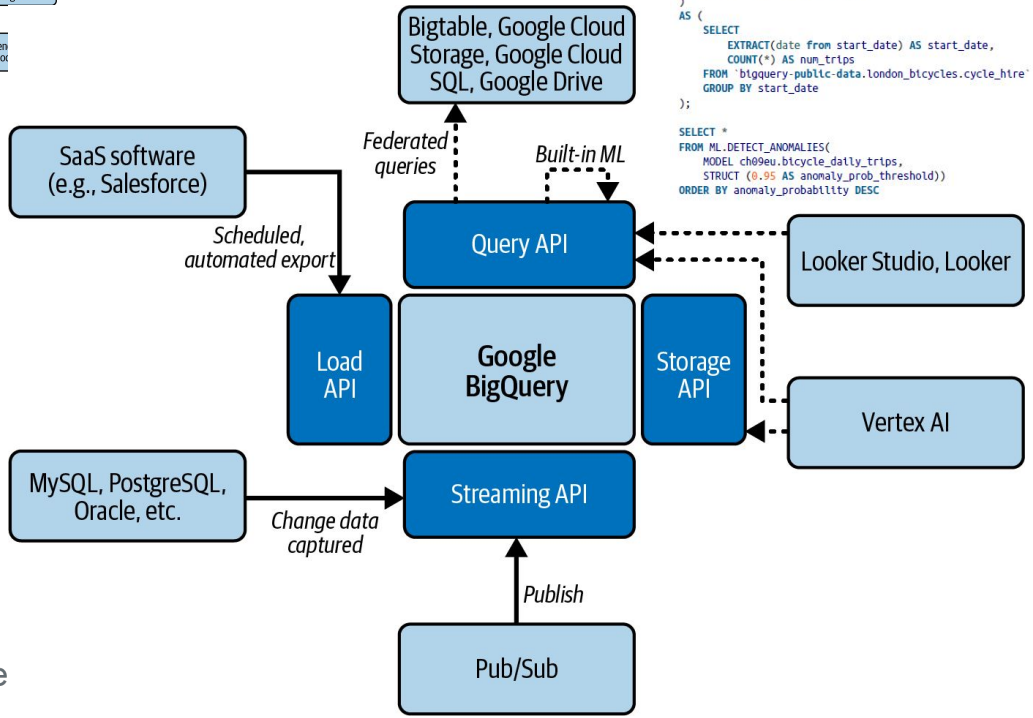

'Simple Statement or URL',
style: TextStyle(
color: Colors.green[200],
),
),
devfest
s.star,
r: Colors.green[500],
Text('23'),
6.
Innovating with a data warehouse

Hub-and-spoke is the ideal architecture for analyst-orgs



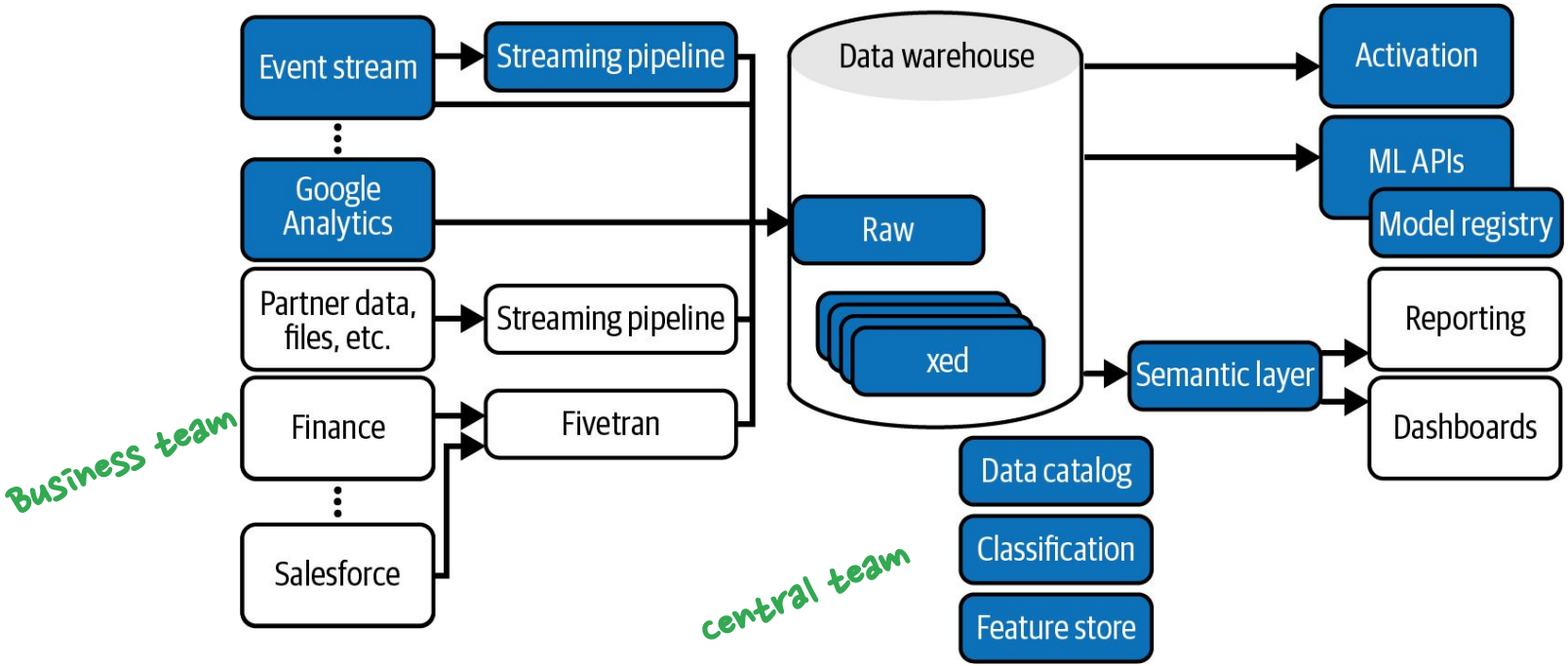
Example 6-8. Anomaly detection model developed in SQL in BigQuery

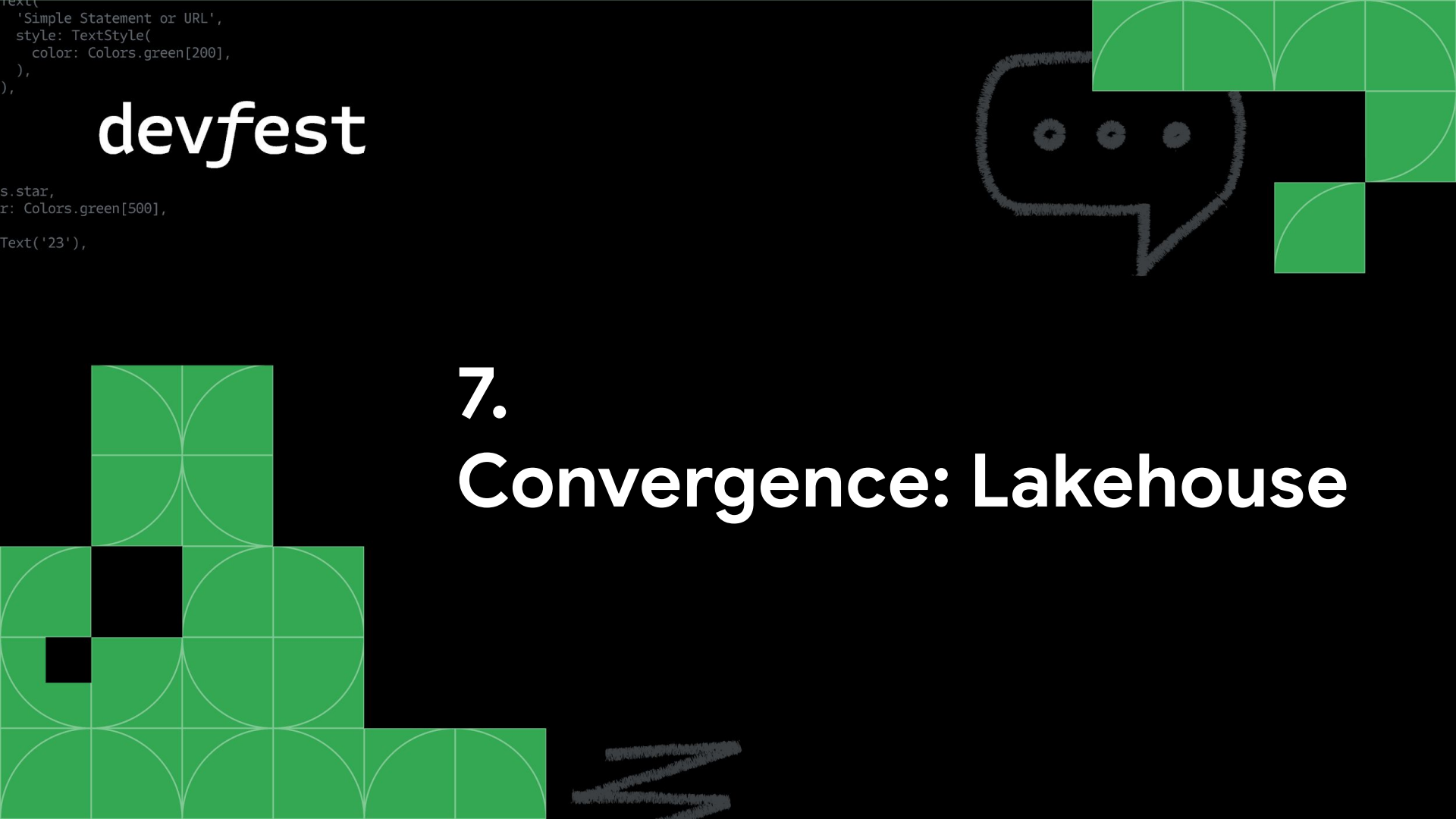
```
CREATE OR REPLACE MODEL ch09eu.bicycle_daily_trips
OPTIONS(
    model_type='arima_plus',
    TIME_SERIES_DATA_COL='num_trips',
    TIME_SERIES_TIMESTAMP_COL='start_date',
    DECOMPOSE_TIME_SERIES=TRUE
)
AS (
    SELECT
        EXTRACT(date from start_date) AS start_date,
        COUNT(*) AS num_trips
    FROM `bigquery-public-data.london_bicycles.cycle_hire`
    GROUP BY start_date
);

SELECT *
FROM ML.DETECT_ANOMALIES(
    MODEL ch09eu.bicycle_daily_trips,
    STRUCT (0.95 AS anomaly_prob_threshold))
ORDER BY anomaly_probability DESC
```



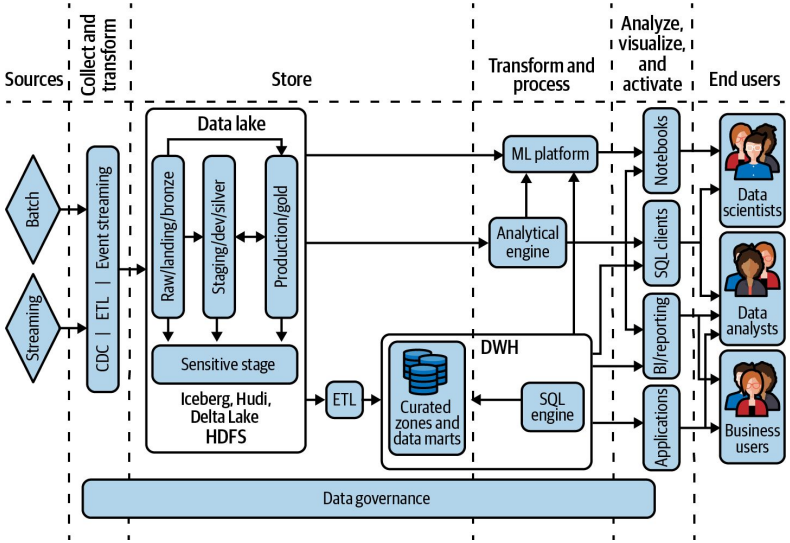
Google Develope

Central data eng vs. business: roles & responsibilities



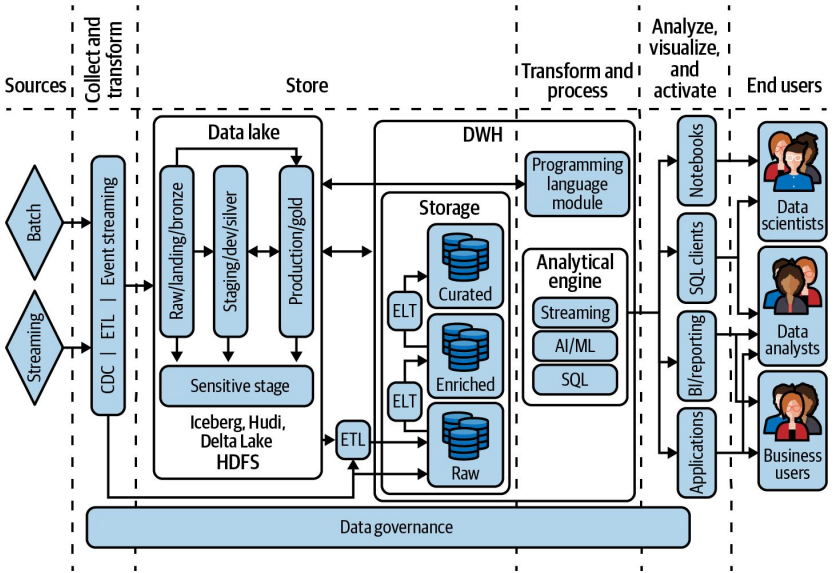

'Simple Statement or URL',
style: TextStyle(
color: Colors.green[200],
),
),
devfest
s.star,
r: Colors.green[500],
Text('23'),
7.
Convergence: Lakehouse

You can build a lakehouse on storage, or SQL-first



Transformations using Spark

Query data stored in Iceberg using SQL

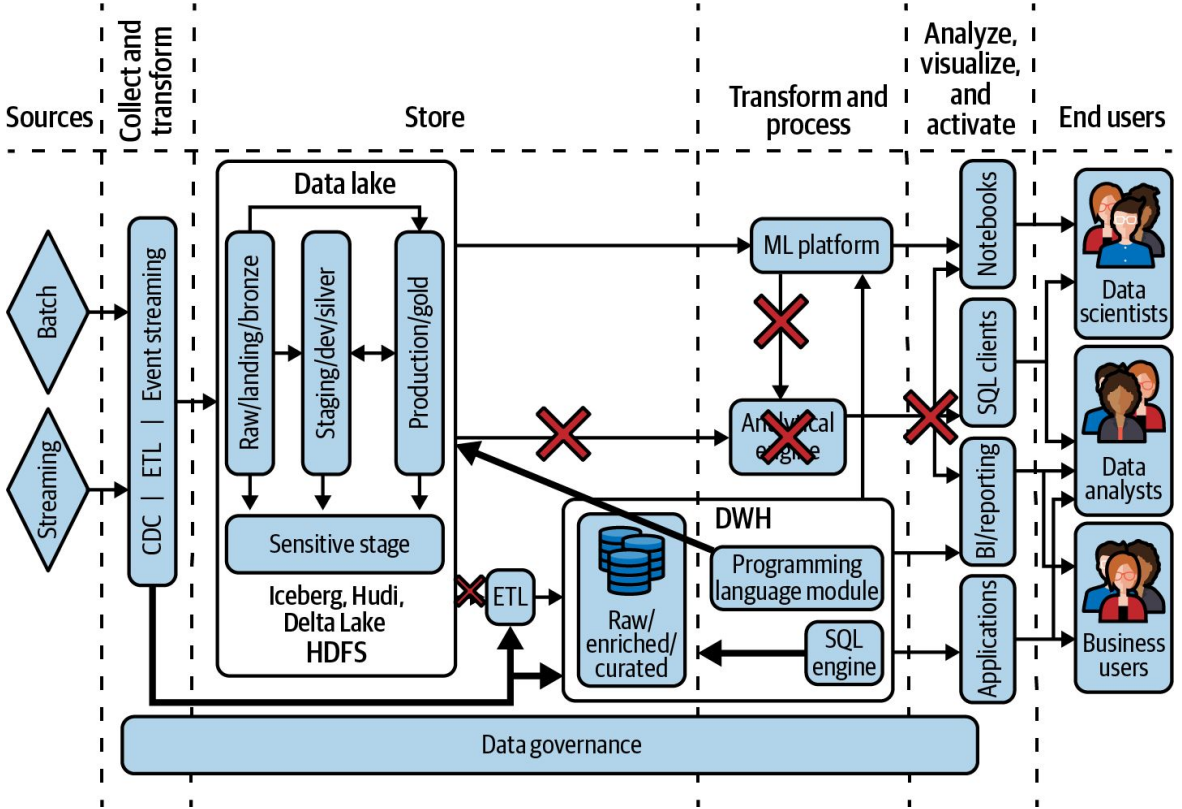


Transformations using SQL (dbt?)

Train models on data stored in DW native storage using Tensorflow

Google Developer Groups

You can migrate from legacy to either lakehouse



A lakehouse is a great destination because …

The Benefits of Convergence

If you are a startup or in the fortunate situation of doing greenfield development, start with a pure data lake or a pure DWH, depending on your use case and skill set (see Chapter 3). For everyone else, we recommend the lakehouse architecture. Regardless of whether you choose a lakehouse-on-storage or a SQL-first lakehouse, choosing a lakehouse architecture provides the following benefits:

Time to market
: You can ingest and use data straightaway, whether from batch or real-time data sources. Rather than employing complex ETL pipelines to process data, data is "staged" either in a messaging bus or through object storage. Then it is transformed within the converged DWH/data lakes, enabling users to act as the data is received.

Reduced risk
: You can continue leveraging existing tools and applications without rewriting them. This reduces the risk and costs associated with change.

Predictive analytics
: Moving away from the traditional view of data marts and data mining to real-time decision making using fresh data increases business value. This is only possible because the governance and strictness around DWHs have come down, reducing barriers to entry.

Data sharing
: The converged environment is now the one-stop shop for all the types of users (i.e., data analyst, data engineer, and data scientist) you may have. They can all access the same managed environment, getting access to different stages of data when they need it. At the same time, different roles can have access to the same data through different layers, and this is governed by platform-wide access rights. This not only increases the data governance but also allows simpler access management and auditing throughout the data ecosystem.

ACID transactions
: In a typical DWH, the data integrity is maintained, and multiple users reading and writing the data see a consistent copy of the data. Although ACID is a key feature in the majority of the databases, traditionally it has been rather difficult to provide the same guarantees when it comes to traditional HDFS-based data lakes. There are schemes such as Delta Lake and Apache Iceberg that try to maintain ACID semantics (refer to "The Evolution of Data Lake with Apache Iceberg, Apache Hudi, and Delta Lake" on page 136); they store a transaction log with the aim of keeping track of all the commits made to a data source.

Multimodal data support
: Semistructured and structured data are key differentiators with the DWHs and data lakes. Semistructured data has some organizational properties such as semantic tags or metadata to make it easier to organize, but data still does not conform to a strict schema. In the converged world, this is accommodated with extended semistructured data support. On the other hand, for unstructured use cases, data lakes are still required apart from edge cases.

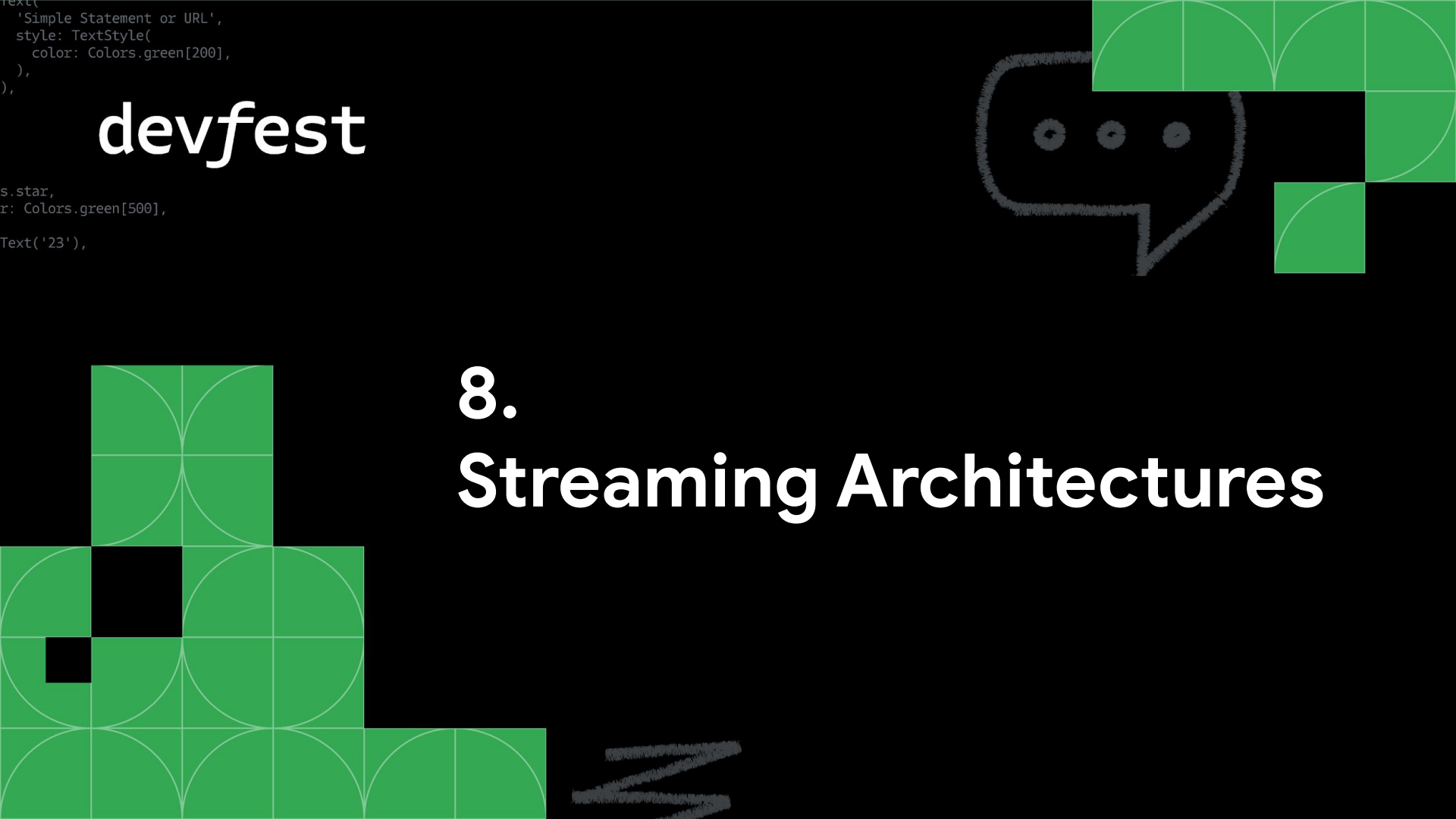
Unified environment
: Traditionally, different tools and environments, usually orchestrated by ETLs, manage data capture, ingest, storage, processing, and serving. In addition, processing frameworks such as Spark, Storm, Beam, etc., provide built-in ETL templates to enable organizations to build ETL pipelines. However, with capable cloud EDWs and integrated cloud tools, this pipeline is now all handled by a single environment. ELT does most of the traditional ETL tasks such as cleanse, dedupe, join, and enrich. This is made possible at different stages of the data lake implementation within the DWH. Furthermore, with the support of core DWHs, you can have access through a united environment to concepts such as stored procedures, scripting, and materialized views.

Schema and governance
: In reality, business requirements and challenges evolve over time. As a result, the associated data changes and accumulates, either by adapting to new data or by introducing new dimensions. As the data changes, applying data quality rules becomes more challenging and requires schema enforcement and evolution. Furthermore, PII data governance becomes more important as new data sources are added. There needs to be a data governance solution allowing organizations to have a holistic view of their data environment. In addition, it is paramount to have the ability to identify and mask PII data for different purposes and personas.

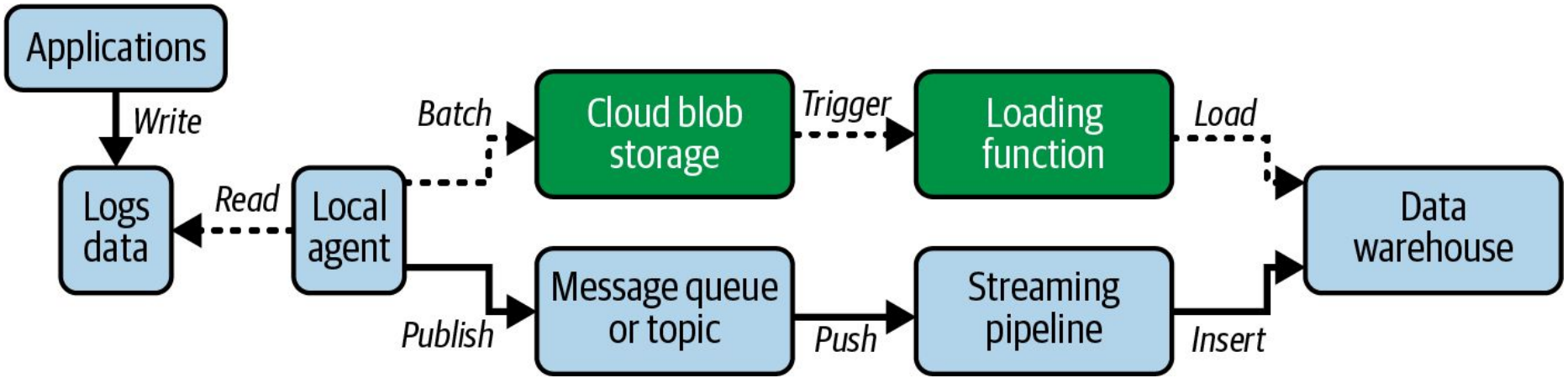
Streaming analytics
: Real-time analytics enables immediate responses, and there would be specific use cases where an extremely low-latency anomaly detection application is required to run. In other words, business requirements would be such that it has to be acted upon as the data arrives on the fly. Processing this type of data or application requires transformation done outside of the warehouse.

Having a single system to manage simplifies the enterprise data infrastructure and allows users to work more efficiently.

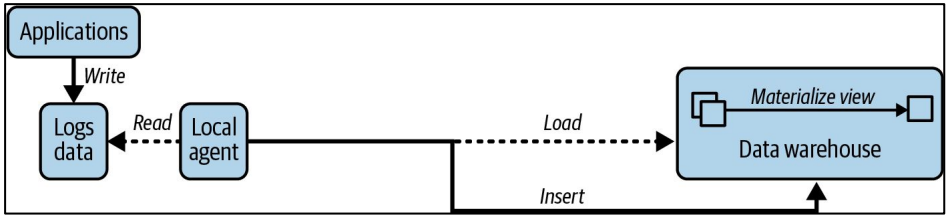
Single system for all workloads

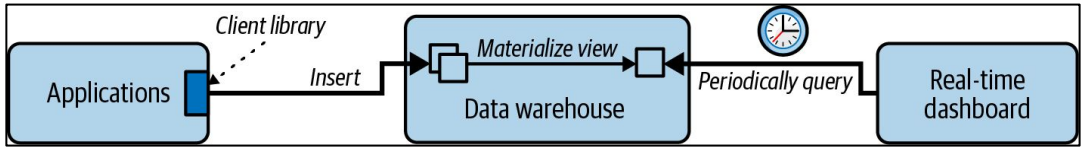

devfest
8.
Streaming Architectures

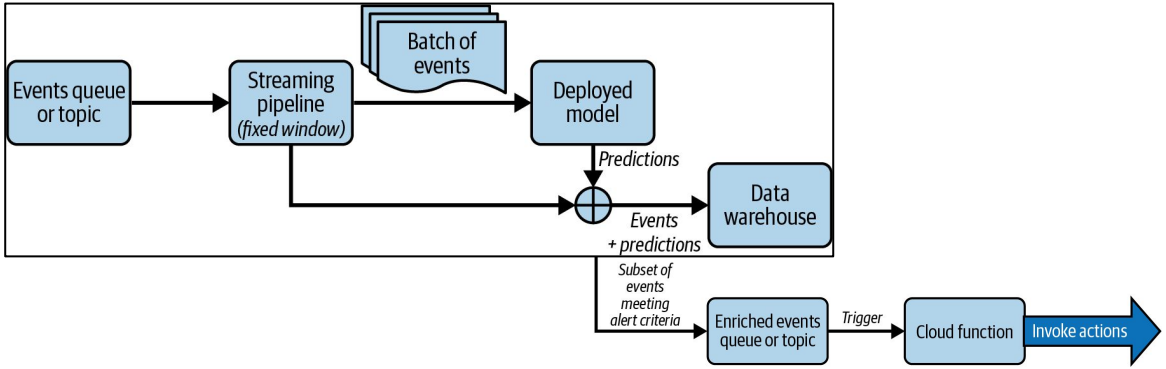
Streaming ETL: microbatch or full streaming



Streaming ELT/Insert, Live Querying, Streaming ML








'Simple Statement or URL',
style: TextStyle(
color: Colors.green[200],
),
),
devfest
s.star,
r: Colors.green[500],
Text('23'),
9.
Hybrid and Edge

The ideal architecture is a Single Pane of Glass

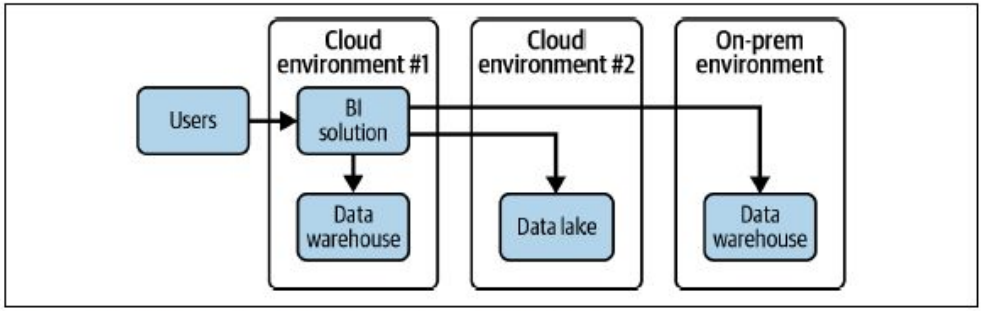


Figure 9-1. Single pane of glass leveraging a BI solution

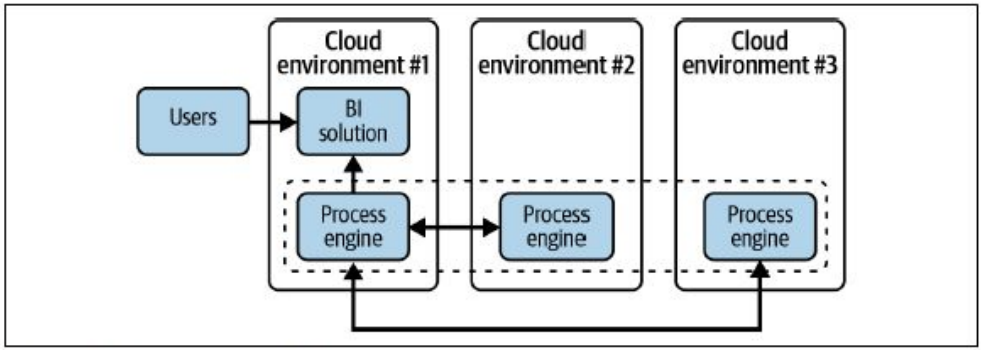
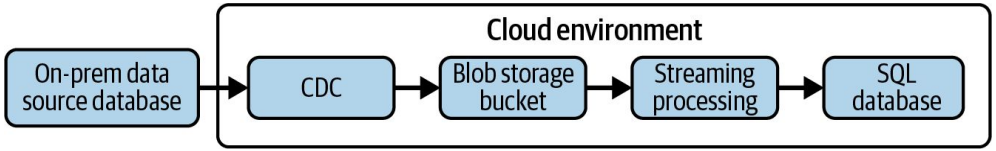
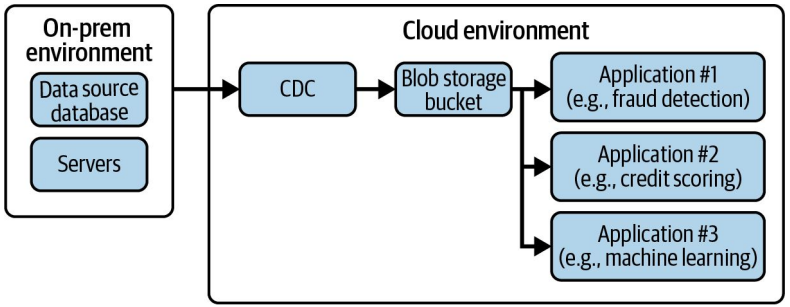


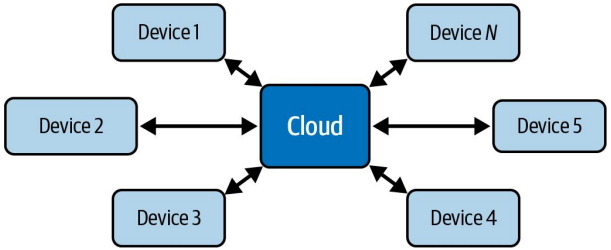
Figure 9-2. Single pane of glass leveraging a distributed process engine solution

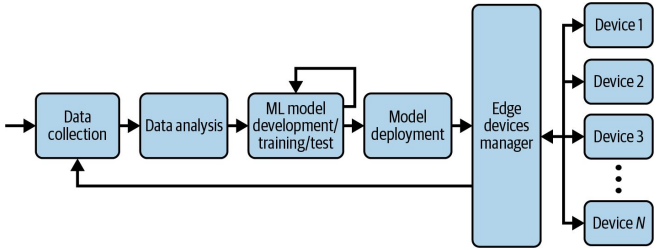
Extend environments through CDC

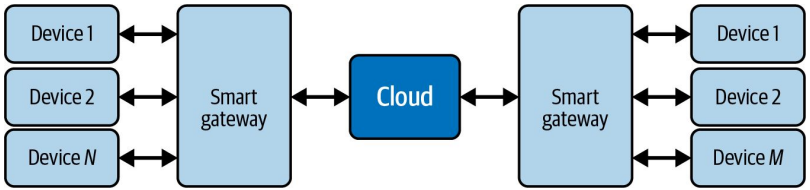




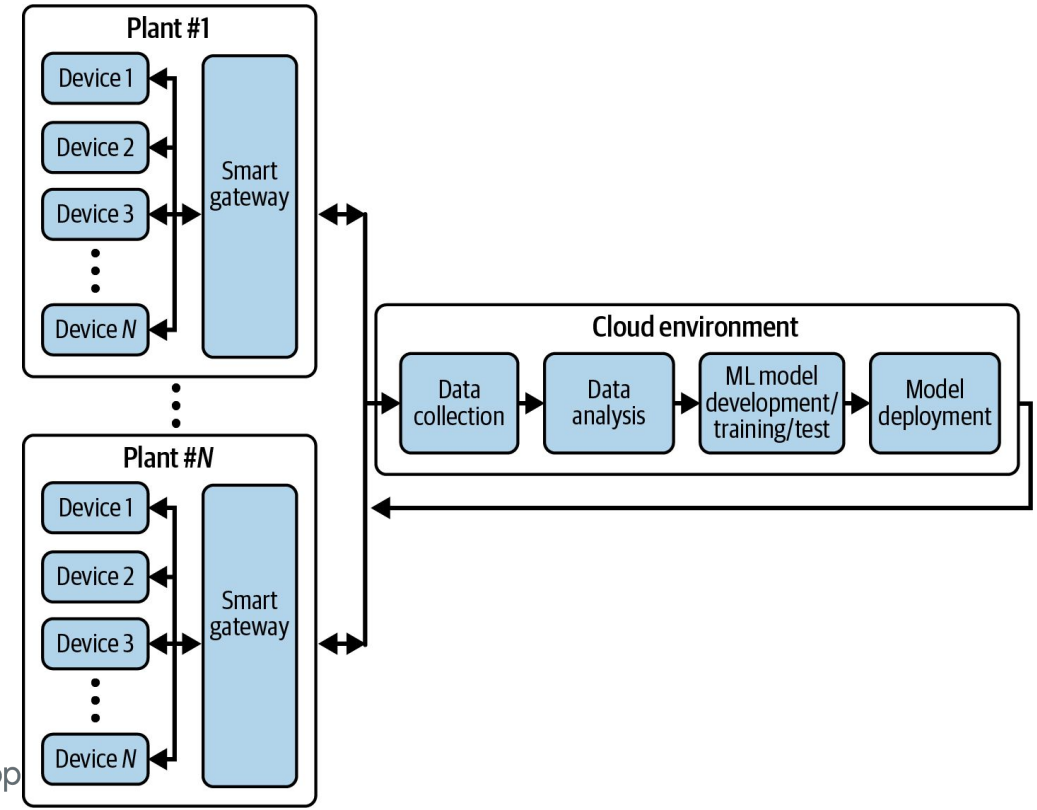
Smart Devices vs. Smart Gateways vs. Device Managers

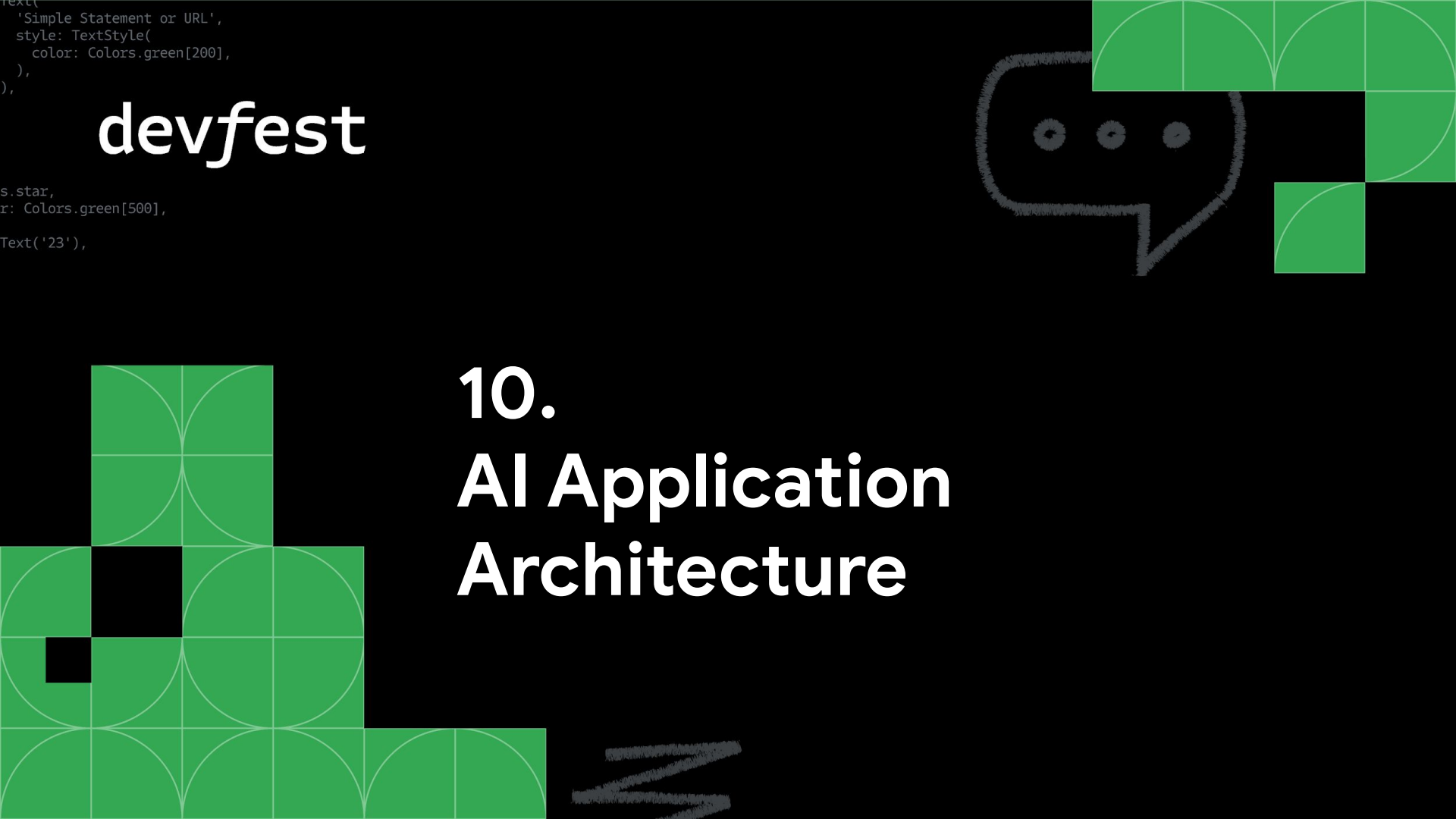






In reality, you'll have a mix




devfest
10.
AI Application
Architecture

Different AI applications are designed differently

- Chapter 10. AI Application Architecture
  - Is This an AI/ML Problem?
  - Buy, Adapt, or Build?
  - AI Architectures
    - Understanding Unstructured Data
    - Generating Unstructured Data
    - Predicting Outcomes
    - Forecasting Values
    - Anomaly Detection
    - Personalization
    - Automation
  - Responsible AI
  - Summary

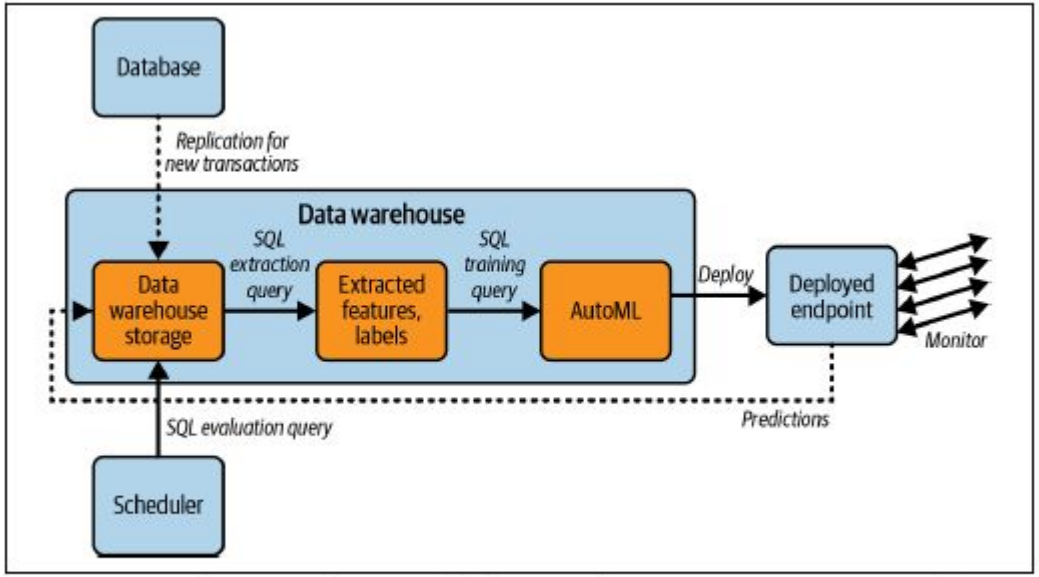
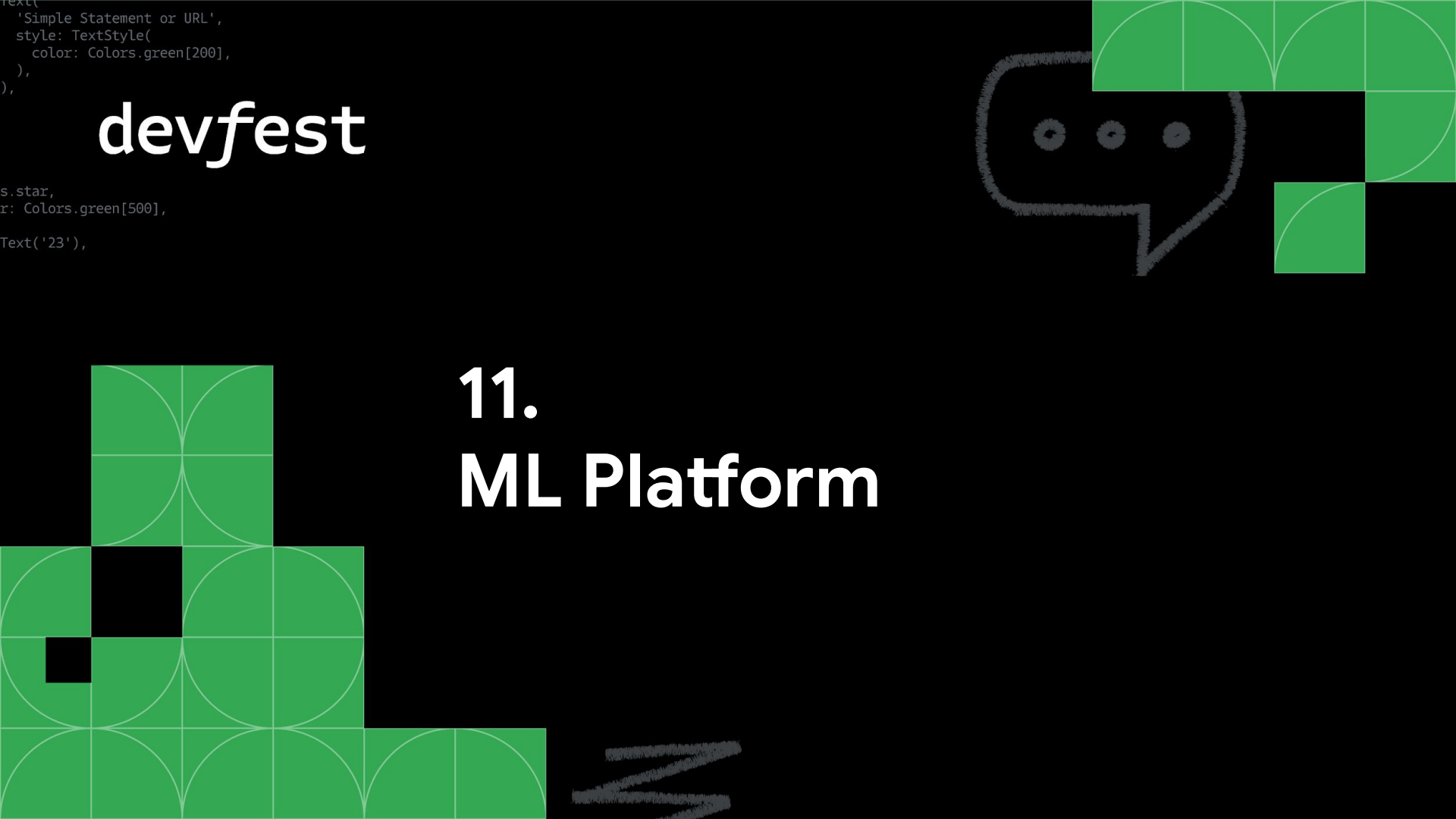
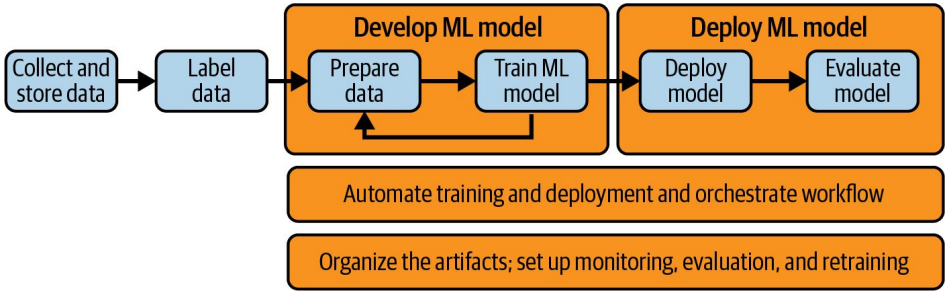
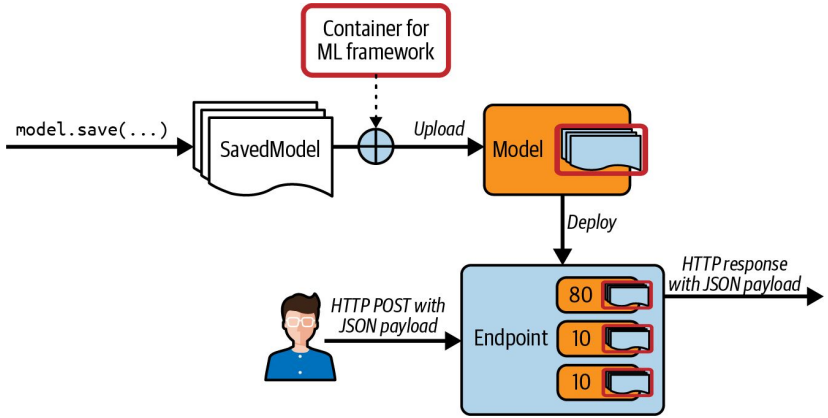


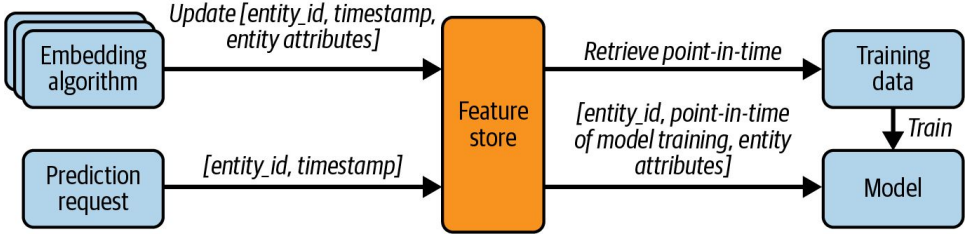
Figure 10-5. Architecture of training, deploying, and monitoring an outcome prediction model

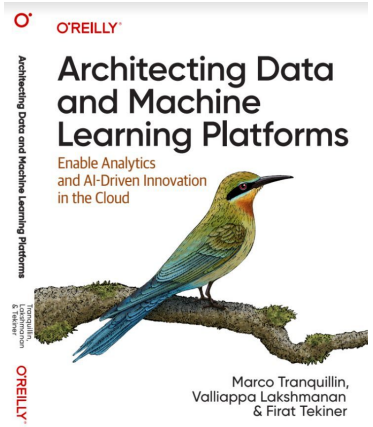

devfest
11.
ML Platform

Your ML platform needs to support developing and deploying all those types of AI applications









Book (Amazon)

Link to Slides

Thank you!

Grazie!
நன்றி!
ευχαριστώ!

Google Developer Groups

Follow me on LinkedIn