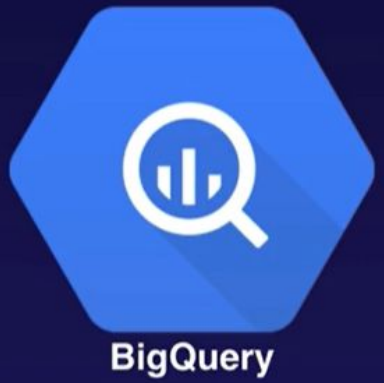
GCP

Data Analytics with BigQuery

BigQuery basics

Introducing BigQuery



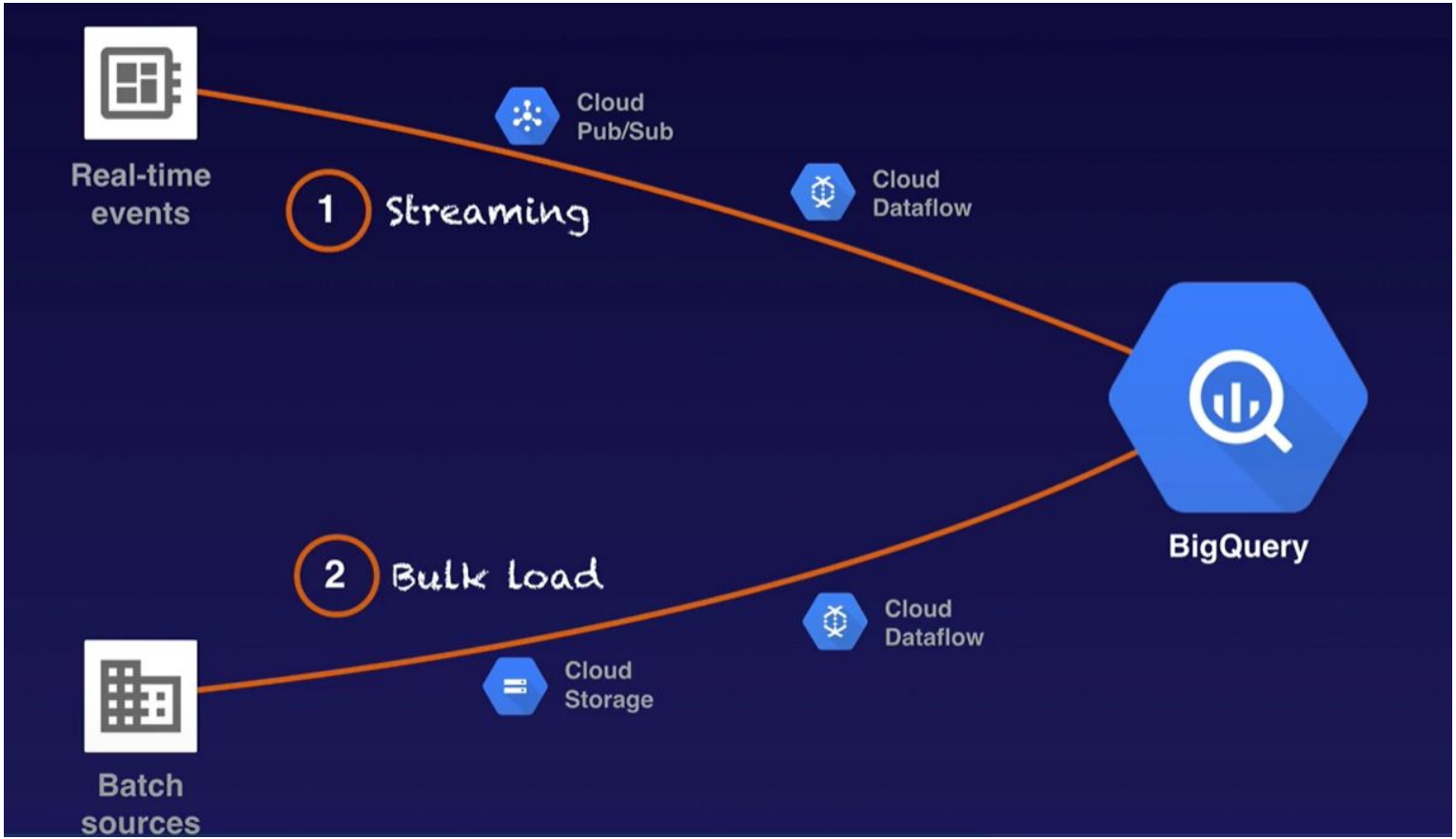
- Peta-byte scale, serverless, highly-scalable cloud enterprise data warehouse
- In-memory BI Engine (BigQuery BI Engine)
- Machine-learning capabilities (BigQuery ML)
- Support for geospatial data storage and processing

key features

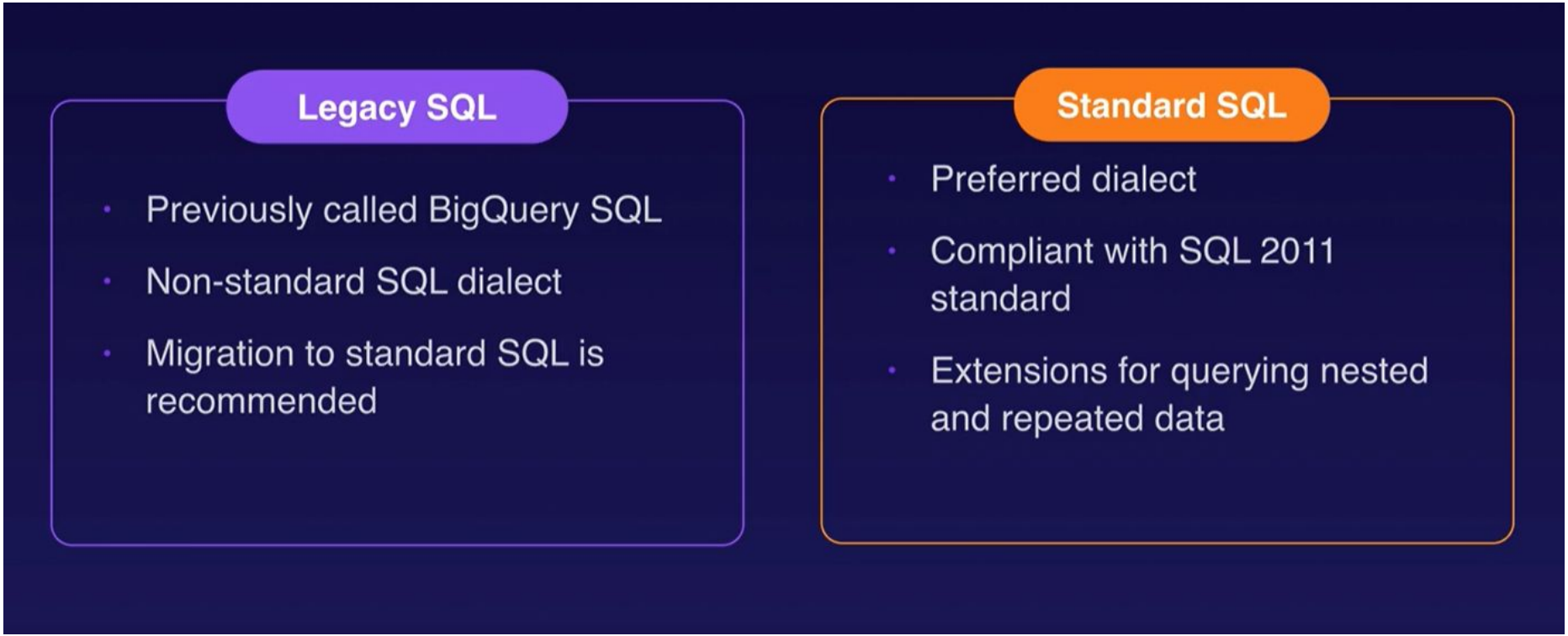
Interacting with BigQuery

Managing Data

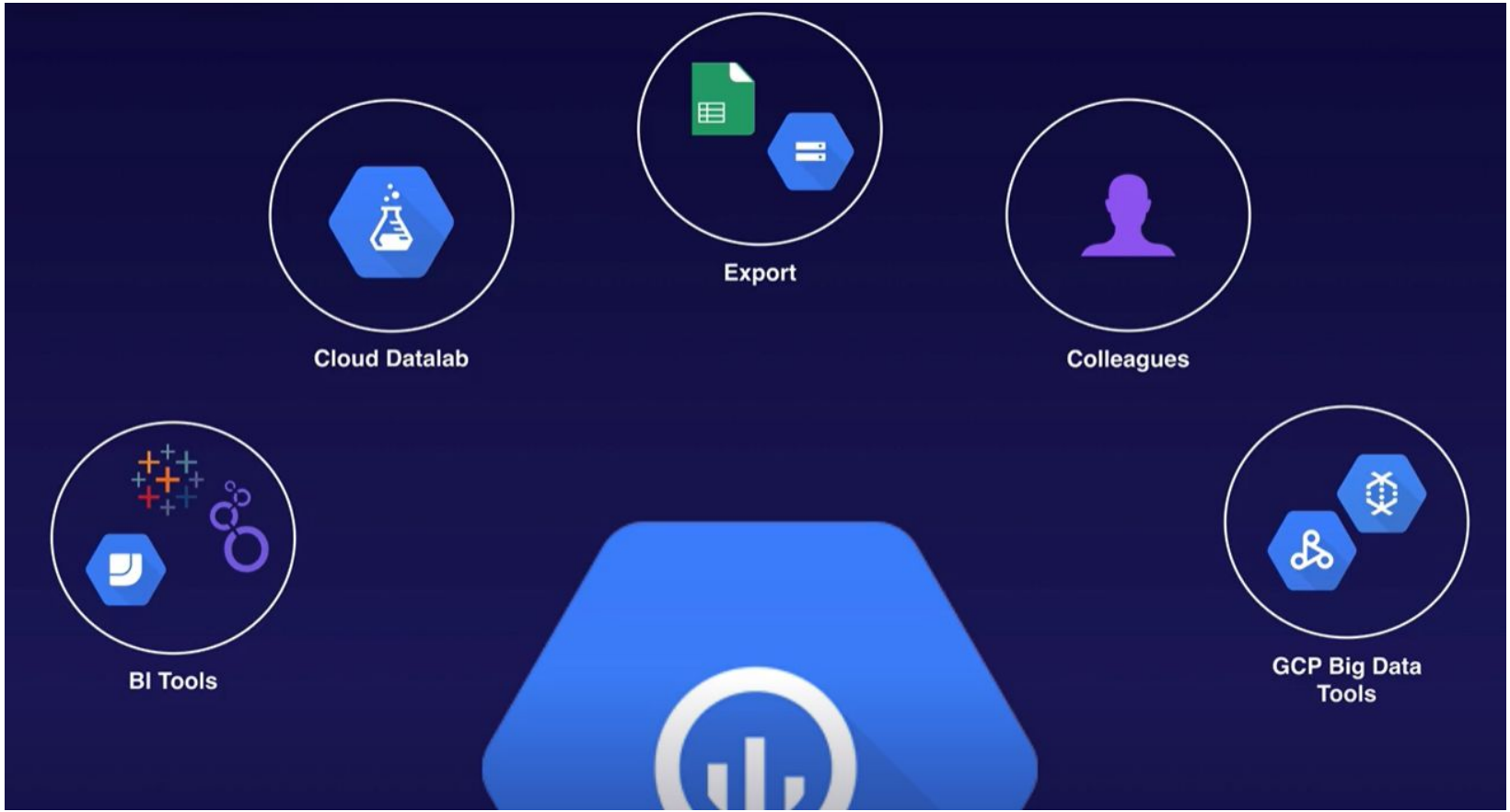
Data ingestion



TheBigQuery SQL Dialect



Using Data in BigQuery



BigQuery Jobs and Operations

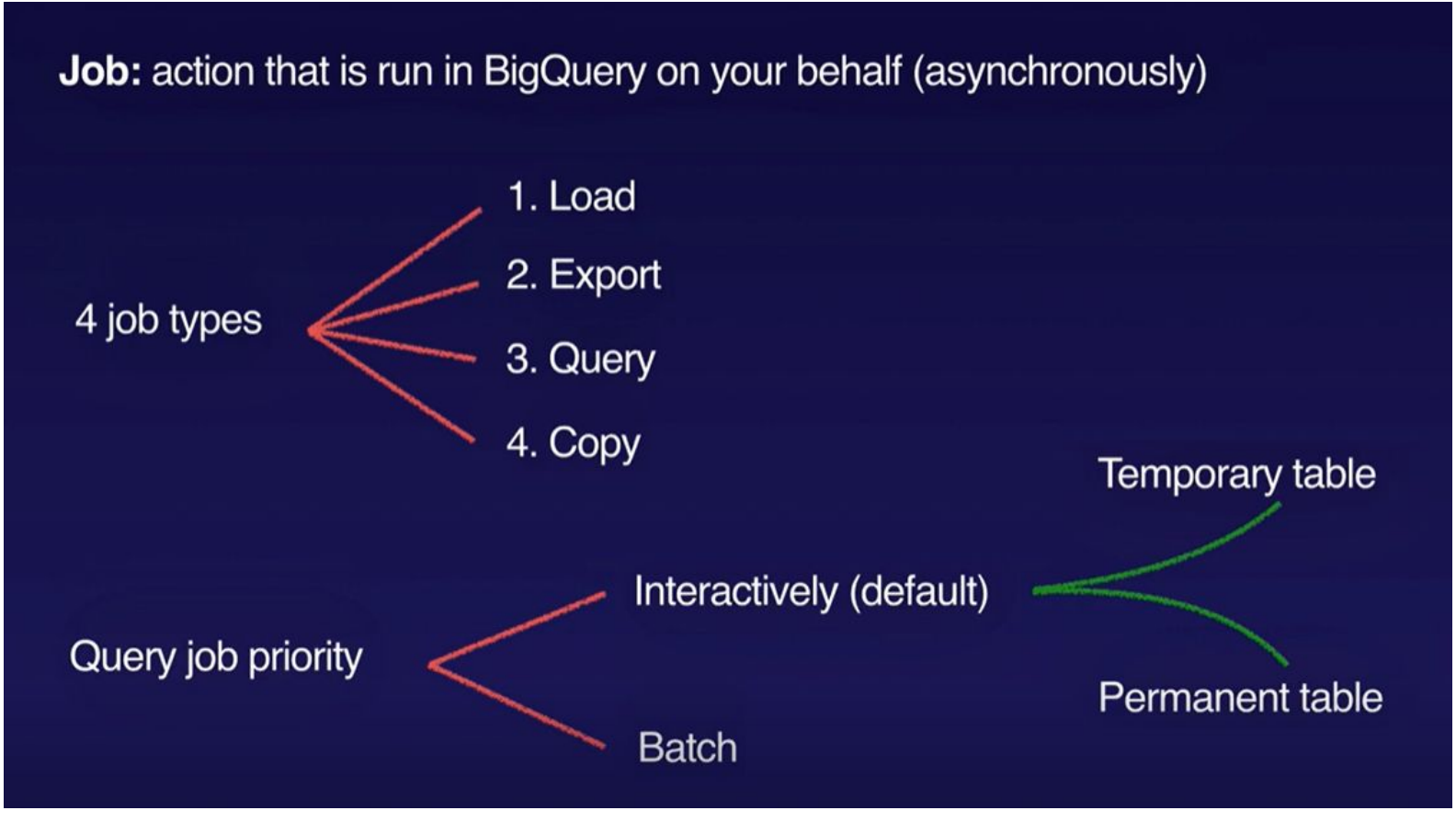


Table Storage in BigQuery

- **Capacitor** columnar data format
- Tables can be partitioned
- Individual records exist as rows
- Each record is composed of columns
- Table schemes specified at creation of table or at data load

Table A - Partition 1

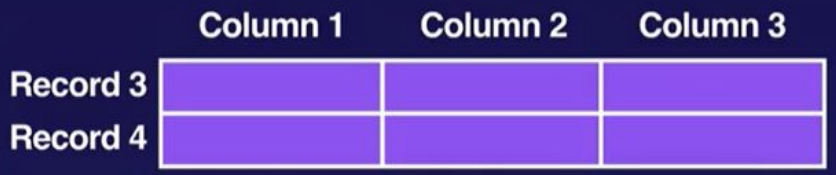
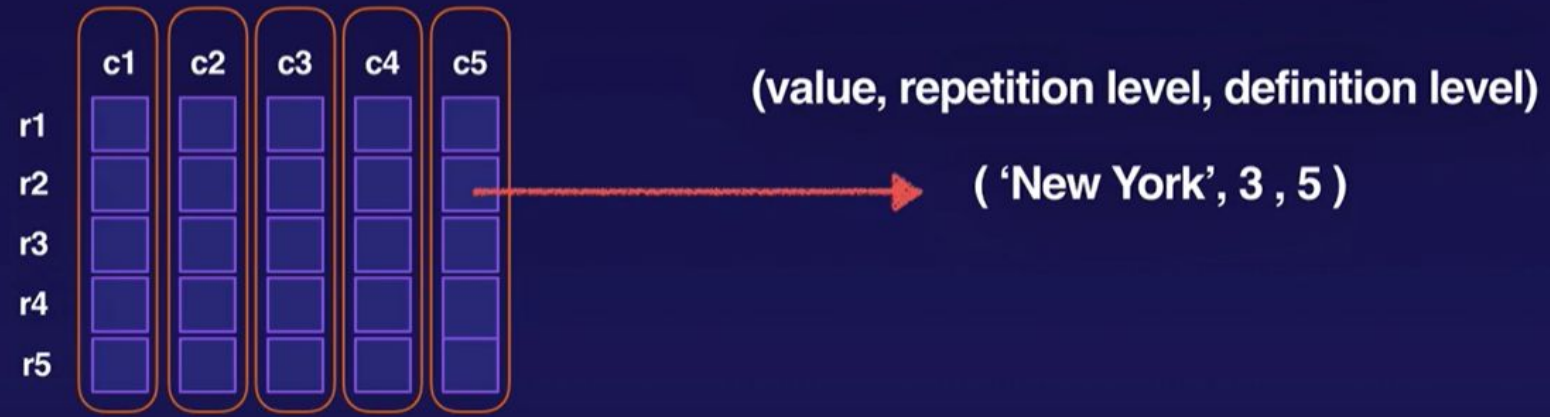


Table A - Partition 2

BigQuery Capacitor

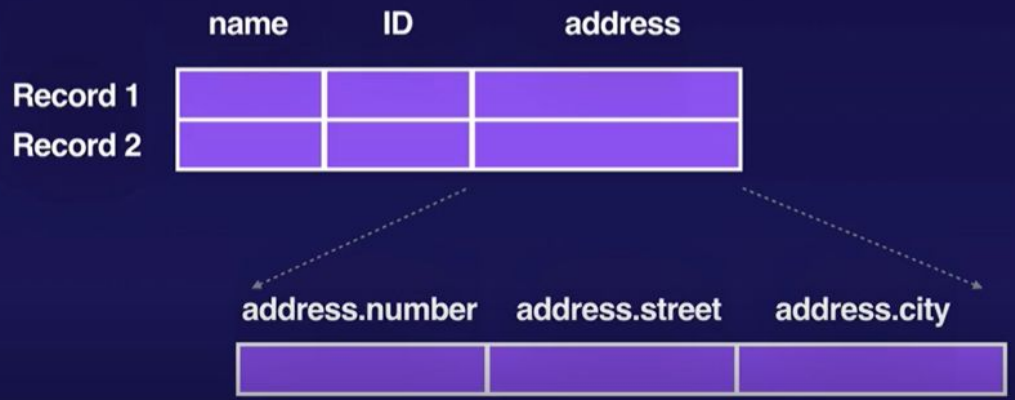
Capacitor storage system

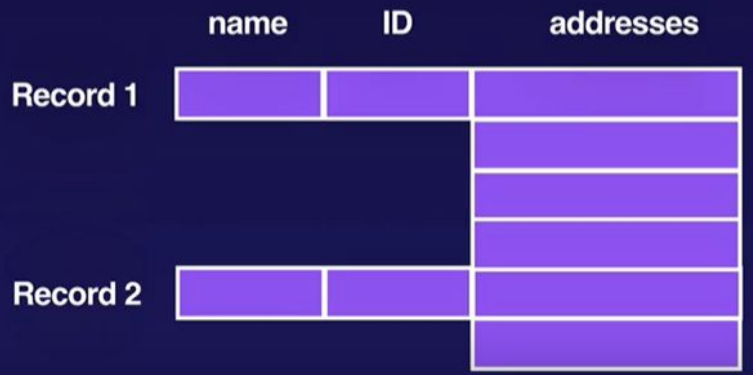
- Proprietary columnar data storage that supports semi-structured data (nested and repeated fields)
- Each **value** stored together with a **repetition level** and a **definition level**



Denormalisation

- BigQuery performance optimised when data is denormalised appropriately
- Nested and repeated columns
- Maintain data relationships in an efficient manner
- **RECORD (STRUCT)** data type



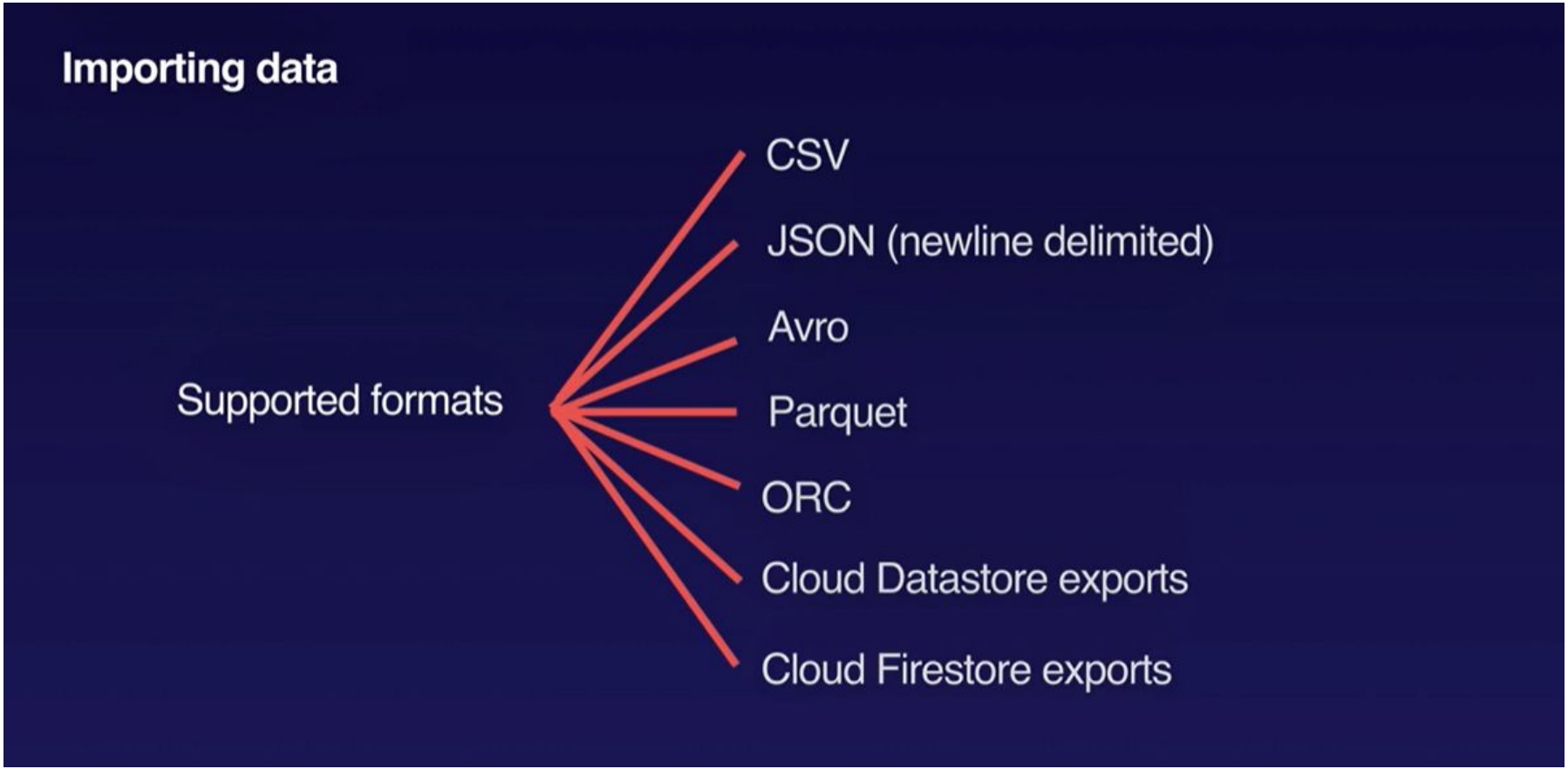


Nested and repeated fields

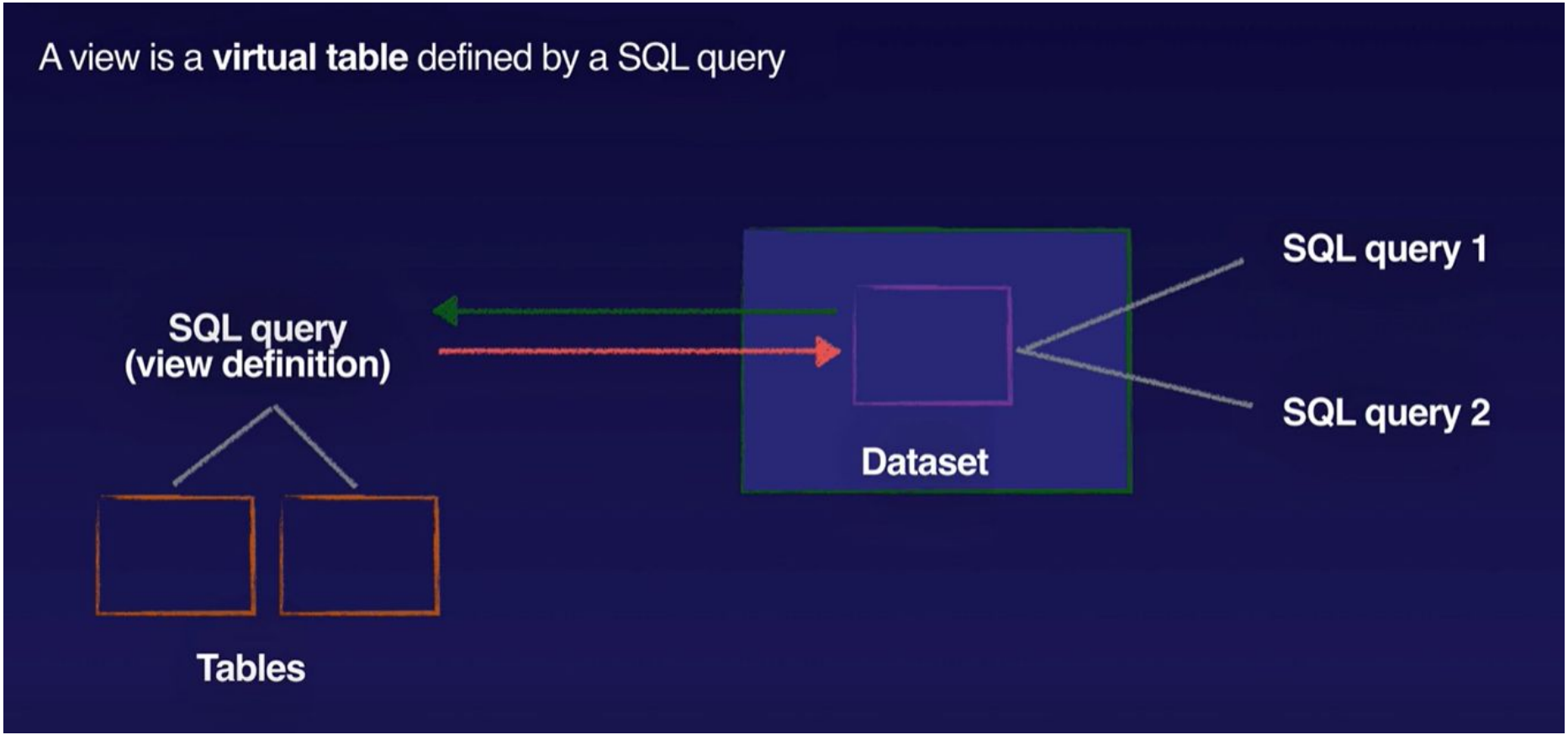
- `id`
- `first_name`
- `last_name`
- `dob` (date of birth)
- `addresses` (a nested and repeated field)
  - `addresses.status` (current or previous)
  - `addresses.address`
  - `addresses.city`
  - `addresses.state`
  - `addresses.zip`
  - `addresses.numberOfYears` (years at the address)

{“id":"1","first_name":"John","last_name":"Doe","dob":"1968-01-22","addresses":[
{"status":"current","address":"123 First Avenue”,"city":"Seattle","state":"WA","zip":"11111","numberOfYears":"1"},
{"status":"previous","address":"456 Main Street","city":"Portland","state":"OR","zip":"22222","numberOfYears":"5"}]}

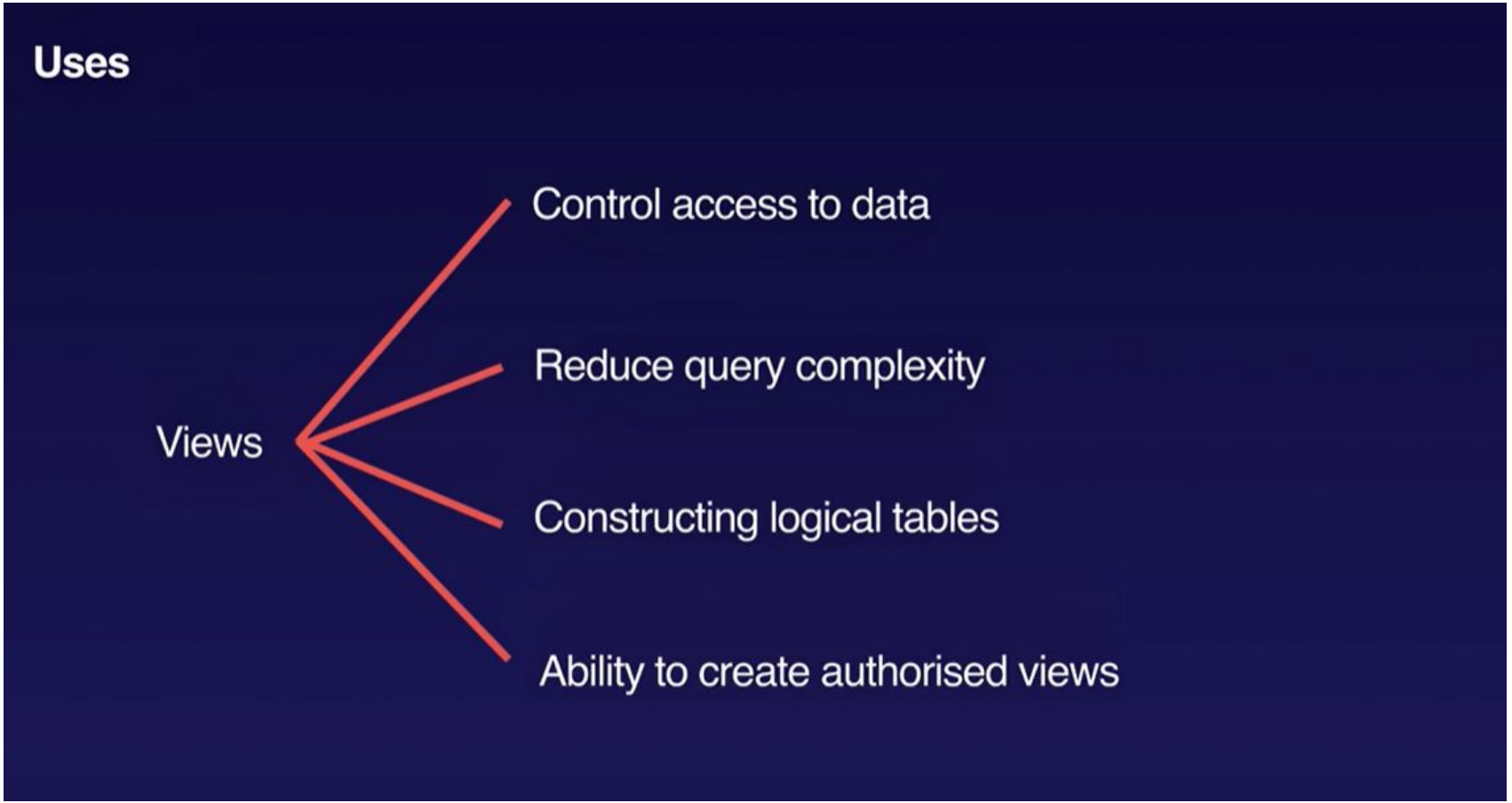
data Formats



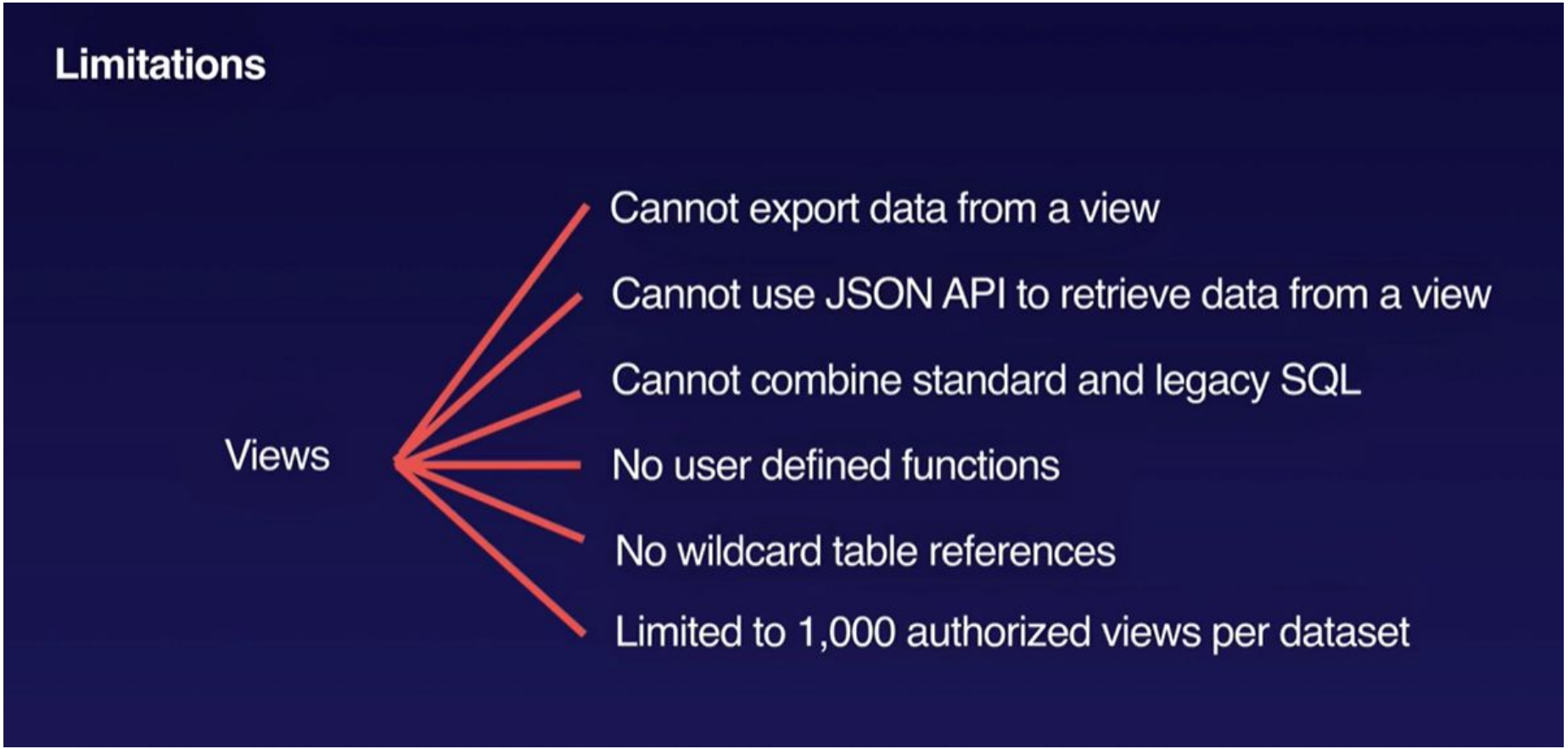
BigQuery Views



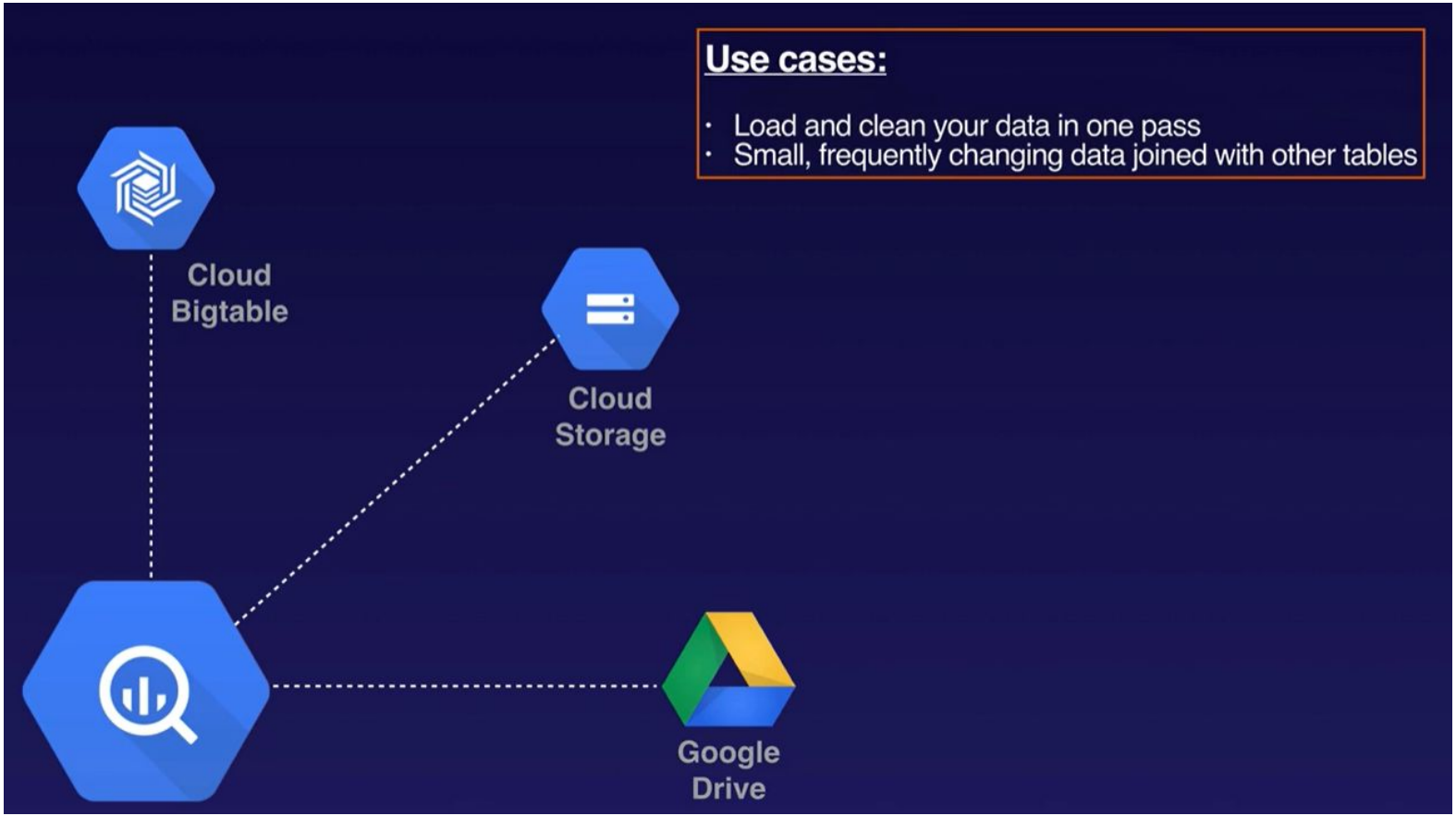
BigQuery Views



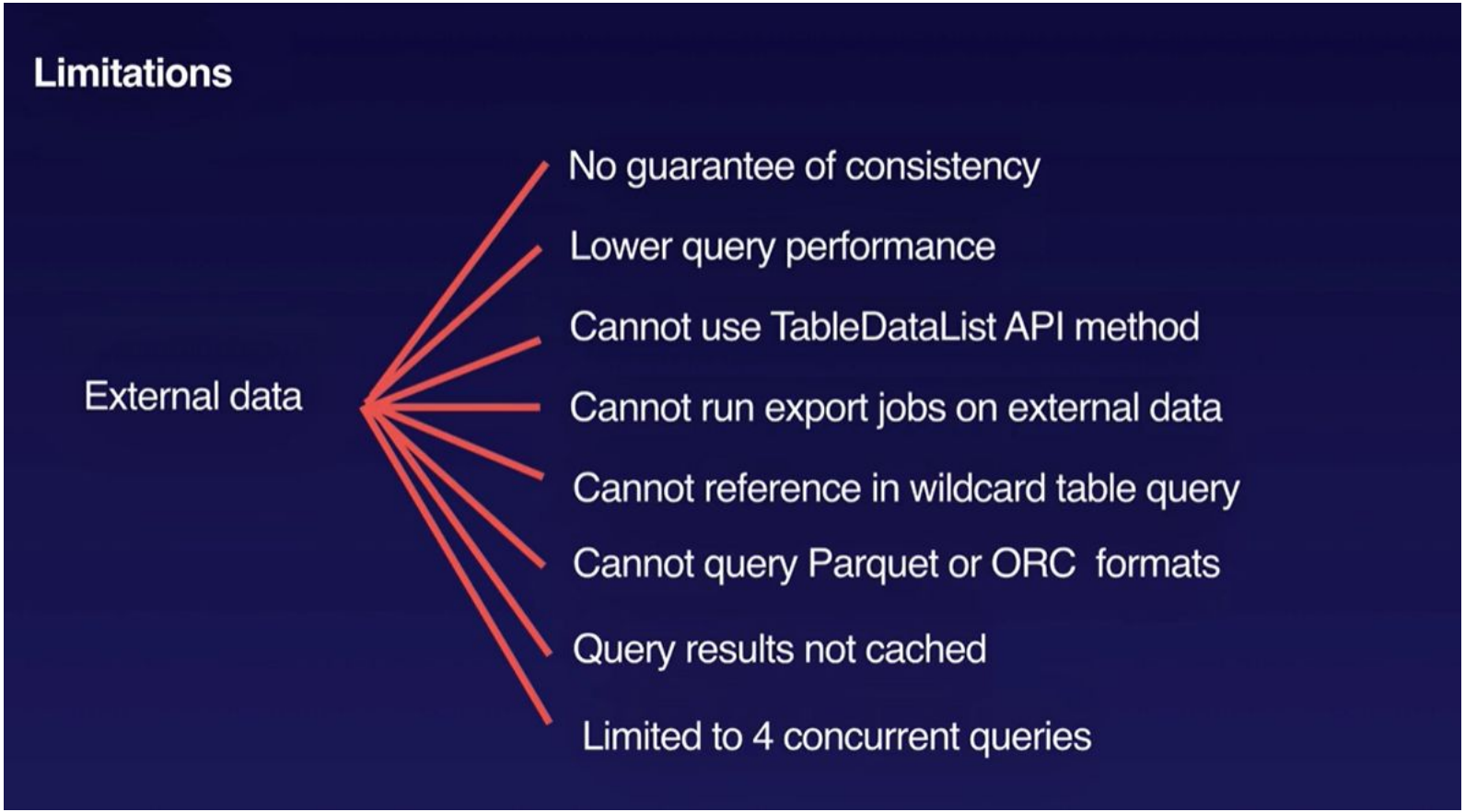
BigQuery Views



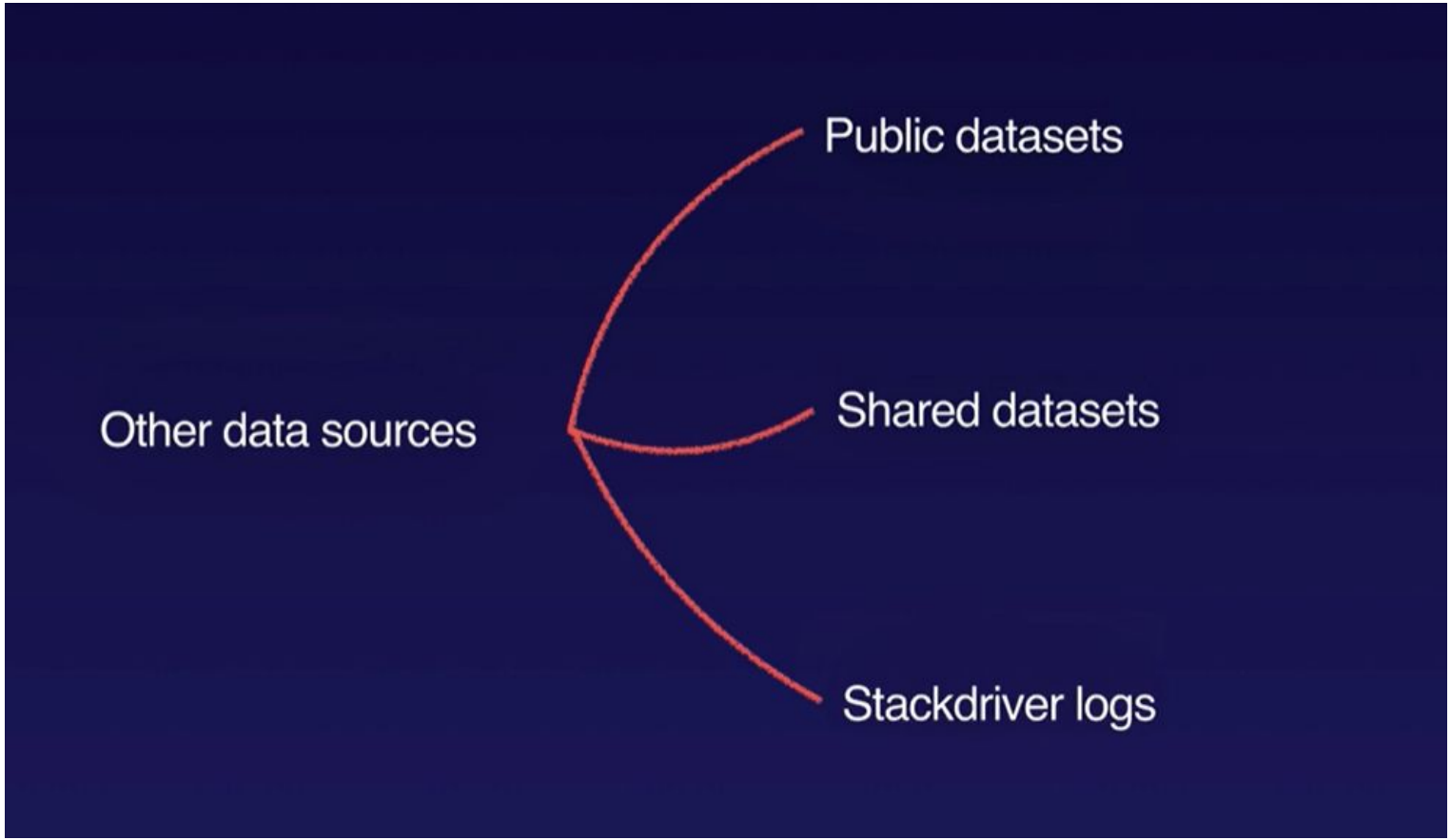
External Data



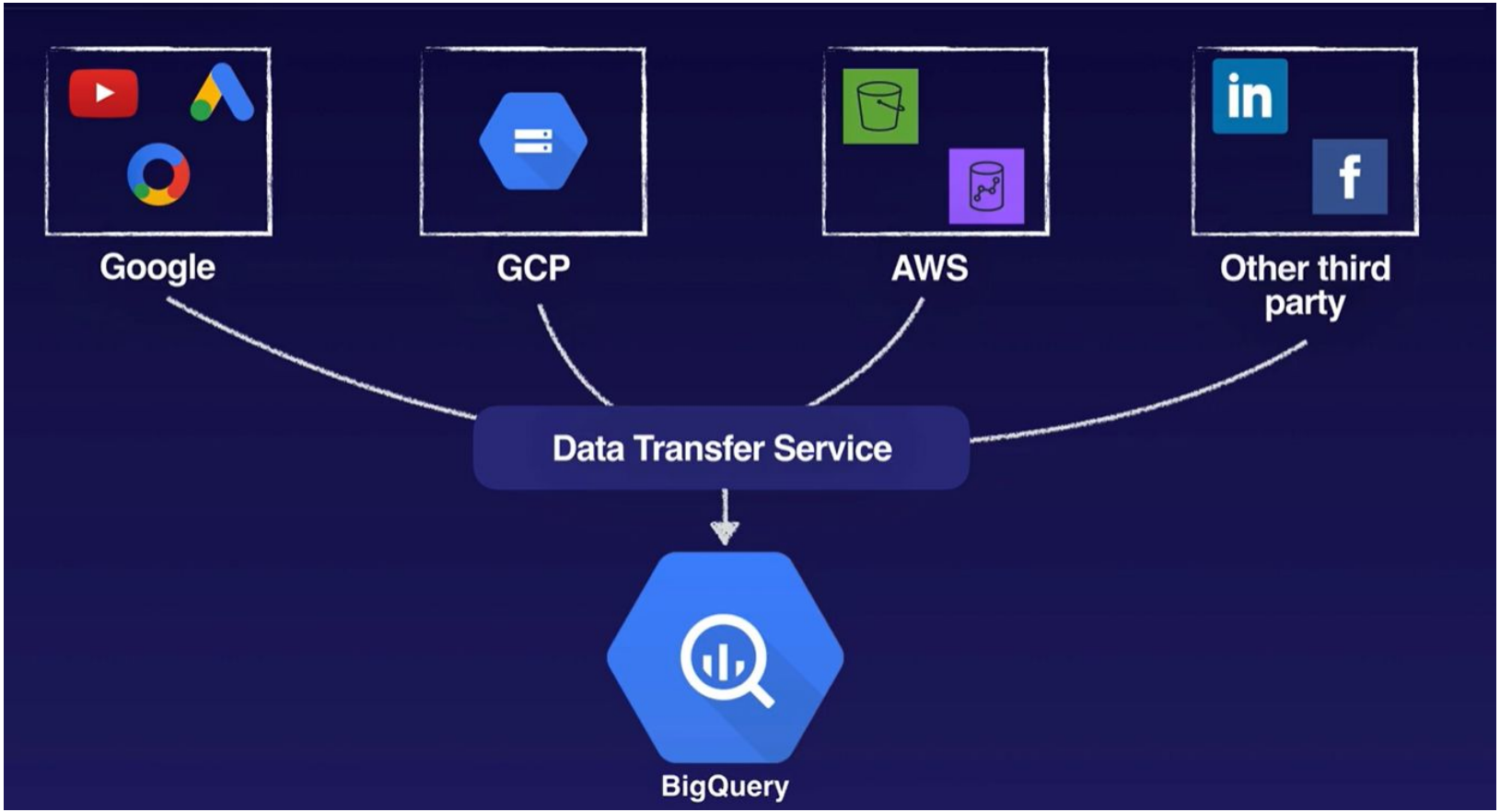
External Data



Other Data sources

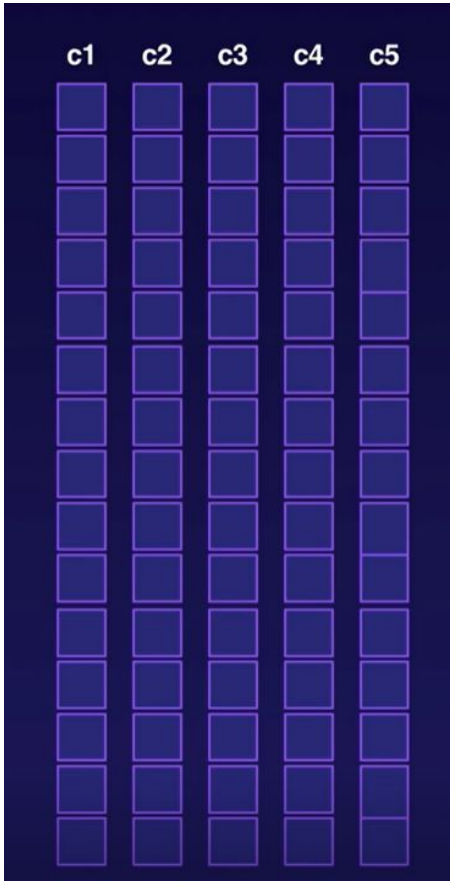


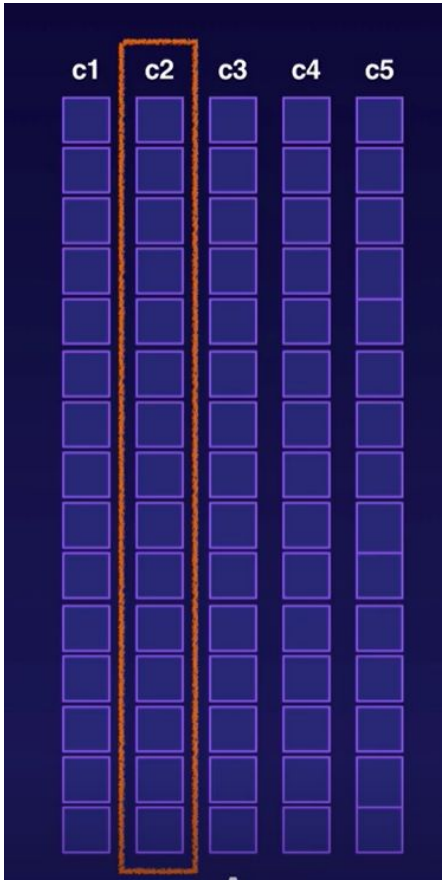
Data Transfer service



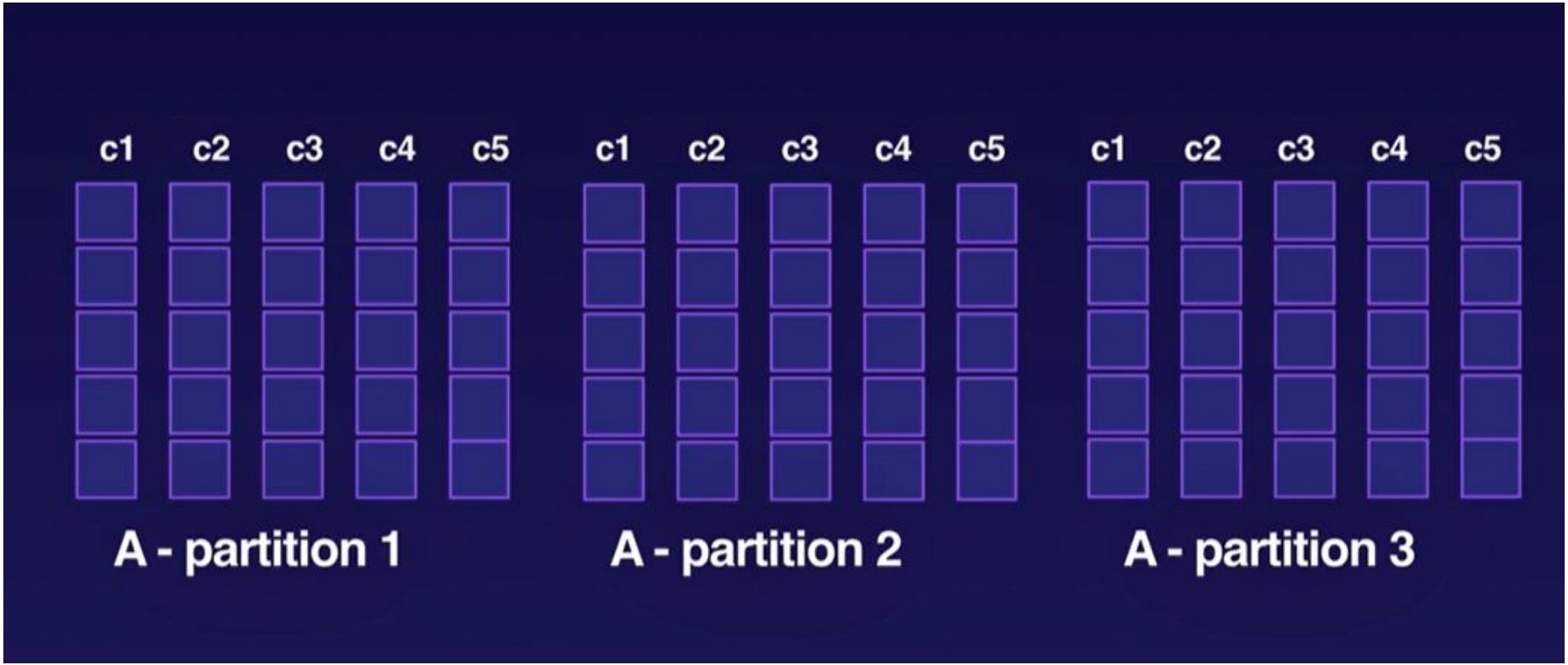
Partitioning and Clustering

Partitioning and Clustering

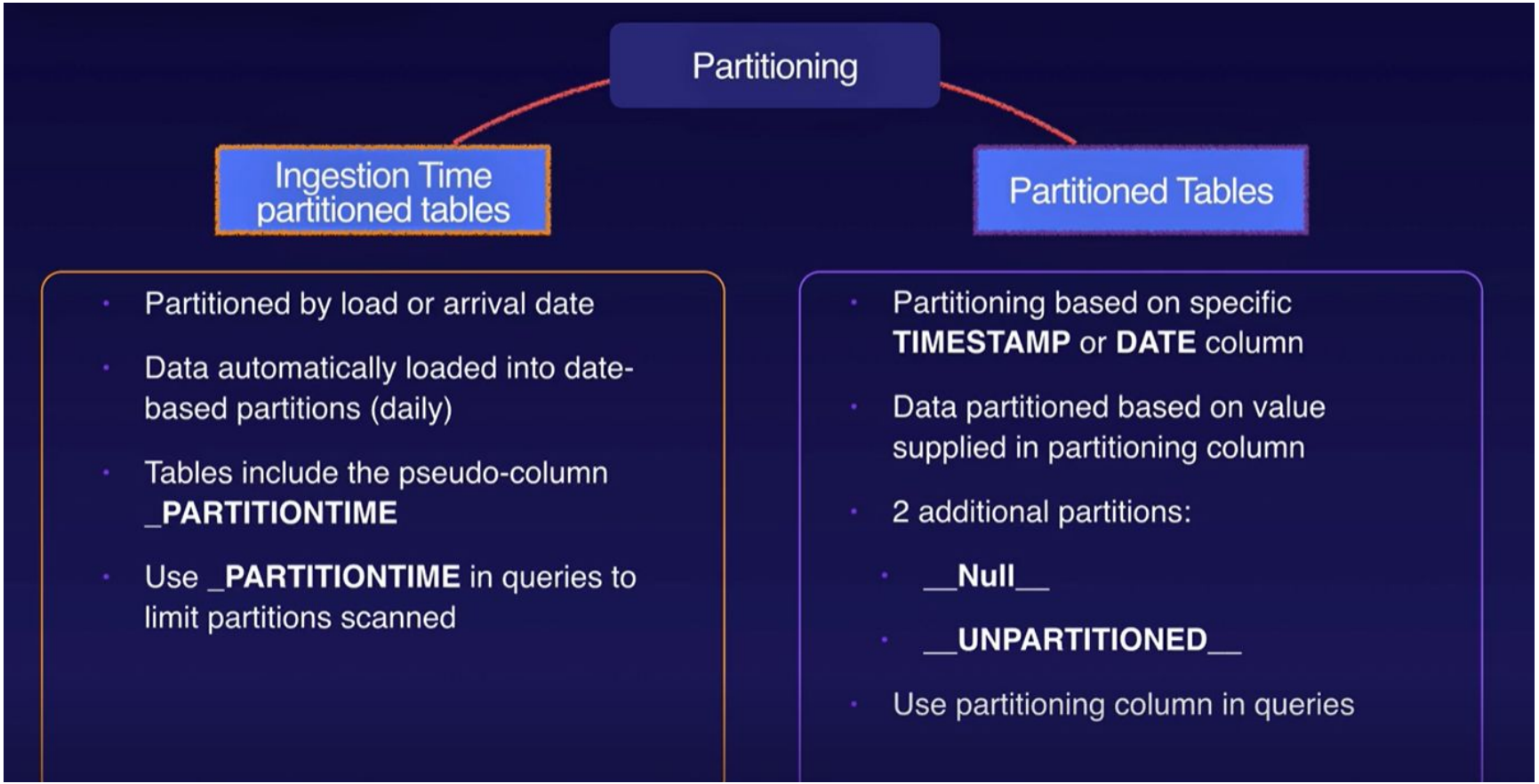




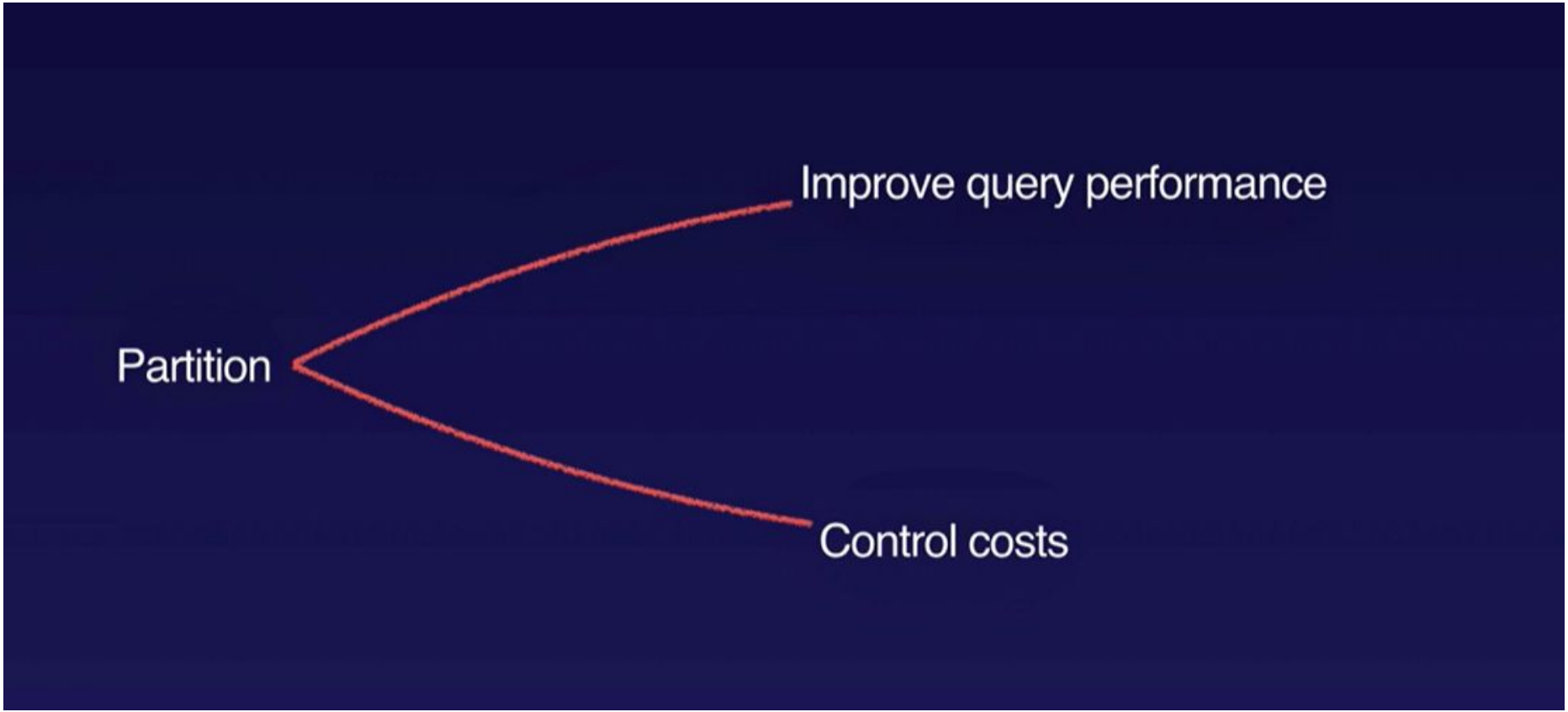
BigQuery Jobs and Operations



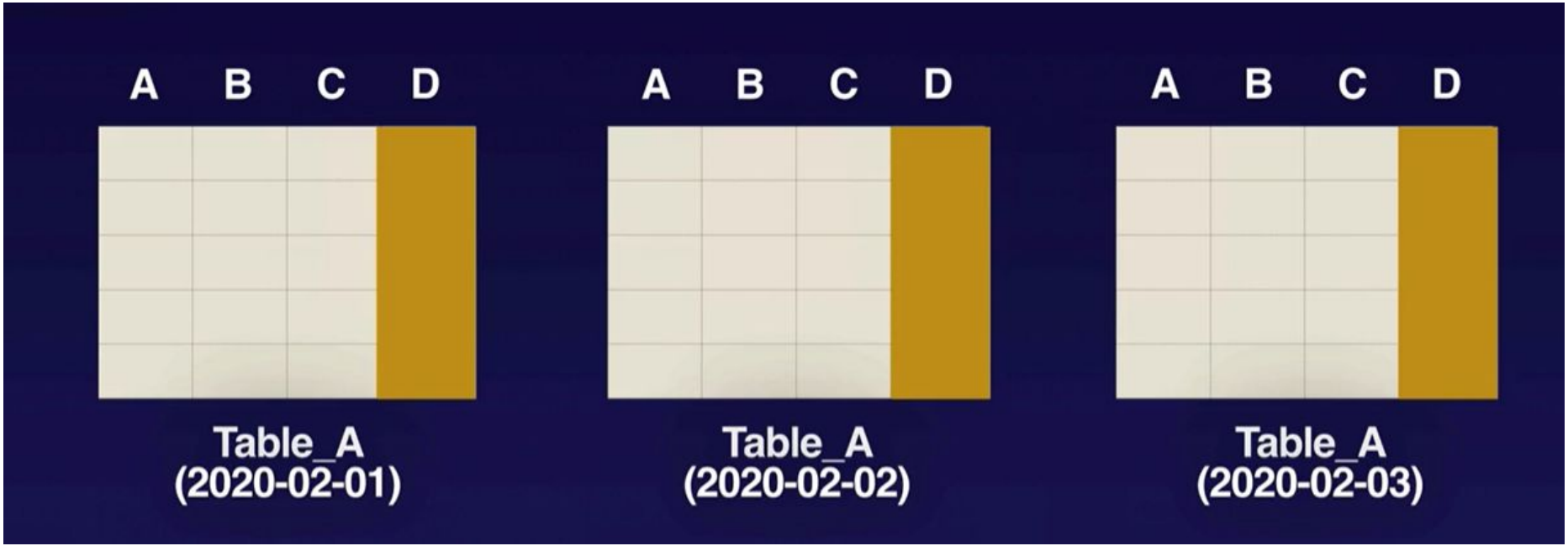
BigQuery Jobs and Operations



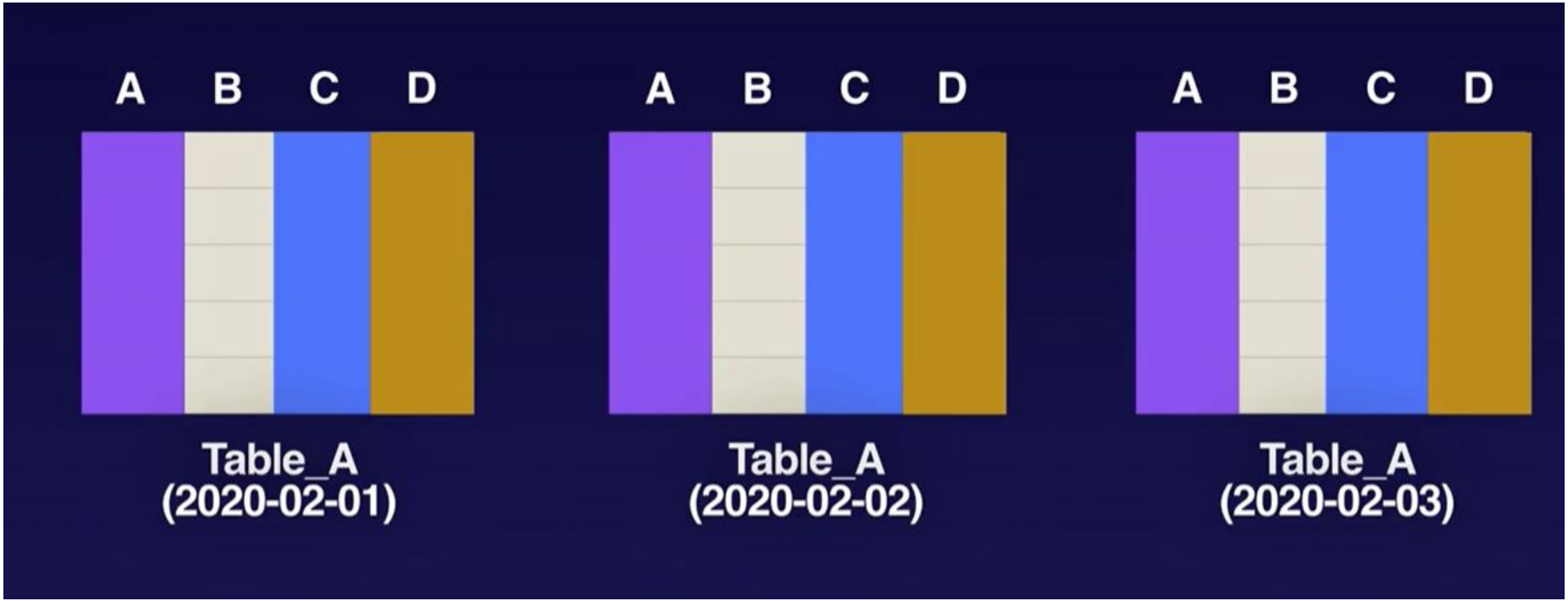
BigQuery Jobs and Operations



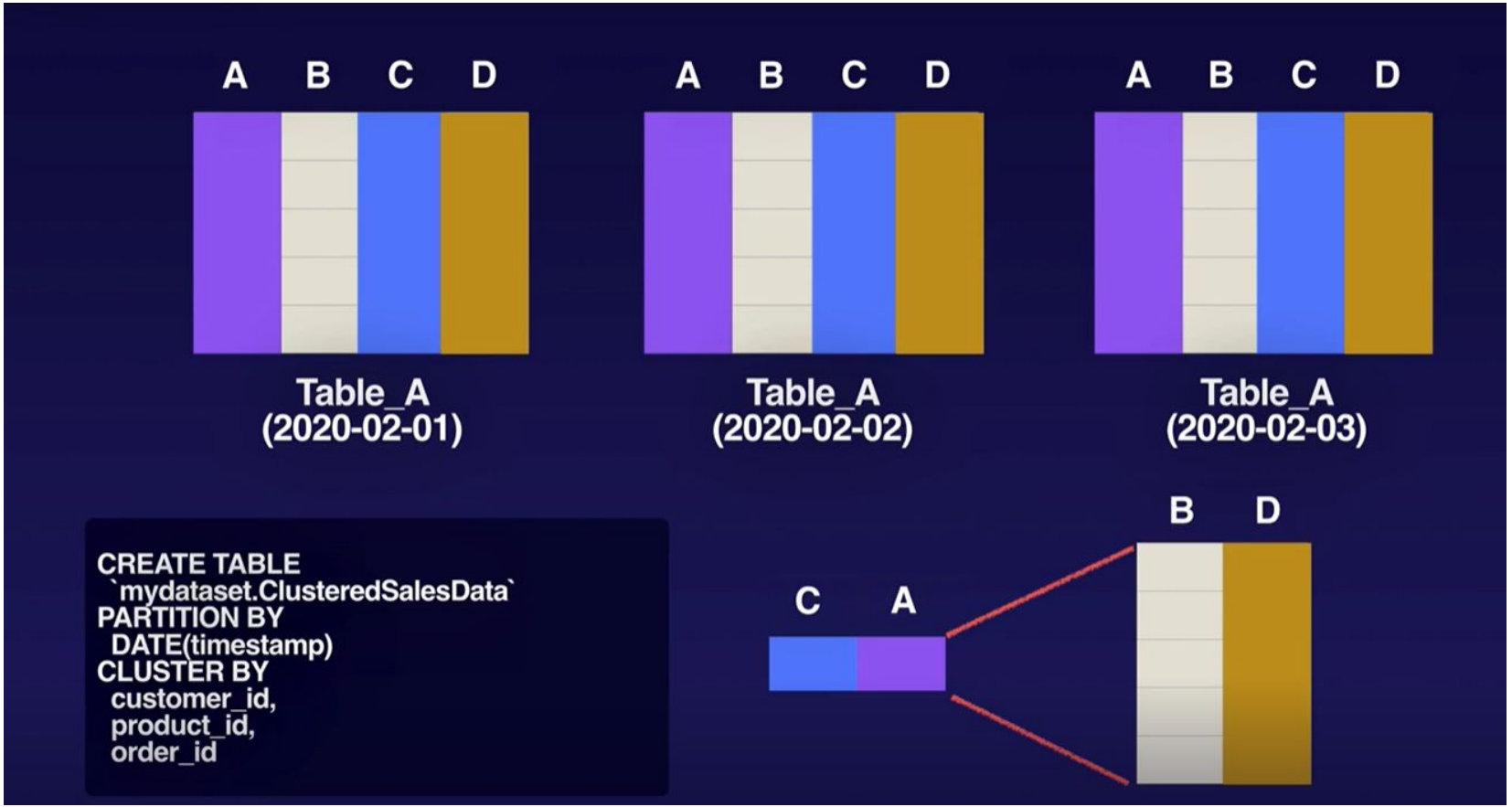
BigQuery Jobs and Operations



BigQuery Jobs and Operations

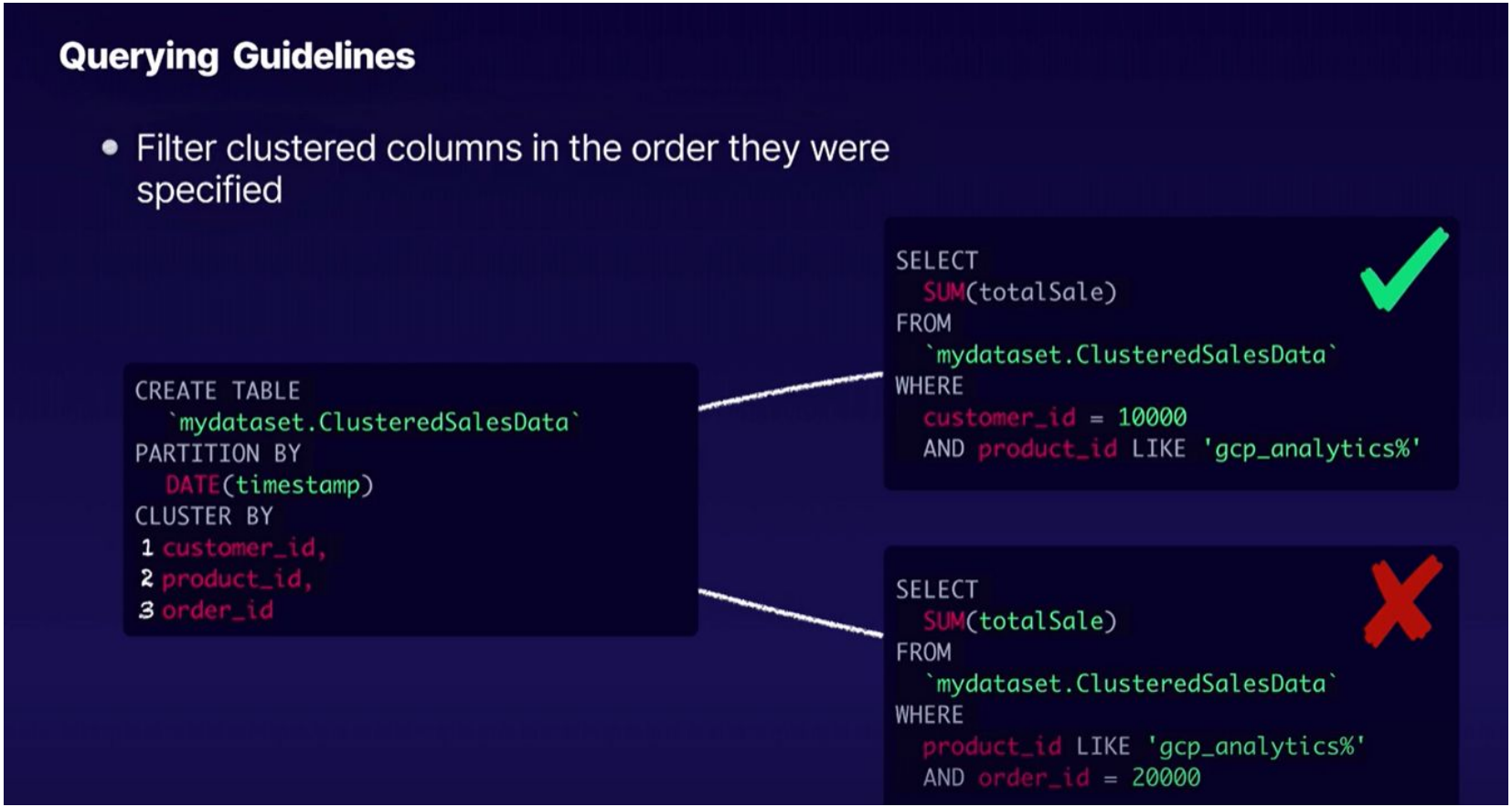


BigQuery Jobs and Operations



BigQuery Jobs and Operations

BigQuery Jobs and Operations



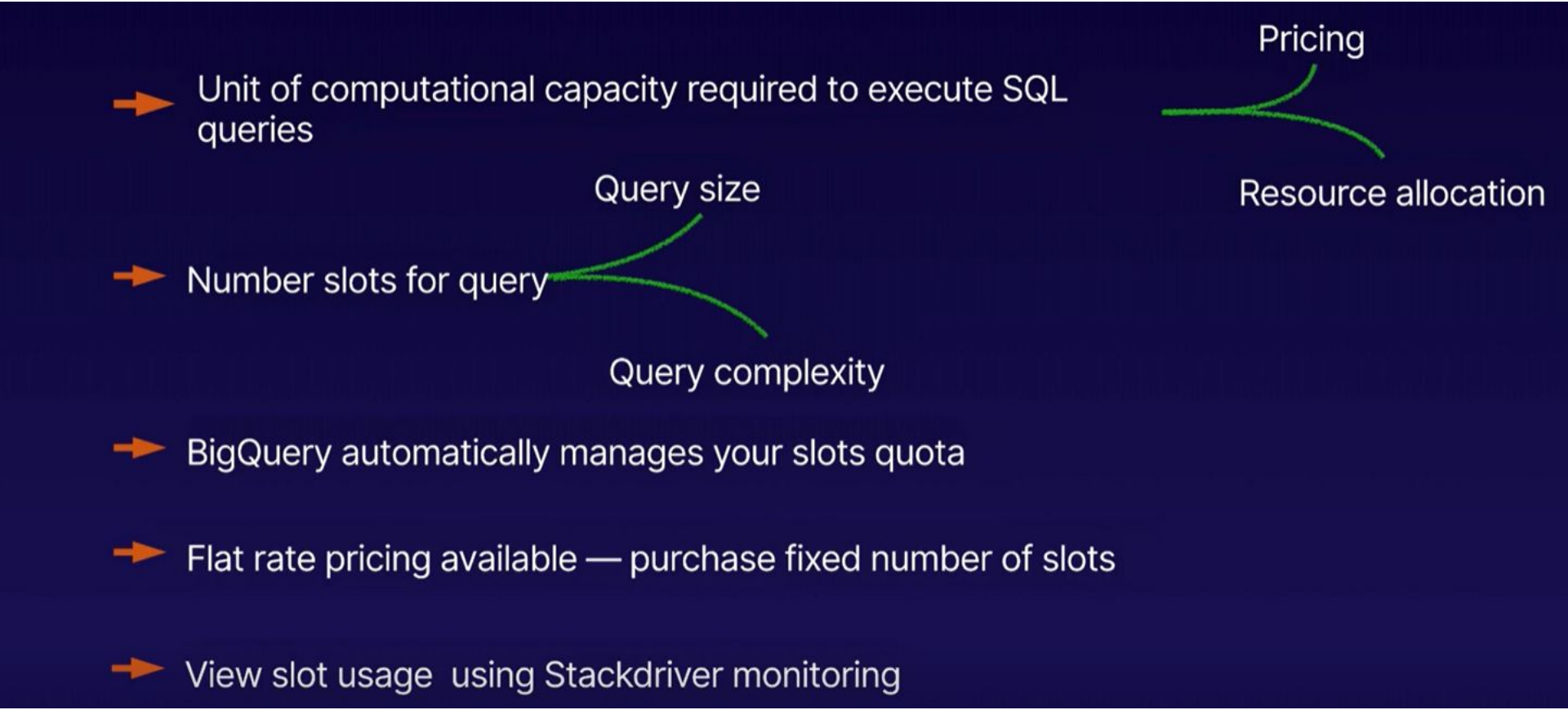
BigQuery Jobs and Operations

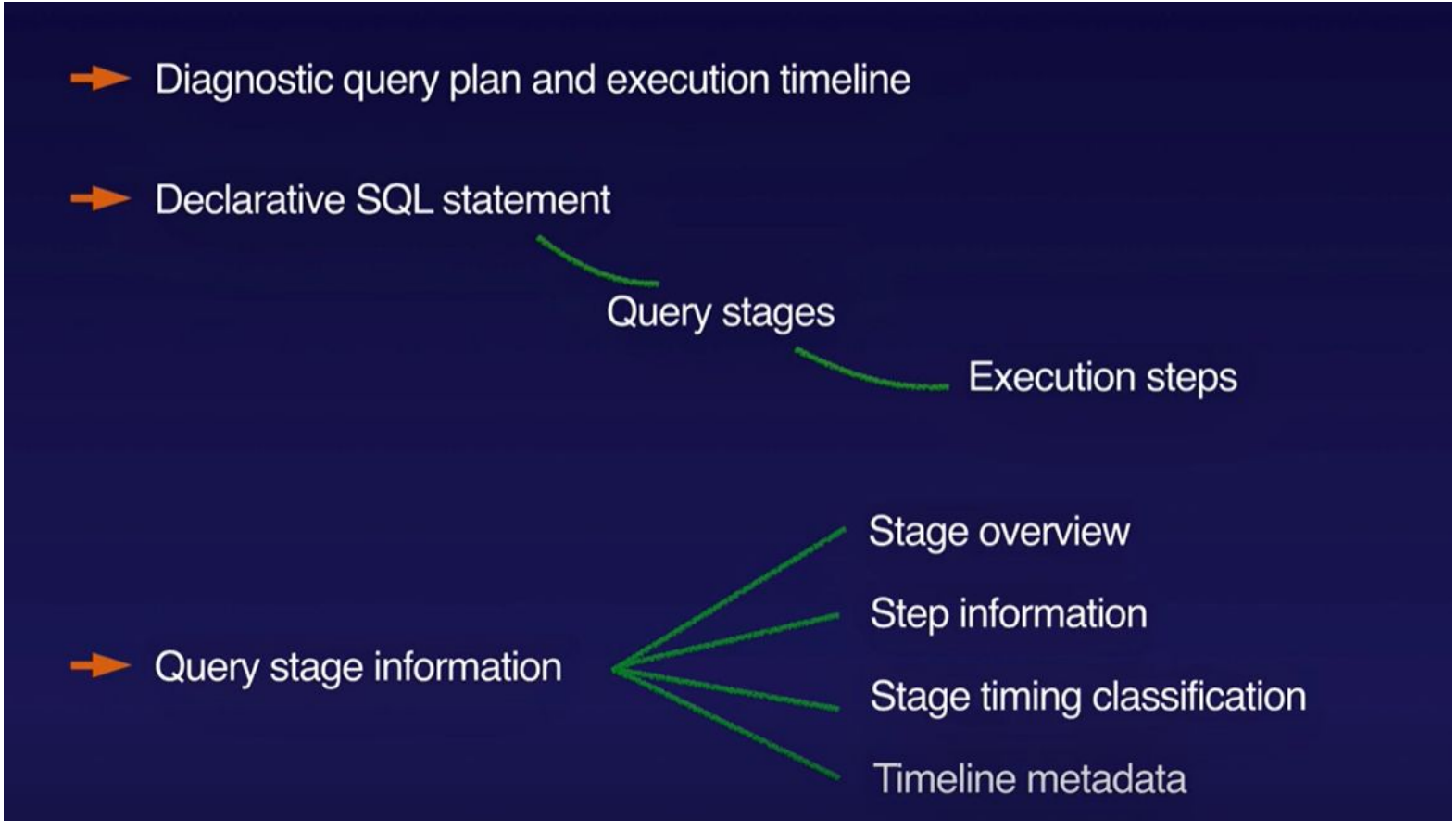
Querying Guidelines (continued)

- Avoid using clustered columns in complex filter expressions
- Avoid comparing cluster columns to other columns

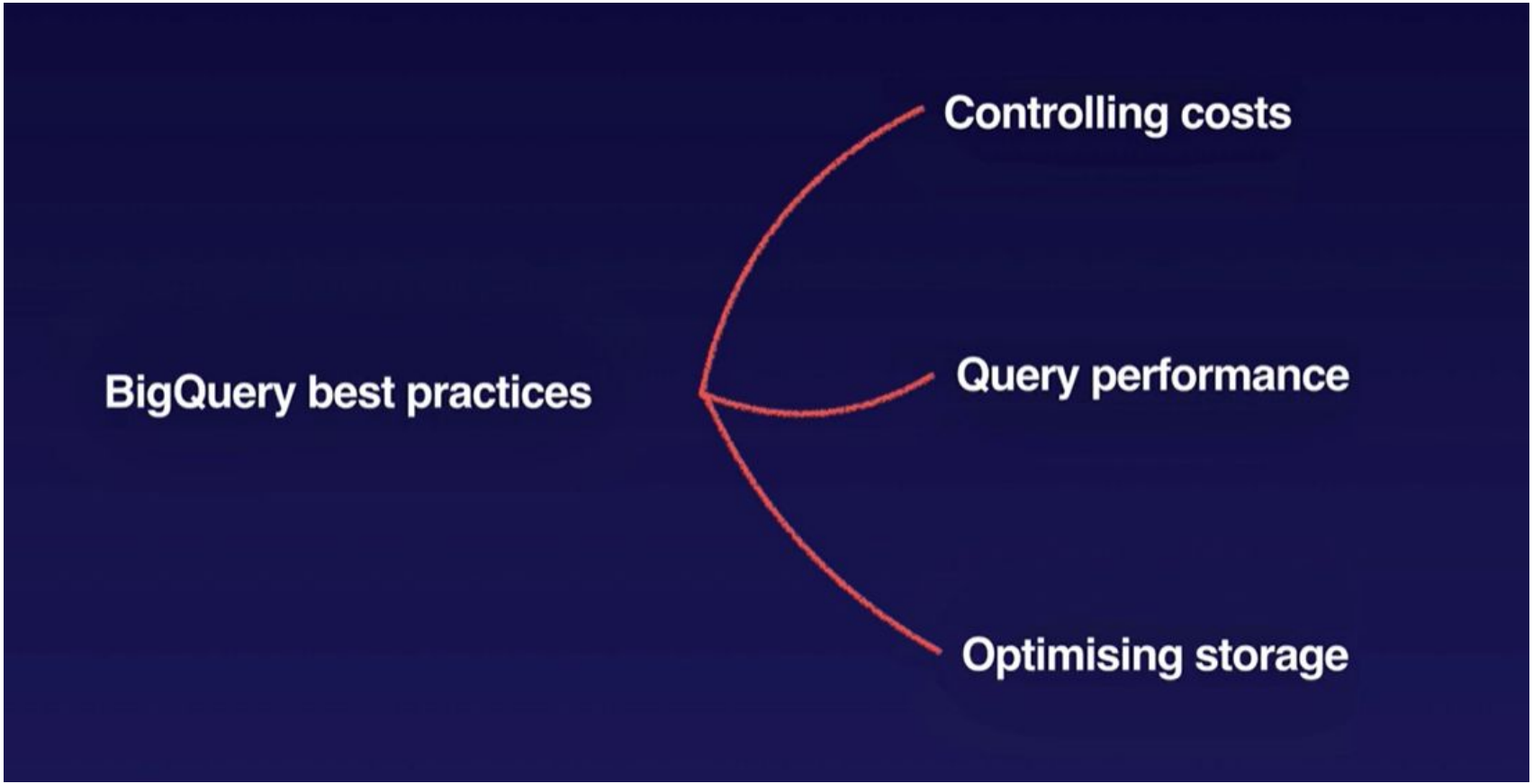
BigQuery Best Practices

Slots

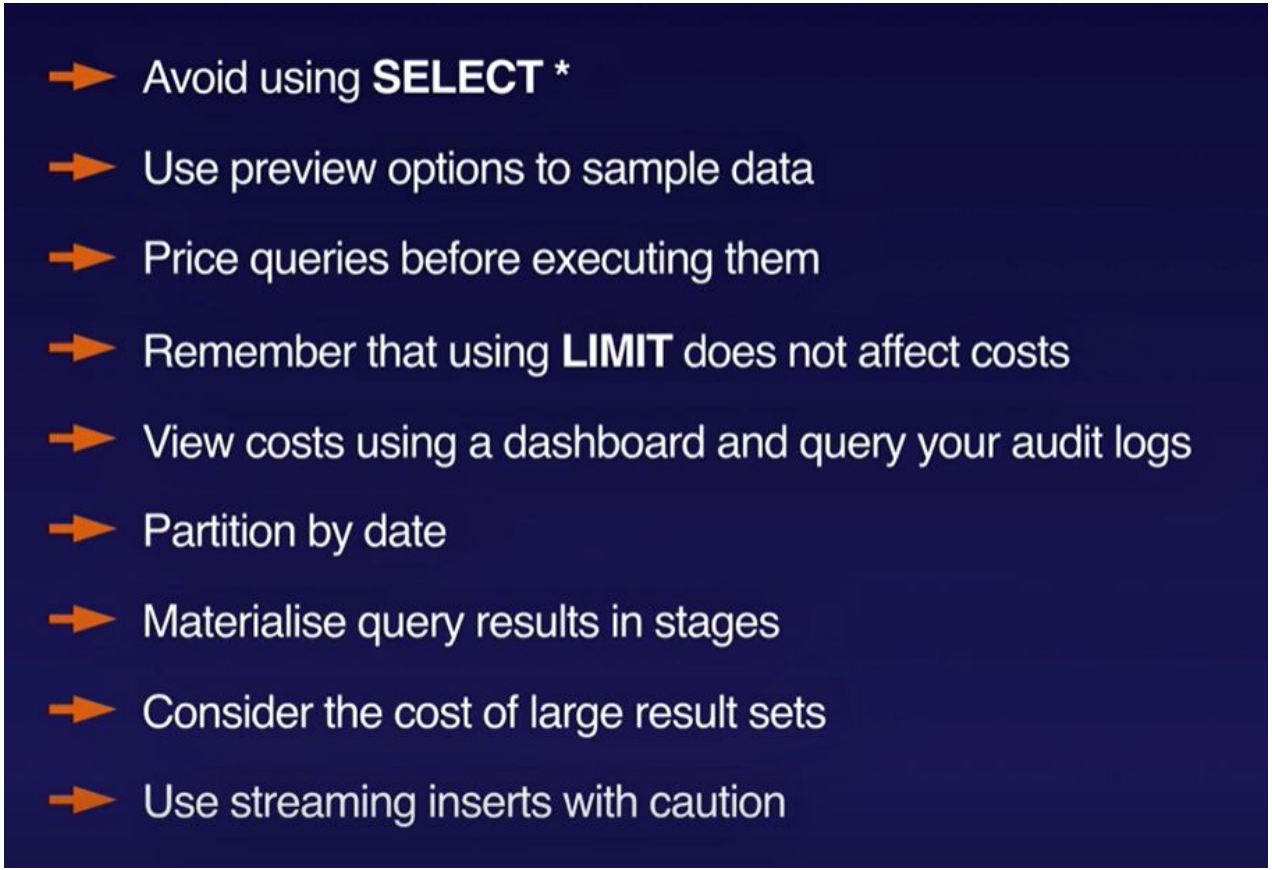

Pricing
Unit of computational capacity required to execute SQL queries
Resource allocation
Query size
Number slots for query
Query complexity
BigQuery automatically manages your slots quota
Flat rate pricing available — purchase fixed number of slots
View slot usage using Stackdriver monitoring


Diagnostic query plan and execution timeline
Declarative SQL statement
Query stages
Execution steps
Query stage information
Stage overview
Step information
Stage timing classification
Timeline metadata

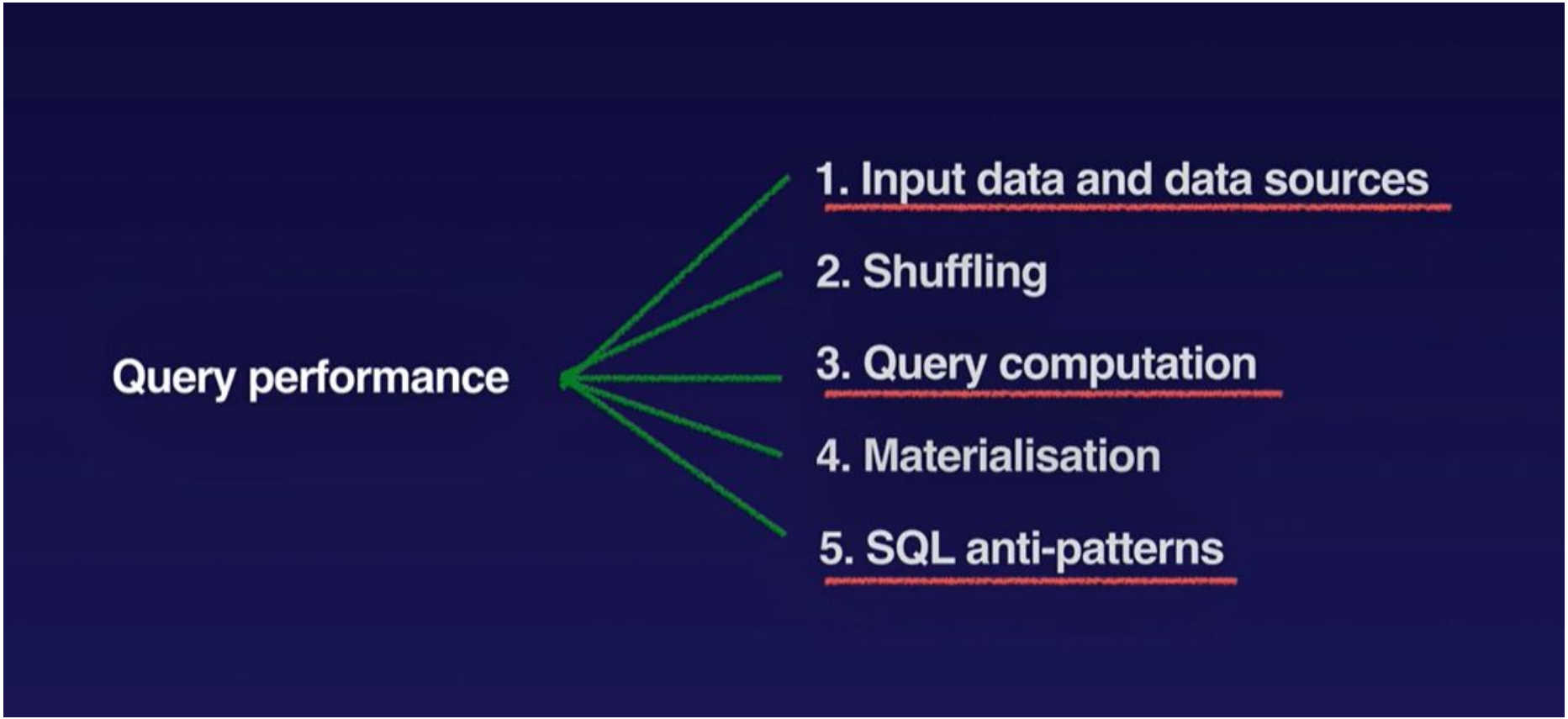
Best practices



Cost Controls



Query Performance Dimensions



Query Performance: Input Data and Data Sources

- Prune partitioned queries
- Denormalise data whenever possible
- Use external data sources appropriately
- Avoid excessive wildcard tables

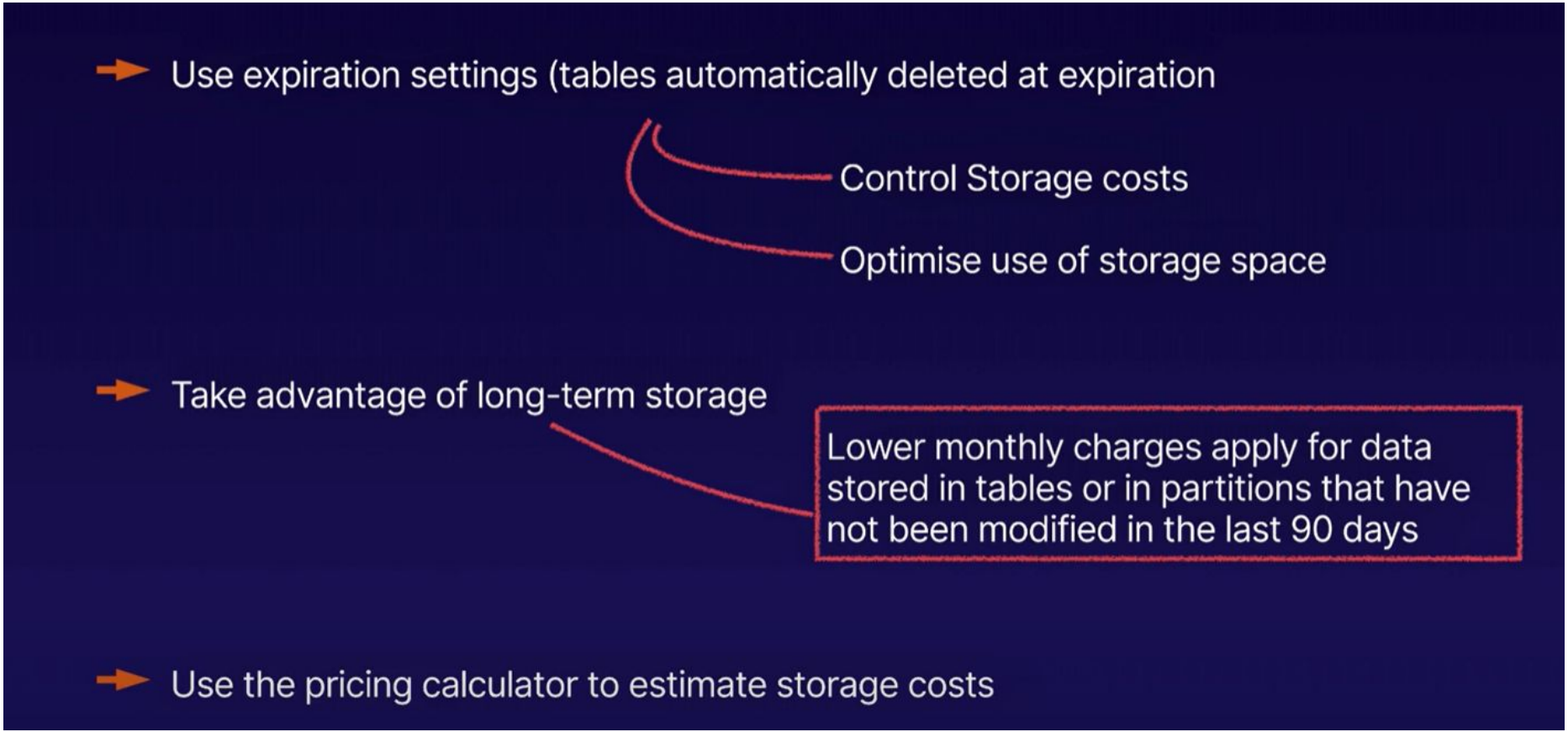
Query Performance: Query Computation

- Avoid repeatedly transforming data via SQL queries
- Avoid JavaScript user-defined functions
- Order query operations to maximise performance
- Optimise **JOIN** patterns

Query Performance: SQL Anti-patterns

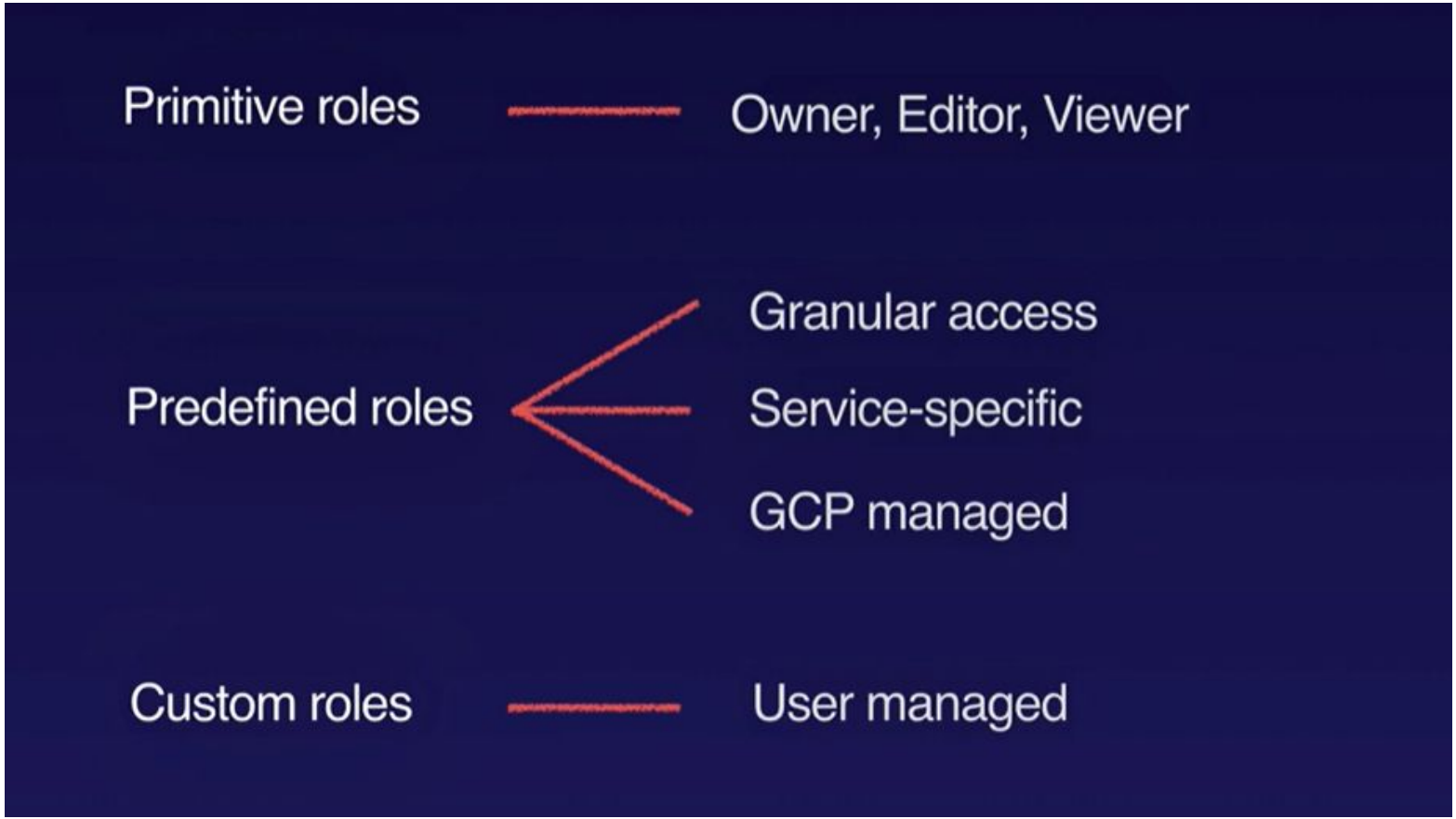
- Avoid self-joins
- Avoid data skew
- Avoid unbalanced joins
- Avoid joins that generate more outputs than inputs (Cartesian product)
- Avoid DML statements that update or insert single rows

Optimising Storage

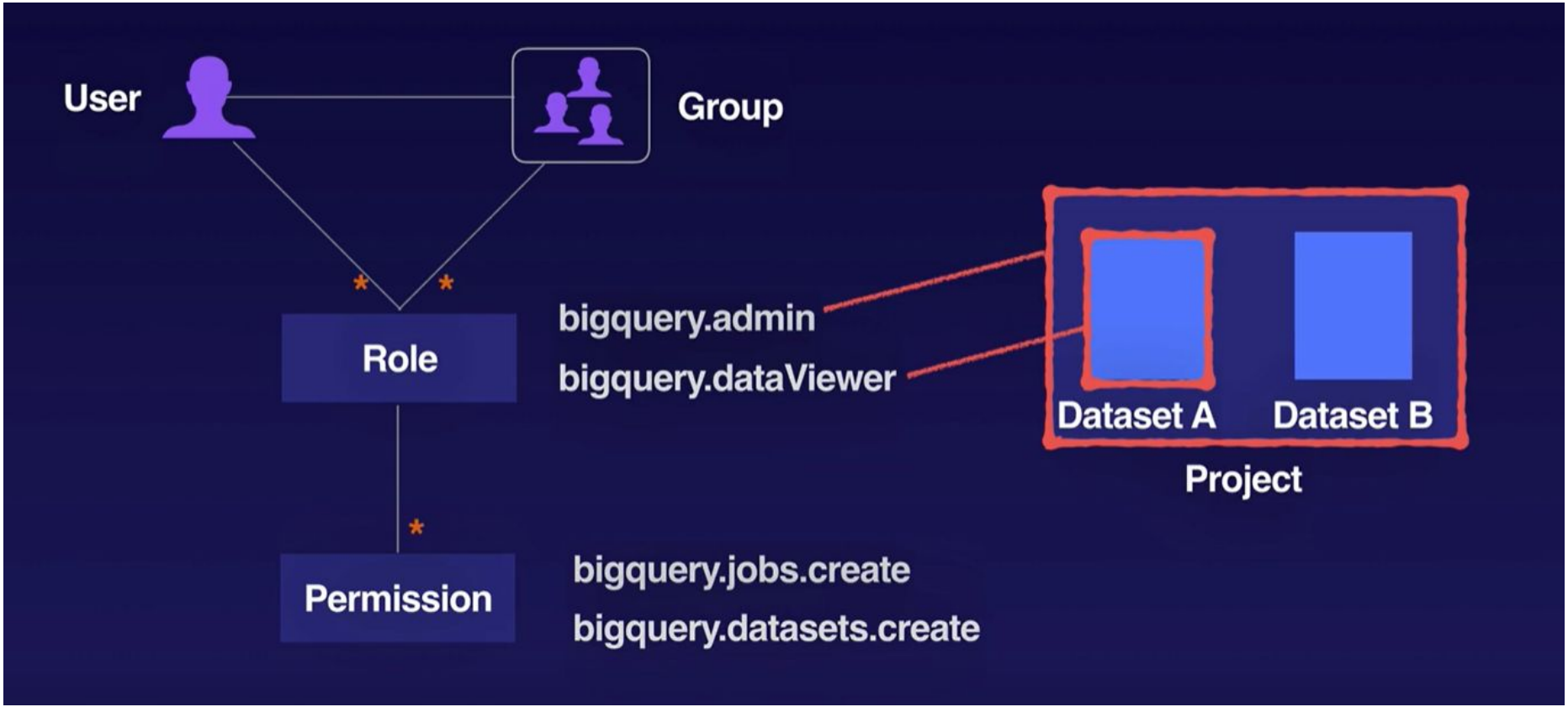


Securing BigQuery

Roles


Primitive roles
Owner, Editor, Viewer
Predefined roles
Granular access
Service-specific
GCP managed
Custom roles
User managed

Roles


User
Group
Role
bigquery.admin
bigquery.dataViewer
Dataset A
Dataset B
Project
Permission
bigquery.jobs.create
bigquery.datasets.create

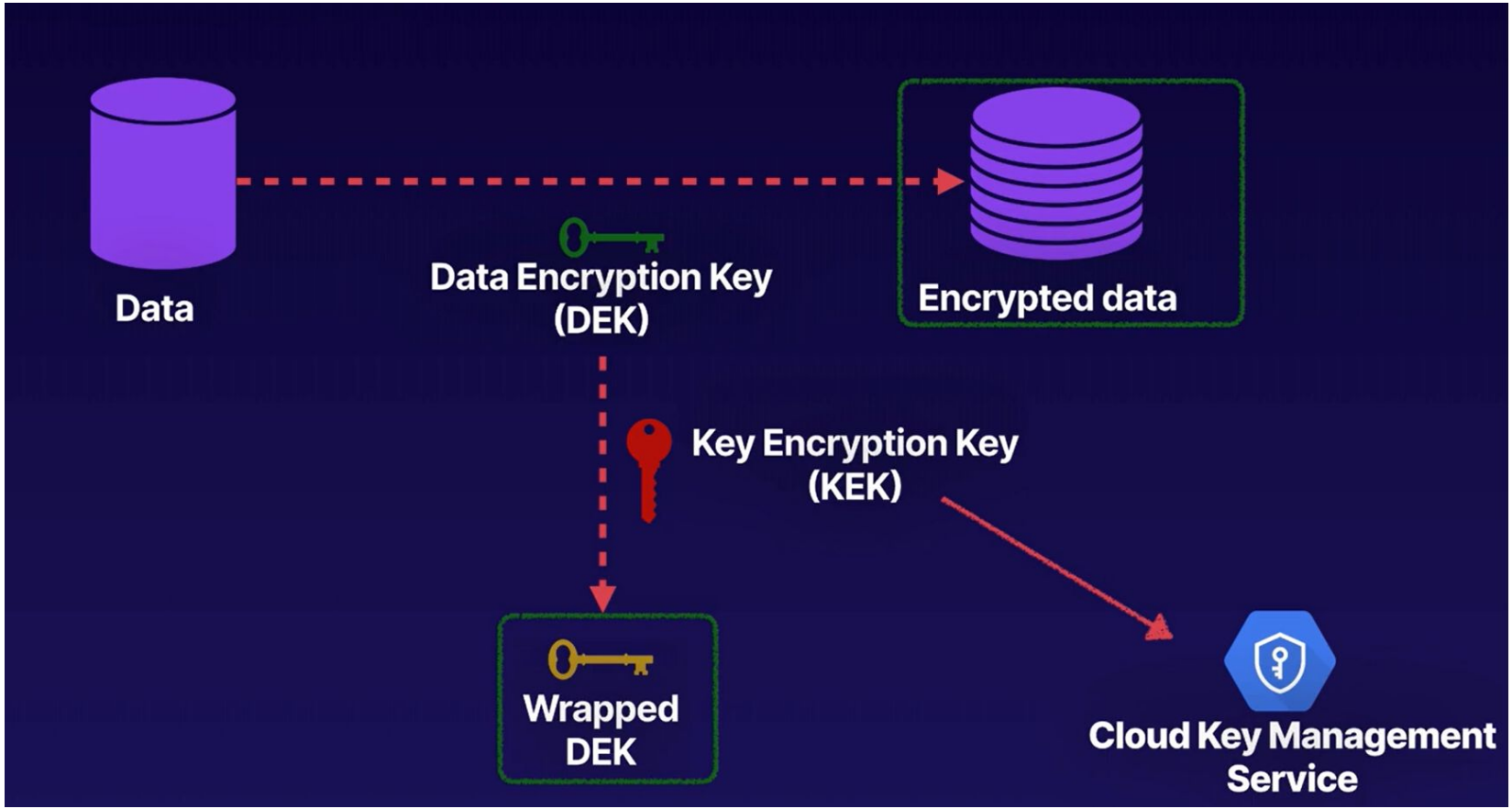
Handling Sensitive Data

Cloud DLP

Data Loss Prevention API

- Fully managed service
- Identify and protect sensitive data at scale
- Over 100 predefined detectors to identify patterns, formats, and checksums
- De-identifies data using masking, tokenisation, pseudonymisation, date shifting and more

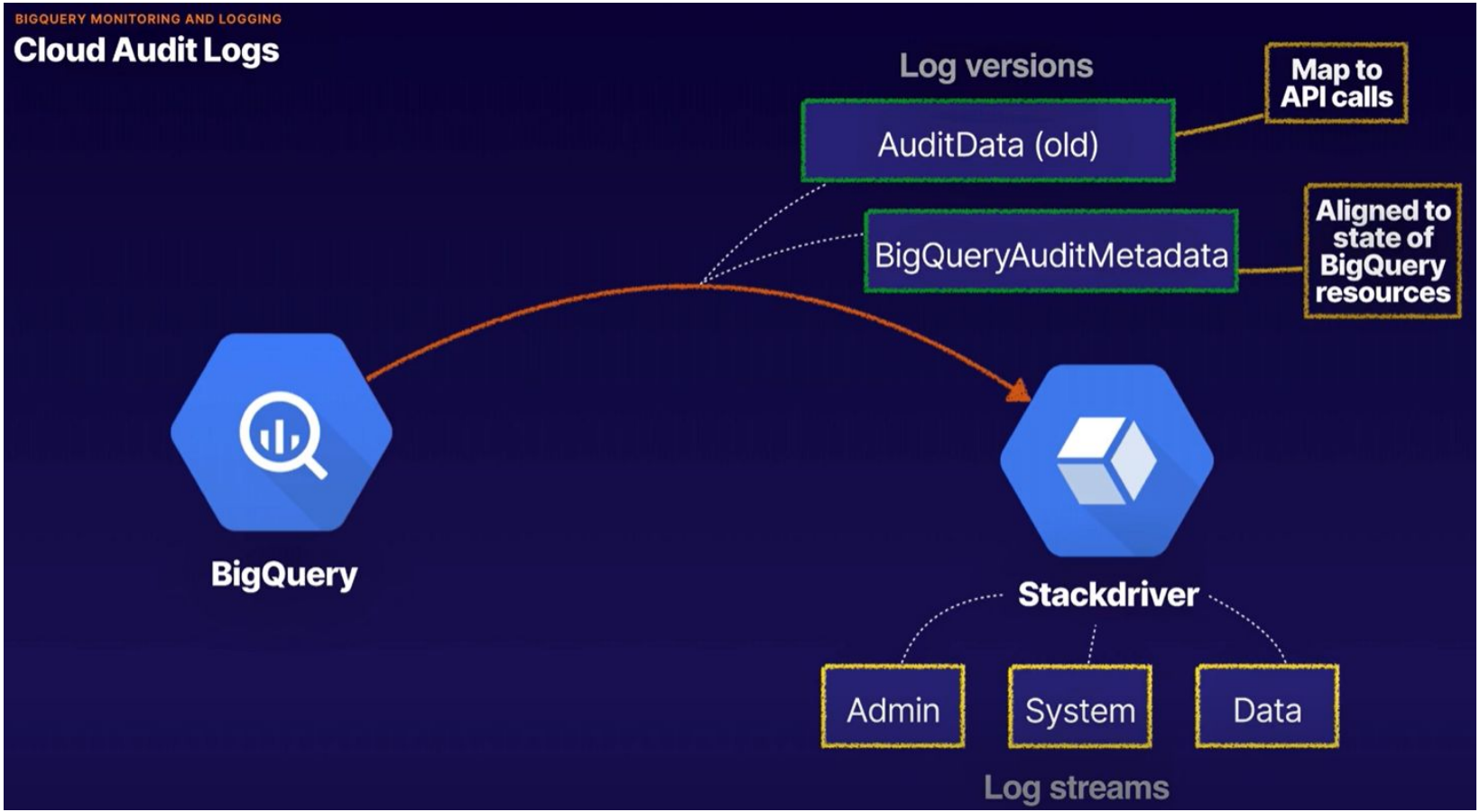
Cloud Key Management Service



BigQuery Monitoring and Logging

BigQuery Monitoring

Cloud Audit Logs

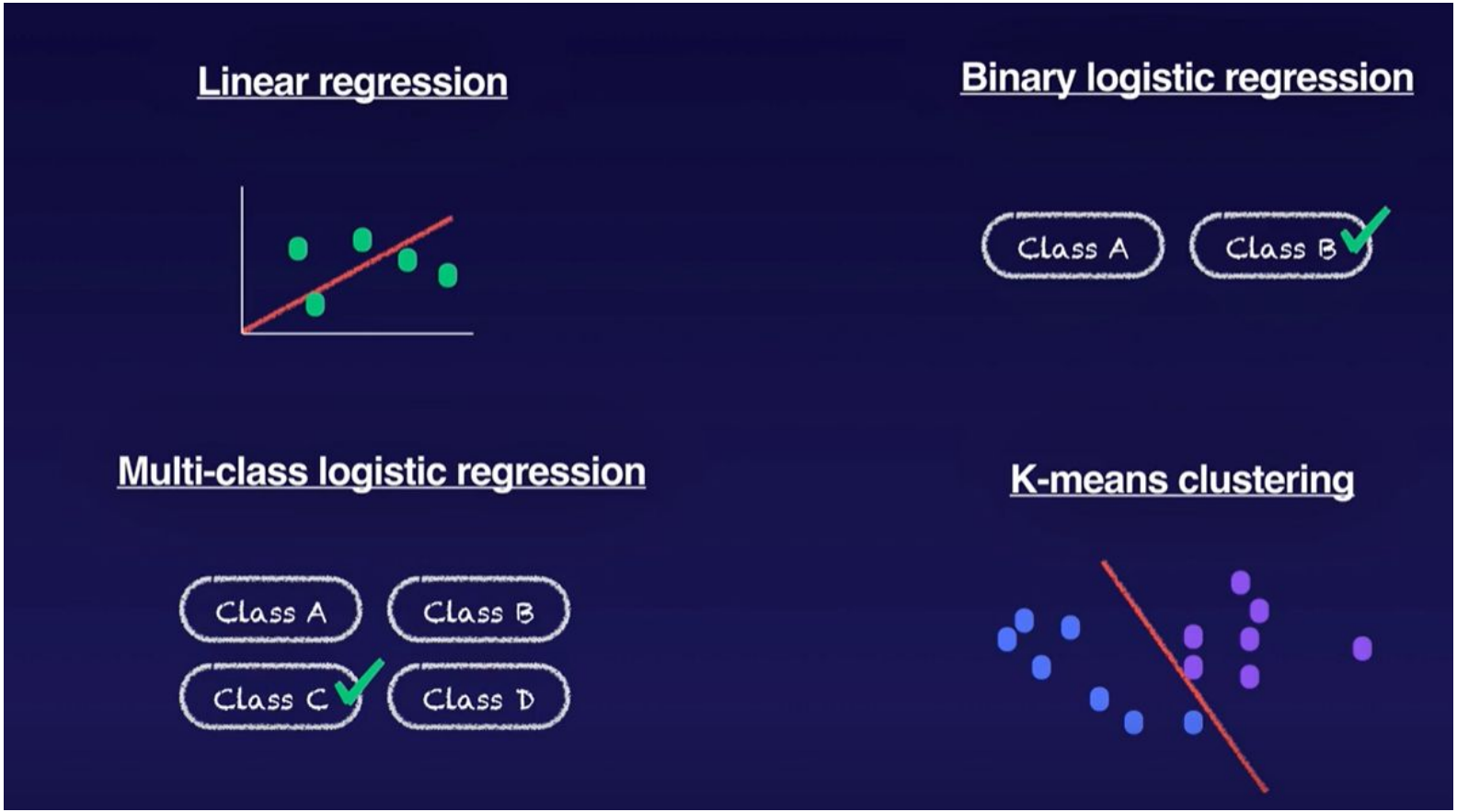


Machine Learning with BigQuery ML

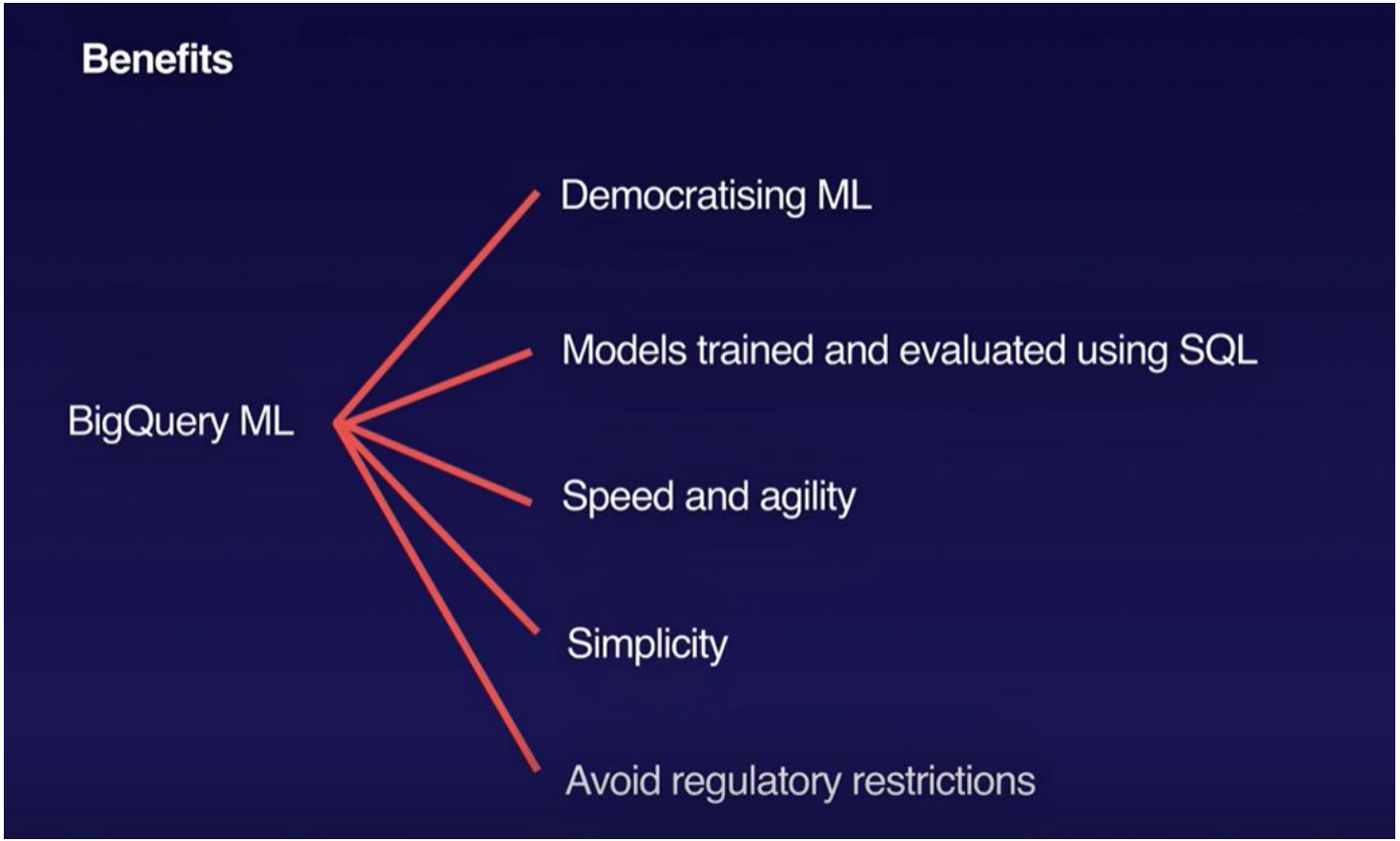
BigQuery ML

- Web console (UI)
- bq command line tool
- BigQuery rest API
- Jupyter notebooks (Cloud Datalab) and other external BI tools

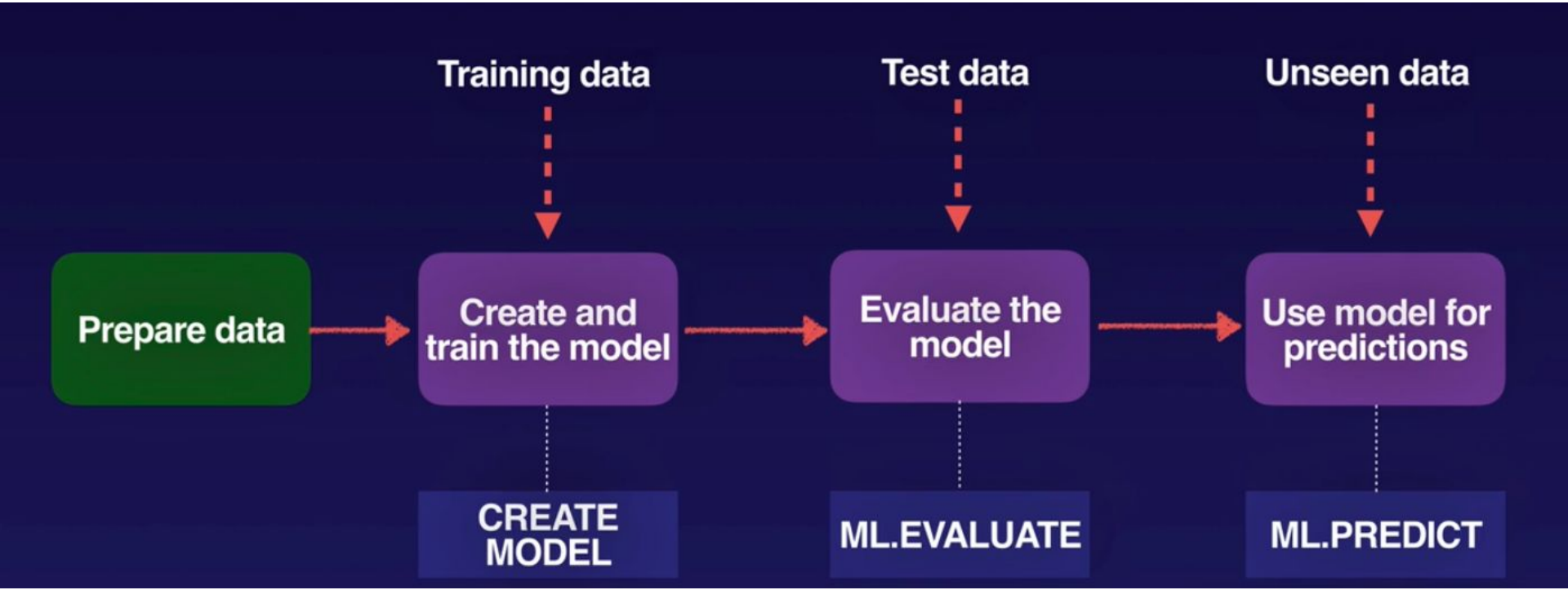
ML Models


Linear regression
Binary logistic regression
Class A
Class B
Multi-class logistic regression
K-means clustering
Class A
Class B
Class C
Class D

BigQuery ML


Benefits
BigQuery ML
Democratising ML
Models trained and evaluated using SQL
Speed and agility
Simplicity
Avoid regulatory restrictions

BigQuery ML process



Thank you