



PHASE 1 FOUNDATIONS

# MULTI-CLOUD DATA ENGINEERING: LEVEL **100** FOUNDATIONS

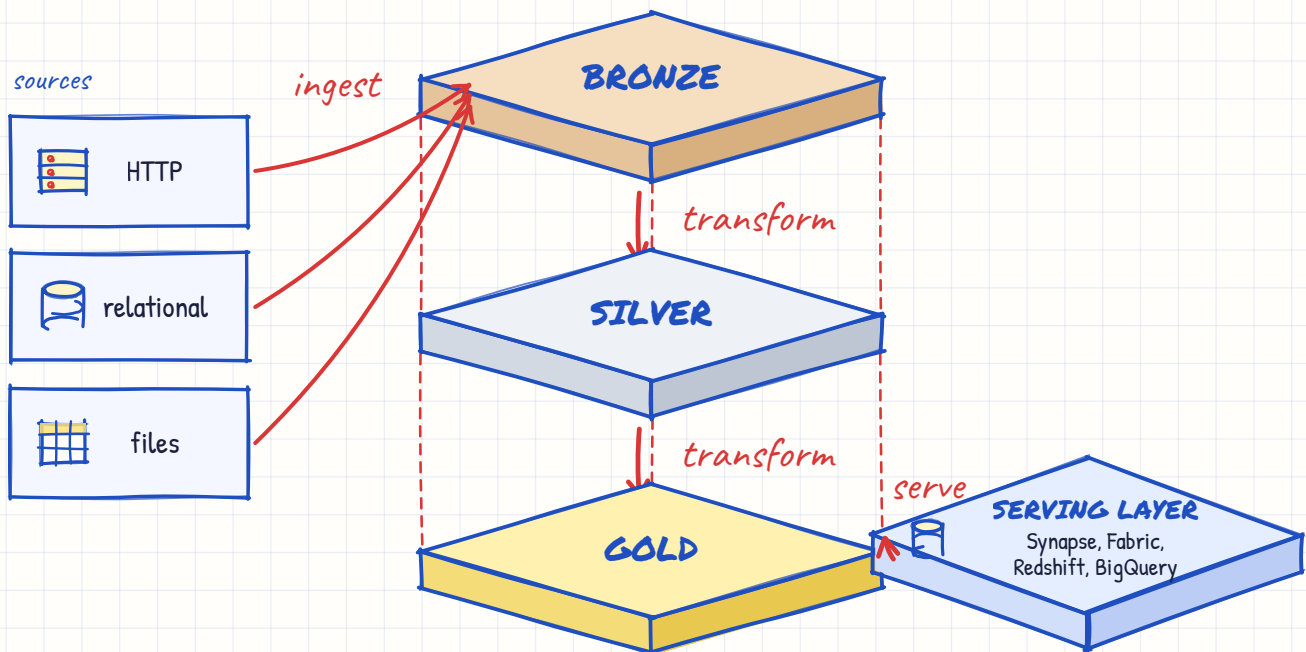
Azure-primary, mapped to AWS and GCP. From SQL and Python to a production-style medallion pipeline.

5 WEEKS

25 SESSIONS

50 CLASS + 45 LAB HOURS

Outcome: Data Engineer



AZURE-PRIMARY. CONCEPTS MAPPED TO AWS AND GCP.



# ABOUT THIS PROGRAM

## OUTCOME

Data Engineer

## DURATION

5 weeks | 25 class sessions | 50 class hours + 45 lab hours

## APPROACH

Azure is the teaching vehicle: each concept is built in Azure first, then mapped to AWS and GCP in the same session.

## CLASS TIMING

18:00 to 20:00 IST, Monday to Friday. Labs are self-paced from written lab guides.

## ENTRY BAR

- working SQL (joins, aggregations, subqueries; window functions are taught)
- basic Python (functions, loops, dicts, reading a file; PySpark is taught from zero)
- comfortable on a command line with Git basics
- No prior cloud experience assumed.

## SESSION FORMAT (120 MINUTES)

|         |                      |                   |                |              |
|---------|----------------------|-------------------|----------------|--------------|
| 40 min  | 35 min               | 15 min            | 20 min         | 10 min       |
| Concept | Azure implementation | AWS + GCP mapping | Structured Q&A | Lab briefing |

Concept (40 min) | Azure implementation (35 min) | AWS + GCP mapping (15 min) | Structured Q&A (20 min) | Lab briefing (10 min)

## LAB MODES

### Hands-on

Hands-on = learner builds for real

### Mixed

Mixed = partly built, partly designed

### Design

Design = architecture exercise, no provisioning



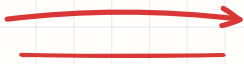
# HOW TO READ THIS BOOK

main topic

Blue pen = main topics

Titles, boxes and key points are drawn in blue.

ingest



Red pen = flows and notes

Arrows, underlines and margin notes show how data moves.

Key Takeaway

Yellow = takeaways

Highlighter swipes mark main topics; yellow boxes carry the Key Takeaway.

3

Numbered circles = callouts

A circle in a diagram points to the session with the same number in that week.



**WHY THIS MATTERS**

Light-blue box



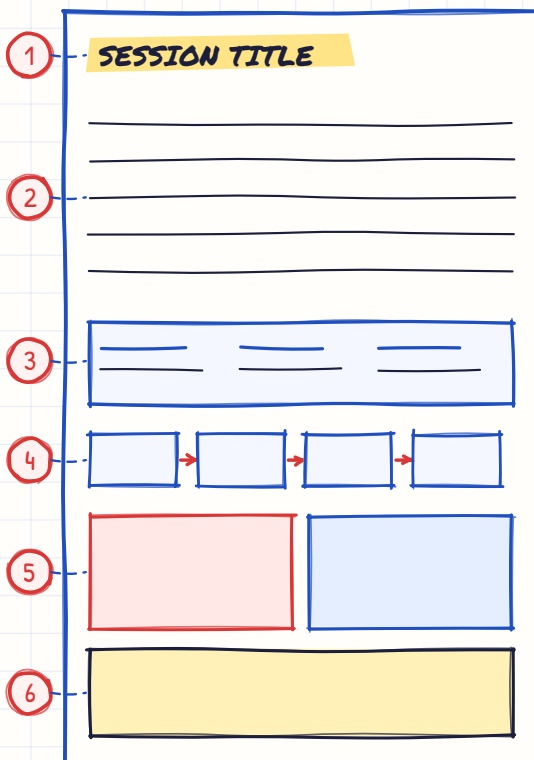
**KEY TAKEAWAY**

Yellow box

**LAB**

Sticky note

## ANATOMY OF A SESSION PAGE



- 1 Session number, day and title
- 2 Topics Covered, as numbered notes
- 3 Azure / AWS / GCP box
- 4 Hand-drawn flow strip
- 5 Lab sticky note and Why this matters
- 6 Key Takeaway



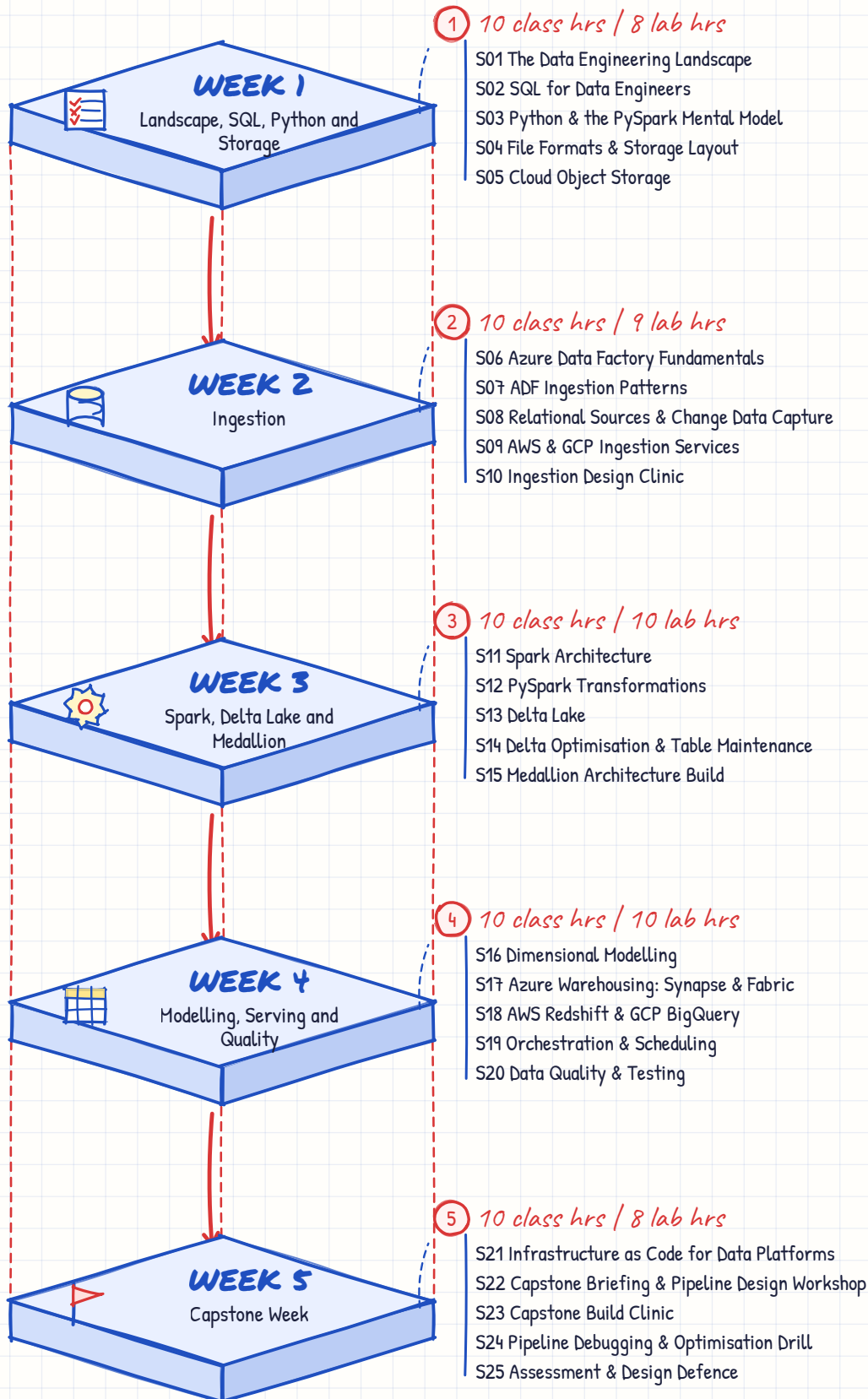
# TABLE OF CONTENTS

|                                                              |           |
|--------------------------------------------------------------|-----------|
| About this program .....                                     | 2         |
| How to read this book .....                                  | 3         |
| Table of contents .....                                      | 4         |
| Level 100 at a glance .....                                  | 5         |
| <b>WEEK 1: LANDSCAPE, SQL, PYTHON AND STORAGE</b> .....      | <b>6</b>  |
| <i>10 class hours / 8 lab hours</i>                          |           |
| S01 Mon   The Data Engineering Landscape .....               | 8         |
| S02 Tue   SQL for Data Engineers .....                       | 9         |
| S03 Wed   Python & the PySpark Mental Model .....            | 10        |
| S04 Thu   File Formats & Storage Layout .....                | 11        |
| S05 Fri   Cloud Object Storage .....                         | 12        |
| <b>WEEK 2: INGESTION</b> .....                               | <b>14</b> |
| <i>10 class hours / 9 lab hours</i>                          |           |
| S06 Mon   Azure Data Factory Fundamentals .....              | 16        |
| S07 Tue   ADF Ingestion Patterns .....                       | 17        |
| S08 Wed   Relational Sources & Change Data Capture .....     | 18        |
| S09 Thu   AWS & GCP Ingestion Services .....                 | 19        |
| S10 Fri   Ingestion Design Clinic .....                      | 20        |
| <b>WEEK 3: SPARK, DELTA LAKE AND MEDALLION</b> .....         | <b>22</b> |
| <i>10 class hours / 10 lab hours</i>                         |           |
| S11 Mon   Spark Architecture .....                           | 24        |
| S12 Tue   PySpark Transformations .....                      | 25        |
| S13 Wed   Delta Lake .....                                   | 26        |
| S14 Thu   Delta Optimisation & Table Maintenance .....       | 27        |
| S15 Fri   Medallion Architecture Build .....                 | 28        |
| <b>WEEK 4: MODELLING, SERVING AND QUALITY</b> .....          | <b>30</b> |
| <i>10 class hours / 10 lab hours</i>                         |           |
| S16 Mon   Dimensional Modelling .....                        | 32        |
| S17 Tue   Azure Warehousing: Synapse & Fabric .....          | 33        |
| S18 Wed   AWS Redshift & GCP BigQuery .....                  | 34        |
| S19 Thu   Orchestration & Scheduling .....                   | 35        |
| S20 Fri   Data Quality & Testing .....                       | 36        |
| <b>WEEK 5: CAPSTONE WEEK</b> .....                           | <b>38</b> |
| <i>10 class hours / 8 lab hours</i>                          |           |
| S21 Mon   Infrastructure as Code for Data Platforms .....    | 40        |
| S22 Tue   Capstone Briefing & Pipeline Design Workshop ..... | 41        |
| S23 Wed   Capstone Build Clinic .....                        | 42        |
| S24 Thu   Pipeline Debugging & Optimisation Drill .....      | 43        |
| S25 Fri   Assessment & Design Defence .....                  | 44        |
| Capstone and assessment .....                                | 46        |
| Key takeaways .....                                          | 47        |
| About ExaGuru .....                                          | 48        |



# LEVEL 100 AT A GLANCE

5 weeks, 5 layers, 25 sessions: 50 class hours + 45 lab hours





## WEEK 1

**LANDSCAPE, SQL, PYTHON AND STORAGE**

10 CLASS HOURS

8 LAB HOURS

**THIS WEEK, SESSION BY SESSION**

|     |     |                                   |          |                 |
|-----|-----|-----------------------------------|----------|-----------------|
| S01 | Mon | The Data Engineering Landscape    | Design   | 2 class   1 lab |
| S02 | Tue | SQL for Data Engineers            | Hands-on | 2 class   2 lab |
| S03 | Wed | Python & the PySpark Mental Model | Hands-on | 2 class   2 lab |
| S04 | Thu | File Formats & Storage Layout     | Hands-on | 2 class   1 lab |
| S05 | Fri | Cloud Object Storage              | Hands-on | 2 class   2 lab |

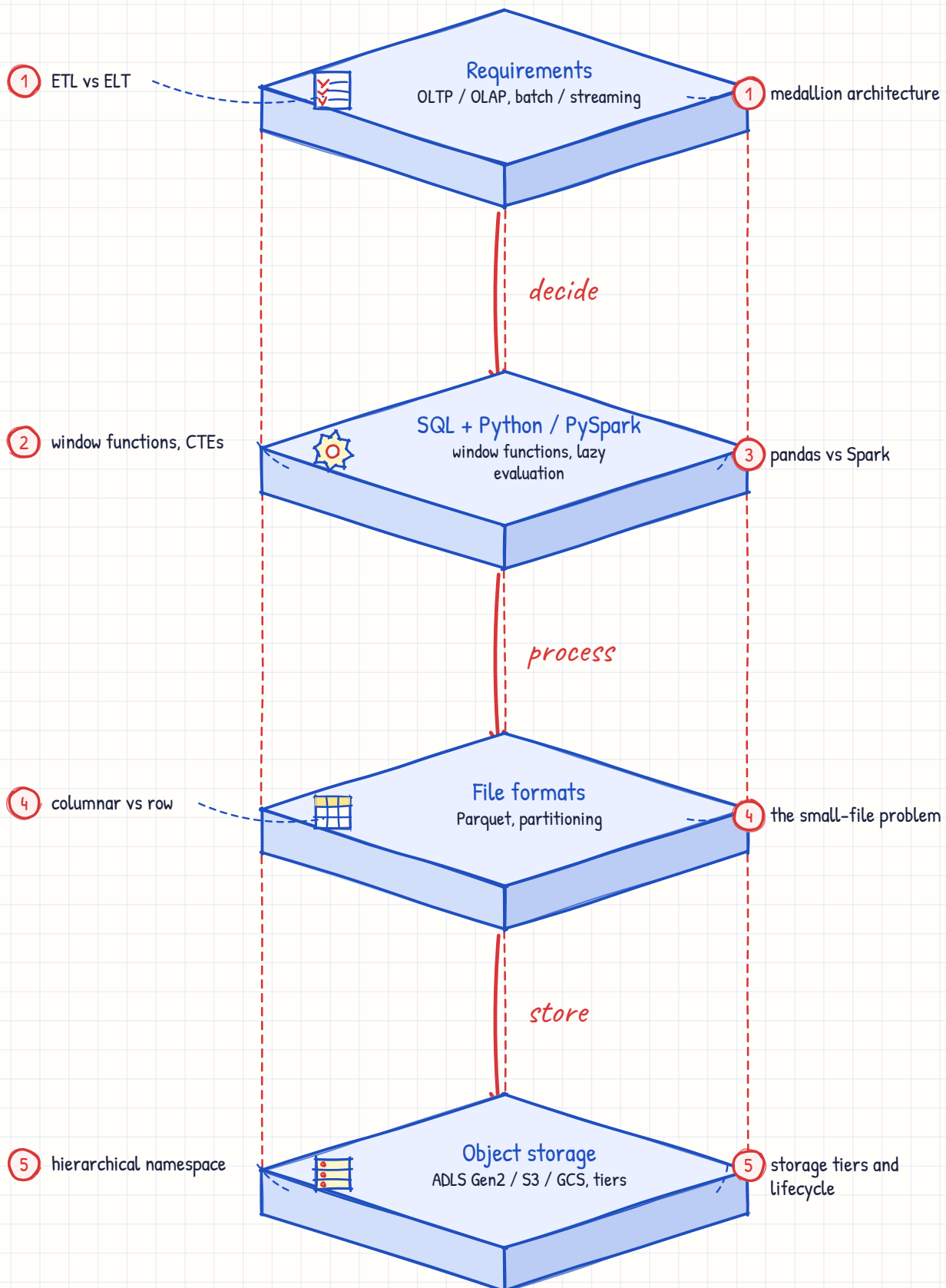
**LABS THIS WEEK**

- S01 Map a real reporting requirement to an architecture on paper
- S02 Solve 15 window-function and plan-reading problems on a real dataset
- S03 Rewrite a pandas script as PySpark; measure the crossover point
- S04 Convert a CSV dataset to partitioned Parquet; measure scan cost and size
- S05 Set up all three accounts WITH budget alerts; build an identical medallion folder layout

**EVERY SESSION, 120 MINUTES**

Class runs 18:00 to 20:00 IST, Monday to Friday. Labs are self-paced from written lab guides. Each concept is built in Azure first, then mapped to AWS and GCP in the same session.

*Next: the exploded architecture for Week 1.*

**WEEK 1 EXPLODED ARCHITECTURE**

Numbered circles = the session in this week (1 to 5). Find its section on the pages that follow.



1

# THE DATA ENGINEERING LANDSCAPE

Focus: Cloud-neutral

S01 / Monday / The Data Engineering Landscape

## TOPICS COVERED

- 1 **OLTP vs OLAP** - OLTP systems run day-to-day transactions: many small reads and writes. OLAP systems answer analytical questions over large volumes of history. Different jobs call for different designs.
- 2 **Batch vs streaming** - Batch processes data in scheduled chunks. Streaming processes records as they arrive. The choice follows how fresh the answer needs to be.
- 3 **ETL vs ELT and why ELT won** - ETL transforms data before loading it. ELT loads raw data first and transforms it inside the platform. ELT won because raw data stays available to reprocess.
- 4 **Lake vs warehouse vs lakehouse** - A lake stores raw files, a warehouse serves structured and modelled tables, and a lakehouse aims to give one platform both roles.
- 5 **Medallion architecture** - Data moves through Bronze, Silver and Gold layers. Each layer is cleaner and more ready for the business than the one before.
- 6 **Where the DE role stops and analyst/scientist begins** - Data engineers build reliable pipelines and platforms. Analysts and scientists use the prepared data. Knowing the boundary clarifies who owns what.

## AZURE / AWS / GCP

| AZURE                                     | AWS                               | GCP                             |
|-------------------------------------------|-----------------------------------|---------------------------------|
| Fabric / Databricks / Synapse positioning | Redshift / EMR / Glue positioning | BigQuery / Dataflow positioning |

## CONCEPT FLOW



### LAB

Map a real reporting requirement to an architecture on paper

Design

2 class hrs | 1 lab hr



### WHY THIS MATTERS

Every later choice, from storage to serving, starts from this vocabulary and these trade-offs.



### KEY TAKEAWAY

Architecture follows requirements. Know OLTP vs OLAP, batch vs streaming, ELT and the medallion layers before you pick tools.



2

## SQL FOR DATA ENGINEERS

Focus: Cross-cloud

S02 / Tuesday / SQL for Data Engineers

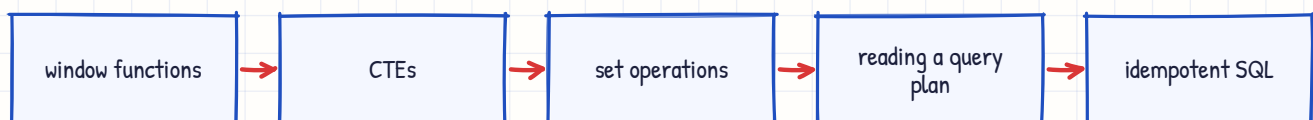
### TOPICS COVERED

- 1 **Beyond joins: window functions, CTEs, grouping sets, QUALIFY** – Window functions calculate across related rows without collapsing them. CTEs name intermediate steps. Grouping sets give several aggregations in one query. QUALIFY filters on window results.
- 2 **Set operations** – Set operations combine or compare whole result sets, row by row.
- 3 **Reading a query plan** – The plan shows how the engine will run your query: scans, joins and ordering. Reading it shows where the work really happens.
- 4 **Why your query is slow** – Slow queries usually come from how much data is read or moved. Start from the plan, not from guesses.
- 5 **Idempotent SQL** – Idempotent SQL gives the same result when run again, so a re-run after a failure cannot create duplicates.

### AZURE / AWS / GCP

| AZURE                             | AWS    | GCP      |
|-----------------------------------|--------|----------|
| Azure SQL, Synapse serverless SQL | Athena | BigQuery |

### CONCEPT FLOW



#### LAB

Solve 15 window-function and plan-reading problems on a real dataset

Hands-on

2 class hrs | 2 lab hrs



#### WHY THIS MATTERS

Most pipeline logic is SQL. Plan-reading and re-runnable SQL separate a working query from a dependable one.



#### KEY TAKEAWAY

Go beyond joins: window functions, CTEs and QUALIFY. Read the plan to find why a query is slow, and write SQL that is safe to re-run.



3

## PYTHON AND THE PYSPARK MENTAL MODEL

Focus: Cloud-neutral

S03 / Wednesday / Python &amp; the PySpark Mental Model

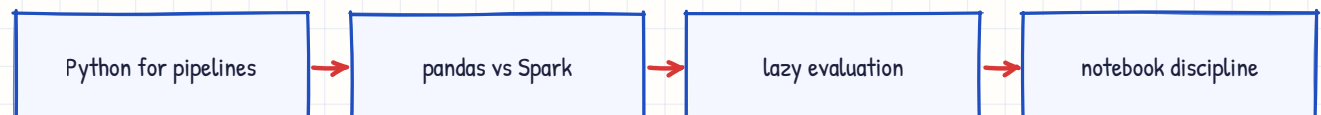
### TOPICS COVERED

- 1 Python for pipelines, not scripts – Pipeline code is structured, repeatable and testable, not a one-off script. Functions with clear inputs and outputs make it dependable.
- 2 Pandas vs Spark and when NOT to use Spark – pandas works on one machine; Spark spreads work across many. Spark adds overhead, so smaller data is often better without it.
- 3 Lazy evaluation – Spark records your transformations and only runs them when a result is needed. This lets it plan the whole job first.
- 4 Notebook discipline and why notebooks are not production – Notebooks suit exploration. Keep them clean and ordered, and move dependable logic into tested code for production.

### AZURE / AWS / GCP

| AZURE                      | AWS                  | GCP                         |
|----------------------------|----------------------|-----------------------------|
| Azure Databricks notebooks | EMR / Glue notebooks | Dataproc / Colab Enterprise |

### CONCEPT FLOW



#### LAB

Rewrite a pandas script as PySpark; measure the crossover point

Hands-on

2 class hrs | 2 lab hrs



#### WHY THIS MATTERS

Choosing the right engine, and writing code that can run unattended, is the core habit of a data engineer.



#### KEY TAKEAWAY

Spark is not always the answer. Understand lazy evaluation, and treat notebooks as exploration, not production.



4

# FILE FORMATS AND STORAGE LAYOUT

Focus: Cross-cloud

S04 / Thursday / File Formats &amp; Storage Layout

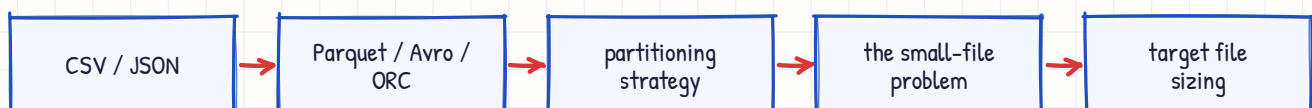
## TOPICS COVERED

- 1 **CSV / JSON / Parquet / Avro / ORC** - Each format trades readability against efficiency. CSV and JSON are text; Parquet, Avro and ORC are built for data platforms.
- 2 **Columnar vs row** - Row formats keep each record together. Columnar formats keep each column together, which suits analytics that read a few columns.
- 3 **Compression codecs** - Codecs shrink files to save storage and read time, at some processing cost. The codec choice affects size and speed.
- 4 **Predicate pushdown** - Filters are pushed down to the file reader so unneeded data is skipped instead of being loaded and then discarded.
- 5 **Partitioning strategy** - Partitioning splits data into folders by a column value so queries read only what they need. Pick columns that queries actually filter on.
- 6 **The small-file problem** - Many tiny files slow reads because each file adds overhead. Fewer, larger files are usually better.
- 7 **Target file sizing** - Aim for files that are neither tiny nor huge, so engines can read them efficiently and in parallel.

## AZURE / AWS / GCP

| AZURE                        | AWS                   | GCP                    |
|------------------------------|-----------------------|------------------------|
| ADLS Gen2 layout conventions | S3 layout conventions | GCS layout conventions |

## CONCEPT FLOW



### LAB

Convert a CSV dataset to partitioned Parquet; measure scan cost and size

Hands-on

2 class hrs | 1 lab hr



### WHY THIS MATTERS

File format and layout decide how much data every later query has to read, and so what it costs.



### KEY TAKEAWAY

Prefer columnar formats, partition on columns you filter by, and avoid small files. Layout is performance.



5

# CLOUD OBJECT STORAGE

Focus: Azure-led

SOS / Friday / Cloud Object Storage

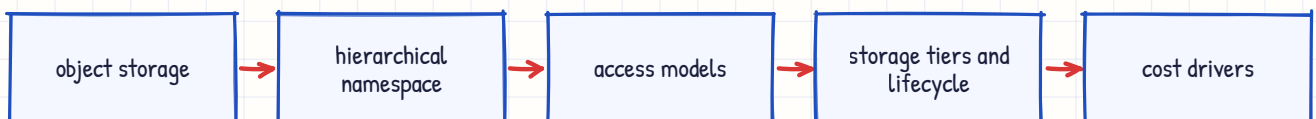
## TOPICS COVERED

- 1 **Object storage as the platform foundation** - Object storage holds data as objects in buckets or containers. It is the base layer the rest of the data platform builds on.
- 2 **Hierarchical namespace and why it matters for analytics** - A hierarchical namespace gives real folders and folder-level operations, which helps when analytics organises and secures data.
- 3 **Access models** - Access models decide who can read or write what, such as RBAC vs POSIX ACLs, bucket policies and IAM.
- 4 **Storage tiers and lifecycle** - Tiers trade storage price against access speed. Lifecycle rules move data between tiers as it ages.
- 5 **Cost drivers** - Cost comes from how much you store, which tier it sits in and how it is accessed. Budget alerts catch surprises early.

## AZURE / AWS / GCP

| AZURE                                      | AWS                                             | GCP                         |
|--------------------------------------------|-------------------------------------------------|-----------------------------|
| ADLS Gen2 (HNS, RBAC vs POSIX ACLs, tiers) | S3 (prefixes, storage classes, bucket policies) | GCS (buckets, classes, IAM) |

## CONCEPT FLOW



### LAB

Set up all three accounts WITH budget alerts;  
build an identical medallion folder layout

Hands-on

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

Every layer of the platform sits on object storage, so its access rules and costs touch everything above it.



### KEY TAKEAWAY

Object storage is the foundation. Learn namespaces, access models and tiers, and set budget alerts from day one.



## WEEK 1 RECAP

10 class hours / 8 lab hours

### I CAN EXPLAIN...

- ☐ I can explain OLTP vs OLAP, batch vs streaming, ETL vs ELT and the medallion architecture.
- ☐ I can explain window functions, CTEs, QUALIFY and why a query is slow.
- ☐ I can explain pandas vs Spark, lazy evaluation and why notebooks are not production.
- ☐ I can explain columnar vs row formats, partitioning and the small-file problem.
- ☐ I can explain the hierarchical namespace, access models and storage tiers.
- ☐ I can explain how ADLS Gen2, S3 and GCS map to each other.

### LABS I HAVE COMPLETED

- ☐ **S01** Map a real reporting requirement to an architecture on paper (Design)
- ☐ **S02** Solve 15 window-function and plan-reading problems on a real dataset (Hands-on)
- ☐ **S03** Rewrite a pandas script as PySpark; measure the crossover point (Hands-on)
- ☐ **S04** Convert a CSV dataset to partitioned Parquet; measure scan cost and size (Hands-on)
- ☐ **S05** Set up all three accounts WITH budget alerts; build an identical medallion folder layout (Hands-on)

### THE WEEK IN FIVE STEPS





# 2

## WEEK 2

# INGESTION

10 CLASS HOURS

9 LAB HOURS

### THIS WEEK, SESSION BY SESSION

|     |     |                                          |          |                 |
|-----|-----|------------------------------------------|----------|-----------------|
| S06 | Mon | Azure Data Factory Fundamentals          | Hands-on | 2 class   2 lab |
| S07 | Tue | ADF Ingestion Patterns                   | Hands-on | 2 class   2 lab |
| S08 | Wed | Relational Sources & Change Data Capture | Hands-on | 2 class   2 lab |
| S09 | Thu | AWS & GCP Ingestion Services             | Hands-on | 2 class   2 lab |
| S10 | Fri | Ingestion Design Clinic                  | Design   | 2 class   1 lab |



### LABS THIS WEEK

- S06 Build a first ADF pipeline copying from HTTP source to ADLS bronze
- S07 MP-A: Metadata-driven incremental ingestion framework in ADF
- S08 Configure CDC on a source table; land change feed into bronze
- S09 Rebuild the S06 ingestion pipeline in Glue and in Data Fusion; compare effort
- S10 Design ingestion for three written enterprise briefs and defend the choices

### EVERY SESSION, 120 MINUTES

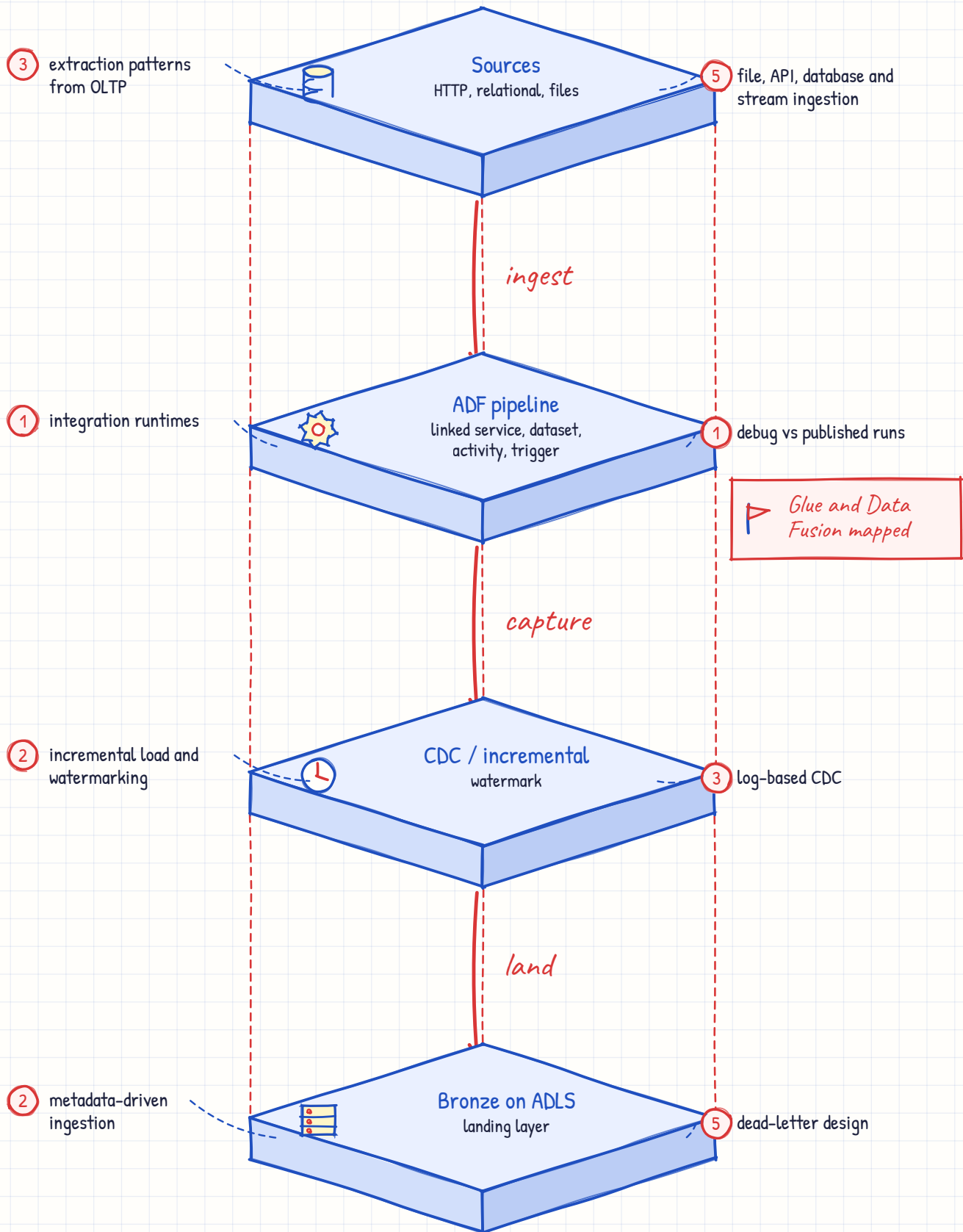


Class runs 18:00 to 20:00 IST, Monday to Friday. Labs are self-paced from written lab guides. Each concept is built in Azure first, then mapped to AWS and GCP in the same session.

*Next: the exploded architecture for Week 2.*



## WEEK 2 EXPLODED ARCHITECTURE



Numbered circles = the session in this week (1 to 5). Find its section on the pages that follow.



1

**AZURE DATA FACTORY FUNDAMENTALS**

Focus: Azure-led

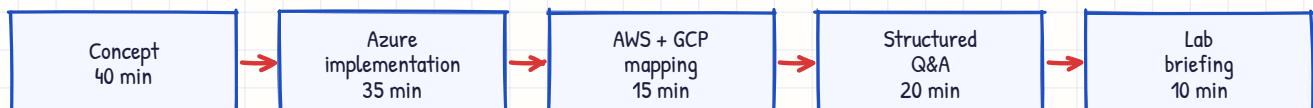
S06 / Monday / Azure Data Factory Fundamentals

**TOPICS COVERED**

- 1 Pipeline / activity / linked service / dataset model – A pipeline groups activities. A linked service holds the connection to a system. A dataset points to the data inside it.
- 2 Integration runtimes and when you need self-hosted – The integration runtime is the compute that moves data. A self-hosted one is needed when the source sits where the cloud runtime cannot reach.
- 3 Triggers – Triggers start pipeline runs, for example on a schedule.
- 4 Debug vs published runs – Debug runs test your pipeline while you build. Published runs use the saved, published version.

**AZURE / AWS / GCP**

| AZURE              | AWS                                   | GCP                  |
|--------------------|---------------------------------------|----------------------|
| ADF (full session) | Glue Studio / Step Functions (mapped) | Data Fusion (mapped) |

**CONCEPT FLOW****THIS SESSION, 120 MINUTES****LAB**

Build a first ADF pipeline copying from HTTP source to ADLS bronze

Hands-on

2 class hrs | 2 lab hrs

**WHY THIS MATTERS**

ADF is the first ingestion tool in the program. Its model of pipelines, activities and datasets recurs everywhere.

**KEY TAKEAWAY**

Learn the ADF model: linked service, dataset, activity, pipeline, trigger. Know debug from published runs.



2

## ADF INGESTION PATTERNS

Focus: Azure-led

S07 / Tuesday / ADF Ingestion Patterns

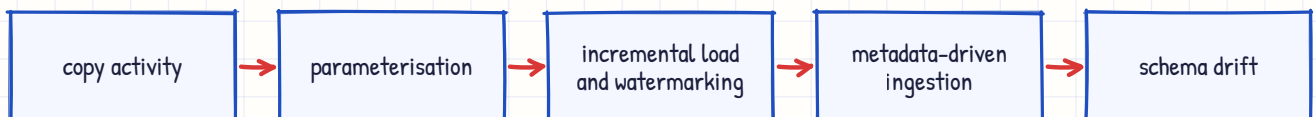
### TOPICS COVERED

- 1 **Copy activity at scale** – The copy activity moves data between stores. At scale, you design it to handle many sources and larger volumes.
- 2 **Parameterisation** – Parameters let one pipeline serve many sources by passing in names, paths or values at run time.
- 3 **Incremental load and watermarking** – An incremental load copies only new or changed rows. A watermark remembers the last value loaded so the next run continues from there.
- 4 **Metadata-driven ingestion frameworks** – Metadata drives a generic pipeline, so adding a source means adding a row of metadata, not building a new pipeline.
- 5 **Handling schema drift** – Schema drift is when source columns change over time. Plan how pipelines react instead of failing unexpectedly.

### AZURE / AWS / GCP

| AZURE                                       | AWS                         | GCP                               |
|---------------------------------------------|-----------------------------|-----------------------------------|
| ADF parameterised metadata-driven framework | Glue job bookmarks (mapped) | Data Fusion / Datastream (mapped) |

### CONCEPT FLOW



#### LAB

MP-A: Metadata-driven incremental ingestion framework in ADF

Hands-on

2 class hrs | 2 lab hrs



#### WHY THIS MATTERS

Real ingestion covers many sources. Parameters and metadata keep that manageable.



#### KEY TAKEAWAY

Parameterise, load incrementally with a watermark, and drive ingestion from metadata.



3

**RELATIONAL SOURCES AND CHANGE****DATA CAPTURE**

Focus: Azure-led

S08 / Wednesday / Relational Sources &amp; Change Data Capture

**TOPICS COVERED**

- 1 **Extraction patterns from OLTP** - Pulling data from transactional databases needs care so that reporting loads do not disturb the application.
- 2 **Full vs incremental vs CDC** - A full load copies everything. An incremental load copies what changed since last time. CDC captures each change as it happens.
- 3 **Log-based CDC concepts** - Log-based CDC reads the database's change log instead of querying tables, so it can see inserts, updates and deletes.
- 4 **Handling deletes** - A simple incremental query cannot see deleted rows. CDC records deletes so they can be applied downstream.
- 5 **Minimising source impact** - Choose extraction methods that put the least load on the source system.

**AZURE / AWS / GCP**

| AZURE                                  | AWS             | GCP                   |
|----------------------------------------|-----------------|-----------------------|
| Azure SQL CDC, ADF change data capture | DMS, Aurora CDC | Datastream, Cloud SQL |

**CONCEPT FLOW****LAB**

Configure CDC on a source table; land change feed into bronze

Hands-on

2 class hrs | 2 lab hrs

**WHY THIS MATTERS**

Business data lives in transactional databases. Extracting it without harming them is a daily concern.

**KEY TAKEAWAY**

Choose full, incremental or CDC deliberately. Log-based CDC sees deletes and spares the source.



4

# AWS AND GCP INGESTION SERVICES

Focus: AWS + GCP

S09 / Thursday / AWS &amp; GCP Ingestion Services

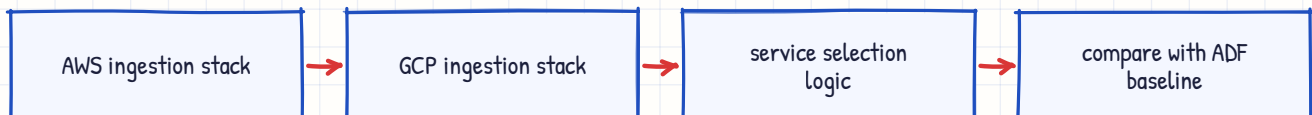
## TOPICS COVERED

- 1 The AWS and GCP ingestion stacks in depth – Each cloud offers its own set of ingestion services. This session walks through the AWS and GCP stacks listed in the box above.
- 2 Service selection logic – Choose a service by the kind of source, the volume and how fresh the data must be.
- 3 What each does better than ADF and where it is worse – Compare each service with ADF, the baseline, honestly: its strengths and its gaps.

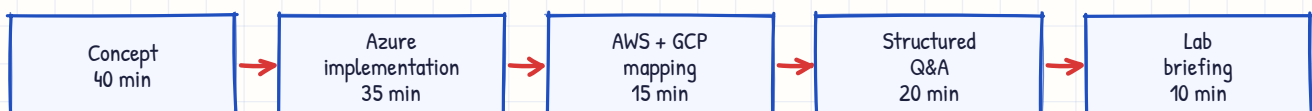
## AZURE / AWS / GCP

| AZURE                 | AWS                                 | GCP                                                           |
|-----------------------|-------------------------------------|---------------------------------------------------------------|
| (comparison baseline) | Glue, DMS, AppFlow, Transfer Family | Data Fusion, Datastream, Storage Transfer, Dataflow templates |

## CONCEPT FLOW



## THIS SESSION, 120 MINUTES



### LAB

Rebuild the S06 ingestion pipeline in Glue and in Data Fusion; compare effort

Hands-on

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

Teams rarely stay on one cloud. Comparing services against a known baseline makes the choice rational.



### KEY TAKEAWAY

Know each cloud's ingestion stack and where it beats, or trails, ADF.



5

**INGESTION DESIGN CLINIC**

Focus: Cross-cloud

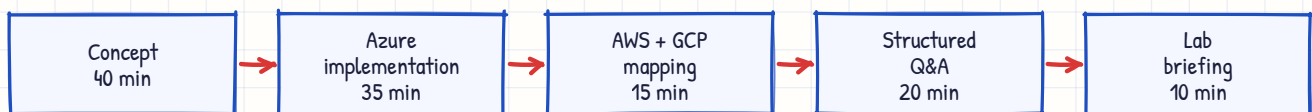
S10 / Friday / Ingestion Design Clinic

**TOPICS COVERED**

- 1 Choosing between file, API, database and stream ingestion – Each source type needs a different approach. Match the ingestion method to how the source delivers data.
- 2 Idempotency and replay – Idempotent loads can be repeated safely. Replay means re-running ingestion for a period without creating duplicates.
- 3 Late-arriving and out-of-order data – Records can arrive after their expected time or in the wrong order. Pipelines need rules for both.
- 4 Schema drift policy – Decide in advance how the pipeline responds when a source schema changes.
- 5 Dead-letter design – A dead-letter location collects records that fail processing, so bad records do not block good ones and can be reviewed later.

**AZURE / AWS / GCP**

|                              |                             |                                    |
|------------------------------|-----------------------------|------------------------------------|
| <b>AZURE</b><br>ADF patterns | <b>AWS</b><br>Glue patterns | <b>GCP</b><br>Data Fusion patterns |
|------------------------------|-----------------------------|------------------------------------|

**CONCEPT FLOW****THIS SESSION, 120 MINUTES****LAB**

Design ingestion for three written enterprise briefs and defend the choices

Design

2 class hrs | 1 lab hr

**WHY THIS MATTERS**

Ingestion design is where reliability is won or lost. A clinic builds the habit of defending each choice.

**KEY TAKEAWAY**

Design for replay, late data, drift and failure from the start. Defend your choices.



## WEEK 2 RECAP

10 class hours / 9 lab hours

### I CAN EXPLAIN...

- ☐ I can explain the ADF model: pipeline, activity, linked service, dataset and trigger.
- ☐ I can explain incremental load, watermarking and metadata-driven ingestion.
- ☐ I can explain full vs incremental vs CDC, and how log-based CDC handles deletes.
- ☐ I can explain how the AWS and GCP ingestion services compare with ADF.
- ☐ I can explain idempotency, replay, late-arriving data and dead-letter design.

### LABS I HAVE COMPLETED

- ☐ **S06** Build a first ADF pipeline copying from HTTP source to ADLS bronze (Hands-on)
- ☐ **S07** MP-A: Metadata-driven incremental ingestion framework in ADF (Hands-on)
- ☐ **S08** Configure CDC on a source table; land change feed into bronze (Hands-on)
- ☐ **S09** Rebuild the S06 ingestion pipeline in Glue and in Data Fusion; compare effort (Hands-on)
- ☐ **S10** Design ingestion for three written enterprise briefs and defend the choices (Design)

### THE WEEK IN FIVE STEPS





## WEEK 3

**SPARK, DELTA LAKE AND  
MEDALLION**

10 CLASS HOURS

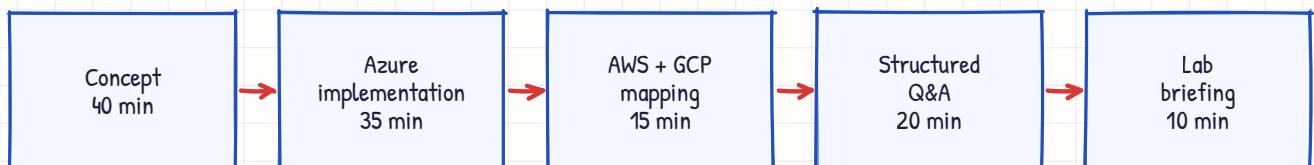
10 LAB HOURS

**THIS WEEK, SESSION BY SESSION**

|     |     |                                        |          |                 |
|-----|-----|----------------------------------------|----------|-----------------|
| S11 | Mon | Spark Architecture                     | Hands-on | 2 class   2 lab |
| S12 | Tue | PySpark Transformations                | Hands-on | 2 class   2 lab |
| S13 | Wed | Delta Lake                             | Hands-on | 2 class   2 lab |
| S14 | Thu | Delta Optimisation & Table Maintenance | Hands-on | 2 class   2 lab |
| S15 | Fri | Medallion Architecture Build           | Hands-on | 2 class   2 lab |

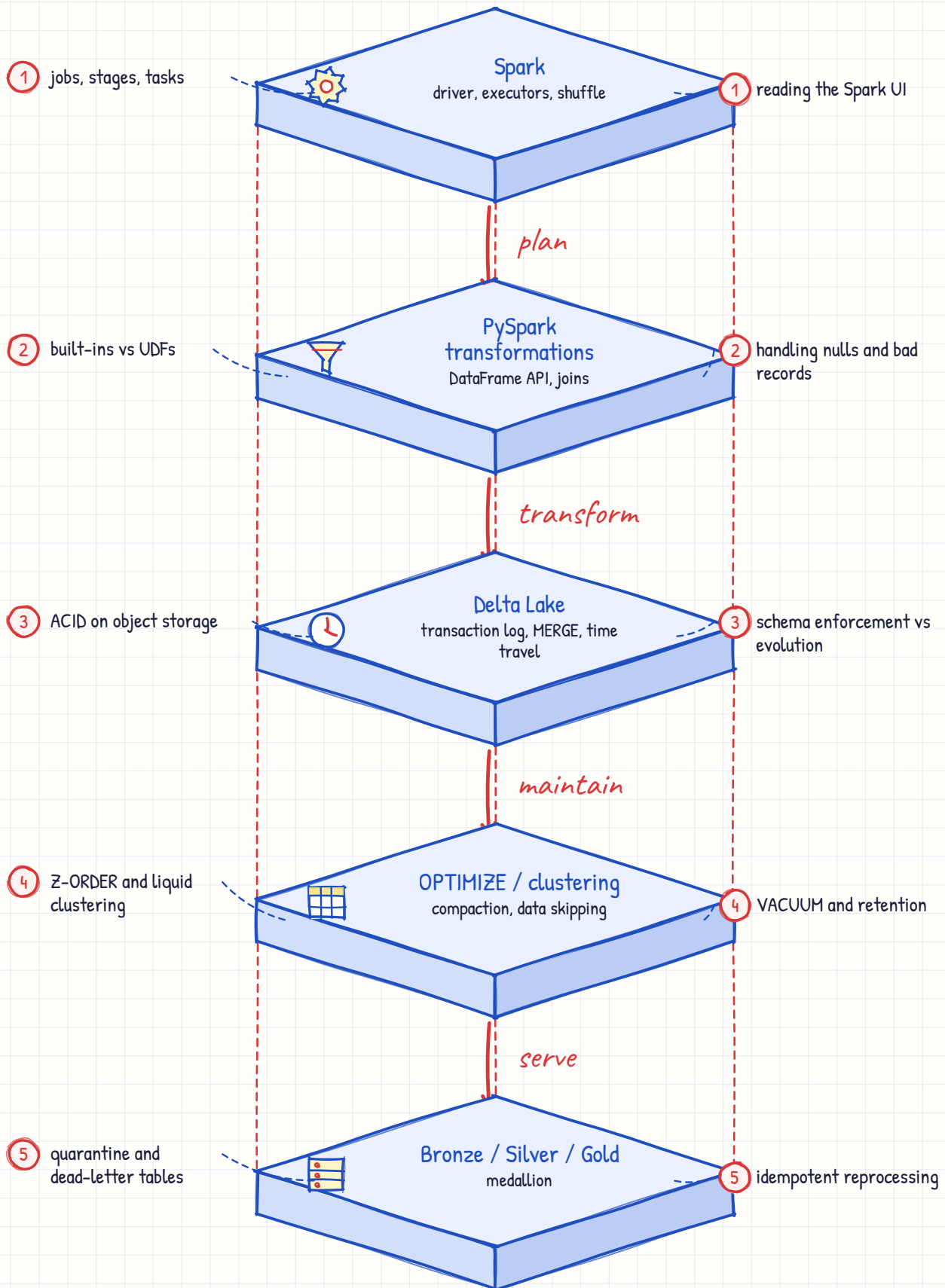
**LABS THIS WEEK**

- S11 Read the Spark UI for a deliberately bad query; identify the shuffle
- S12 MP-B: Transform raw bronze into cleaned, conformed silver tables
- S13 Implement SCD Type 2 with MERGE; use time travel to recover a bad load
- S14 Measure query time before and after OPTIMIZE and clustering on the same table
- S15 MP-B: Complete bronze to silver to gold pipeline, reprocessable end to end

**EVERY SESSION, 120 MINUTES**

Class runs 18:00 to 20:00 IST, Monday to Friday. Labs are self-paced from written lab guides. Each concept is built in Azure first, then mapped to AWS and GCP in the same session.

*Next: the exploded architecture for Week 3.*

**WEEK 3 EXPLODED ARCHITECTURE**

Numbered circles = the session in this week (1 to 5). Find its section on the pages that follow.



1

# SPARK ARCHITECTURE

Focus: Cloud-neutral

S11 / Monday / Spark Architecture

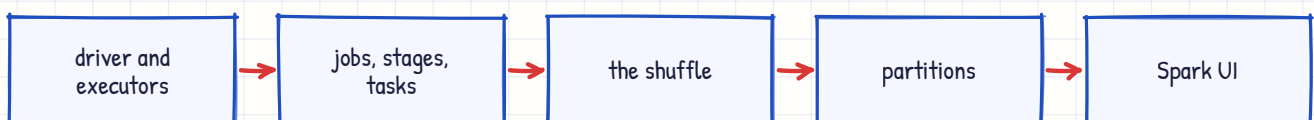
## TOPICS COVERED

- 1 **Driver and executors** – The driver plans and coordinates the job. Executors do the work on partitions of the data.
- 2 **Jobs, stages, tasks** – An action creates a job. A job splits into stages, and stages split into tasks that run in parallel.
- 3 **The shuffle and why it dominates runtime** – A shuffle moves data between executors, for example for joins and aggregations. Moving data across the cluster is often the costliest step.
- 4 **Partitions** – Partitions are the chunks of data Spark processes in parallel. Their number and size affect speed.
- 5 **Catalyst and adaptive query execution** – Catalyst optimises query plans. Adaptive query execution adjusts the plan at run time using what Spark learns.
- 6 **Reading the Spark UI** – The Spark UI shows jobs, stages and tasks, so you can see where time is spent and spot shuffles.

## AZURE / AWS / GCP

| AZURE                           | AWS             | GCP                |
|---------------------------------|-----------------|--------------------|
| Azure Databricks / Fabric Spark | EMR, Glue Spark | Dataproc, Dataflow |

## CONCEPT FLOW



### LAB

Read the Spark UI for a deliberately bad query; identify the shuffle

Hands-on

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

You cannot tune what you cannot read. Spark's architecture explains almost every slow job.



### KEY TAKEAWAY

Learn driver, executors, stages and the shuffle. Use the Spark UI to find where time goes.



2

# PYSPARK TRANSFORMATIONS

Focus: Azure-led

S12 / Tuesday / PySpark Transformations

## TOPICS COVERED

- 1 **DataFrame API in depth** – DataFrames hold tabular data with named columns. Their API expresses selects, filters and transformations clearly.
- 2 **Joins and join strategies** – Spark can run a join in different ways. The strategy chosen affects how much data is shuffled.
- 3 **Aggregations** – Aggregations group rows and summarise them, which usually involves a shuffle.
- 4 **Built-ins vs UDFs and the performance cliff** – Built-in functions are optimised by Spark. UDFs are custom code the optimiser cannot see into, so they can slow jobs sharply.
- 5 **Handling nulls and bad records** – Decide how to treat missing values and records that do not fit, so they do not corrupt silver tables.

## AZURE / AWS / GCP

| AZURE                             | AWS                 | GCP               |
|-----------------------------------|---------------------|-------------------|
| Databricks Free Edition notebooks | EMR / Glue (mapped) | Dataproc (mapped) |

## CONCEPT FLOW



### LAB

MP-B: Transform raw bronze into cleaned, conformed silver tables

Hands-on

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

Transformations turn raw bronze data into trustworthy silver tables. The details decide quality and speed.



### KEY TAKEAWAY

Prefer built-ins to UDFs, understand join strategies, and handle nulls and bad records on purpose.



3

**DELTA LAKE**

S13 / Wednesday / Delta Lake

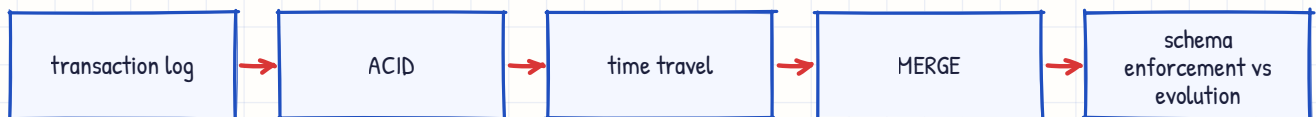
Focus: Azure-led

**TOPICS COVERED**

- ① **The transaction log and how ACID works on object storage** – Delta Lake keeps a transaction log beside the data files. The log records each commit, giving ACID guarantees on object storage.
- ② **Time travel** – Because the log keeps history, you can read the table as it was at an earlier version.
- ③ **MERGE for upserts** – MERGE updates matching rows and inserts new ones in a single operation.
- ④ **Schema enforcement vs evolution** – Enforcement rejects data that does not match the table schema. Evolution lets the schema change deliberately.
- ⑤ **Concurrency and conflict** – Several writers can work on a table at once. The log detects conflicting changes.

**AZURE / AWS / GCP**

| AZURE                        | AWS                                            | GCP                                     |
|------------------------------|------------------------------------------------|-----------------------------------------|
| Delta on Databricks + Fabric | Delta / Iceberg on EMR, Glue<br>Iceberg tables | Delta / Iceberg on Dataproc,<br>BigLake |

**CONCEPT FLOW****THIS SESSION, 120 MINUTES****LAB**

Implement SCD Type 2 with MERGE; use time travel to recover a bad load

Hands-on

2 class hrs | 2 lab hrs

**WHY THIS MATTERS**

Delta Lake brings reliable, transactional tables to plain object storage. Recovery and upserts become routine.

**KEY TAKEAWAY**

The transaction log gives ACID, time travel and safe MERGE upserts on object storage.



4

## DELTA OPTIMISATION AND TABLE MAINTENANCE

Focus: Azure-led

S14 / Thursday / Delta Optimisation &amp; Table Maintenance

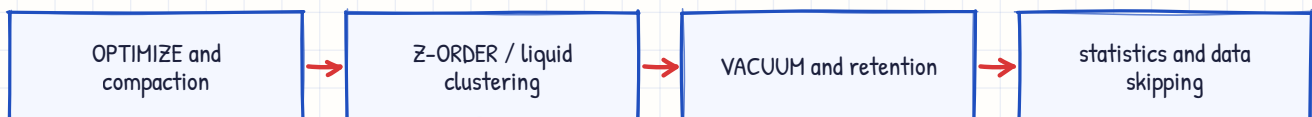
### TOPICS COVERED

- 1 **OPTIMIZE and compaction** - OPTIMIZE compacts many small files into fewer, larger ones to speed up reads.
- 2 **Z-ORDER and liquid clustering** - Both organise data by chosen columns so related rows sit together, helping queries that filter on them.
- 3 **VACUUM and retention** - VACUUM removes old files that are no longer needed. The retention setting controls how long they are kept.
- 4 **Partition strategy and over-partitioning** - Too many partitions create many small files. Partition only where it helps.
- 5 **Statistics and data skipping** - File-level statistics let the engine skip files that cannot match a filter.

### AZURE / AWS / GCP

| AZURE                                  | AWS                                   | GCP                           |
|----------------------------------------|---------------------------------------|-------------------------------|
| Databricks OPTIMIZE, liquid clustering | Iceberg compaction, Glue optimisation | BigLake / Iceberg maintenance |

### CONCEPT FLOW



### LAB

Measure query time before and after OPTIMIZE and clustering on the same table

Hands-on

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

Tables degrade as data lands. Maintenance keeps queries fast and files tidy.



### KEY TAKEAWAY

Compact, cluster and vacuum on purpose. Measure before and after, and avoid over-partitioning.



5

# MEDALLION ARCHITECTURE BUILD

Focus: Azure-led

S15 / Friday / Medallion Architecture Build

## TOPICS COVERED

- 1 **Bronze / silver / gold responsibilities** – Bronze keeps raw data, silver holds cleaned and conformed data, and gold serves business-ready tables.
- 2 **Idempotent reprocessing** – Re-running a layer produces the same result, so you can reprocess safely.
- 3 **Quarantine and dead-letter tables** – Bad records are set aside in separate tables instead of failing the whole load.
- 4 **Full vs incremental refresh** – A full refresh rebuilds a table from scratch. An incremental refresh updates only what changed.
- 5 **Reprocessing without downtime** – Design so that readers keep seeing valid data while a table is being rebuilt.

## AZURE / AWS / GCP

| AZURE                       | AWS                 | GCP                      |
|-----------------------------|---------------------|--------------------------|
| Databricks + ADLS medallion | S3 + Glue medallion | GCS + BigQuery medallion |

## CONCEPT FLOW



### LAB

MP-B: Complete bronze to silver to gold pipeline, reprocessible end to end

Hands-on

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

The medallion pipeline is the program's central build. Everything before it feeds it, everything after serves from it.



### KEY TAKEAWAY

Give each layer a clear job, keep reprocessing idempotent, and quarantine bad records.



## WEEK 3 RECAP

10 class hours / 10 lab hours

### I CAN EXPLAIN...

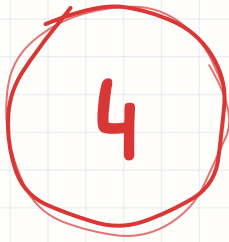
- ☐ I can explain driver, executors, stages and why the shuffle dominates runtime.
- ☐ I can explain built-ins vs UDFs and join strategies.
- ☐ I can explain the Delta transaction log, time travel and MERGE.
- ☐ I can explain OPTIMIZE, liquid clustering, VACUUM and over-partitioning.
- ☐ I can explain bronze, silver and gold responsibilities and idempotent reprocessing.

### LABS I HAVE COMPLETED

- ☐ S11 Read the Spark UI for a deliberately bad query; identify the shuffle (Hands-on)
- ☐ S12 MP-B: Transform raw bronze into cleaned, conformed silver tables (Hands-on)
- ☐ S13 Implement SCD Type 2 with MERGE; use time travel to recover a bad load (Hands-on)
- ☐ S14 Measure query time before and after OPTIMIZE and clustering on the same table (Hands-on)
- ☐ S15 MP-B: Complete bronze to silver to gold pipeline, reprocessible end to end (Hands-on)

### THE WEEK IN FIVE STEPS





WEEK 4

**MODELLING, SERVING AND QUALITY**

10 CLASS HOURS

10 LAB HOURS

**THIS WEEK, SESSION BY SESSION**

|     |     |                                     |          |                 |
|-----|-----|-------------------------------------|----------|-----------------|
| S16 | Mon | Dimensional Modelling               | Design   | 2 class   2 lab |
| S17 | Tue | Azure Warehousing: Synapse & Fabric | Hands-on | 2 class   2 lab |
| S18 | Wed | AWS Redshift & GCP BigQuery         | Mixed    | 2 class   2 lab |
| S19 | Thu | Orchestration & Scheduling          | Hands-on | 2 class   2 lab |
| S20 | Fri | Data Quality & Testing              | Hands-on | 2 class   2 lab |

**LABS THIS WEEK**

S16 Model a retail scenario into a star schema and justify the grain

S17 MP-C: Serve the gold layer through a Fabric Lakehouse and a Warehouse; compare

S18 Load the same gold dataset into BigQuery and Redshift Serverless; compare cost and speed

S19 Orchestrate the full medallion pipeline with dependencies, retries and failure alerts

S20 MP-C: Add a quality gate that blocks bad data from reaching gold

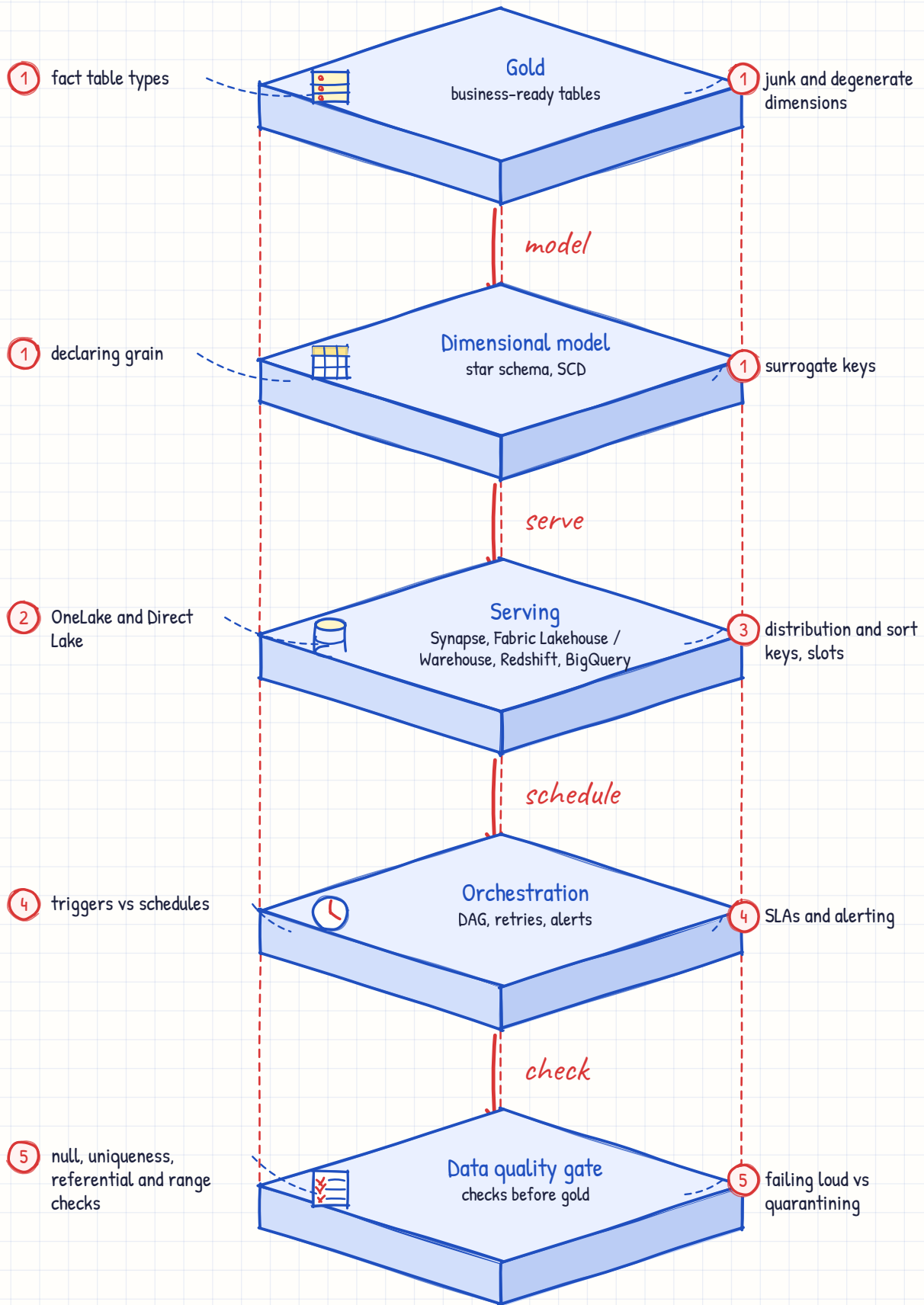
**EVERY SESSION, 120 MINUTES**

Class runs 18:00 to 20:00 IST, Monday to Friday. Labs are self-paced from written lab guides. Each concept is built in Azure first, then mapped to AWS and GCP in the same session.

*Next: the exploded architecture for Week 4.*



## WEEK 4 EXPLODED ARCHITECTURE



Numbered circles = the session in this week (1 to 5). Find its section on the pages that follow.



1

# DIMENSIONAL MODELLING

Focus: Cloud-neutral

S16 / Monday / Dimensional Modelling

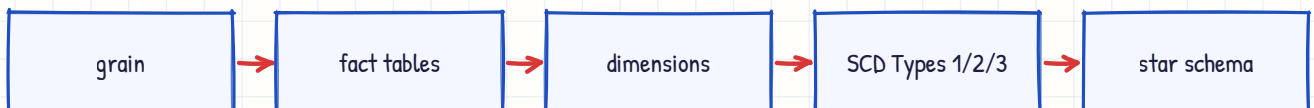
## TOPICS COVERED

- 1 **Star schema** – A central fact table surrounded by dimension tables. It is easy to query and easy to understand.
- 2 **Declaring grain** – Grain states exactly what one row in the fact table represents. Declare it first.
- 3 **Fact table types (transaction, periodic, accumulating)** – A transaction fact records each event. A periodic fact records a snapshot per period. An accumulating fact tracks a process through its stages.
- 4 **SCD Types 1/2/3** – Slowly changing dimensions handle changed attributes. Type 1 overwrites, Type 2 keeps history as new rows, and Type 3 keeps limited history in extra columns.
- 5 **Surrogate keys** – Surrogate keys are system-made identifiers for dimension rows, independent of source keys.
- 6 **Conformed dimensions** – Conformed dimensions are shared consistently across fact tables so results line up.
- 7 **Junk and degenerate dimensions** – A junk dimension groups small, miscellaneous flags. A degenerate dimension is an identifier kept in the fact table.

## AZURE / AWS / GCP

|                                                |                                      |                                      |
|------------------------------------------------|--------------------------------------|--------------------------------------|
| <b>AZURE</b><br>Modelling for Synapse / Fabric | <b>AWS</b><br>Modelling for Redshift | <b>GCP</b><br>Modelling for BigQuery |
|------------------------------------------------|--------------------------------------|--------------------------------------|

## CONCEPT FLOW



### LAB

Model a retail scenario into a star schema and justify the grain

Design

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

The gold layer is only as useful as its model. Grain and dimensions decide what questions the business can ask.



### KEY TAKEAWAY

Declare the grain first. Choose fact types and SCD types deliberately, and share conformed dimensions.



2

**AZURE WAREHOUSING: SYNAPSE AND FABRIC**

Focus: Azure-led

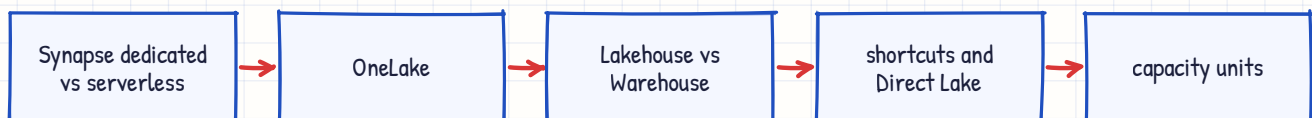
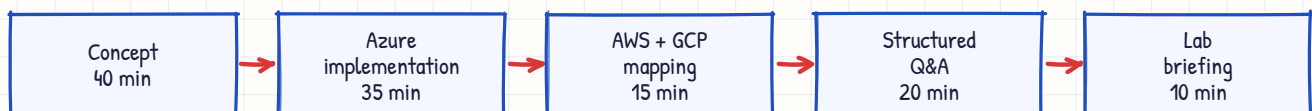
S17 / Tuesday / Azure Warehousing: Synapse &amp; Fabric

**TOPICS COVERED**

- 1 Synapse dedicated vs serverless SQL pool – Dedicated pools are provisioned capacity for warehouse workloads. Serverless pools query files in the lake on demand.
- 2 Microsoft Fabric: OneLake, Lakehouse vs Warehouse, shortcuts, Direct Lake – Fabric centres on OneLake. Lakehouse and Warehouse are two ways to serve data. Shortcuts reference data without copying it, and Direct Lake lets reports read lake data directly.
- 3 Capacity units – Fabric work consumes capacity units, so sizing and usage matter.
- 4 When to pick which – Choose between these options by workload and by who will use the data.

**AZURE / AWS / GCP**

| AZURE                                        | AWS             | GCP             |
|----------------------------------------------|-----------------|-----------------|
| Synapse + Microsoft Fabric<br>(full session) | (mapped in S18) | (mapped in S18) |

**CONCEPT FLOW****THIS SESSION, 120 MINUTES****LAB**

MP-C: Serve the gold layer through a Fabric Lakehouse and a Warehouse; compare

Hands-on

2 class hrs | 2 lab hrs

**WHY THIS MATTERS**

Serving is where gold data meets users. Knowing Azure's options lets you pick the right one.

**KEY TAKEAWAY**

Compare Synapse pools and Fabric's Lakehouse and Warehouse, and pick by workload.



3

# AWS REDSHIFT AND GCP BIGQUERY

Focus: AWS + GCP

S18 / Wednesday / AWS Redshift &amp; GCP BigQuery

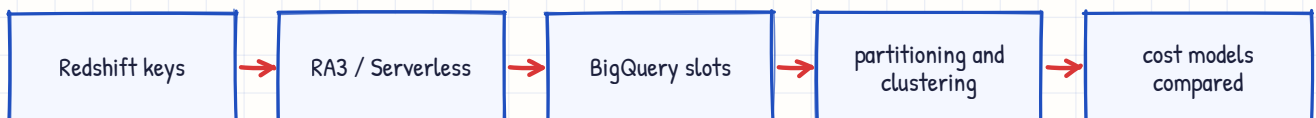
## TOPICS COVERED

- 1 Redshift architecture, distribution and sort keys, RA3, Serverless – Redshift is the AWS warehouse. Distribution and sort keys control how data is placed and ordered. RA3 and Serverless are node and deployment options.
- 2 BigQuery architecture, slots, partitioning and clustering – BigQuery is the GCP warehouse. Slots are units of compute capacity. Partitioning and clustering organise tables so queries scan less.
- 3 Cost models compared – The two services charge in different ways. Compare how each model behaves for your workload.
- 4 When each genuinely wins – Neither is best everywhere. Match each to the workloads where it is strongest.

## AZURE / AWS / GCP

| AZURE                 | AWS                           | GCP                 |
|-----------------------|-------------------------------|---------------------|
| (comparison baseline) | Redshift, Redshift Serverless | BigQuery, BI Engine |

## CONCEPT FLOW



### LAB

Load the same gold dataset into BigQuery and Redshift Serverless; compare cost and speed

Mixed

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

Comparing the same dataset on two warehouses turns opinion into evidence.



### KEY TAKEAWAY

Know Redshift and BigQuery architecture and cost models, and when each genuinely wins.



4

# ORCHESTRATION AND SCHEDULING

Focus: Azure-led

S19 / Thursday / Orchestration &amp; Scheduling

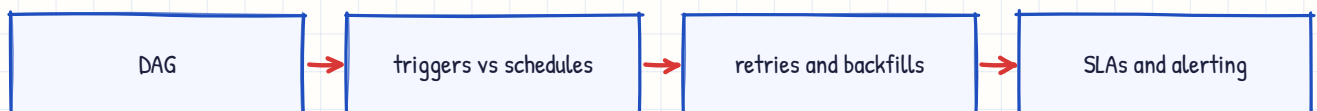
## TOPICS COVERED

- 1 **DAGs and dependency management** - A DAG lays out tasks and their dependencies, so each task runs only when what it needs has finished.
- 2 **Triggers vs schedules** - A schedule runs on the clock. A trigger runs when an event or condition occurs.
- 3 **Retries, backfills and catch-up** - Retries rerun failed tasks. Backfills and catch-up process periods that were missed.
- 4 **SLAs and alerting** - An SLA sets when data must be ready. Alerts tell you quickly when a run fails or runs late.
- 5 **Idempotent task design** - Each task should be safe to repeat, so retries and backfills do not duplicate data.
- 6 **Parameter passing** - Pass values such as dates between tasks and pipelines instead of hard-coding them.

## AZURE / AWS / GCP

| AZURE                               | AWS                            | GCP                       |
|-------------------------------------|--------------------------------|---------------------------|
| ADF triggers, Fabric Data Pipelines | Airflow / MWAA, Step Functions | Cloud Composer, Workflows |

## CONCEPT FLOW



### LAB

Orchestrate the full medallion pipeline with dependencies, retries and failure alerts

**Hands-on**

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

A pipeline that needs someone to press a button is not a pipeline. Orchestration makes it run, recover and report.



### KEY TAKEAWAY

Model dependencies as a DAG, design tasks to be idempotent, and alert on failure.



5

## DATA QUALITY AND TESTING

Focus: Cross-cloud

S20 / Friday / Data Quality &amp; Testing

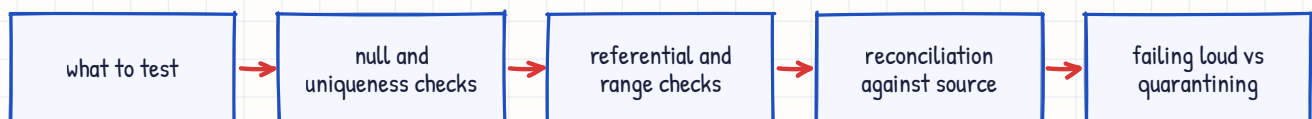
### TOPICS COVERED

- 1 What to test in a pipeline – Decide which checks protect the data: its shape, its values and its totals.
- 2 Null, uniqueness, referential and range checks – Check for missing values, duplicate keys, orphan references and values outside expected limits.
- 3 Reconciliation against source – Compare counts or totals with the source to show nothing was lost or doubled.
- 4 Expectations frameworks – Frameworks let you declare expectations about data and run them automatically.
- 5 Failing loud vs quarantining – Failing loud stops the pipeline when a check fails. Quarantining sets bad records aside while good ones continue.

### AZURE / AWS / GCP

| AZURE                                        | AWS                      | GCP                   |
|----------------------------------------------|--------------------------|-----------------------|
| Great Expectations / dbt tests on Databricks | Deequ, Glue Data Quality | Dataplex data quality |

### CONCEPT FLOW



#### LAB

MP-C: Add a quality gate that blocks bad data from reaching gold

Hands-on

2 class hrs | 2 lab hrs



#### WHY THIS MATTERS

Trust in data is earned by testing it. A quality gate protects everything downstream.



#### KEY TAKEAWAY

Test nulls, uniqueness, references and ranges, reconcile with the source, and decide: fail loud or quarantine.



## WEEK 4 RECAP

10 class hours / 10 lab hours

### I CAN EXPLAIN...

- ☐ I can explain star schema, grain, fact table types and SCD Types 1/2/3.
- ☐ I can explain Synapse dedicated vs serverless and Fabric Lakehouse vs Warehouse.
- ☐ I can explain Redshift and BigQuery architecture, and their cost models.
- ☐ I can explain DAGs, retries, backfills, SLAs and alerting.
- ☐ I can explain null, uniqueness, referential and range checks, and failing load vs quarantining.

### LABS I HAVE COMPLETED

- ☐ **S16** Model a retail scenario into a star schema and justify the grain (Design)
- ☐ **S17** MP-C: Serve the gold layer through a Fabric Lakehouse and a Warehouse; compare (Hands-on)
- ☐ **S18** Load the same gold dataset into BigQuery and Redshift Serverless; compare cost and speed (Mixed)
- ☐ **S19** Orchestrate the full medallion pipeline with dependencies, retries and failure alerts (Hands-on)
- ☐ **S20** MP-C: Add a quality gate that blocks bad data from reaching gold (Hands-on)

### THE WEEK IN FIVE STEPS





5

WEEK 5

## CAPSTONE WEEK

10 CLASS HOURS

8 LAB HOURS

## THIS WEEK, SESSION BY SESSION

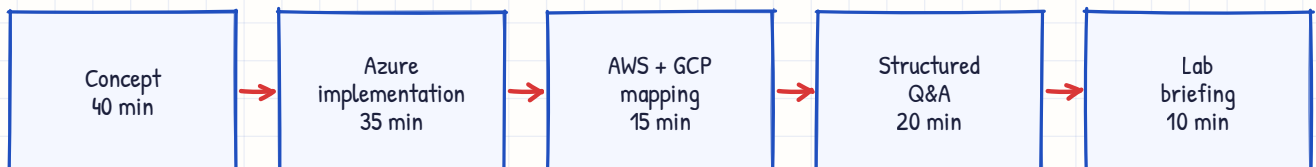
|     |     |                                              |          |                 |
|-----|-----|----------------------------------------------|----------|-----------------|
| S21 | Mon | Infrastructure as Code for Data Platforms    | Hands-on | 2 class   2 lab |
| S22 | Tue | Capstone Briefing & Pipeline Design Workshop | Design   | 2 class   1 lab |
| S23 | Wed | Capstone Build Clinic                        | Hands-on | 2 class   2 lab |
| S24 | Thu | Pipeline Debugging & Optimisation Drill      | Hands-on | 2 class   2 lab |
| S25 | Fri | Assessment & Design Defence                  | Mixed    | 2 class   1 lab |



## LABS THIS WEEK

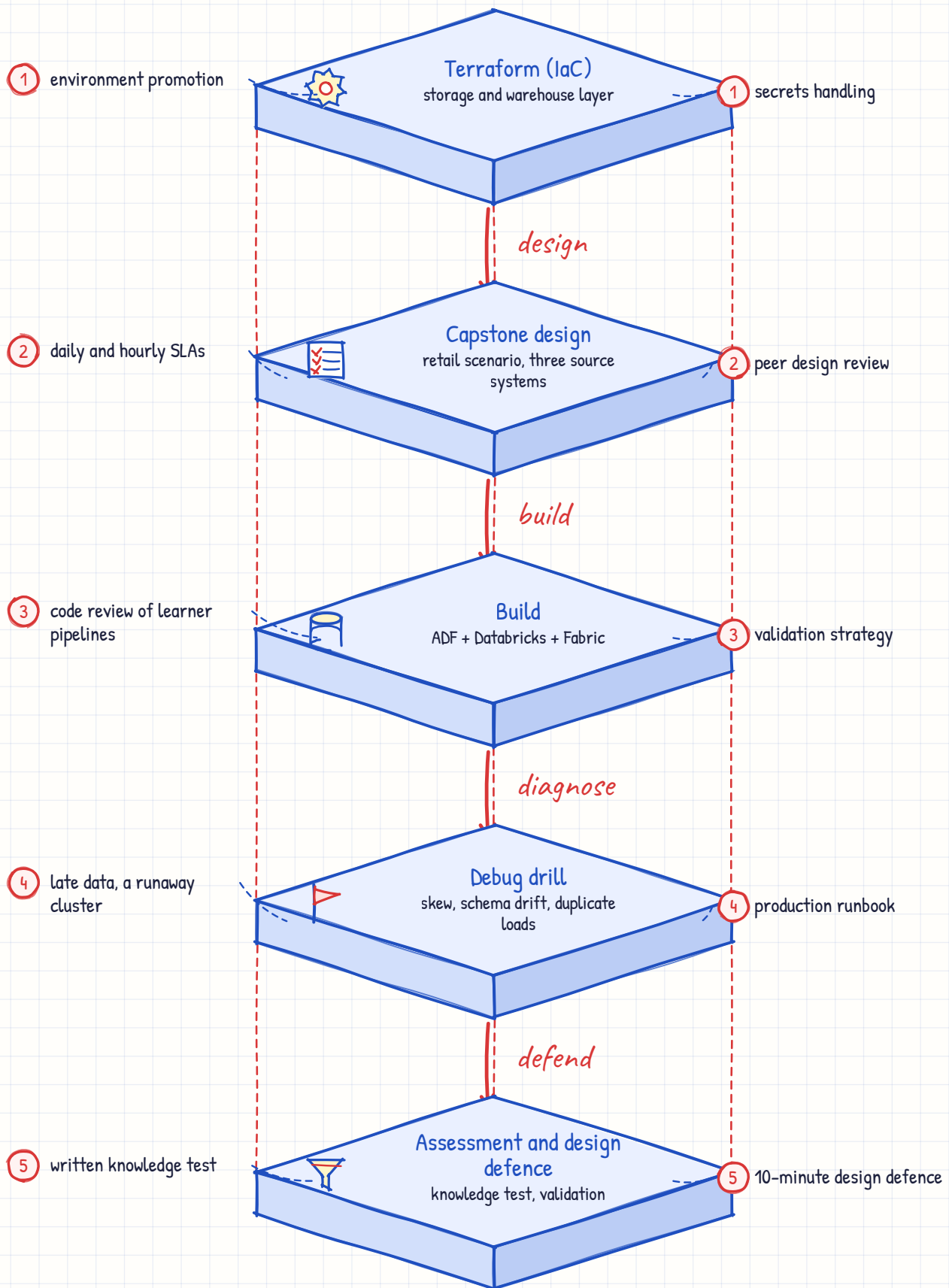
- S21 Codify the storage and warehouse layer in Terraform
- S22 CAP-1 DESIGN: architecture document, data model and SLA plan
- S23 CAP-1: guided build; bulk of construction happens in self-paced lab hours
- S24 Timed fault diagnosis; produce a production runbook
- S25 Assessment + 10-minute design defence per learner

## EVERY SESSION, 120 MINUTES



Class runs 18:00 to 20:00 IST, Monday to Friday. Labs are self-paced from written lab guides. Each concept is built in Azure first, then mapped to AWS and GCP in the same session.

*Next: the exploded architecture for Week 5.*

**WEEK 5 EXPLODED ARCHITECTURE**

Numbered circles = the session in this week (1 to 5). Find its section on the pages that follow.



# INFRASTRUCTURE AS CODE FOR DATA PLATFORMS

Focus: Azure-led

S21 / Monday / Infrastructure as Code for Data Platforms

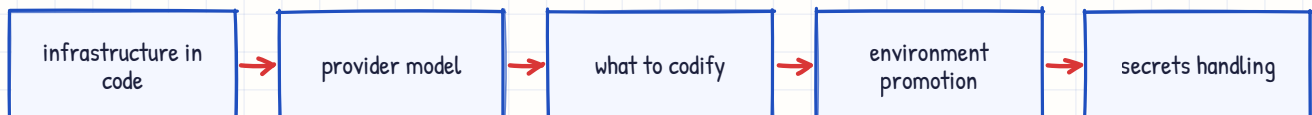
## TOPICS COVERED

- 1 Why data infrastructure belongs in code – Infrastructure written as code can be reviewed, repeated and rebuilt, instead of clicked together by hand.
- 2 Provider model – Terraform uses providers to manage each cloud's resources.
- 3 What to codify and what not to – Codify what must be repeatable. Not everything needs to be in code.
- 4 Environment promotion – The same definitions move from one environment to the next, with only the settings changing.
- 5 Secrets handling – Keep secrets out of code and out of source control. Reference them from a secure place.

## AZURE / AWS / GCP

| AZURE                                       | AWS                    | GCP                         |
|---------------------------------------------|------------------------|-----------------------------|
| Terraform / Bicep for ADLS, ADF, Databricks | Terraform for S3, Glue | Terraform for GCS, BigQuery |

## CONCEPT FLOW



### LAB

Codify the storage and warehouse layer in Terraform

Hands-on

2 class hrs | 2 lab hrs



### WHY THIS MATTERS

Code-defined infrastructure makes environments repeatable and reviewable.



### KEY TAKEAWAY

Codify what must be repeatable, promote across environments, and keep secrets out of code.



2

## CAPSTONE BRIEFING AND PIPELINE DESIGN WORKSHOP

Focus: Cross-cloud

S22 / Tuesday / Capstone Briefing &amp; Pipeline Design Workshop

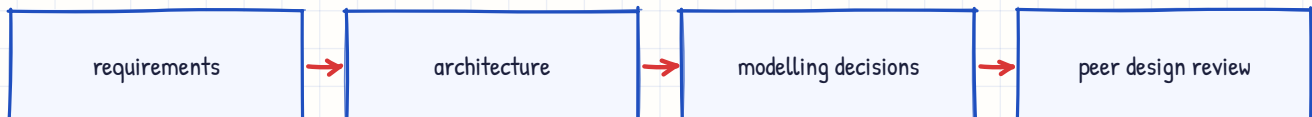
### TOPICS COVERED

- 1 Scenario: retail company, three source systems, daily and hourly SLAs – The capstone scenario is a retail company with three source systems and both daily and hourly SLAs.
- 2 Requirements to architecture – Turn the stated requirements into an architecture, step by step.
- 3 Modelling decisions – Choose and justify the data model, including the grain.
- 4 Peer design review – Learners review each other's designs and give feedback.

### AZURE / AWS / GCP

|                                    |                           |                           |
|------------------------------------|---------------------------|---------------------------|
| <b>AZURE</b><br>Azure design track | <b>AWS</b><br>AWS mapping | <b>GCP</b><br>GCP mapping |
|------------------------------------|---------------------------|---------------------------|

### CONCEPT FLOW



#### LAB

CAP-1 DESIGN: architecture document, data model and SLA plan

Design

2 class hrs | 1 lab hr



#### WHY THIS MATTERS

Design comes before build. A reviewed design makes the build faster and cleaner.



#### KEY TAKEAWAY

Move from requirements to architecture, justify the model, and take feedback in peer review.



3

**CAPSTONE BUILD CLINIC**

Focus: Azure-led

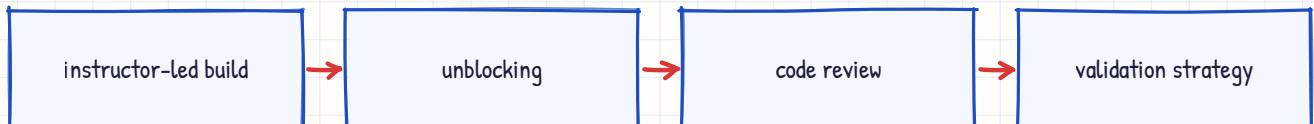
S23 / Wednesday / Capstone Build Clinic

**TOPICS COVERED**

- 1 **Instructor-led build session** - The instructor builds alongside you on the capstone pipeline.
- 2 **Unblocking** - Stuck points are cleared so construction can keep moving.
- 3 **Code review of learner pipelines** - Learner pipelines are reviewed for correctness and clarity.
- 4 **Validation strategy** - Plan how you will show that the pipeline works.

**AZURE / AWS / GCP**

|                                           |                                    |                                    |
|-------------------------------------------|------------------------------------|------------------------------------|
| <b>AZURE</b><br>ADF + Databricks + Fabric | <b>AWS</b><br>(optional AWS track) | <b>GCP</b><br>(optional GCP track) |
|-------------------------------------------|------------------------------------|------------------------------------|

**CONCEPT FLOW****LAB**

CAP-1: guided build; bulk of construction happens in self-paced lab hours

Hands-on

2 class hrs | 2 lab hrs

**WHY THIS MATTERS**

Most of the capstone is built in self-paced lab hours. The clinic keeps it moving and on track.

**KEY TAKEAWAY**

Build, ask for help early, take code review, and plan how you will validate.



4

## PIPELINE DEBUGGING AND OPTIMISATION DRILL

Focus: Azure-led

S24 / Thursday / Pipeline Debugging &amp; Optimisation Drill

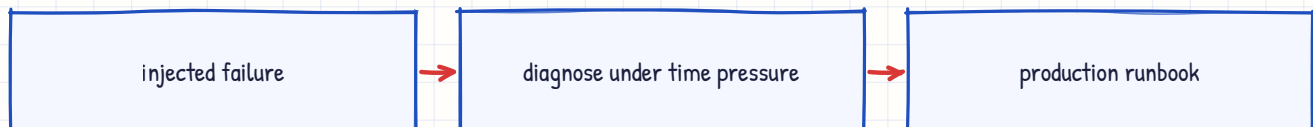
### TOPICS COVERED

- 1 Instructor injects failures: skew, schema drift, duplicate loads, late data, a runaway cluster – Each of these is a realistic production fault. The instructor creates them on purpose so you can practise finding the cause.
- 2 Learners diagnose under time pressure – You diagnose each fault against the clock, using the monitoring tools listed above.

### AZURE / AWS / GCP

| AZURE                    | AWS                      | GCP                           |
|--------------------------|--------------------------|-------------------------------|
| Spark UI, ADF monitoring | CloudWatch, Glue metrics | Dataproc / BigQuery job stats |

### CONCEPT FLOW



#### LAB

Timed fault diagnosis; produce a production runbook

Hands-on

2 class hrs | 2 lab hrs



#### WHY THIS MATTERS

Production pipelines fail. Practising diagnosis under pressure builds the skill you need on the day.



#### KEY TAKEAWAY

Read the monitoring tools, diagnose quickly, and write the runbook.



5

## ASSESSMENT AND DESIGN DEFENCE

*S25 / Friday / Assessment & Design Defence*

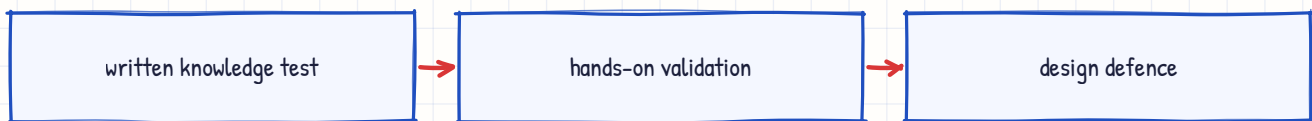
### TOPICS COVERED

- 1 Written knowledge test – A written test of the concepts covered across the level.
- 2 Hands-on validation of the capstone – Your capstone pipeline is validated hands-on.
- 3 Each learner defends their pipeline design and its trade-offs – You explain and defend your pipeline design and the trade-offs you made.

### HOW THE SESSION IS ASSESSED

| WRITTEN TEST           | VALIDATION                          | DEFENCE                                                       |
|------------------------|-------------------------------------|---------------------------------------------------------------|
| Written knowledge test | Hands-on validation of the capstone | Each learner defends their pipeline design and its trade-offs |

### CONCEPT FLOW



### THIS SESSION, 120 MINUTES



#### LAB

Assessment + 10-minute design defence per learner

**Mixed**

2 class hrs | 1 lab hr



#### WHY THIS MATTERS

The assessment checks both knowledge and the ability to build and defend a pipeline.



#### KEY TAKEAWAY

Show the pipeline works, and explain your design and its trade-offs.



## WEEK 5 RECAP

10 class hours / 8 lab hours

### I CAN EXPLAIN...

- ☐ I can explain why data infrastructure belongs in code.
- ☐ I can explain the retail capstone scenario, from requirements to architecture.
- ☐ I can explain how to validate a pipeline.
- ☐ I can explain skew, schema drift, duplicate loads, late data and a runaway cluster.
- ☐ I can explain my design and its trade-offs.

### LABS I HAVE COMPLETED

- ☐ S21 Codify the storage and warehouse layer in Terraform (Hands-on)
- ☐ S22 CAP-1 DESIGN: architecture document, data model and SLA plan (Design)
- ☐ S23 CAP-1: guided build; bulk of construction happens in self-paced lab hours (Hands-on)
- ☐ S24 Timed fault diagnosis; produce a production runbook (Hands-on)
- ☐ S25 Assessment + 10-minute design defence per learner (Mixed)

### THE WEEK IN FIVE STEPS





## CAPSTONE AND ASSESSMENT

S22 to S25 and CAP-1

### THE CAPSTONE SCENARIO

Retail company, three source systems, daily and hourly SLAs. Learners go from requirements to architecture, make modelling decisions, and take part in a peer design review.

#### CAP-1 DESIGN (S22)

CAP-1 DESIGN: architecture document, data model and SLA plan

**Design** 2 class hrs | 1 lab hr

#### CAP-1 BUILD (S23)

CAP-1: guided build; bulk of construction happens in self-paced lab hours

**Hands-on** 2 class hrs | 2 lab hrs



#### DEBUG DRILL (S24)

Instructor injects failures: skew, schema drift, duplicate loads, late data, a runaway cluster. Learners diagnose under time pressure. Timed fault diagnosis; produce a production runbook



#### ASSESSMENT (S25)

- Written knowledge test
- Hands-on validation of the capstone
- Each learner defends their pipeline design and its trade-offs

Assessment + 10-minute design defence per learner

### WHAT YOU PRODUCE

- ☐ Architecture document, data model and SLA plan (CAP-1 DESIGN)
- ☐ The capstone pipeline, validated hands-on (CAP-1)
- ☐ A production runbook (S24)
- ☐ A 10-minute design defence (S25)

### THE WEEK 5 PATH



### LEVEL 100 TOTAL

50 class hours | 45 lab hours. Outcome: Data Engineer.



# KEY TAKEAWAYS

Level 100: the whole program on one page

1

## WEEK 1: LANDSCAPE, SQL, PYTHON AND STORAGE

- Architecture follows requirements: OLTP vs OLAP, batch vs streaming, ELT, medallion.
- Window functions, CTEs and plan-reading make SQL production-grade.
- Spark is not always the answer; notebooks are not production.
- Columnar formats, partitioning and storage tiers shape speed and cost.

2

## WEEK 2: INGESTION

- ADF model: pipeline, activity, linked service, dataset, trigger.
- Load incrementally with a watermark; drive ingestion from metadata.
- Choose full, incremental or CDC; log-based CDC sees deletes.
- Design for replay, late data, schema drift and dead letters.

3

## WEEK 3: SPARK, DELTA LAKE AND MEDALLION

- The shuffle dominates runtime; read the Spark UI.
- Built-ins beat UDFs; handle nulls and bad records on purpose.
- Delta gives ACID, time travel and MERGE on object storage.
- OPTIMIZE, clustering and VACUUM keep tables fast; bronze, silver, gold each have a job.

4

## WEEK 4: MODELLING, SERVING AND QUALITY

- Declare the grain; choose fact types and SCD types deliberately.
- Serve gold through Synapse, Fabric, Redshift or BigQuery, by workload.
- Orchestrate with DAGs, retries, backfills, SLAs and alerts.
- Test and gate data: fail loud, or quarantine.

5

## WEEK 5: CAPSTONE WEEK

- Codify infrastructure; keep secrets out of code.
- Design first, then build, then debug under pressure.
- Defend your design and its trade-offs.

**OUTCOME: DATA ENGINEER**



## ABOUT EXAGURU

ExaGuru offers expert-designed technical training, taught through live sessions and hands-on labs.

Level 100 pairs live classes with self-paced labs, so learners build as they learn.

## READY TO BUILD YOUR DATA ENGINEERING FOUNDATION?

Talk to us about the Multi-Cloud Data Engineering program

[www.exaguru.com](http://www.exaguru.com)

+91 63940 49607 / +91 91611 11705

**KEEP LEARNING. KEEP BUILDING.**

