
COMP5339 Data Engineering

Independent Study Review Notes

Weeks 1–11 | Version 1.0 | Semester 1, 2026

Concept-focused revision notes with diagrams and worked exam-style questions, aligned to past-paper question patterns.

Independent study notice

These are independently authored review notes prepared from the author's study of COMP5339 (Data Engineering) at the University of Sydney in Semester 1, 2026. They are not official University of Sydney materials, are not endorsed by or affiliated with the University, and do not reproduce or redistribute university lecture slides, readings, assessments, answer keys, or other course resources. Explanations, examples, and diagrams were created by the author, with external sources identified where applicable. University/course names are used only to identify the subject context.

Scope. This product covers Weeks 1–11 only. Security/privacy material associated with Week 12 is intentionally excluded because it was not examinable in Semester 1, 2026. Refer to separately maintained material for that topic; it is outside this product.

How to use these notes

- **Blue boxes** hold formal definitions worth memorising.
- **Green boxes** explain *why a concept matters* in practice.
- **Gold “Q” boxes** give an exam-style question, a model answer, and a marker's note at the end of each week — use them to self-test.
- **Diagrams** accompany every architecture and data-flow concept; being able to redraw them is a fast way to revise.
- The **capstone scenario** near the end shows how to assemble many weeks' ideas into one integrated design answer.

0 Contents

1	Week 1: Introduction to Data Engineering	4
1.1	Definition and Role	4
1.1.1	Type A vs Type B Engineers	4
1.2	Data Science Hierarchy of Needs	4
1.3	Data Engineering Lifecycle	4
1.4	Cross-Cutting Undercurrents	5
2	Week 2: Data Acquisition and Data Cleaning	5
2.1	Data Transfer Patterns: Push, Pull, Poll	5
2.2	Bounded vs Unbounded Data	6
2.3	ETL vs ELT	6
2.4	Batch vs Streaming Ingestion	6
2.5	Data Quality Fundamentals	7
2.6	Four Data Cleaning Patterns	7
3	Week 3: Databases, Data Warehouses, and Data Lakes	7
3.1	OLTP vs OLAP	7
3.2	Relational Model and SQL	8
3.3	OLAP Cube Operations	8
3.4	Star Schema	8
3.5	Data Warehouse	9
3.6	Data Lake, Warehouse, and Lakehouse	9
3.7	Database as Source and Sink	10
4	Week 4: Web Scraping and Web APIs	10
4.1	Web Scraping Workflow	10
4.2	HTML, the DOM, and Page Structure	10
4.2.1	Four Content Extraction Methods	11
4.3	HTTP and REST APIs	11
4.4	REST, SOAP, and Safe API Access	11
4.5	Web Data Categories and Tools	12
5	Week 5: Semistructured Data and NoSQL Databases	12
5.1	Semistructured Data	12
5.2	XML: Well-Formed vs Valid	12
5.3	XPath Selection	13
5.4	JSON vs XML	13
5.5	Schema-First vs Schema-Late	13
5.6	NoSQL Flavours	13
5.7	MongoDB	14
5.8	Neo4j and Cypher	14
5.9	PostgreSQL JSONB vs JSON	14
6	Week 6: Spatial Data Engineering	15
6.1	Spatial vs Geospatial; SDBMS vs GIS	15
6.2	Geometry Types and Models	15
6.3	Coordinate Systems and Projections	15
6.4	Spatial Operations	15
6.5	9-Intersection Model	15
6.6	PostGIS	16
6.7	GeoPandas	16
7	Week 7: Temporal Data Engineering	16

7.1	Why Temporal Data Matters	16
7.2	Temporal Data Types	17
7.3	Three Kinds of Time	17
7.4	Four Table Types	17
7.5	Representing Open-Ended Intervals	17
7.6	Three Temporal Statement Types	18
7.7	SQL:2011 and PostgreSQL Support	18
7.8	Time Series Data and Storage	18
8	Week 8: Unstructured Data and Feature Extraction	18
8.1	Unstructured Data Pipeline	19
8.2	Text Preprocessing	19
8.3	Text Feature Extraction	19
8.4	Image Feature Extraction	19
8.5	ML Task Types and Feature Engineering	19
9	Week 9: Stream Data Processing	20
9.1	Why Stream Processing	20
9.2	DBMS vs DSMS	20
9.3	Stream Processing Architecture	20
9.4	MQTT	21
9.5	Apache Kafka	21
9.6	Stateless vs Stateful Processing	21
9.7	Window Types	21
9.8	Three Notions of Time and Watermarks	22
9.9	Stream Joins and Engines	22
10	Week 10: Scalable Data Engineering	22
10.1	Scale-Up vs Scale-Out	23
10.2	Column Store vs Row Store	23
10.3	Wide-Column NoSQL and Apache Parquet	23
10.4	Data Partitioning and Sharding	24
10.5	Data Replication	24
10.6	CAP Theorem	24
10.7	HDFS	25
10.8	MapReduce	25
10.9	Apache Spark and Apache Flink	25
11	Week 11: DataOps, Data Architectures, and ML	26
11.1	Data Architecture	26
11.2	Architecture Patterns	26
11.3	DataOps	27
11.4	Data Observability: Five Pillars	27
11.5	CI/CD in Data Engineering	27
11.6	Scaling ML: In-Database vs Feature Stores	27
11.7	Feature Stores	27
12	Exam Question Playbook	28
12.1	Capstone Scenario (authored practice scenario)	29
13	Quick Reference Comparison Tables	30
13.1	ETL vs ELT	30
13.2	OLTP vs OLAP	30
13.3	Data Lake vs Data Warehouse	31
13.4	NoSQL Flavours	31

13.5 Window Types in Stream Processing	31
13.6 Spark Streaming vs Apache Flink	31
13.7 Replication Strategies	32
13.8 CAP Theorem Summary	32
13.9 Architecture Patterns Summary	32

1 Week 1: Introduction to Data Engineering

1.1 Definition and Role

Data engineering is the discipline of **designing, implementing, and maintaining systems that produce high-quality data** for downstream consumers. Unlike data science, which focuses on extracting insight, data engineering focuses on the infrastructure and pipelines that make reliable data available in the first place. A data engineer sits at the intersection of software engineering, systems design, and domain knowledge.

Data engineering

The development, implementation, and maintenance of systems and processes that take in raw data and produce high-quality, consistent information that supports downstream use cases such as analysis and machine learning.

Type A vs Type B Engineers

Organisations distinguish two archetypes. **Type A (Abstraction)** engineers use managed services and frameworks, avoiding low-level infrastructure management; they value speed of delivery. **Type B (Build)** engineers design and build custom data systems from scratch when off-the-shelf tools are insufficient. Most real roles blend both, leaning one way depending on the organisation's maturity and needs.

1.2 Data Science Hierarchy of Needs

Effective machine learning rests on a foundation of solid data work. The *hierarchy of needs* makes the point that you cannot do reliable AI/ML without first solving collection, movement, storage, and transformation.

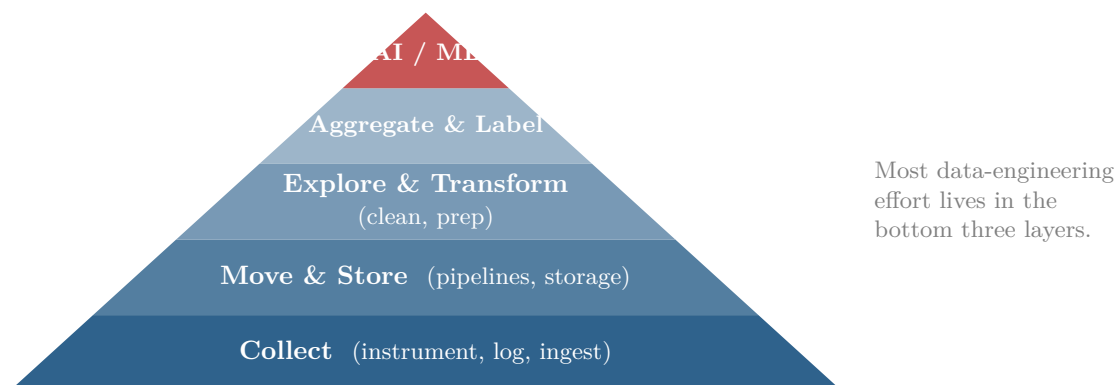


Figure 1: Data science hierarchy of needs: ML sits on top of foundational data work.

1.3 Data Engineering Lifecycle

The lifecycle describes the journey of data through a system: it is generated by source systems, ingested into the platform, transformed to be fit for purpose, and served to consumers. **Storage** is not a sequential stage but an underpinning layer present throughout, and six **undercurrents** cut across every stage.