**1. SQL & Databases**

**Q1. What is the difference between INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN?**  
**:**

* INNER JOIN → Only matching rows.
* LEFT JOIN → All rows from left + matching right.
* RIGHT JOIN → All rows from right + matching left.
* FULL OUTER JOIN → All rows from both, matching where possible.

**Q2. How do you optimize a SQL query with large datasets?**  
**:**  
Use indexes, avoid SELECT \*, proper joins, partitioning, denormalization if needed, analyze execution plan.

**Q3. What are indexes in databases? Difference between clustered and non-clustered?**  
**:**

* Indexes speed up lookups.
* Clustered → data is stored in order of index (only one per table).
* Non-clustered → separate structure pointing to data (many allowed).

**Q4. Explain ACID properties.**  
**:**

* Atomicity → all or nothing.
* Consistency → valid state after transaction.
* Isolation → concurrent transactions don’t conflict.
* Durability → changes persist after commit.

**Q5. Difference between OLTP and OLAP systems?**  
**:**

* OLTP → Transactional (banking, e-commerce).
* OLAP → Analytical (data warehouses, BI reports).

**Q6. Write a query to find the second highest salary from a table.**  
SELECT MAX(salary)

FROM employees

WHERE salary < (SELECT MAX(salary) FROM employees);

Q7. What are window functions in SQL? Give an example.

: Functions that operate across sets of rows without collapsing them.

SELECT name, salary, RANK() OVER (ORDER BY salary DESC) AS rank

FROM employees;

Q8. Difference between partitioning and sharding?

 Partitioning → splitting data within one DB instance.

 Sharding → splitting data across multiple DB instances.

**Q9. What are star and snowflake schemas?**  
**:**

* Star schema → fact table + denormalized dimensions.
* Snowflake schema → fact table + normalized dimensions.

**Q10. What are isolation levels in SQL?**  
**:**

* Read Uncommitted → dirty reads possible.
* Read Committed → no dirty reads.
* Repeatable Read → no dirty or non-repeatable reads.
* Serializable → strictest, prevents all anomalies.

**Q11. Difference between SQL and NoSQL databases?**  
**:**

* SQL → structured, relational, ACID.
* NoSQL → flexible schema, distributed, scalable (key-value, doc, graph, columnar).

**Q12. When would you use columnar vs row storage?**  
**:**

* Columnar → analytics, aggregations (faster reads).
* Row-based → transactional workloads (fast inserts/updates).

**2. Data Modeling & Warehousing**

**Q13. What are fact tables and dimension tables?**  
**:**

* Fact = measurable metrics (sales, revenue).
* Dimension = descriptive attributes (date, product, customer).

**Q14. Explain Slowly Changing Dimensions (SCDs).**  
**:**

* Type 1 → overwrite old value.
* Type 2 → add new row, keep history.
* Type 3 → add new column for limited history.

**Q15. What is denormalization and why use it in warehouses?**  
**:**  
Combining tables to reduce joins → improves query performance at the cost of redundancy.

**Q16. Batch vs real-time data processing?**  
**:**

* Batch → large chunks, periodic (e.g., daily ETL).
* Real-time → continuous (e.g., streaming logs).

**Q17. What is CAP theorem?**  
**:**  
In distributed systems, can only guarantee 2 out of 3:

* Consistency, Availability, Partition tolerance.

**Q18. Difference between data lake, data warehouse, and data mart?**

* Data Lake → raw, unstructured.
* Data Warehouse → structured, cleaned for analytics.
* Data Mart → subject-specific subset of a warehouse.

**Q19. ETL vs ELT?**  
**:**

* ETL → Extract → Transform → Load (before DB).
* ELT → Extract → Load → Transform (inside DB). Preferred in cloud.

**Q20. How to handle schema evolution in pipelines?**  
**:**  
Schema-on-read, versioning, Avro/Parquet formats, data contracts, optional fields.

**3. Big Data & Distributed Systems**

**Q21. What is HDFS?**  
**:**  
Hadoop Distributed File System → stores large files across cluster with replication for fault tolerance.

**Q22. How does MapReduce work?**  
**:**

* Map → splits input, processes in parallel.
* Reduce → aggregates results.

**Q23. Spark RDDs, DataFrames, Datasets?**  
**:**

* RDD → low-level, unstructured.
* DataFrame → structured, optimized.
* Dataset → type-safe, combines both.

**Q24. Spark vs Hadoop?**  
**:**

* Spark → in-memory, faster.
* Hadoop → disk-based, slower.

**Q25. Transformations vs Actions in Spark?**  
**:**

* Transformations → lazy, define pipeline (map, filter).
* Actions → trigger execution (collect, count).

**Q26. How to optimize Spark jobs?**  
**:**  
Partitioning, caching, broadcast joins, avoiding shuffles, tuning executors.

**Q27. What is Kafka?**  
**:**  
Distributed messaging system for high-throughput event streaming.

**Q28. Compare Kafka Streams, Flink, Spark Streaming.**  
**:**

* Kafka Streams → lightweight, built-in with Kafka.
* Flink → low-latency, event-time processing.
* Spark Streaming → micro-batches.

**4. Cloud & Data Platforms**

**Q29. Compare AWS Redshift, Snowflake, BigQuery.**  
**:**

* Redshift → AWS-managed, cluster-based.
* Snowflake → multi-cloud, separation of storage/compute.
* BigQuery → serverless, pay-per-query.

**Q30. What is serverless data processing?**  
**:**  
Run code without managing servers (e.g., AWS Lambda, GCP Cloud Functions).

**Q31. How does AWS Glue work?**  
**:**  
Serverless ETL → crawls schema, transforms, loads data.

**Q32. S3 vs Redshift for data?**  
**:**

* S3 → raw data lake.
* Redshift → analytics-focused data warehouse.

**Q33. What is Airflow?**  
**:**  
Orchestration tool → manages workflows using Directed Acyclic Graphs (DAGs).

**Q34. How to design a data ingestion pipeline in cloud?**  
**:**  
Source → Ingestion (Kafka, Kinesis) → Processing (Spark, Glue) → Storage (S3, Snowflake) → BI tools.

**Q35. What is Delta Lake?**  
**:**  
Storage layer that adds ACID transactions and schema enforcement to data lakes.

**Q36. What are IAM roles and policies?**  
: Identity & Access Management → define who can access what resources in cloud securely.

**5. Programming & Data Structures**

**37. Python: remove duplicates from a list.**

my\_list = list(set(my\_list))

**Q38. Difference between shallow and deep copy in Python?**

* Shallow → references nested objects.
* Deep → copies nested objects fully.

Q39. How to handle exceptions in Python?

try:

risky\_code()

except Exception as e:

print(e)

finally:

cleanup()  
  
Q40. Reverse a linked list in Python.  
prev, curr = None, head

while curr:

nxt = curr.next

curr.next = prev

prev = curr

curr = nxt

return prev  
  
**Q41. Mutable vs Immutable types in Python?**  
**:**

* Mutable → list, dict, set.
* Immutable → str, tuple, int.

**Q42. Process a large CSV (100GB) efficiently?**  
**:**  
Use pandas.read\_csv(..., chunksize=), dask, or PySpark.

**6. System Design for Data Engineering**

**Q43. Design a streaming pipeline.**  
**:**  
Producers → Kafka → Stream Processor (Spark/Flink) → Storage (S3/Redshift) → BI/Analytics.

**Q44. Design a warehouse schema for e-commerce.**  
**:**

* Fact tables → sales, orders.
* Dimensions → customer, product, date, store.

**Q45. What are bottlenecks in ETL pipelines?**  
**:**  
Network I/O, skewed partitions, poor indexing. Fix via scaling, caching, optimization.

**Q46. How do you ensure data quality in pipelines?**  
**:**  
Validation checks, deduplication, logging, anomaly detection.

**Q47. Design a real-time analytics system (e.g., stock prices).**  
**:**  
Producers → Kafka → Flink/Spark → NoSQL DB (Cassandra) → Dashboard (Grafana).

**Q48. How to design for scalability in data systems?**  
**:**  
Partitioning, distributed storage, horizontal scaling, caching layers.

**7. Governance & Security**

**Q49. How to handle PII/PHI in pipelines?**  
**:**  
Encryption, masking, anonymization, RBAC, GDPR/HIPAA compliance.

**Q50. How to track data lineage in pipelines?**  
**:**  
Metadata catalogs & lineage tools (Apache Atlas, Collibra, OpenLineage).