Research Assignment

Section A

1. ~~Let~~ Relational Databases RDBMS, NoSQL Databases, Object database, Graph databases.
2. RDMS stores and manages data in relations (tables), supports SQL for queries, enforces schema, constraints, transactions and relationships between tables.
3. ~~Primary key a column in one table that references~~
3. Primary key a column or set of columns that uniquely identifies each row in a table (no NULLS)
- Foreign key: a column in one table that references a primary key in another table to establish a relationship.
4. It is important as it process of organizing data to reduce redundancy and improve integrity
- It improves consistency, simplifies updates and reduces storage waste and anomalies.
5. The structure of a ~~db~~ database (tables, columns, types, constraints, relationships and indexes
6. Structured - data that is highly organized, typically stored in relational databases with predefined schemas.
- Unstructured data - data without predefined format or organization, making it challenging to process with traditional tools.
- Semi-structured - data that has some organizational properties but lacks a rigid schema, using tags or markers for structure
7. Fact table - stores measureable, numeric business events and ~~liking~~ linking to dimensions.
- Dimension table descriptive attributes used to

to slice facts.

8. A representation of data structures and my relationships. Ensuring consistent data organization, supports requirements and guides implementation and queries.

9. Database - OLTP - focused, supports transations and application data (operational)

- Data Warehouse - OLAP - focused, curated and structured historical data optimized for reporting/analytics.

- Data lake stores vast raw data in native formats for later processing and analytics.

10. Data marts is a smaller, focused subset of a data warehouse designed for a specific business unit or purpose

- It differs from a data warehouse ~~es~~ is a ~~eee~~ enterprise-wide and larger and a data mart is a smaller/narrower in scope.

~~B~~.

Section B

11. A query language retrieves/manipulates data from databases.

- SQL is a standard, declaratire, widely adopted, powerful for relational data, supports join, aggregating, transactions, and many tools intergrate with it

12. Indexes are data structures that speed up lookup and sorting on columns.

- They reduce full-table scans at the cost of extra ~~stroag~~ storage and slower writes/updates.

13. A transactions is a sequence of databases operations treated as one unit.

- the ~~part~~ properties are Atomicity
- Atomicity - ensures that transaction is treated as a sig single, indivisible unit. example bank transfer must be debit one account.
- Consistency ensures that database remains in a consistent, valid state before and after the transaction example balance can't go negative if business rule prohibits overdrafts.
- Isolation transactions execute independently of each other ex
- Durability - committed changes persist even after crashes.

14. The engine handles storage, indexing, caching, query planning and execution.
- Engine design strongly affects throughput, latency and scalability.

15. View. a virtual table defined by a query
- Stored procedure - precompiled routine executed on the DB server
- Trigger - automated code that runs on specific events to enforce rules or audit.

16. ETL (Extract -> Transform -> Load) transform data before loading into the target.
- ELT (Extract, Load, Transform) load raw data first and transform there.

17. Batch - processes data in chunks at intervals.
- Stream - processes continuous data in real-time

18. A joins combines row from two + tables based on related columns.

① INNER JOIN - rows with matching keys in both tables

② LEFT JOIN - all rows from left table, matched

rows from right or NULLS
④ RIGHT JOIN - all rows from right table, matched rows from left
⑤ FULL OUTER JOIN - rows matched or unmatched from both sides (NULLS where missing)
⑤ ~~OUTER~~ JOIN - Cartesian product (all combinations) (CROSS
⑥ SELF JOIN - table joined to itself
19. Ensures relationships between tables remain consistent
- Prevents ~~apa~~ orphaned records and maintains data correctness.
20. Redundancy increases storage use can cause inconsistent updates, and may slow writes. Controlled redundancy can be used to improve read performance at a trade-off.

Section C

21. Cloud - managed services, pay-as-you-go, easier scaling and global reach.
- On-premise - full control over hardware/ security/configuration, higher setup/maintenance cost/ slower to scale.
22. Policies, processes roles and controls ensuring data quality, security, privacy and compliant use
- Ensures trusted data, accountability, regulatory compliance and consistent data practices
23. Accuracy and consistency of data over its lifecycle. Maintained via constraints, validation rules/ transactions, audits and access controls.

24. Data quality measures correctness, completeness, timeliness, consistency and relevance. Poor quality leads to incorrect insights, bad decisions and lost trust.

25. Collects, clean, explores, analyzes data, creates reports/ dashboards, derives insights to support decisions, communicates findings and works with stakeholders to define metrics,

26. Install/configure DBMS, manage backups and recovery, tune performance, manage security and access, apply patches and ensure availability and integrity of databases.

27. Define requirements and sources → extract data → transform/clean/enrich → validate/quality checks → load into target → orchestrate/schedule → monitor & alert → document.

28. Scaling, performance tuning, data consistency, backups and recovery time, cost management, schema evolution, security and ensuring low-latency access for many users.

29. MySQL - web apps, OLTP
- PostgreSQL - advanced SQL features, geospatial, analytics and OLTP
- Oracle DB - enterprise OLTP, heavy transactional systems, advanced features
- SQL Server - enterprise apps, BI intergration.
- MongoDB - flexible document data, rapid development

30. CSV - simple, human-readable, widely supported
- JSON - semi-structured, good for documents and nested data
- Parquet - columnar, compressed, efficient for analytics

and big data.
- Avro - row-based, schema evolution friendly, used in streaming
- ORC - columnar, optimized for Hadoop ecosystems.