- Research and describe technologies that provide schema-on-read and schema-on-write. Provide examples of each.

Schema-on-Write

When the data structure is defined before the data is written. The data is validated against the schema to ensure it conforms to the predefined structure.

1. Relational Databases (SQL, MySQL, PostgreSQL)
2. Data Warehouses (Snowflake, BigQuery)

Schema-on-Read

When the data structure is applied only when data is read or queried. Allows for semi-structured data.

1. Hadoop (Spark, Cloudera)
2. NoSQL (MongoDB, Cassandra)

- Discuss the advantages of schema-on-read and how this aligns with the characteristics and common uses of big data.

Schema-on-read offers a lot of advantages, the primary benefit being flexibility and agility. The lack of schema validation on write allows for quick data ingestion, which is perfect for high throughput systems like data warehouses and lakes. The schema on read structure allows the data structure to be flexible. The storage is also cost-effective in that the data is structured without schema, or raw. This provides strong support for nested and complex data structures.

- Discuss, as appropriate, how increased database parallelism occurs with the schema-on-read technologies versus traditional schema-on-write.

Schema-on-write is a strong solution for strict data integrity and validation. The structured schema defined before write allows for indexes, partitions, and optimizations to make querying efficient. The structured data also makes the data predictable, which makes writing queries easier for developers and data engineers. Schema-on-write is best suited for environments that require fast and reliable transactions with strict adherence to data rules. This is good for traditional systems like banking, e-commerce, and retail systems.

- Discuss other techniques that may complement schema-on-write data preparation, such as key/value pair programming offered via MapReduce

ETL Pipelines

Systems like data warehouses and data lakes receive data from many locations. ETL pipelines Extract that data from these sources, Transform the data into a format ingestible by the warehouse, and Load the data into the warehouse. Hence the acronym "ETL".

Views and Indexing

Views are precomputed, potentially complex queries to make commonly accessed data requests more performant. Indexes can be created on columns that are often used in query sorting/filtering operations. This will make those operations more performant.

Data partitioning and sharding

Partitioning is the division of large tables into smaller more manageable pieces, allowing databases to only retrieve the relevant pieces when executing queries. Sharding is the splitting of databases across multiple servers so that each server hosts a portion of the data, supporting horizontal scaling.

Upsolver. (2020, November 25). *Data Management: Schema-on-Write Vs. Schema-on-Read*. Upsolver. https://www.upsolver.com/blog/manage-your-data-schema-on-read-vs-schema-on-write