**Multi-stage pipeline Question**

**4. Devise a database schema to hold the data and apply it to a MySQL or SQLite database?**

CREATE TABLE BaseJoinedTable (

\_id INT PRIMARY KEY,

OrderID BIGINT UNIQUE,

ExecutedVolume BIGINT

);

CREATE TABLE TransactionDetails(

OrderID BIGINT PRIMARY KEY

SettlementCurrency VARCHAR(255),

MarketName VARCHAR(255),

Portfolio VARCHAR(255),

TradeID INT,

\_full\_text VARCHAR(255)

);

CREATE TABLE BmrsWindForecastPair(

OrderID BIGINT PRIMARY KEY,

recordType VARCHAR(255),

startTimeOfHalfHrPeriod DATE,

settlementPeriod INT,

initialForecastPublishingPeriodCommencingTime TIMESTAMP,

initialForecastSpnGeneration FLOAT,

latestForecastPublishingPeriodCommencingTime TIMESTAMP,

latestForecastSpnGeneration FLOAT,

outTurnPublishingPeriodCommencingTime TIMESTAMP,

fuelTypeGeneration FLOAT,

activeFlag INT,

EFA FLOAT

);

CREATE TABLE CircuitBreaker(

OrderID BIGINT PRIMARY KEY,

1P INT,

1V INT,

2V INT,

2P INT,

3P INT,

3V INT,

4V INT,

4P FLOAT

);

CREATE TABLE DeliveryDetails(

OrderID BIGINT PRIMARY KEY,

DeliveryEnd TIMESTAMP,

OrderEntryUser VARCHAR(255),

OrderPeriodID VARCHAR(255),

BiddingLevelName VARCHAR(255),

MemberName VARCHAR(255),

DeliveryStart TIMESTAMP,

EFA VARCHAR(255)

);

**A diagram of a computer

Description automatically generated**

Figure 1 – Entity Relationship Diagram for the Power Purchase Order placement.

**Architecture Questions**

1. **How would you do this in a production cloud environment?**

It is more important to understand the exact requirement and business case to work backwards towards the best architecture solution. These would the standard procedure that I usually follow while proposing a cloud production solution,

* Usability and maintainability – How easy to deploy and operate, how easy to manage the solution and what is the skillset required etc.
* Reliability – how can the team manage in case of disruption, is there SLAs in place to handle such incidences, how would is the production hand over get taken care, what are the risks and who would provide the required sign offs, what testing needed to be done to move to production etc.
* Performance – How to scale the architecture, what are all the current resources that are underutilised, how to improve the utilisation, which design pattern and software architecture would suit the best for this use case.
* Cost – What would the estimated cost and whether the cost is within the project scope, how to perform cost optimisation without affecting the performance etc.
* Security – How compliant is the solution with regulations like GDPR, would check for proper encryption, assigning appropriate permissions to users, enforcing higher protection to the resources to prevent cyber-attacks like DDOS, Ransomware by identifying the vulnerability and preparing for zero day exploit etc.

Also, I would create three different environment Development, UAT and Production to ensure proper stability. This would be orchestrated using CI / CD pipeline. I would finally inspect for point of failures and try to minimise the impact.

1. **How would your solution change if the data inputs were to be received in the form of streamed messages?**

For our assumption let us take the current solution is batch job that ingests data in a certain interval and now the business requirement is to access the data in stream or real time or near-real time. I would propose usage of stream processing engines like Apache Kafka, Apache Flink, Apache Spark Streaming, Apache Flume, Google Cloud Pub/Sub Logstash or AWS Kinesis. This would next lead to the storage of the data, usage of database or storage solution that supports real-time data ingestion and querying, such as Apache Cassandra, MongoDB, Elasticsearch, or Amazon DynamoDB can be explored. The reason for choosing these databases is due to the CAP theorem, which states that the database must be available all the time and has to be partitioned to handle such high volume and velocity.

1. **How would your solution change if the data was 100x the scale?**

The general recommendation would explore the horizontal and vertical scaling option. The horizontal scaling would be less expensive and distributed system is theoretically capable of infinite processing power. As I explained above, first would exploring the right architecture design, since this is a data architecture, I would recommend increasing the number of nodes with better compute power. Also, I would explore the possibility of using GPUs to potentially improve the performance of certain types of workloads, that involve highly parallelizable tasks such as machine learning, data analytics. Not all databases or distributed processing engines support GPUs and I would also keep in mind about the cost scope.

Other well-known strategy is the usage of microservices architecture, containerization, and serverless computing. I would explore the Kubernetes based orchestration to significantly improve the performance. Data partitioning technique such as sharding is another method to achieve the scaling. Data compression and serialization can also be used while storing and transferring the data. Network bottleneck is a major issue in the distributed system and serializing, compressing the data before transmission is advisable.

1. **What kind of architecture and approach would you use to serve this data for dashboards and reporting, used by multiple teams with differing requirements?**

I would store all the data in a centralized repository such as a data warehouse or data lake or data lakehouse to serve multiple teams quickly at scale. Here there would be a major debate on the Inmon and Kimbal method of designing the system. Since the requirement is that the different team has various requirement, it would be better to implement Kimball’s approach by providing the teams with flexibility to create dashboards using BI tools like Power BI, Looker, Tableau based on the business requirements to enable them to deliver at a faster pace. There has been more emphasis on the self-serve analytics to avoid data replication and again Kimbal method would help to achieve it. The Data marts can be derived from the centralized data warehouse and tailored to meet the specific requirements of each team. ETL has to managed in such a way to support the dashboard with consistent format suitable for quick reporting and analysis.