CHALLENGES - [BigQuery, Airflow/Composer, Data Fusion, Pub/Sub, Dataproc, Dataflow]

CHALLENGES ---> [Environment, Production deployment, Development, Data, Performance]

BIGQUERY

BIGQUERY [Environment] :

-> Ensuring consistency in schemas, data models
-> Access controls across environments- crucial for development and testing
-> Limited control over resources
-> Network latency and bandwidth limitations

VAARAHI CLOUD TECHNOLOGIES
HYDERABAD
★

BIGQUERY [Production Deployment] :

-> Data Inconsistencies, Inaccuracies, Incompleteness
-> Inefficient query execution plans, suboptimal indexing strategies, resource contention
-> Resource over-provisioning, Under-utilization.
-> Version control issues, deployment errors, unintended consequence of changes
-> Setting up alerts for query failures, unexpected costs, data quality issues

BIGQUERY [Development] :

-> Designing effective data models
-> Data types consideration, field names, schema nesting structures, schema evolution, migration
-> Ensuring data consistency, integrity, timeliness during ingestion process
-> Debugging errors& performance issues
-> Conflicting challenges & maintaining version history
-> data set compatibility -> file formats like CSV, JSON, Avro, Parquet, ORC(optimized Row Columnar)

BIGQUERY [Data] :

-> Inconsistent(or)Incomplete data, Data errors, Duplicates, Outliers
-> Schema Mapping, Data transformation, Data synchronization
-> Data ownership, Data privacy, Security & Compliance requirements
-> Excessive data storage costs, Data retention policies
-> Inconsistent Formatting (dates, timestamp, units)

BIGQUERY [Storage & Processing] :

-> As data grows storage & processing time increases
-> Determining optimal partitioning & clustering keys, especially for evolving data patterns
-> Complex query logic, inefficient join aggregation strategies
-> Choosing slots based on our workload

AIRFLOW/COMPOSER

AIRFLOW [Environment] :

-> Differences in environment setup
-> Software versions & Infrastructure configuration
-> Dependency management
-> Configuration inconsistencies

AIRFLOW [Production Deployment] :

-> Doesn't offer built-in versioning for DAG'S
-> Centralized log storage- Crucial to identify errors
-> Setting up reduction components, implementing failover mechanisms
-> Lack of Automation tools; difference between development & production environment

AIRFLOW [Development] :

-> Often involve multiple tasks with dependencies, schedules & retries
-> Managing dependencies between tasks within a DAG
-> Configuring task parameters, retries, timeouts & execution dates
-> Promoting code reusability & modularity in airflow dag's

AIRFLOW [Data] :

-> Dealing with different data formats, sources & Ingestion frequencies
-> Implementing efficient data process algorithms, handling data skew & distribution
-> Inefficient join algorithms, data shuffling

AIRFLOW [Storage & Processing] :

-> Managing data storage configurations, ensuring data consistency, handling data retention & archival policies
-> Optimizing task execution execution times, minimizing parallelism & concurrenty
-> Optimizing resource utilization, inefficient resource allocation, under-utilization of resources

DATA FUSION

DATA FUSION [Environment] :

-> Authorization & Authentication requirements, Network connectivity issues & Access restrictions
-> handling missing values, outliers, duplicates, data normalization

DATA FUSION [Production Deployment] :

-> Data synchronization issues, data conflicts, data-quality problems
-> Managing data-streams, ensuring low-latency processing, integrity across real-time & batch processing pipelines
-> Tracking changes, resolving conflicts, reproducibility in production deployment

DATA FUSION [Development] :

-> Schema mis-matches, inconsistencies, varying levels of data-quality
-> Selecting informative features, handling high-dimesional data
-> Developers need to design workflows- Computationally efficient, parallelizable

DATA FUSION [Data] :

-> Aligning data-semantics, resolving schema conflicts
-> Managing data transformations & mapping between different sources
-> Handling data latency, synchronization delays, data-freshness requirements
-> Managing data scalability, low-latency processing, processing bottlenecks

DATA FUSION [Storage & Processing] :

-> Dealing with data formats, structures & storage systems
-> Resource limitations, CPU, Memory & Storage in data processing
-> Determining optimal partitioning & shading strategies
-> Appropriate compression & encoding techniques

PUB/SUB

PUB/SUB [Environment] :

-> Designing scalable architectures
-> Optimizing message routing & delivery
-> Ensuring proper logging & tracing mechanism
-> Handling out-of-order messages,duplicate messages & message-loss (or) corruption

PUB/SUB [Production Deployment] :

-> Robust logging & tracing mechanism- To pinpoint where messages are being dropped(or)errors occur within Pub/sub pipeline
-> Pub/Sub system typically dont gurantee message delivery order
-> Implementing encryption, access control & authentication mechanism to safeguard message privacy & integrity

PUB/SUB [Development] :

-> Ensuring message ordering while achievinh high-throughput
-> Designing systems to handle network partitions,retries handle
-> Handle failures without introducing duplicate messages accidentally
-> Efficiently serializing & de-serializing messages
-> Implementing flow-control mechanisms,handling busy traffic

VAARAHI CLOUD TECHNOLOGIES
HYDERABAD
★

PUB/SUB [Dataside] :

-> Schema evolution,data transformation,compatability issues between data sources & subscribers
-> Network latency,processing delays & system over-head
-> Identifying obselete(stale) data
-> Implementing data archival & deletion process effectively

PUB/SUB [Storage & Processing] :

-> Implementing reliable message storage mechanism,handling message replication
-> Determining optimal retention periods,handling message expiration
-> Designing efficient indexing structures, optimizing query performance

DATA PROC

DATA PROC [Environment] :

-> Selecting the appropriate compute, storage & networking resources
-> Configuring security settings & access controls
-> Under-provsioning(or) over-provisioning resources,unnecessary expenses
-> Implementing auto-scaling mechanisms,designing scalable architectures
-> Implementing fault-tolerant mechanisms,checkpointing & job retries

DATA PROC [Production Deployment] :

-> Managing data movement efficiently,minimizing data transfer costs
-> Optimizing data transfer speed over network connections
-> Implementing encryption,access controls,audit logging mechanisms
-> Ensuring consistency between development,testing,production environment

DATA PROC [Development] :

-> Managing dependencies between pipeline stages;ensuring fault-tolerance & data consistency
-> Implementing data validation checks & quality assurance processes
-> Comprehensive testcases, validating piepiline behaviour

DATA PROC [Dataside] :

-> Incomplete,inaccurate and inconsistent data
-> Managing data shuffling,optimizing resource utilization
-> Capturing & maintaining metadata; tracking data dependencies

DATA PROC [Storage and Processing] :

-> Scaling storage systems to accomodate large datasets
-> Optimizing data access patterns,minimizing data transfer times
-> Managing data movement efficiently,minimizing data transfer costs
-> Implementing data replication and redundancy mechanisms

DATAFLOW

DATA FLOW [Environment] :

-> Dataflow offers autoscaling capabilities to handle varying workloads, but improper configuration can lead to under-provisioning or over- provisioning of resources.
-> Network latency can impact the performance of Dataflow jobs, especially in streaming scenarios where low latency is critical.
-> Insufficient network bandwidth can lead to delays in data processing and increased job runtimes, especially when dealing with large volumes of data.
-> Managing compatibility between different versions of Dataflow APIs and SDKs, and ensuring that updates do not break existing jobs, is crucial.
-> Debugging Dataflow jobs, especially in distributed environments, requires effective tools and strategies to trace and resolve issues.

DATA FLOW[Production Deployment] :

-> Achieving consistent throughput requires balancing resource allocation and optimizing data partitioning and processing logic.

-> Setting up automated deployment pipelines for Dataflow jobs to ensure smooth and reliable rollouts can be complex.

-> Implementing strong security measures, including encryption and access controls, to protect data in transit and at rest.

-> Implementing efficient data transformation logic to ensure that data is correctly processed and formatted for downstream systems.

DATA FLOW[Development] :

-> Debugging Dataflow pipelines locally is challenging because they are designed to run in a distributed environment.

-> Implementing windowing strategies and triggers correctly to handle time-based aggregations and event-time processing adds another layer of complexity.

-> Accurately estimating the cost of running Dataflow pipelines can be difficult due to variable data volumes and processing complexities.

-> Integrating Dataflow pipelines with various data sources and sinks can be complex, especially when dealing with different data formats and consistency requirements.

DATA FLOW[Dataside] :

-> Designing appropriate windowing strategies (e.g., tumbling, sliding, session windows) to aggregate and analyze data over specific periods.

-> Managing data skew to prevent certain partitions from becoming bottlenecks due to uneven data distribution.

-> Maintaining the correct order of events, especially in streaming pipelines, to ensure accurate processing and analytics.

-> Implementing robust data validation to detect and handle anomalies, missing values, and corrupted records.

DATA FLOW[Storage and Processing] :

-> Efficiently managing compute resources and configuring autoscaling to handle varying workloads without incurring excessive costs.

-> Choosing the right data format and compression methods to balance storage costs and read/write performance.

-> Defining appropriate data retention policies and managing the lifecycle of data, including archiving and deletion.

-> Managing and storing large volumes of data efficiently, and ensuring the system scales seamlessly with increasing data loads.