Data Architecture Design

We consider 3 components to the data architecture design:

1) Data Ingestion
2) Data Processing and Storage
3) Data Visualisation
4) Orchestration and Monitoring

1) Data Ingestion
   Datasets A and B are similar in nature but with dataset A being larger in volume. The concern would be the ability to ingest and process the high-throughput data quickly, with duplicate event handling. Kafka is a reliable open-source tool for this purpose. The available client libraries in languages like Java, Python, or .NET also allow for greater flexibility. Since high-throughput data (10000 events per seconds) for higher complexity data processing tasks are required, Structured Streaming may also be considered.

2) Data Processing and Storage
   The datasets in question and the use case likely require high performance and low latency for data access. S3 store is capable of storing both raw and processed data with durability and scalability. MSSQL may also be considered if OLTP and OLAP use cases are to be integrated.

3) Data Visualisation
   Tableau can be used to create dashboards for OLAP purposes. It also supports real-time dashboarding to visualise by fetching data from the processed data store to fetch and display real-time insights.

4) Orchestration and Monitoring
   Airflow can be used to orchestrate and schedule data ingestion, processing, and storage workflows. Given the many plug-and-play operators, it is readily executable for on-premise software and cloud provider subscriptions.

De-duplication can be carried out at various stages, such as in the data ingestion stage with the Kafka Idempotent Producer to reduce the likelihood of duplicates. The data processing stage with MSSQL may also be used to target upstream retries, as well as the data visualization stage through Tableau. Further data processing may also be carried out at Tableau to fit specific business use-cases. This ensures that veracity of data is maintained.

The proposed architecture stream is as follows:

Kafka (Data ingestion) → MSSQL (Data Processing and Storage) → Tableau (Visualisation) → Airflow (Workflow Orchestration)

Assumptions:

1) Unique identifiers must be present.
2) The primary keys are set appropriately and meaningfully to unique identifiers within the data, e.g., the geographical_location_oid must be available.
3) Volume of data is consistent. There are compute and storage hardware catered to support the high-volume data flow and subsequent scaling (if any).
4) Stable reference data (dataset B) can be cached in memory for more efficient joins.

Considerations for end users:

1) Scalability
   Influx of data volume may be experienced if more video cameras are installed. Besides the hardware support, the PM must consider the eventual velocity and volume of data flow.
   a. Does the architecture support horizontal scaling or vertical scaling?
   b. How will the system handle non-linear growth in data volume? Are there limits to the amount of data that can be stored or processed?
   c. Should the system be designed to be elastic? i.e., automatically scale up and down based on demand.

2) Reliability and stability
   The system is likely purposed for live surveillance monitoring rather than backend analytics. Reliability and stability are key concerns to ensure high availability of the system and to minimise unscheduled downtime.
   a. What mechanisms need to be in place to ensure the system remains operational in the event of hardware or software failures?
   b. How is data redundancy handled to prevent data loss? What level of data redundancy are we considering?
   c. What are levels of RTO and RPO intended for the system?
   d. What security strategies are in place for data-in-transit and at rest?

3) Data consistency
   The join between datasets A and B must be carefully monitored to maintain consistent data.
   a. Are transactions managed to ensure data integrity across the system?
   b. How are data conflicts handled in a distributed environment?

4) Performance
   Real-time data streaming and processing capabilities require low-latency data processing and retrieval. The storage system must be able to handle high read and write throughput.
   a. What is the expected latency for different operations? e.g., read/write operations.
   b. What is the expected throughput the system can handle? e.g., transactions per second.
   c. What strategies are employed to optimise query performance?

5) Cost efficiency

   The large volumes of data for real-time/almost real-time use case will inflate the cost. A balance is needed between performance and cost effectiveness.

   a. How efficiently should the system utilise computational and storage resources?
   b. Are there tools and processes in place to monitor and manage costs effectively? If not, is there manual monitoring and intervention in place?
   c. What are the costs associated with the underlying infrastructure? e.g., Cloud subscription services.