**Current System Analysis: Audit, Sustainability, and Maintenance Requirements**

1. System architecture, dependencies, and interconnections
2. Assessment of sustainability and required upgrades
3. Identifying bottlenecks, bugs, and areas for optimization
4. Decoupling from Comotion and connecting to AWS RDS

**System Architecture, Dependencies, and Interconnections**

The existing system architecture is established within the AWS infrastructure, overseen by the integrated Comotion database management system. Comotion takes charge of overseeing the existing data pipeline framework, as elaborated upon below. Operational data generated by the Mendix low-code system, OpsLife, is transmitted at 10-minute intervals through an API conduit to the designated AWS server. This data storage operation is invoked synchronously every 10 minutes, orchestrating the transfer of the accumulated data into the designated data lake repository.

Subsequently, the data derived from these operations is harnessed by the internally developed ComoDash system, devised by Comotion, to facilitate sophisticated data querying for generating insightful outcomes. The entire spectrum of data visualization and insight extraction processes relies exclusively on the ComoDash sandbox environment. This environment stands as the sole avenue for querying the data lake and extracting insights, with no alternative methodologies currently available.

[Diagram placeholder for Data Pipeline]

**Sustainability Evaluation and Required Enhancements**

Presently, the viability of the prevailing data system is subject to scrutiny, revealing shortcomings in its ability to sustainably support robust data analytics endeavours. The process of querying data through the ComoDash sandbox environment falls short of delivering the requisite depth of data-driven insights, thereby diminishing user confidence within the OpsLife system's capability to furnish meaningful, precise, and coherent data-derived insights. This predicament is attributed to a confluence of contributing factors:

1. The direct access of the Lion Africa Data Analytics (DA) team to the database remains unattainable.

2. The necessity to select one schema from a pool of five during data querying introduces confusion and fosters the potential for erroneous data extraction.

3. The design and upkeep of the data lake structures and records exhibit inadequacies, culminating in the emergence of duplicates and data omissions during querying processes. This, in turn, compels the DA team to engage in supplementary post-query data transformation measures.

4. The querying of the data lake is executed via two distinct methods:

* Utilization of the ComoDash sandbox when the queried record volume remains under 5000.
* Engagement of a separate API mechanism for queries exceeding the 5000-record threshold.

In light of these challenges, a series of essential updates are mandated to reinforce the sustainability of the current data pipeline:

* Augmented pre-processing and data transformation methodologies must be incorporated, pre-emptively applied before the integration of new data into the data lake.
* The imperative for multiple data schemas during querying endeavours must be rectified by establishing a universal, all-encompassing data schema that can be employed for all querying endeavours.
* Comprehensive adjustments are necessitated within the ComoDash system to accommodate more expansive querying parameters, transcending the current constraint of 5000 records per query.

**Identifying bottlenecks, bugs, and areas for optimization**

* **Performance Profiling:** Utilize specialized profiling tools to pinpoint performance bottlenecks, such as excessive processing times or memory leaks, across the data pipeline components. This assessment will provide insights into areas requiring immediate attention.
* **Thorough Testing:** Execute rigorous testing scenarios to uncover any latent bugs or irregularities within the system. Employ techniques such as stress testing, load testing, and boundary testing to identify vulnerabilities that might compromise system stability.
* **Code Review:** Undertake a meticulous review of the codebase to identify inefficiencies, redundancy, or code smells that could hinder optimal performance. This includes analysing algorithms, data processing logic, and data transformation procedures.
* **Monitoring Infrastructure:** Implement a robust monitoring infrastructure to capture real-time data on system behaviour and resource utilization. This will allow for proactive identification of performance degradation and timely intervention.
* **Benchmarking:** Establish performance benchmarks to gauge system efficiency and set clear targets for improvement. Regularly compare system performance against these benchmarks to track progress and ascertain the effectiveness of optimization efforts.

**Decoupling from Comotion and Connecting to AWS RDS**

The transition from a dependence on Comotion to a direct connection with AWS RDS requires careful planning and meticulous execution to ensure a seamless integration. This process involves the following key steps:

* **Data Mapping and Transformation:** Map the existing data structure and format from Comotion to align with the schema requirements of AWS RDS. This may necessitate data transformation to facilitate smooth migration.
* **Schema Design:** Design a robust database schema within AWS RDS that accommodates the data from the existing data lake and serves the analytical needs of the OpsLife system. Ensure that the schema is optimized for querying performance.
* **Data Migration:** Develop a data migration strategy to facilitate the transfer of data from the data lake managed by Comotion to the AWS RDS instance. Employ ETL processes to ensure data integrity during migration.
* **API Integration:** Establish secure and reliable API connections between the Mendix low-code system (OpsLife) and AWS RDS. This entails creating appropriate API endpoints, implementing authentication mechanisms, and adhering to best practices for data transmission.
* **Testing and Validation:** Rigorously test the end-to-end data flow and interactions between OpsLife and AWS RDS. Validate data accuracy, query performance, and system stability under varying workloads and scenarios.
* **Data Governance and Security:** Implement robust data governance practices within AWS RDS, including access controls, encryption, and audit trails. Ensure compliance with relevant data protection regulations and industry standards (Ifris 17).
* **Change Management:** Execute a comprehensive change management strategy to guide stakeholders through the transition process. Provide training and documentation to facilitate a smooth shift from the Comotion-based environment to the AWS RDS infrastructure.