**Notes from the Conversation**

**Overview**

The discussion focuses on improving data pipelines for **Slingshot's system** with a specific emphasis on handling **query history, job credits**, and **data processing pipelines**. The goal is to ensure accurate computation, better performance, and reduced dependency on customer accounts.

**Key Topics**

**1. Job Credits Pipeline Update**

* **Old Workflow**:
  + Job credits were computed **in the customer’s account** using the load\_job\_credits procedure.
  + Data was fetched from three Snowflake views:
    - Query History
    - Warehouse Metering History
    - Query Acceleration History
  + Computation was triggered **daily** for the entire day.
  + Challenges:
    - High customer computational cost.
    - Manual interventions for large datasets.
    - Maintenance overhead for debugging and updates.
* **New Workflow**:
  + All computations moved **in-house** to Slingshot's Hub account.
  + Raw data (query history, metering history, acceleration history) is fetched **hourly**.
  + Job credits are computed on Slingshot’s side, reducing customer dependency.
  + Benefits:
    - Improved scalability and debugging.
    - Reduced computational costs for customers.
    - Easier maintenance and flexibility.

**2. Handling Query History**

* **Key Issues**:
  + **Long-running queries** (e.g., 12-hour queries) can be missed if data is fetched based on start time.
  + Snowflake adds queries to **query history** with a latency of **up to 45 minutes** after their end time.
* **Solution**:
  + Fetch query history data based on **end time**, ensuring that long-running queries are captured.
  + Range truncation:
    - Truncate **min start time** to the hour before.
    - Truncate **max start time** to the hour after.
  + Data is recomputed from the truncated range for accurate and complete processing.

**3. Persistent Database Setup**

* Each **tenant has a persistent database per region** containing metadata tables (e.g., roles, job credits, warehouse info).
* The new sync\_share\_to\_persistent\_DB procedure automates object creation and data merges for roles and other metadata.

**4. Benefits of the New Approach**

* **Reduced Customer Dependency**:
  + Customers no longer need to run procedures or maintain scripts for job credits.
  + Debugging and reprocessing are handled entirely in-house by Slingshot.
* **Improved Efficiency**:
  + Hourly processing ensures smaller, manageable chunks of data.
  + Eliminates performance bottlenecks caused by large daily computations.
* **Easier Maintenance**:
  + Updates and improvements to pipelines are now centralized in Slingshot’s Hub account.
* **Future-Proof**:
  + Data is prepared for machine learning use cases, such as query pattern analysis and cost optimization.

**5. Monitoring and Alerts**

* Existing alerts:
  + Monitor **query history failures**.
* Recommended additions:
  + Set up alerts for the new **job credits task** to monitor failures or data gaps.
  + Proactive monitoring of missing data ensures smoother system operations.

**6. Production Readiness**

* The updated pipeline is being tested in **lower environments**.
* Pending tasks:
  + Final validation to ensure new job credits match the old results or perform better.
  + Documentation updates for the new pipeline and processes.
  + Alert setup for new job credits.

**Key Learnings**

1. Using **end time** ensures long-running queries are captured correctly.
2. Hourly data fetching improves performance and reduces computational strain.
3. Moving computations in-house provides better control, scalability, and ease of maintenance.
4. Clear documentation and monitoring are essential for smooth pipeline transitions.

**Action Items**

1. **Validate the New Job Credits Pipeline**:
   * Ensure it matches or improves the old system’s accuracy.
2. **Complete Documentation**:
   * Document the updated workflows, tasks, and pipeline structure.
3. **Set Up Alerts**:
   * Monitor the new job credits task for failures or data gaps.
4. **Prepare for Production**:
   * Address any remaining testing issues and ensure a smooth transition.

**Key Takeaways**

* The Slingshot system is becoming more efficient and independent by centralizing computations in its Hub account.
* The new pipeline reduces customer impact, improves maintainability, and ensures scalability for future use cases.