Orchestration Design

The pipeline would be implemented as a daily Airflow DAG with sequential task: first parallel extraction from Zuora and Stripe sources, followed by transformation (only if extraction succeeds), and finally loading to destination, and with each task having explicit dependencies. Airflow's built-in features would handle scheduling, retries, and execution monitoring. Each task would be designed to be idempotent to support reruns without incurring side effects.

Airflow's built-in monitoring would track task statuses. Failures would trigger Slack/email alerts through webhook integrations; logged error details would be used for investigations. In case of failures, the pipeline can restart from failure points by leveraging Airflow's checkpointing (no need for full reruns). External monitoring tools like Datadog can be integrated for better observability.

System Design Brief

To handle 100M daily rows, the architecture would shift to distributed processing and cloud-native services. Data extraction would leverage change data capture (CDC) tools like Fivetran or Debezium for efficient replication from PostgreSQL and Salesforce. Transformation would be done in Spark or Snowpark for distributed processing, with results stored in cloud storage (S3/ADLS/GCS). Incremental processing would replace full loads where possible.

The target Snowflake warehouse would utilize auto-scaling compute. For query optimization, the tables in the warehouse would be partitioned by date and clustered by customer dimension (e.g. customer_email). Materialized views would also be used for frequently accessed aggregations. Assumptions include: source systems support CDC & incremental loads, and budget for scalable cloud resources.