Rental Marketplace Analytics Pipeline

This project is an end-to-end data pipeline for a rental marketplace, enabling analytical reporting on rental listings and user interactions. The platform stores application data in an AWS Aurora MySQL database, and the goal of this pipeline is to extract these data, transform them to generate key business metrics, and load them into a data warehouse (Amazon Redshift) for business intelligence and reporting.

Setup and Troubleshooting Guide

Prerequisites

- Python 3.8+
- AWS Account with appropriate permissions
- MySQL database instance
- AWS Redshift cluster
- AWS Glue service access
- AWS Step Functions access

Environment Setup

1. Clone the repository and install dependencies:
2. git clone <repository-url>
3. cd Batch-Data-Processing-for-Rental-Marketplace-Analytics
4. pip install -r requirements.txt
5. Create a .env file with the required credentials:
6. HOST=<mysql-host>
7. USER=<mysql-username>
8. PASSWORD=<mysql-password>
9. DATABASE=<database-name>
10. DB_PORT=3306

AWS Configuration

Step Functions Setup

1. Create an AWS Step Functions state machine using code/glue_scripts/step_functions_code.json.
2. Ensure IAM roles have permissions for:
    - AWS Glue job execution
    - S3 read/write access
    - Redshift Data API access

Glue Jobs Configuration

1. Create the following Glue jobs using the scripts provided in the code/glue_scripts folder:
    - rds_to_s3_extraction
    - Curated_to_s3
    - Loading data to redshift
2. Required job parameters:
    - For rds_to_s3_extraction:
    - RDS_HOST
    - RDS_PORT
    - RDS_DB_NAME
    - RDS_USERNAME
    - RDS_PASSWORD
    - S3_OUTPUT_BUCKET
    - TABLES_TO_EXTRACT
    - For Curated_to_s3:
    - OUTPUT_BUCKET
    - DATABASE_NAME
    - For Loading data to redshift:
    - REDSHIFT_CONNECTION
    - DATABASE
    - DATABASE_USER

    - IAM_ROLE_ARN
    - S3_RAW_PATH
    - S3_CURATED_PATH
    - S3_PRESENTATION_PATH

Database Setup

MySQL Setup

1. Execute ingestion scripts in order:
2. python code/ingestion_scripts/apartments.py
3. python code/ingestion_scripts/apartments_attributes.py
4. python code/ingestion_scripts/bookings.py
5. python code/ingestion_scripts/user_viewing.py

Redshift Setup

1. Create required schemas:
    - raw_layer
    - curated_layer
    - presentation_layer
2. Execute schema and table creation scripts from code/sql/

Pipeline Components

Data Flow

1. MySQL → S3 (raw data)
2. S3 → Glue Catalog (via crawler)
3. Glue Catalog → S3 (curated data)
4. S3 → Redshift (final destination)

Monitoring Points

- Step Functions execution console

- Glue job runs
- Redshift query history
- S3 bucket contents

Common Issues and Solutions

Extraction Job Failures

Issue: rds_to_s3_extraction job fails
Solutions:

- Verify MySQL connectivity
- Check RDS credentials
- Ensure S3 bucket permissions
- Validate table names in TABLES_TO_EXTRACT

Crawler Issues

Issue: Glue crawler fails or hangs
Solutions:

- Verify S3 path exists
- Check IAM roles
- Confirm S3 bucket permissions
- Wait for retry (configured for 30 attempts)

Curated Layer Processing

Issue: Curated_to_s3 job fails
Solutions:

- Check Glue catalog tables exist
- Verify data formats
- Monitor memory usage
- Check for schema mismatches

Redshift Loading

Issue: Loading data to redshift job fails
Solutions:

- Verify Redshift connection
- Check IAM role permissions
- Validate table schemas
- Monitor COPY command errors in SVL_QLOG

General Troubleshooting Steps

1. Check CloudWatch logs for detailed error messages
2. Verify IAM roles and permissions
3. Monitor resource usage
4. Check network connectivity
5. Validate data formats and schemas

Support and Maintenance

For additional support:

1. Review CloudWatch logs
2. Check Step Functions execution history
3. Monitor Glue job metrics
4. Review Redshift STL_LOAD_ERRORS table