

# AWS ETL Pipeline Pricing Analysis

## Executive Summary

This document provides a comprehensive cost analysis for the serverless ETL pipeline solution designed to transform Parquet files to JSON format. The architecture leverages AWS Lambda, AWS Glue, Amazon S3, DynamoDB, and API Gateway to create a scalable, cost-effective data processing solution.

## Architecture Overview

The ETL pipeline consists of the following components: - **Frontend:** React.js application (locally hosted) - **API Layer:** AWS API Gateway with Lambda integration - **Processing Layer:** AWS Glue Jobs for ETL operations - **Storage Layer:** Amazon S3 for input/output files, DynamoDB for metadata - **Orchestration:** AWS Lambda functions for workflow management

## Pricing Analysis by Service

### 1. AWS Lambda

**Service:** AWS Lambda Functions (4 functions) - upload-handler - job-trigger - status-monitor - download-handler

**Pricing Model:** Pay-per-use (requests + compute time)

**Unit Pricing:** - **Requests:** \$0.0000002 per request - **Compute Time:** - Tier 1 (0-6B GB-seconds): \$0.0000166667 per GB-second - Tier 2 (6B-15B GB-seconds): \$0.0000150000 per GB-second - Tier 3 (15B+ GB-seconds): \$0.0000133334 per GB-second

**Usage Scenarios:**

#### Low Usage (100 jobs/month)

- **Requests:** 400 requests/month (4 functions × 100 jobs)
- **Compute Time:** ~2 seconds per function × 512MB = 0.4 GB-seconds per job
- **Total Compute:** 40 GB-seconds/month

**Monthly Cost:** - Requests:  $400 \times \$0.0000002 = \$0.00008$  - Compute:  $40 \times \$0.0000166667 = \$0.00067$  - **Total Lambda Cost:** \$0.00075/month

#### Medium Usage (1,000 jobs/month)

- **Requests:** 4,000 requests/month
- **Total Compute:** 400 GB-seconds/month

**Monthly Cost:** - Requests:  $4,000 \times \$0.0000002 = \$0.0008$  - Compute:  $400 \times \$0.0000166667 = \$0.0067$  - **Total Lambda Cost:**  $\$0.0075/\text{month}$

#### High Usage (10,000 jobs/month)

- **Requests:** 40,000 requests/month
- **Total Compute:** 4,000 GB-seconds/month

**Monthly Cost:** - Requests:  $40,000 \times \$0.0000002 = \$0.008$  - Compute:  $4,000 \times \$0.0000166667 = \$0.067$  - **Total Lambda Cost:**  $\$0.075/\text{month}$

## 2. AWS Glue

**Service:** AWS Glue ETL Jobs (Python Shell)

**Pricing Model:** Pay-per-use (DPU-hours)

**Unit Pricing:** - **Standard ETL Jobs:** \$0.44 per DPU-hour - **Flex ETL Jobs:** \$0.29 per DPU-hour (recommended for intermittent workloads)

**Usage Scenarios:**

#### Low Usage (100 jobs/month)

- **Job Duration:** ~5 minutes per job (0.083 hours)
- **DPU Usage:** 1 DPU per job (Python Shell default)
- **Total DPU-hours:**  $100 \times 0.083 \times 1 = 8.3$  DPU-hours/month

**Monthly Cost:** - Standard:  $8.3 \times \$0.44 = \$3.65$  - **Flex (Recommended):**  $8.3 \times \$0.29 = \$2.41$

#### Medium Usage (1,000 jobs/month)

- **Total DPU-hours:** 83 DPU-hours/month

**Monthly Cost:** - Standard:  $83 \times \$0.44 = \$36.52$  - **Flex (Recommended):**  $83 \times \$0.29 = \$24.07$

#### High Usage (10,000 jobs/month)

- **Total DPU-hours:** 830 DPU-hours/month

**Monthly Cost:** - Standard:  $830 \times \$0.44 = \$365.20$  - **Flex (Recommended):**  $830 \times \$0.29 = \$240.70$

## 3. Amazon S3

**Service:** Object Storage for input/output files

**Pricing Model:** Pay-per-use (storage + requests)

**Unit Pricing:** - **Standard Storage:** \$0.023 per GB-month (first 50TB)  
- **PUT Requests:** \$0.0005 per 1,000 requests - **GET Requests:** \$0.0004 per 1,000 requests

**Usage Scenarios:**

#### Low Usage (100 jobs/month)

- **Storage:** 10GB average (5GB input + 5GB output)
- **PUT Requests:** 200/month (upload + Glue output)
- **GET Requests:** 300/month (download + processing)

**Monthly Cost:** - Storage:  $10 \times \$0.023 = \$0.23$  - PUT:  $0.2 \times \$0.0005 = \$0.0001$  - GET:  $0.3 \times \$0.0004 = \$0.00012$  - **Total S3 Cost:** \$0.23/month

#### Medium Usage (1,000 jobs/month)

- **Storage:** 50GB average
- **PUT Requests:** 2,000/month
- **GET Requests:** 3,000/month

**Monthly Cost:** - Storage:  $50 \times \$0.023 = \$1.15$  - PUT:  $2 \times \$0.0005 = \$0.001$  - GET:  $3 \times \$0.0004 = \$0.0012$  - **Total S3 Cost:** \$1.15/month

#### High Usage (10,000 jobs/month)

- **Storage:** 200GB average
- **PUT Requests:** 20,000/month
- **GET Requests:** 30,000/month

**Monthly Cost:** - Storage:  $200 \times \$0.023 = \$4.60$  - PUT:  $20 \times \$0.0005 = \$0.01$  - GET:  $30 \times \$0.0004 = \$0.012$  - **Total S3 Cost:** \$4.62/month

## 4. Amazon DynamoDB

**Service:** NoSQL database for job metadata

**Pricing Model:** On-Demand (pay-per-request)

**Unit Pricing:** - **Read Request Units:** \$0.125 per million RRUs - **Write Request Units:** \$0.625 per million WRUs - **Storage:** \$0.25 per GB-month (after 25GB free tier)

**Usage Scenarios:**

#### Low Usage (100 jobs/month)

- **Write Operations:** 500/month (job creation + status updates)
- **Read Operations:** 1,000/month (status checks + job queries)
- **Storage:** <1GB (within free tier)

**Monthly Cost:** - Writes:  $0.0005M \times \$0.625 = \$0.0003125$  - Reads:  $0.001M \times \$0.125 = \$0.000125$  - Storage: \$0 (free tier) - **Total DynamoDB Cost:** \$0.0004/month

#### Medium Usage (1,000 jobs/month)

- **Write Operations:** 5,000/month
- **Read Operations:** 10,000/month
- **Storage:** <1GB (within free tier)

**Monthly Cost:** - Writes:  $0.005\text{M} \times \$0.625 = \$0.003125$  - Reads:  $0.01\text{M} \times \$0.125 = \$0.00125$  - Storage: \$0 (free tier) - **Total DynamoDB Cost:** \$0.004/month

#### High Usage (10,000 jobs/month)

- **Write Operations:** 50,000/month
- **Read Operations:** 100,000/month
- **Storage:** ~2GB

**Monthly Cost:** - Writes:  $0.05\text{M} \times \$0.625 = \$0.03125$  - Reads:  $0.1\text{M} \times \$0.125 = \$0.0125$  - Storage:  $(2-25) \times \$0.25 = \$0$  (within free tier) - **Total DynamoDB Cost:** \$0.044/month

## 5. Amazon API Gateway

**Service:** REST API endpoints

**Pricing Model:** Pay-per-request

**Unit Pricing:** - **REST API:** \$3.50 per million requests (first 333M) - **HTTP API:** \$1.00 per million requests (first 300M) - Recommended

**Usage Scenarios:**

#### Low Usage (100 jobs/month)

- **API Requests:** 1,000/month (10 requests per job workflow)

**Monthly Cost:** - REST API:  $0.001\text{M} \times \$3.50 = \$0.0035$  - **HTTP API (Recommended):**  $0.001\text{M} \times \$1.00 = \$0.001$  - **Total API Gateway Cost:** \$0.001/month

#### Medium Usage (1,000 jobs/month)

- **API Requests:** 10,000/month

**Monthly Cost:** - REST API:  $0.01\text{M} \times \$3.50 = \$0.035$  - **HTTP API (Recommended):**  $0.01\text{M} \times \$1.00 = \$0.01$  - **Total API Gateway Cost:** \$0.01/month

#### High Usage (10,000 jobs/month)

- **API Requests:** 100,000/month

**Monthly Cost:** - REST API:  $0.1\text{M} \times \$3.50 = \$0.35$  - **HTTP API (Recommended):**  $0.1\text{M} \times \$1.00 = \$0.10$  - **Total API Gateway Cost:** \$0.10/month

## Total Cost Summary

### Monthly Cost Breakdown

Service	Low Usage (100 jobs)	Medium Usage (1,000 jobs)	High Usage (10,000 jobs)
AWS			

<b>Lambda</b>	\$0.00075	\$0.0075	\$0.075
<b>AWS Glue (Flex)</b>	\$2.41	\$24.07	\$240.70
<b>Amazon S3</b>	\$0.23	\$1.15	\$4.62
<b>DynamoDB</b>	\$0.0004	\$0.004	\$0.044
<b>API Gateway (HTTP)</b>	\$0.001	\$0.01	\$0.10
<b>TOTAL</b>	<b>\$2.64</b>	<b>\$25.24</b>	<b>\$245.54</b>

### Cost Per Job

Usage Level	Total Monthly Cost	Cost Per Job
Low (100 jobs)	\$2.64	\$0.026
Medium (1,000 jobs)	\$25.24	\$0.025
High (10,000 jobs)	\$245.54	\$0.025

## Cost Optimization Recommendations

### 1. Immediate Optimizations

- **Use AWS Glue Flex Jobs:** Save 34% on Glue costs (\$0.29 vs \$0.44 per DPU-hour)
- **Implement HTTP API:** Save 71% on API Gateway costs (\$1.00 vs \$3.50 per million requests)
- **Optimize Lambda Memory:** Right-size Lambda functions to reduce compute costs
- **S3 Lifecycle Policies:** Move processed files to cheaper storage classes after 30 days

### 2. Advanced Optimizations

- **Batch Processing:** Group multiple small files into single Glue jobs
- **Reserved Capacity:** Consider DynamoDB reserved capacity for predictable workloads
- **CloudWatch Optimization:** Reduce log retention periods to minimize storage costs
- **Data Compression:** Implement compression for S3 storage to reduce costs

### 3. Monitoring and Alerting

- **Cost Budgets:** Set up AWS Budgets with alerts at 80% and 100% thresholds
- **Usage Monitoring:** Track DPU-hours and request volumes for optimization opportunities
- **Performance Metrics:** Monitor job execution times to optimize resource allocation

## Free Tier Benefits

## First 12 Months (New AWS Accounts)

- **Lambda:** 1M requests + 400,000 GB-seconds free per month
- **DynamoDB:** 25GB storage + 25 RCU/WCU free
- **S3:** 5GB standard storage + 20,000 GET + 2,000 PUT requests free
- **API Gateway:** 1M HTTP API requests free per month

**Estimated Free Tier Savings:** \$1-2 per month for low usage scenarios

## Scaling Considerations

### Linear Scaling Factors

- AWS Glue costs scale linearly with job volume
- Lambda costs scale with request volume and execution time
- S3 costs scale with data volume and request frequency

### Volume Discounts

- API Gateway: Reduced rates after 300M requests/month
- S3: Reduced rates after 50TB/month
- Lambda: Reduced compute rates at higher usage tiers

## Risk Factors and Mitigation

### Cost Risks

1. **Runaway Jobs:** Implement timeout controls and monitoring
2. **Data Growth:** Monitor storage costs and implement lifecycle policies
3. **Error Handling:** Ensure failed jobs don't consume unnecessary resources

### Mitigation Strategies

1. **Cost Alerts:** Set up billing alerts and budgets
2. **Resource Limits:** Implement service quotas and limits
3. **Regular Reviews:** Monthly cost analysis and optimization reviews

## Conclusion

The serverless ETL pipeline offers excellent cost efficiency with a pay-per-use model. Key findings:

- **Low barrier to entry:** \$2.64/month for 100 jobs
- **Predictable scaling:** ~\$0.025 per job across all usage levels
- **Cost-effective:** 90% of costs come from actual data processing (Glue)
- **Optimization potential:** 30-40% cost reduction through recommended optimizations

The solution provides excellent value for organizations needing flexible, scalable ETL processing without the overhead of managing infrastructure.

---

**Report Generated:** December 1, 2025  
**Pricing Data Source:** AWS Price List API  
**Region:** US East (N. Virginia)  
**Currency:** USD