# Metadata-Driven ETL Implementation Plan

Using AWS Glue, Redshift, and S3

## 1. Establish Metadata Control Table

The foundation of this approach is a control table in Amazon Redshift. This table will define all column mappings between source files in S3 and the target Redshift tables. Each record specifies the source column, target column, data types, and any transformation rules. Business users can update mappings directly in Redshift using SQL, enabling changes without modifying ETL code. The control table therefore becomes the single source of truth for all transformation logic.

## 2. Configure AWS Glue Connections

Set up two key connections in the AWS Glue console:  
- Redshift connection: Provide cluster endpoint, database name, credentials, and ensure it runs in the same VPC/security group.  
- S3 connection: Grant Glue permission to read from the raw data bucket and write to staging/output locations.  
  
Both connections should be tested to confirm that Glue can access the necessary resources. Any networking or IAM permission issues should be resolved before proceeding.

## 3. Create IAM Roles and Permissions

Define an IAM role that the Glue job will assume. This role should:  
- Include the AWSGlueServiceRole managed policy.  
- Allow GetObject and ListBucket on source S3 buckets, and PutObject on staging buckets.  
- Permit Redshift actions for database connection and data loading.  
- Enable CloudWatch logging for job monitoring.  
  
This role ensures the Glue job can securely read, transform, and write data across all required services.

## 4. Develop the Glue ETL Job

Create a Glue job using Spark and Python. The job should:  
1. Query the Redshift control table to retrieve active mappings for the current target table.  
2. Load raw data files from S3 into dynamic frames.  
3. Apply transformations dynamically based on the control table rules (renaming, type casting, formatting, and custom logic).  
4. Write transformed data into the target Redshift tables using Glue’s Redshift connector.  
  
This design allows one job to handle multiple source-to-target mappings without hardcoding logic.

## 5. Implement Error Handling and Validation

Introduce validation steps to ensure data integrity:  
- Verify all source columns referenced in the control table exist in incoming files.  
- Check for data type mismatches before applying conversions.  
- Log any failed transformations with detailed messages (column name, rule applied, error details) to CloudWatch.  
- Write failed records to a dedicated S3 bucket for later review and correction.  
  
This ensures resilience and transparency in data processing.

## 6. Automate Job Scheduling and Monitoring

For automation:  
- Use Amazon EventBridge or S3 event notifications to trigger Glue jobs when new data files arrive.  
- Configure CloudWatch metrics and alarms to track job status, duration, and error counts.  
- Send failure or performance alerts via Amazon SNS, notifying responsible teams immediately.  
  
This creates a fully managed and monitored ETL pipeline.

## 7. Manage and Govern Mappings

Provide procedures for business users to manage mappings:  
- Add new mappings with INSERT statements.  
- Modify mappings with UPDATE.  
- Deactivate obsolete mappings by setting the is\_active flag to FALSE.  
  
Consider implementing an approval process before changes go live, and log all modifications with user IDs and timestamps for audit purposes. This balances flexibility with governance.

## 8. Testing and Performance Optimization

Before production deployment:  
- Test with small representative datasets to confirm transformation accuracy.  
- Simulate edge cases such as missing columns, null values, or invalid data types.  
- Perform load tests with production-scale data to identify bottlenecks.  
- Optimize Glue configurations (worker type, number of workers, partitions, memory).

## Key Benefits

- Flexibility: Update mappings without code deployments.  
- Reusability: Single job handles multiple tables and formats.  
- Auditability: Metadata-driven changes logged and tracked.  
- Scalability: Designed for large and growing datasets.  
- User Empowerment: Business teams manage mappings directly.

