MySQL to Elasticsearch Integration – Detailed Explanation

# 1. AWS DMS (Database Migration Service) with Change Data Capture (CDC)

AWS Database Migration Service (DMS) is a fully managed service by Amazon Web Services that enables seamless data replication from relational databases like MySQL to other systems, including Elasticsearch (or AWS OpenSearch). It supports both full-load migration and Change Data Capture (CDC) — a mechanism that captures and streams real-time data changes from the source database to the target system.

In the CDC mode, AWS DMS listens to the binary log (binlog) of the MySQL database to detect and replicate real-time changes, such as new inserts, updates, and deletions. These changes are then transformed into JSON documents and sent to the target Elasticsearch index.

## How it works:

1. MySQL Configuration: The source MySQL database must have binary logging enabled, with the binlog\_format set to ROW. This allows DMS to accurately detect row-level changes.  
2. DMS Replication Instance: A replication instance is provisioned within AWS. It acts as the intermediary engine that connects to both source and target endpoints.  
3. Source Endpoint Configuration: The MySQL server is configured as a source endpoint. It must be accessible from the replication instance and a user with replication privileges must be provided.  
4. Target Endpoint Configuration: The Elasticsearch/OpenSearch server is configured as the target endpoint. DMS connects to it and automatically writes documents into the specified index.  
5. CDC Task Creation: A migration task is created in DMS with either “CDC only” or “Full Load + CDC” mode. The CDC mode ensures continuous replication of changes as they happen in MySQL.

## Strengths of AWS DMS:

- Real-time Replication  
- No Infrastructure Maintenance  
- Resilience  
- Scalability

## Limitations:

- No Native Support for SQL Joins  
- Initial Setup Complexity  
- Cost

## When to Choose AWS DMS:

- You need live data to appear in your Kibana dashboard as soon as it is entered or modified in MySQL.  
- Your system is hosted on AWS or has a stable connection to AWS.  
- You want a managed and resilient solution without building custom ETL jobs.  
- You can pre-process or flatten your data using views or intermediate tables to overcome the lack of joins in DMS.

# 2. Logstash JDBC Plugin with Scheduled SQL Polling

Logstash, a data processing pipeline tool from the Elastic Stack, offers a JDBC input plugin that enables pulling data from relational databases like MySQL into Elasticsearch. It works by executing a SQL query on a fixed interval (e.g., every minute), retrieving the result set, and indexing the data into Elasticsearch.

Unlike AWS DMS, Logstash does not rely on the binary log for change tracking. Instead, it works based on periodic polling — meaning it checks the database every X minutes and extracts the data as defined in the SQL query. It can be configured to fetch only new or updated rows using a timestamp column or primary key logic.

## How it works:

1. JDBC Connection: Logstash uses a JDBC driver to connect to the MySQL database. The connector is configured with the database URL, username, and password.  
2. Custom SQL Query: A SQL query is defined to extract the desired data from one or more tables. This query can include joins, filters, transformations, and aggregations.  
3. Polling Schedule: The polling frequency is set (e.g., every 1 minute). On each run, Logstash executes the SQL query, retrieves the result set, and prepares the data for indexing.  
4. Data Output: The result is transformed (if needed) and sent to Elasticsearch. Each record becomes a document in the specified index.

## Strengths of Logstash:

- Flexibility  
- Open-source and free  
- Lightweight and Transparent  
- No Binlog Required

## Limitations:

- Not Real-Time  
- No Native Change Tracking  
- Potential Duplication or Skips

## When to Choose Logstash JDBC:

- Your dashboard can tolerate 1–5 minute delays between data updates.  
- You want to join multiple tables or apply SQL-based logic before indexing.  
- You prefer full control over how and what data is moved.  
- You are not yet on AWS or do not want to rely on AWS-managed services.  
- You are working on a prototype, MVP, or analytical use case that doesn’t require millisecond updates.