Extract: extract source data in **real-time,** as soon as it appears in the source.

### Transform:  raw data extracted from the source is transformed into data that is compatible with the specific destination

### Load: The last step of the ETL process is loading the data into the target system, or destination.

### Transformation Isn’t Necessarily Done In Data Pipelines

### AWS Glue

### Azure Data Factory

### Google Cloud Dataflow

### 3 ways: batch processing, stream processing, Automated Cloud Data Warehouse

### Stream processing: To build a stream processing ETL pipeline with Kafka, you need to:

### Extract data into Kafka: the Confluent JDBC connector pulls each row of the source table and writes it as a key/value pair into a Kafka topic (a feed where records are stored and published). Applications interested in the state of this table read from this topic. As client applications add rows to the source table, Kafka automatically writes them as new messages to the Kafka topic, enabling a real-time data stream. Note you can implement a database connection yourself without Confluent’s commercial product.

### Pull data from Kafka topics: the ETL application extracts messages from the Kafka topic as Avro records, creates an Avro schema file, and deserializes them. Then it creates KStream objects from the messages.

### Transform data in KStream objects: with the Kafka Streams API, the stream processor receives one record at a time, processes it, and produces one or more output records for downstream processors. These processors can transform messages one at a time, filter them based on conditions, and perform data operations on multiple messages such as aggregation.

### Load data to other systems: the ETL application still holds the enriched data, and now needs to stream it into target systems, such as a data warehouse or data lake. In Confluent’s example, they propose using their S3 Sink Connector to stream the data to Amazon S3. You can also integrate with other systems such as a Redshift data warehouse using Amazon Kinesis.

### 