Steps to Design of Data Pipeline for Offloading E-commerce Transaction Data

Data Source and Format

For capturing the transaction data, we can use various sources like logs, API, messaging systems, or databases. In this case, we can use a messaging system like Apache Kafka to capture the transaction data in real-time. The transaction data format can be structured data like JSON or CSV.

Type of Data Pipeline

We will use a stream processing pipeline to handle the high volume of real-time transaction data generated by the website. Stream processing is an ideal choice as it can handle the data in real-time with low latency, and we can perform continuous processing of data.

Data Warehouse

We will use a distributed data warehouse like Amazon Redshift, Google BigQuery, Teradata or Snowflake to store the transaction data. These data warehouses can handle large volumes of data and are scalable and efficient.

Data Transformation and Cleaning

Before storing the transaction data in the data warehouse, we need to perform some transformation and cleaning. The transformation includes enriching the data with additional attributes, mapping, or aggregating the data. Cleaning involves filtering out irrelevant data, fixing data quality issues, and normalizing the data.

Scheduling Mechanism

To ensure the data in the data warehouse is up-to-date, we need to schedule the pipeline to run at a specific time. For example, we can schedule the pipeline to run every second to capture the new transaction data generated by the website.

Potential Issues and Bottlenecks

During the offloading process, we may face some issues and bottlenecks like network latency, data skew, data duplication, or hardware failure. To address these issues, we can use techniques like data partitioning, load balancing, data replication, and fault tolerance. We can also monitor the pipeline's performance and troubleshoot any issues in real-time using monitoring tools like Prometheus, Grafana, or Datadog.