**Anatomy of a Fetch Request**

**Anatomy of a Fetch Request"** in **Apache Kafka**, showing how data is fetched by consumers and replica followers from Kafka brokers.

**What's a Fetch Request?**

A **fetch request** is how Kafka consumers and replica followers **retrieve messages** from Kafka topics and partitions.

**Step-by-Step Explanation:**

**1. Kafka Consumers and Replica Followers**

* **Kafka Consumers**: Request data from Kafka to process.
* **Replica Followers**: Request data from the leader partition to replicate it.

**2. Network**

* The fetch request travels over the network to a **Kafka Broker**.
* Handled by **Network Threads**, which listen for incoming requests.

**3. Request Queue**

* The fetch request is placed in the **Request Queue**, waiting for processing.

**4. Worker Threads / IO Threads**

* **Worker Threads** pick up the request and begin reading data from disk (via Page Cache).
* They handle I/O-intensive work and load the data efficiently.

**5. Page Cache**

* Like with produce requests, Kafka uses the **Page Cache** to read data from disk efficiently without frequent physical disk access.

**6. Purgatory Map**

* If the request can’t be fulfilled immediately (e.g., no new data available yet), it enters **Purgatory**.
* This is an internal Kafka mechanism to **wait for specific conditions** to be met before replying.

**7. Readiness Check**

* Kafka checks:
  + Is the consumer or replica ready to receive data?
  + Conditions include:
    - fetch.min.bytes (minimum data size)
    - fetch.max.wait.ms (maximum wait time)
    - Availability of new messages

**8. Response Queue**

* Once the fetch request is satisfied (data is available or timeout is reached), the response goes into the **Response Queue**.
* Network threads pick it up and send the response back to the **consumer or replica**.

**Important Notes (from the diagram):**

* **Same Queues and Threads**: Kafka reuses the **same network threads, request queue, response queue, and IO threads** as with produce requests.
* **Separate Purgatories**: Kafka maintains **separate purgatories** for **produce and fetch requests** to handle their distinct waiting conditions independently.