**Error Handling Strategies**

**Error Handling Strategies** in streaming pipelines, specifically with **Redpanda Connect**. Let’s break it down:

**1. Dead-Letter Queue (DLQ)**

* **Concept**: A special storage (queue or topic) for failed messages.
* **Purpose**: Isolate problematic records so they don’t affect the main pipeline. These can later be reprocessed, analyzed, or corrected.
* **Redpanda Feature**: Any Kafka topic can serve as a DLQ. You configure manual routing so that failed messages go there.

*Example*: If inserting into a database fails due to a schema mismatch, that record is sent to a DLQ topic for later inspection.

**2. Fallback Output**

* **Concept**: A backup output channel.
* **Purpose**: If the main output fails, the system automatically sends errored messages to an alternate sink. This ensures the main pipeline isn’t blocked.
* **Redpanda Feature**: fallback output with **failover sinks** (as seen in your previous YAML example).

*Example*: Primary output → Snowflake. If Snowflake is down, failed messages go to S3 bucket or another Kafka topic.

**3. Poison Pill Isolation**

* **Concept**: A "poison pill" message is one that always causes failures (e.g., corrupted JSON).
* **Purpose**: Prevents one bad message from blocking the entire pipeline.
* **Redpanda Feature**: Redpanda Connect uses **error flags + routing to DLQ/fallback** so that poison pills are quarantined, not retried forever.

*Example*: A malformed event is detected → flagged as errored → routed to DLQ → main pipeline continues.

**In summary:**

* **DLQ** = safe storage for bad messages.
* **Fallback Output** = automatic rerouting when primary sink fails.
* **Poison Pill Isolation** = keeps one bad record from breaking the entire system.