databricks

Cloud Storage with LakeFlow Connect
Standard Connectors

LECTURE

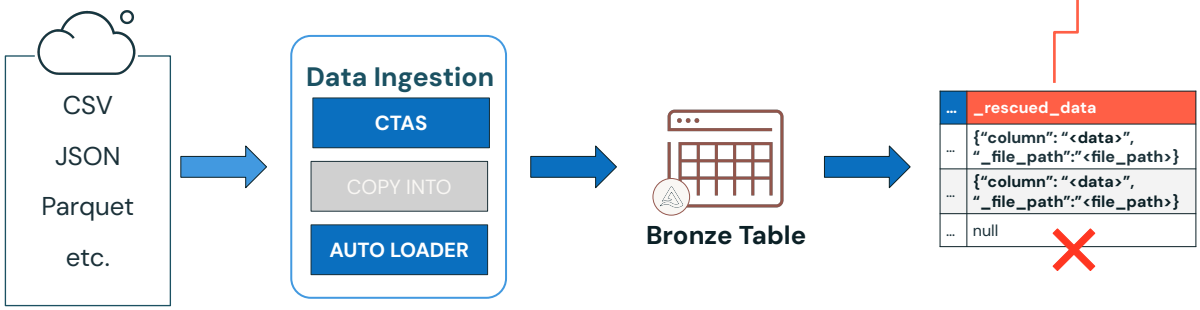
Working with the Rescued Data Column

In data ingestion, the rescued data column (_rescued_data) captures mismatched or unparseable fields as JSON—preserving non-conforming input values in your Lakehouse tables instead of dropping them.

Working with the Rescued Data Column

Rescuing Malformed Rows on Ingestion

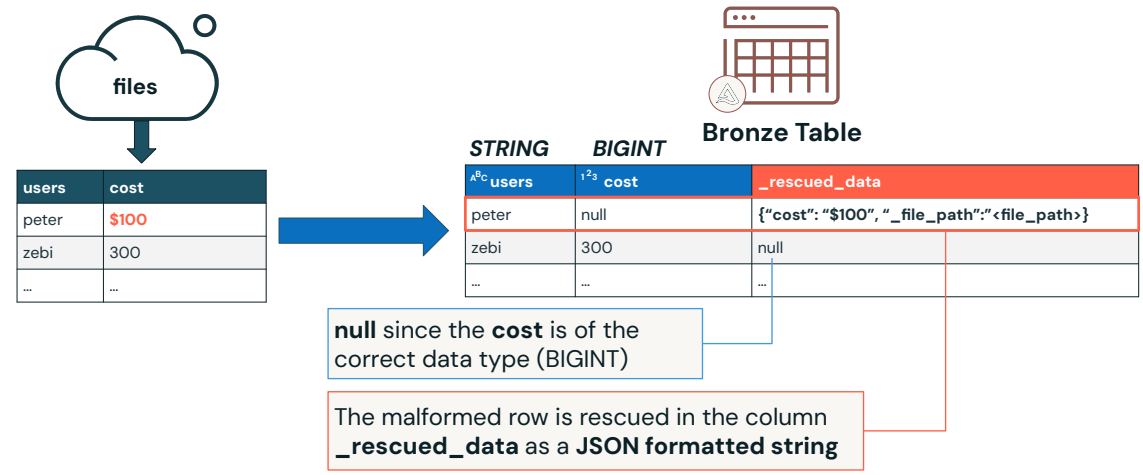
read_files(), spark.read or **Auto Loader** provides a **rescued data column** if the raw data does not match the schema



During data ingestion there are times when the input data doesn't match with the schema in your table. Ingestion techniques like read_files(), spark.read or Auto Loader provide a rescued data column during ingestion. The rescued data column ensures that columns that don't match with the schema are rescued instead of being dropped.

Working with the Rescued Data Column

Rescued Data Column Example



For example, suppose your cloud storage location contains a set of raw files (these could be CSV, TXT, JSON, or other formats) and it contains a users and cost column.

When ingesting this data, the users column must be read into the table as a STRING, and the cost column must be read as a BIGINT.

In the first row of data, the value "Peter" will be read into the users column of the bronze table correctly. However, since the cost column contains a string like $100, it does not match the expected BIGINT type. As a result, this value will not be inserted into the cost column. Instead, it will be captured in the _rescued_data column, stored as a JSON-formatted string.

In the second row, both values are valid: the value "zebi" is a STRING, and 300 is a BIGINT. Therefore, this row will be read into the bronze table without issues, and the _rescued_data column will remain empty for that row.

Thank you for completing this lesson and continuing your journey to develop your skills with us.