Data Validation

In pyspark….!

Validation Types in pyspark...

- **Schema Validation**
- **Null/Empty Checks**
- **Data Types Validation**
- **Range Checks**
- **Uniqueness Constraints**
- **Pattern Matching(Regex)**
- **Referential Integrity Checks**
- **Date Validation**

1. Schema Validation

- **Verifies data types and structure as per the defined schema.**

eg. Example: Ensure `Sales` is of type `IntegerType`, `Date` is `DateType`.

2. Null/Empty Checks

- **Identify columns with null or empty values.**

Example: `df.filter(df["column"].isNull())

3. Data Types Validation

- **Ensure correct data types using schema or `cast()` methods.**

Example: `df.withColumn("amount", df["amount"].cast("Double"))`

4. Range Checks

- **Validate numerical values lie within an acceptable range.-**

Example: `df.filter((df["age"] >= 0) & (df["age"] <= 120))

5. Uniqueness Constraints

- **Ensure specific columns have unique values.-**

Example: Detect duplicates via `groupBy().count().filter("count > 1")`

6. Pattern Matching(Regex)

- **For validating strings like emails, phone numbers.-**

Example: `df.filter(df["email"].rlike("^[\w.-]+@[\w.-]+\.\w+$"))

7. Referential Integrity Checks

- **Validate foreign keys exist in the related dataset.-**

Use joins to verify presence

8. Date Validation

- **Ensure dates are not null and within logical bounds.-**

Example: `df.filter(df["date"] < current_date())

These validations can be incorporated into ETL pipelines using PySpark DataFrame transformations or custom validation functions