# Prepare Your Data

# Concept | Recipe

In Dataiku, recipes are fundamental components within the Flow that encapsulate data transformation logic. They play a crucial role in separating processing logic from datasets, offering several advantages:

1. **Flexibility with Data Storage**: Recipes allow you to change underlying data storage technologies (e.g., switching cloud providers) without affecting the processing logic defined in the Flow's recipes. This flexibility ensures that your data processing workflows remain intact despite changes in data infrastructure.
2. **Clear Data Lineage**: By examining the Flow, you can trace the entire history of transformations applied to data, from its raw form to the final output dataset. This lineage visibility is critical for understanding how data has been manipulated and transformed over time.

**Recipe Categories**

Recipes in Dataiku are categorized based on their functionality and purpose:

* **Visual Recipes (Yellow circles)**:
  + These recipes offer a graphical user interface (GUI) for performing common data transformations. Users can apply operations like cleaning, grouping, or filtering data without needing to write code manually.
* **Code Recipes (Orange circles)**:
  + Code recipes provide the flexibility to define custom processing logic using languages such as Python, R, or SQL. They are ideal for tasks that require specific or complex transformations beyond what visual recipes offer.
* **Plugin Recipes (Red circles)**:
  + Plugin recipes extend Dataiku's capabilities through custom plugins. They combine the advantages of code recipes (freedom to define custom logic) with the accessibility of visual recipes (easy GUI-based interaction). Plugin recipes appear in the Flow only after relevant plugins have been manually installed.

**Choosing the Right Recipe**

* **Visual Recipes**: Suitable for common data transformation tasks that can be efficiently handled through a GUI, promoting ease of use and accessibility across team members who may not be proficient in coding.
* **Code Recipes**: Provide maximum flexibility for advanced transformations and custom logic, allowing data scientists and analysts to implement specific processing requirements using their preferred programming languages.
* **Plugin Recipes**: Offer a hybrid approach, where the power of custom code can be encapsulated within a visual interface, combining ease of use with the capability for advanced customization.

**Benefits of Recipe-Based Approach**

* **Modularity and Reusability**: Recipes can be reused across different projects or within the same project, promoting consistency and efficiency in data processing workflows.
* **Version Control and Collaboration**: Changes made to recipes are tracked, facilitating version control and collaborative work among team members working on the same project.

By leveraging recipes in Dataiku's Flow, organizations can streamline data transformation processes, maintain data lineage transparency, and adapt seamlessly to evolving data storage technologies and business requirements.

Top of Form

Bottom of Form

## Concept | Prepare Recipe

The "Prepare Recipe" in Dataiku is a powerful tool for creating data transformation scripts visually and interactively. Here's a breakdown of how you can use it effectively:

### 1. Adding Transformation Steps

To prepare your data, you add transformation steps to the recipe script. These steps can be added in several ways:

* **Using the Processor Library**: Dataiku provides a library of around 100 data processors. Each processor is designed to handle specific tasks such as filtering rows, rounding numbers, extracting regular expressions, concatenating or splitting columns, and more. For instance, there's a Formula language processor similar to spreadsheet functions for creating new columns.
* **Column Context Menu**: When working with a specific column, Dataiku suggests relevant steps based on the column's data type and context. For example, it might suggest parsing date columns or applying string transformations like converting to lowercase.
* **Analyze Window**: This window guides you through data preparation tasks such as merging categorical values, offering a structured approach to refining your data.
* **Manual Adjustments**: You can manually adjust column order by dragging columns, switch between table and column views for batch operations, and more.

### 2. Previewing and Applying the Script

When you add steps to the script, you can immediately preview their effect on a sample of the dataset in the Explore tab. This allows for quick iteration and modification of transformation steps. However, these changes are applied only when you run the recipe.

### 3. Managing Complexity

As scripts grow more complex, Dataiku offers features to manage them effectively:

* **Disabling Steps**: Temporarily disable steps to test different scenarios without deleting them.
* **Organizing Steps**: Group individual steps logically and add colors or comments to enhance readability and collaboration.
* **Copying and Pasting**: Easily copy steps within the same recipe or to another recipe, even across different projects or Dataiku instances.

### 4. Execution and Output

When you're ready, running the recipe applies all transformation steps to the entire input dataset, generating a new output dataset. The original input dataset remains unchanged.

# Concept | Date handling

In Dataiku, handling date parsing and working with date columns involves utilizing the Prepare recipe, which simplifies the process of recognizing and parsing date formats correctly. Here’s a step-by-step overview based on the provided text:

**Challenges with Date Handling**

Working with dates in data often presents challenges due to:

* **Multiple Formats:** Dates can be represented in various formats such as "1/5/19", "2019-01-05", or "1 May, 2019".
* **Time Zones:** Dates may include time zone information.
* **Extracting Components:** Extracting specific components like day of the week can be complex.

**Parsing Dates in Dataiku**

1. **Identifying Unparsed Date Columns:**
   * Dataiku's Prepare recipe helps in recognizing columns that contain unparsed dates.
2. **Using the Parse Date Processor:**
   * Dataiku suggests the Parse date processor when it identifies a column as containing unparsed dates.
   * This processor helps convert ambiguous date strings into a format that the system can understand as a date.
3. **Applying the Processor:**
   * Once identified, you can add the Parse date processor to the script.
   * This processor requires specifying the format of the date and optionally the time zone to ensure accurate parsing.
4. **Previewing and Confirming Changes:**
   * After applying the Parse date processor, Dataiku immediately shows a preview of how the date format has changed.
   * The column transforms from an unparsed string to a structured date format.
5. **Further Data Manipulations:**
   * With parsed dates, Dataiku suggests additional steps like computing time differences, extracting date components (e.g., day of the week), or filtering based on dates.

# Concept | Dataiku formulas

Dataiku's Formula language provides a powerful toolset similar to traditional spreadsheet formulas, enabling users to perform calculations, manipulate data, and enhance the functionality of visual recipes within the platform. Here’s a comprehensive overview of its capabilities and practical applications:

**Formula Processor in Dataiku**

The Formula processor within the Prepare recipe allows you to incorporate formulas to transform and analyze data:

* **Formula Step**: Add a Formula step to create new columns or modify existing ones based on specified expressions.
* **Editor Panel**: Offers support features like code completion and error alerts, enhancing ease of use and accuracy.

**Key Features of Dataiku Formula Language**

1. Mathematical Functions: Includes round(), sum(), and max() for numerical operations.
2. Comparison Operators: Supports >, <, >=, <= for comparing values.
3. Logical Operators: Includes AND, OR for combining conditions.
4. Handling Missing Values: Functions such as isBlank() or isNULL() assist in managing null or empty values.
5. Conditional Statements: Supports if-then statements (if(condition, value\_if\_true, value\_if\_false)) for creating conditional logic.
6. String Functions: Includes length() to measure string lengths and substring() to extract parts of strings based on specified positions.
7. Random Number Generation: Utilizes rand(min, max) to generate random integers within defined ranges, useful for simulations.
8. Column Name Handling: Functions like strval() and numval() facilitate handling column names with spaces by ensuring correct syntax and data type handling.

**Practical Use Case: Fraud Detection**

An example scenario involves flagging large credit card purchases for potential fraud detection:

* **Objective**: Compare each transaction's amount (purchase\_amount) against the average purchase amount for the corresponding credit card (card\_purchase\_amount\_avg).
* **Formula Example**: Create a new column (higher\_than\_avg\_purchase) using an if-then statement:

if(purchase\_amount > card\_purchase\_amount\_avg, "yes", "no")

This flags transactions where the purchase amount exceeds the average as "yes", indicating potential fraud.

See the [Formula language](https://doc.dataiku.com/dss/latest/formula/index.html) page in the reference documentation.

# Tutorial | Prepare Your Data (text)

At the end of the tutorial in [Basics 101](https://academy.dataiku.com/basics-101), we realized our categories of t-shirts needed to be consistently named. We can make this happen with a Prepare recipe.

**Objectives**

In this tutorial, you will:

1. **Create a Prepare Recipe**: Start a new data preparation workflow.
2. **Add Various Processor Steps**: Use different methods to add and configure steps within the Prepare recipe.
3. **Parse Date Columns**: Convert a date column from string format to a proper date type.
4. **Use the Formula Language**: Compute a new column using formulas.
5. **Run the Recipe**: Execute the preparation steps on the entire dataset.

**Step-by-Step Guide**

**1. Create a Prepare Recipe**

* Open the orders dataset in Dataiku.
* Click on the **Actions** button or the plus sign at the top-right.
* Under **Visual Recipes**, select **Prepare**.
* In the **New data preparation recipe** window:
  + Keep **orders** as the input dataset.
  + Accept the default output dataset name (**orders\_prepared**).
  + Click **Create Recipe**.

**2. Add Steps in Different Ways**

* **Drag Columns**: Drag the order\_id column in front of the pages\_visited column to reorder columns.
* **Analyze Window**: Cleanse the tshirt\_category column by merging similar categories using the Analyze window's Mass Actions feature.
* **Parse Date**: Convert the order\_date column from string format to date format using the **Parse date** processor.

**3. Use the Formula Language**

* Add a **Formula** step to compute the dollar value of each t-shirt order:
  + Click the yellow **+ Add a New Step** button.
  + Select **Formula** from the processors library.
  + Enter total as the name of the new output column.
  + In the expression field, type tshirt\_price \* tshirt\_quantity.
  + Click anywhere to see the total column appear.

**4. Rename/Delete Columns**

* After parsing the order\_date column:
  + Delete the original order\_date column.
  + Rename the parsed date column (order\_date\_parsed) to order\_date.

**5. Run the Recipe**

* Once all steps are defined:
  + Click **Run** in the lower-left corner of the page.
  + Wait for the job to complete.
  + Click **Explore dataset orders\_prepared** to view the output dataset.

### Group the Data

# Concept | Group recipe