

Data Cleaning Guide - Learner Instructions

This guide reminds you of key data cleaning steps, such as handling missing values, removing duplicates, fixing inconsistent formats, and aligning column names.

It uses an employee dataset to illustrate examples of common data cleaning tasks.

## **Hypothetical Database: `employee\_data`**

| **EmployeeID** | **Name** | **Department** | **Salary** | **HireDate** | **JobTitle** |
| --- | --- | --- | --- | --- | --- |
| **1** | Alice Johnson | HR | 50000 | 2020-03-01 | HR Manager |
| **2** | Bob Smith | Sales | 55000 | 2021/05/15 | Sales Executive |
| **3** | Charlie Brown | Marketing |  | 2021-06-10 | Marketing Lead |
| **4** | Alice Johnson | HR | 50000 | 2020-03-01 | HR Manager |
| **5** | Eve Davis | Tech Support | 48000 | March 12, 2022 | Support Analyst |
| **6** | Frank Wright | Tech support | 48000 | 2022-03-12 | Support analyst |

## **1. Missing Values**

**Problem**: The Salary column has a missing value for Charlie Brown.

### **Identify Missing Values:**

# Check for missing values  
 print(employee\_data.isnull().sum())

### **Fix Missing Values:**

Consider filling them with:

* The **mean** of the column for a balanced approach.
* The **median**, which is robust against outliers.
* A **fixed** value if there's a logical default, such as 0 or a common price.

# Fill missing Salary with the average salary  
 employee\_data['Salary'] = employee\_data['Salary'].fillna(employee\_data['Salary'].mean())

## **2. Duplicates**

**Problem**: The entry for Alice Johnson appears twice, creating duplicate rows.

### **Identify Duplicates:**

# Check for duplicate rows  
 print(employee\_data.duplicated().sum())  
  
 # View duplicate rows  
 print(employee\_data[employee\_data.duplicated()])

### **Fix Duplicates:**

# Remove duplicate rows  
 employee\_data = employee\_data.drop\_duplicates()

## **3. Inconsistent Formats**

**Identify Mismatched Column Types.**

Use **.dtypes** or **.info()** to Check Data Types

Inspect the data types of all columns in the dataset.

# Check data types of all columns

print(df.dtypes)

# Alternatively, use .info() for an overview of the dataset

print(df.info())

**What to look for:**

* **Numeric columns**: Should be **int64** or **float64**, but may appear as object if values are stored as strings.
* **Date columns**: Should be **datetime64**, but may appear as **object** if stored as strings or in mixed formats.

### **a. Numeric Column: Salary**

**Problem**: Salary should always be numeric, but it contains invalid entries.

#### **Identify and Fix:**

# Convert Salary to numeric and handle invalid entries  
 employee\_data['Salary'] = pd.to\_numeric(employee\_data['Salary'], errors='coerce')  
  
 # Fill invalid entries with the average salary  
 employee\_data['Salary'] = employee\_data['Salary'].fillna(employee\_data['Salary'].mean())

### **b. Date Column: HireDate**

**Problem**: HireDate has inconsistent formats (`YYYY-MM-DD`, `YYYY/MM/DD`, `Month Day, Year`).

#### **Identify and Fix:**

# Convert HireDate'to a consistent datetime format  
 employee\_data['HireDate'] = pd.to\_datetime(employee\_data['HireDate'], errors='coerce')

# Fill invalid dates with a default value  
 employee\_data['HireDate'] = employee\_data['HireDate'].fillna(pd.Timestamp('2020-01-01'))

### **c. Text Columns: Department and JobTitle**

**Problem**: Department and JobTitle contain inconsistent capitalisation and extra spaces.

#### **Identify and Fix:**

# Standardise text columns by stripping spaces and converting to lowercase  
 employee\_data['Department'] = employee\_data['Department'].str.strip().str.lower()  
 employee\_data['JobTitle'] = employee\_data['JobTitle'].str.strip().str.lower()  
  
 # View unique values after cleaning  
 print(employee\_data['Department'].unique())  
 print(employee\_data['JobTitle'].unique())

## **4. Mismatched Columns**

**Problem**: Column names between datasets do not match. For instance, another dataset might use Dept instead of Department.

### **Identify and Fix:**

# Rename mismatched column to match the current dataset  
 employee\_data.rename(columns={'Dept': 'Department'}, inplace=True)

## **Final Cleaned Dataset**

| **EmployeeID** | **Name** | **Department** | **Salary** | **HireDate** | **JobTitle** |
| --- | --- | --- | --- | --- | --- |
| **1** | Alice Johnson | hr | 50000.0 | 2020-03-01 | hr manager |
| **2** | Bob Smith | sales | 55000.0 | 2021-05-15 | sales executive |
| **3** | Charlie Brown | marketing | 51500.0 | 2021-06-10 | marketing lead |
| **5** | Eve Davis | tech support | 48000.0 | 2022-03-12 | support analyst |
| **6** | Frank Wright | tech support | 48000.0 | 2022-03-12 | support analyst |