Absolutely. Below is a **full-length, deeply analytical, and well-structured report** explaining the **entire data cleaning and preprocessing pipeline** implemented in your Python script. Every step is organized by logical sections, and every decision is fully justified, including column-level treatments, code logic, and reasoning.

# Flight Dataset Cleaning & Preprocessing Report

**Prepared for:** Business Intelligence Project **Contributors:** Zehra Ahmed, Farah Inayat, Kisa Fatima, Zuha Aqib **Date:** 18-May-2025 **Dataset:** flights\_sample\_3m.csv

## 1. Introduction

This report documents and analyzes the complete data cleaning and preprocessing workflow applied to the flight dataset. The goal was to transform a raw, inconsistent dataset into a reliable, analysis-ready form for BI visualization and reporting. Every cleaning decision is guided by data-driven logic, domain knowledge, and visual audit.

## 2. Data Loading & Initial Exploration

* The dataset was loaded using pandas.read\_csv().
* The default display setting was updated to show **all columns**.
* Initial statistics included:
  + df.shape – to check row and column count.
  + df.dtypes – to inspect data types.
  + df.isnull().sum() – to assess missing values.

## 3. Data Type Conversion

Several columns were explicitly converted for readability and consistency:

| Column | From | To | Justification |
| --- | --- | --- | --- |
| FL\_DATE | object | datetime | Enables time-based operations |
| AIRLINE, ORIGIN, DEST, CANCELLATION\_CODE, etc. | object | string | Clean categorical comparison and grouping |
| CANCELLED, DIVERTED | float64 | int64 | Logical binary columns |

## 4. Redundant Column Removal

* AIRLINE\_DOT was **dropped** as it is simply a concatenation of AIRLINE and AIRLINE\_CODE.
* Decision based on:
  + Identical record count.
  + No added value.
  + Duplicate representation.

## 5. Manual Audit for Categorical Consistency

For string fields (AIRLINE, ORIGIN, DEST, CANCELLATION\_CODE), a value count was performed to identify:

* Typos
* Capitalization differences
* Unexpected categories

All values were found to be clean — **no manual replacements needed**.

## 6. Missing Values: Strategy by Column Group

### 6.1 Cancelled Flights

Identified using:

df[df['DEP\_TIME'].isna()][['CANCELLED', 'DIVERTED']].value\_counts()

Result: All 77,615 rows with time-related nulls were **cancelled flights**.

**Strategy:** Filled these columns with 0:

* DEP\_TIME, DEP\_DELAY, TAXI\_OUT, WHEELS\_OFF, WHEELS\_ON, TAXI\_IN, ARR\_TIME, ARR\_DELAY
* ELAPSED\_TIME, AIR\_TIME
* Delay reason columns: DELAY\_DUE\_\*

### 6.2 DEP\_DELAY (29 nulls)

Logic:

DEP\_DELAY = DEP\_TIME - CRS\_DEP\_TIME

* Both times are in hhmm; converted to minutes using a helper function.
* 29 missing rows were updated accordingly.

### 6.3 TAXI\_OUT (1,191 nulls)

* All 1,191 rows were **cancelled flights**.
* However, 400 cancelled flights had non-null TAXI\_OUT, indicating taxi started but flight cancelled before takeoff.

Handled with caution:

* Verified values.
* Filled nulls in TAXI\_OUT, WHEELS\_OFF, WHEELS\_ON, ARR\_TIME, TAXI\_IN, etc. with 0 where consistent.

### 6.4 WHEELS\_ON (802 nulls)

* 800 flights were **diverted**.
* 2 flights were **not diverted or cancelled** → considered data errors → dropped.

For diverted\_only:

* All relevant columns were filled with 0, acknowledging rerouted or incomplete journey.

## 7. Derived and Computed Fields

### 7.1 DEP\_DELAY & ARR\_DELAY

Computed using the formula:

minutes = (hhmm // 100) \* 60 + (hhmm % 100)

* DEP\_DELAY: DEP\_TIME - CRS\_DEP\_TIME
* ARR\_DELAY: ARR\_TIME - CRS\_ARR\_TIME

Overnight flights were handled using:

if delay < 0: delay += 1440

### 7.2 CRS\_ELAPSED\_TIME

Same strategy as delays:

CRS\_ARR\_TIME - CRS\_DEP\_TIME

### 7.3 ELAPSED\_TIME

ARR\_TIME - DEP\_TIME

### 7.4 AIR\_TIME

WHEELS\_ON - WHEELS\_OFF

All three accounted for overnight flights and data edge cases.

## 8. Delay Reasons

Columns:

* DELAY\_DUE\_CARRIER
* DELAY\_DUE\_WEATHER
* DELAY\_DUE\_NAS
* DELAY\_DUE\_SECURITY
* DELAY\_DUE\_LATE\_AIRCRAFT

**Strategy:**

* All nulls filled with 0.0 as they indicate *no delay due to that reason*.
* Justified over using -1 or NaN to avoid statistical bias.

## 9. Feature Engineering

### 9.1 Date Decomposition

From FL\_DATE:

* year, quarter, month, day created
* month stored as 3-letter string (e.g., “Jan”)

### 9.2 Binary Categorical Mapping

Mapped binary CANCELLED and DIVERTED to:

* cancelled\_c: 'Yes' / 'No'
* diverted\_c: 'Yes' / 'No'

Improves visualization in BI tools.

## 10. Column Renaming

Renamed for clarity and business relevance:

| Original | New |
| --- | --- |
| FL\_DATE | FLIGHT\_DATE |
| CRS\_DEP\_TIME | SCHEDULED\_DEP\_TIME |
| DEP\_TIME | ACTUAL\_DEP\_TIME |
| CRS\_ARR\_TIME | SCHEDULED\_ARR\_TIME |
| ARR\_TIME | ACTUAL\_ARR\_TIME |
| CRS\_ELAPSED\_TIME | SCHEDULED\_ELAPSED\_TIME |
| ELAPSED\_TIME | ACTUAL\_ELAPSED\_TIME |
| AIR\_TIME | IN\_AIR\_DURATION |
| WHEELS\_OFF | WHEELS\_LEFT\_ORIGIN |
| WHEELS\_ON | WHEELS\_REACHED\_DESTINATION |
| TAXI\_IN | TAXI\_TIME\_DESTINATION |
| TAXI\_OUT | TAXI\_TIME\_ORIGIN |
| FL\_NUMBER | FLIGHT\_NUMBER |

## 11. Final Checks

* df.isnull().sum() → All key columns confirmed clean.
* Data types inspected again.
* Shape validated.

**Output:**

* Saved full cleaned dataset: flights\_sample\_3m\_cleaned.csv
* Saved 1000-row sample: flights\_sample\_1k\_cleaned.csv

## 12. Conclusion

This robust data cleaning pipeline systematically addressed:

* Data types
* Redundant fields
* Time-based derivations
* Domain-specific missing value logic
* Categorical mappings
* Feature enrichments

The final dataset is **clean**, **complete**, and **BI-ready**, supporting high-level KPI tracking, time series analysis, delay reason breakdowns, and cancellation patterns.

Let me know if you’d like a visual summary, data profile plots, or inclusion of correlation matrices as appendices!