**DAB July 2025 Week 8 Assignment – Francky Ciceron**

**Pandas Retail Store Analysis**

1. **Basic Information Retrieval**:
   * How many unique products are in the product catalog?
   * What are the top 5 most expensive products?
   * Which store has the largest floor space?
   * What is the distribution of customers by state?
2. **Removing duplicates:**

Resolving Duplicate “customer\_id” in “customers\_df”

Upon checking for duplicates by “customer\_id”, one duplicate was found:

- Two rows shared the same “customer\_id = 26”, with minor differences in email, phone formatting, address spelling, and state abbreviation.

- After reviewing both, I retained the complete and more standardized row.

- The duplicate was removed using “drop\_duplicates(subset=["customer\_id"], keep="last")”.

1. Discussion questions and answers:

3.1. **What are the key advantages of using Pandas for data cleaning compared to other methods?**

* **Intuitive syntax**: Pandas offers readable, chainable commands that make complex cleaning tasks easier to express.
* **Built-in functions**: Methods like .dropna(), .fillna(), .duplicated(), and .astype() simplify common cleaning operations.
* **Flexible indexing**: Label-based and position-based indexing allow precise targeting of rows and columns.
* **Integration**: Pandas works seamlessly with NumPy, Matplotlib, and other libraries, making it ideal for end-to-end workflows.
* **Performance**: Vectorized operations are faster and more memory-efficient than traditional loops or Excel formulas.

3.2. **How would your approach to handling missing values differ if the missing data was not random but had a pattern or meaning?**

* **Investigate the pattern**: First, I’d explore whether missingness correlates with other variables (e.g., missing income only for certain regions).
* **Avoid blind imputation**: Instead of filling with mean or mode, I’d consider domain-specific logic or conditional imputation.
* **Use flags**: Sometimes it’s better to preserve missingness by creating a binary indicator column (e.g., is\_missing\_income) to retain its informational value.
* **Consult stakeholders**: If the pattern reflects business rules or operational gaps, I’d document it and consult with domain experts before cleaning.

3.3. **What types of data quality issues might not be immediately visible through simple DataFrame inspection methods?**

* **Inconsistent formats**: Dates, phone numbers, or addresses may look fine but vary in structure.
* **Hidden duplicates**: Same entity with slight variations (e.g., “Betty Lewis” vs “B. Lewis”).
* **Outliers**: Extreme values that skew analysis but don’t appear as missing or duplicated.
* **Encoding issues**: Special characters or invisible whitespace can cause parsing errors.
* **Logical inconsistencies**: E.g., a registration date before a birthdate, or negative prices.

3.4. **How would you document your data cleaning process to ensure reproducibility?**

* **Use markdown cells**: Clearly explain each step in Jupyter or VS Code notebooks.
* **Comment code**: Add inline comments to describe logic and decisions.
* **Save intermediate outputs**: Use versioned files or checkpoints to track progress.
* **Create reusable functions**: Encapsulate cleaning steps for consistency across datasets.
* **Log changes**: Print summaries after each transformation (e.g., shape, null counts) to verify impact.

3.5. **In what scenarios might it be better to remove rows with missing values rather than imputing them?**

* **High proportion of missing data**: If most of the row is empty, imputation may be unreliable.
* **Critical fields missing**: If key identifiers or outcome variables are missing, the row may be unusable.
* **Small dataset with minimal loss**: Dropping a few rows may be safer than introducing noise.
* **Non-random missingness with bias risk**: Imputation could distort patterns if missingness is systematic.
* **Downstream model requirements**: Some models (e.g., decision trees) may perform better with clean, complete data.