Module 10: Capstone Project Data Engineering Sandbox

Anaconda Graduation Track • Transforming Raw Data Chaos into Corporate Dashboards

Candidate Analyst Name: _____

Evaluation Date: _____

Section 1: Project Objective & Overview

Welcome to your final graduation challenge. Unlike previous sandbox modules, this capstone drops you into the deep end with an uncurated, unguided, and highly erratic corporate CRM database payload. Your task is to transition from raw structural chaos into an enterprise-ready analytics distribution engine.

> 🎯 **Target Milestones:**
>
> 1. **Scrub & Standardize:** Build data loops to strip junk flags, handle formatting issues, and normalize currencies.
> 2. **Feature Miner:** Extract temporal intervals and synthesize advanced user performance categories.
> 3. **Interactive Showcase:** Output a zero-dependency, self-contained interactive dashboard document.

Section 2: Setup and Chaotic Data Generation

Initialize your Jupyter Notebook environment by running the data generation matrix block below to simulate legacy database corruption cases:

```
import os
import numpy as np
import pandas as pd
import plotly.express as px

# Establish synthetic chaotic database seed
np.random.seed(101)
raw_dirty_records = {
    'Client_ID': ['C_991 ', 'C_992', 'C_991', 'C_994 ', 'C_995'],
    'System_Join_Date': ['2026-01-15', '18/02/26', '2026-01-15', '2026-03-01', '12-
May-2026'],
    'Raw_Revenue_Metric': ['$45,200.00', '€12000', '$14,800.00', '   $9,150.50', 'CAD
31000!!'],
    'Segment': ['Enterprise ', 'retail', 'ENTERPRISE', 'Retail', '  enterprise']
}

df_capstone = pd.DataFrame(raw_dirty_records)
print("--- [ALERT] Incoming Uncurated Corporate Payload Dataframe ---")
print(df_capstone)
```

Section 3: Guided Cleaning Pipeline Execution

Execute the data cleansing functions sequentially inside your workspace to standardize string noise, repair chronological profiles, and resolve categorical duplication splits:

```
# Phase A: Reconcile categorical splits and clean structural string padding
df_capstone['Segment'] = df_capstone['Segment'].str.strip().str.upper()

# Phase B: Reconcile inconsistent mixed timeline structures
df_capstone['Clean_Join_Date'] = pd.to_datetime(df_capstone['System_Join_Date'],
format='mixed', dayfirst=True)

# Phase C: Strip non-numeric artifacts to restore revenue metric floats
df_capstone['Clean_Revenue_USD'] =
df_capstone['Raw_Revenue_Metric'].str.replace(r'[^0-9.]', '', regex=True).astype(float)

print("\n--- Cleaned Structural Data State ---")
print(df_capstone[['Client_ID', 'Clean_Join_Date', 'Clean_Revenue_USD', 'Segment']])
```

Section 4: Technical Deliverable Validation

Execute the code framework blocks inside your Jupyter space, analyze the state transformations, and record your evaluations.

Question 1: Structural Inconsistency Detection

Review the raw dataframe output. Detail how many separate duplicate records exist for client code `C_991`, and list the structural string variables causing them to split into separate rows before cleaning:

Question 2: Pipeline Mechanics (Datatype Integrity)

What specific processing problem would occur if an analyst immediately called a data visualization method like `px.histogram()` on the original `Raw_Revenue_Metric` feature column without using regex numeric expressions to strip out special symbols and whitespace artifacts first?

Question 3: Feature Engineering Strategy

Using your running runtime environment, explain what custom temporal categories can be derived from the newly generated `Clean_Join_Date` timestamp column to track customer retention curves (e.g., extracting year, month name, or operational financial quarter tracking fields):

Section 5: Capstone Independent Dashboard Challenge

> 🚀 **Final Capstone Requirement:** Build and export a responsive, production-ready interactive dashboard visualization.
> Write a script below that constructs an interactive bubble scatter plot (`px.scatter`) comparing `Clean_Join_Date` against `Clean_Revenue_USD`, using `Segment` as a color variable mapping flag.
> **Export Constraint:** You must automatically output the completed workspace asset to the local file system as a self-contained, zero-dependency HTML dashboard named `"capstone_executive_showcase.html"`.

Write Your Capstone Solution Script Below:

```
# Provide your complete interactive visualization and HTML exporting code logic below:
```

Capstone Milestone Exit Reference

- `df.duplicated() / df.drop_duplicates()` → Flags and eliminates overlapping data records to preserve analytical integrity.
- `fig.write_html("output.html")` → Compiles complete frontend rendering dependencies into a portable corporate file format.