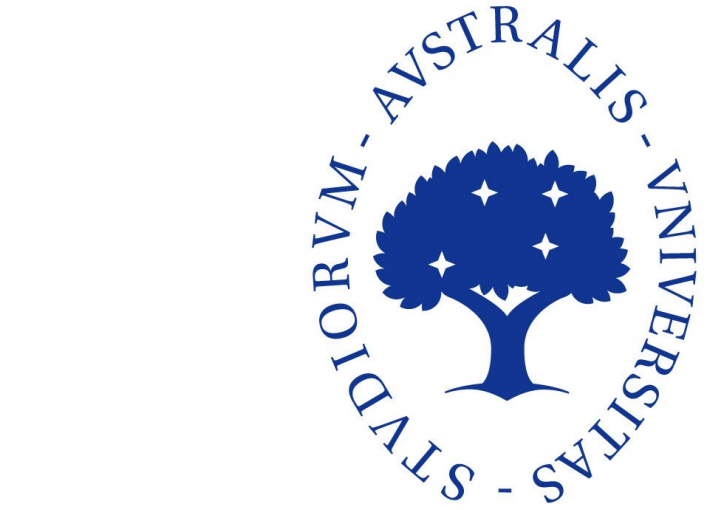

UNIVERSIDAD

AUSTRAL

Maestría en Ciencia de Datos

Materia: Introducción a Data Warehousing

Alumnos: Buonincontro, Brenda

Maslaton, Carlos

Maslaton, Mariano

Molteni, María Pía

Ton Vanerio, Nicolás

Link al repositorio: https://github.com/cmaslaton/tp_datawarehousing

El proyecto se estructura en torno a un script principal, **main.py**, que cumple el rol de orquestador del flujo completo. Este script ejecuta de manera secuencial todos los pasos definidos, desde la etapa inicial hasta la generación final de productos de datos, garantizando la coherencia e integridad del proceso.

El flujo de trabajo se divide en etapas específicas, cada una encapsulada en scripts individuales que van desde step_01 hasta step_10_3. Las primeras etapas están orientadas a la adquisición y preparación de los datos: se crea la base de datos, las tablas temporales y se define la estructura de la metadata. Luego, se cargan los archivos CSV provenientes de la primera ingesta (Ingesta1) y se trasladan a la capa de ingesta, tras lo cual se realiza un enriquecimiento con fuentes externas, como el archivo world_data.csv:

- **step_01**: Crea la base de datos, las tablas temporales (prefijo TMP_) y estructura la Metadata.
- **step_02**: Carga los archivos CSV de Ingesta1 en las tablas temporales.
- **step_03**: Persiste los datos desde la capa temporal hacia las tablas de ingesta (ING_).
- **step_04**: Integra y enriquece los datos con información externa (world_data.csv).

Posteriormente, se construye en **step_05** el modelo dimensional, con la generación de dimensiones y tablas de hechos (DWA_FACT_), incorporando tanto una capa de memoria (DWM_) como una de enriquecimiento, que permite derivar nuevos campos a partir de los datos originales.

Una vez estructurado el modelo, con **step_06** se implementan mecanismos de control de calidad. Esto se realiza mediante la ejecución de un conjunto de reglas definidas en la capa DQM_, que incluyen verificaciones, indicadores e informes descriptivos sobre los datos. Superada esta instancia, se habilita la carga inicial al Data Warehouse, trasladando los datos validados desde la capa de ingesta con el script **step_07**.

El proceso contempla además una segunda ingesta de datos (Ingesta2), que incluye nuevos archivos CSV, la corrección de errores detectados y la actualización de las distintas capas: DWA_, DWM_, enriquecimiento, DQM_ y Metadata (**step_08, step_08b y step_09**).

Finalmente, el flujo culmina con la generación de productos de datos que sintetizan la información procesada. Se desarrollaron tres tableros: un análisis de ventas por país y categoría (DP1), un reporte de desempeño de empleados por trimestre (DP2) y un análisis logístico vinculado al desempeño de los shippers (DP3).

Todos los scripts interactúan con una única base SQLite (**tp_dwa.db**). Dentro de esa base, las tablas están organizadas en capas:

Capa	Prefijo	Función
Temporal	TMP_	staging crudo desde CSV
Ingesta	ING_	datos limpios, listos para el DWA
Modelo DWH	DWA_	modelo dimensional (dimensiones + hechos)
Metadata	MET_	registra todas las tablas del sistema
Calidad	DQM_	guarda controles, métricas, ejecuciones
Memoria y enriquecimiento	(en DWA_)	historial y columnas derivadas
Productos de datos	DPx_	resultados finales para análisis y dashboards

Una vez presentada la estructura general del programa y el funcionamiento del flujo completo, se expone a continuación la resolución de cada punto del trabajo práctico, siguiendo el orden establecido en el enunciado. Para cada consigna se indica el script correspondiente y el criterio aplicado.

2) Comparar la estructura de las tablas y el modelo de entidad relación. Adecuar si fuera necesario. Definir y crear las FOREIGN-KEYS necesarias para verificar la integridad referencial.

Este punto es resuelto específicamente en step_05_create_dwh_model.py, donde se crean relaciones entre dimensiones y hechos usando claves sustitutas (sk_*) y claves naturales (nk_*) cuando corresponde.

Se definen FOREIGN KEYs entre DWA_FACT_Ventas y dimensiones como DWA_DIM_Cliente, DWA_DIM_Producto, DWA_DIM_Tiempo, etc.

3) Considerar también la tabla de países (World-Data-2023) y vincularla con las tablas que correspondan.

Se carga world_data.csv y se vincula con ING_Geografia, agregando datos externos como PBI, densidad y esperanza de vida por país en el step_04_link_world_data.py.

4) Crear un área temporal y persistir el modelo relacional obtenido con los datos de los .CSV.

Se crea el área TMP_ para staging (step_01_setup_staging_area.py), y se cargan los datos originales desde Ingesta1 (step_02_load_staging_data.py), como modelo relacional sin transformar.

5) Crear el soporte para la Metadata y utilizarlo para describir las entidades.

Se crea la tabla MET_entidades también en step_01_setup_staging_area.py y en cada paso se registra cada nueva entidad (tabla creada) con nombre, descripción y tipo de capa (TMP, ING, DWA, DQM, etc.).

6) Definir y crear el Modelo Dimensional del DWA y documentarlo en la Metadata. Debe incluir una capa de Memoria y una de Enriquecimiento (datos derivados).

Con el script llamado step_05_create_dwh_model.py se crean las dimensiones DWA_DIM_Cliente, DWA_DIM_Producto, etc., y los hechos DWA_FACT_Ventas.

La capa de memoria está incluida con SCD Tipo 2 (ej. empleados con historial de dirección) y la capa de enriquecimiento incluye columnas derivadas como edad_al_contratar o monto_total. Además, se documenta cada entidad en MET_entidades.

7) Diseñar y crear el DQM para poder persistir los procesos ejecutados sobre el DWA, los descriptivos de cada entidad procesada y los indicadores de calidad. Documentar el diseño en la Metadata.

Se crean los DQMs:

- DQM_ejecucion_procesos
- DQM_indicadores_calidad
- DQM_descriptivos_entidad

Y se insertan registros en MET_entidades para documentar las tablas del DQM (step_06_create_dqm.py).

8) Realizar la carga inicial del DWA con los datos que se seleccionen de las tablas recibidas y procesadas.

Este punto es resuelto en step_07_initial_dwh_load.py, donde se cargan datos desde ING_ a DWA_, transformando y creando SK y se calcula monto_total, se cargan dimensiones y hechos.

a) Definir los controles de calidad de ingesta para cada tabla, los datos que se persistirán en el DQM y los indicadores y límites para aceptar o rechazar los datasets. Realizar y ejecutar los scripts correspondientes. Tener en cuenta: *outliers*, datos faltantes, valores que no respetan los formatos, etc.

Nuevamente, el script step_06_create_dqm.py es el encargado de registrar los porcentajes de nulos, formatos erróneos, etc., y se define si los datos se aceptan o no según umbrales.

b) Definir los controles de calidad de integración para el conjunto de tablas, los datos que se persistirán en el DQM y los indicadores y límites para aceptar o rechazar los datasets. Realizar y ejecutar los scripts correspondientes. Tener en cuenta: la integridad referencial e indicadores de comparación.

Con los scripts step_06_create_dqm.py y step_07_initial_dwh_load.py se valida integridad referencial entre tablas cargadas al DWA y si hay errores, no se carga.

c) Ingestar los datos de Ingesta1 en el DWA definido. Los datos se deben insertar desde las tablas temporales creadas. Actualizar todas las capas. Siempre y cuando se superen los umbrales de calidad.

Este punto también se resuelve en step_07_initial_dwh_load.py, donde se ingesta desde TMP_ → ING_ → DWA_ y se actualizan dimensiones, hechos, metadata y DQM si se superan umbrales.

9) Actualización:

a) Persistir en área temporal las tablas entregadas como Ingesta2.

Se cargan nuevos archivos en TMP_ con step_08_load_ingesta2_to_staging.py.

b) Repetir los pasos definidos para Ingesta1 que sean adecuados para Ingesta2.

Se hacen controles de calidad, transformaciones, y se decide si se ingesta con step_08b_data_remediation.py y step_09_update_dwh_with_ingesta2.py.

El sistema repite el flujo completo, pero además incluye un bloque específico para detección y remediación de errores antes de la carga al DWA.

c) Considerar altas, bajas y modificaciones. Tener en cuenta el orden de prevalencia para las actualizaciones.

La lógica de SCD Tipo 2 se aplica para preservar el historial en dimensiones como empleados y clientes (step_09_update_dwh_with_ingesta2.py).

d) Si hubiera errores se debe decidir si se cancela toda la actualización, se procesa en parte o en su totalidad. Lo que suceda debe quedar registrado en el DQM.

Para abordar errores e inconsistencias en los datos recibidos durante Ingesta2, se diseñó un sistema de remediación automatizado dividido en cuatro fases. Cada una resuelve un tipo específico de problema y deja trazabilidad en el DQM. A continuación, se describen las estrategias implementadas y los resultados obtenidos:

Fase 1: Corrección de lógica temporal en dimensiones con SCD Tipo 2

Se abordaron posibles registros con inconsistencias en los campos de validez temporal, específicamente cuando la fecha de inicio superaba a la de fin. Se aplicó una estrategia

de detección automática y, en caso de error, se ajustó la fecha de fin a un día posterior a la fecha de inicio. En este caso, no se detectaron problemas: el modelo inicial ya era consistente.

Fase 2: Resolución de regiones faltantes en clientes

Se detectaron registros en TMP2_customers sin región asignada. Se aplicó un mapeo automático de país a región utilizando un diccionario base, y se actualizó la información tanto en la tabla temporal como, en caso de corresponder, en la dimensión DWA_DIM_Clientes. Dos clientes fueron corregidos: ALFKI (Germany → Western Europe) y ANATR (Mexico → North America).

Fase 3: Compleción de datos de envío

Se encontraron órdenes con campos faltantes como ship_region, ship_postal_code y shipped_date. Se completaron los valores utilizando herencia desde los datos del cliente (por ejemplo, asignando la región del cliente al campo ship_region). Las fechas de envío nulas fueron marcadas como "Pending Shipment". En total, se procesaron 25 órdenes.

Fase 4: Remediación geográfica avanzada

Se resolvieron valores nulos en campos geográficos a través de un motor de enriquecimiento que combinó múltiples fuentes. Se mapearon 89 países a 7 regiones y se aplicó fuzzy matching para nombres de países con variaciones ortográficas. Se utilizó world_data_2023 como fuente principal de referencia, y se propagó la información de manera inteligente hacia los campos de envío en base a la relación con los clientes.

e) Se debe considerar además la capa de Memoria para persistir la historia de los campos que han sido modificados.

Incluida en dimensiones con SCD Tipo 2 (empleados, clientes).

f) Se debe considerar además actualizar la capa de Enriquecimiento para persistir los datos derivados que se vean afectados.

Se recalculan columnas como monto_total, edad_al_contratar si cambian.

g) Desarrollar y ejecutar los scripts correspondientes para actualizar el DWA con los nuevos datos.

step_09_update_dwh_with_ingesta2.py aplica lógica de carga incremental.

h) Actualizar DQM si fuera necesario.

Se registran nuevas ejecuciones e indicadores si corresponde.

i) Actualizar Metadata si fuera necesario.

Se agregan entradas si cambian entidades.

10) Publicar un producto de datos resultante del DWA para un caso de negocio particular y un período dado si corresponde.

Cada uno de los siguientes archivos genera un DPx_ que se registra en Metadata y en DQM:

- step_10_1_ventas_mensuales_categoria_pais.py
- step_10_2_performance_empleados_trimestral.py
- step_10_3_analisis_logistica_shippers.py

También se calcula y persiste cada tabla.

11) Explotación.

a) Desarrollar y publicar un tablero para la visualización del producto de datos desarrollado. Dejar huella en el DQM y en Metadata de ser necesario.

b) Desarrollar y publicar un tablero de visualización que permita navegar por los datos persistidos en el DQM. Dejar huella en el DQM y en Metadata de ser necesario.

La resolución de ambos puntos será presentada en clase con los tableros dash_productos.pbix y dash_dqm.pbix