

Caso IATA - Gestión de Datos

Información del Proyecto

Maestría: Ciencia de Datos
Universidad: Pontificia Universidad Javeriana
Materia: Gestión de Datos
Estudiantes: Edwin Silva Salas, Carlos Preciado Cárdenas, Cristian Restrepo Zapata
Fecha de inicio: Noviembre 2025



Descripción del proyecto

Este proyecto corresponde al **Caso IATA** de la materia Gestión de Datos, en el cual se trabaja con una base de datos relacional de un sistema de reservas y vuelos de aerolíneas. El objetivo principal es realizar operaciones de gestión de datos, análisis y potencialmente crear un **Data Mart** para facilitar consultas analíticas sobre el negocio de vuelos.

Objetivos del Proyecto

- 1. **Configurar la base de datos transaccional IATA** con las tablas y datos proporcionados
- 2. **Realizar consultas y análisis** sobre los datos de vuelos, usuarios, aerolíneas e itinerarios
- 3. **Diseñar e implementar un Data Mart dimensional** (Star Schema) para análisis de negocio
- 4. **Generar reportes y métricas** clave del negocio de aerolíneas
- 5. **Visualizar la información** interactiva y dinámica para la toma de decisiones

Modelo de Datos - Base Transaccional (OLTP)

La base de datos IATA contiene las siguientes entidades:

Tablas Principales

Tabla	Descripción	Registros
AEROLINEAS	Aerolíneas disponibles (Avianca, Latam, Wingo)	3
MODELOS	Modelos de aviones (Airbus 320, Boeing 747)	2
AVIONES	Flota de aviones de cada aerolínea	10
CIUDADES	Ciudades origen/destino de los vuelos	12
AEROPUERTOS	Aeropuertos internacionales	12
ITINERARIOS	Rutas y horarios de vuelos	35
USUARIOS	Pasajeros registrados en el sistema	20
VUELOS	Reservas y compras de vuelos (Fact table)	170

Proceso de Implementación (OLTP)

Fase 1: Preparación del Entorno

- ☒ Configurar conexión a Oracle Database local del estudiante
- ☒ Crear usuario IATA en la base de datos local
- ☒ Otorgar privilegios básicos al usuario IATA
- ☒ Preparar scripts de eliminación y verificación

Detalles de Configuración Inicial

Como primera actividad del caso, se procedió a crear el **usuario IATA** en la base de datos Oracle local del computador del estudiante. Esta decisión se tomó debido a que el script **IATA.sql** proporcionado como material de la actividad utiliza la nomenclatura **IATA.OBJECT** para todos los objetos de base de datos (tablas, constraints, índices, etc.).

Pasos ejecutados:

- 1. **Conexión como usuario SYS:** Se estableció conexión a la base de datos local utilizando el usuario administrador SYS con rol SYSDBA
- 2. **Creación del usuario IATA:** Se ejecutó el comando **CREATE USER IATA** con contraseña segura
- 3. **Asignación de privilegios:** Se otorgaron los privilegios básicos necesarios:
 - o **CONNECT:** Para conectarse a la base de datos
 - o **RESOURCE:** Para crear objetos (tablas, vistas, secuencias, etc.)
 - o **QUOTA UNLIMITED ON USERS:** Para utilizar espacio ilimitado en el tablespace USERS

```
-- Comandos ejecutados como SYS
CREATE USER IATA IDENTIFIED BY "passIATA123";
GRANT CONNECT, RESOURCE TO IATA;
ALTER USER IATA QUOTA UNLIMITED ON USERS;
```

Ver detalles completos del script de creación: [Creacion_IATA_OLTP.sql](#)

Conexión al Esquema IATA

Una vez completada la configuración del usuario IATA, se estableció una nueva conexión a la base de datos utilizando las credenciales del usuario IATA. Esta conexión permite trabajar directamente sobre el esquema transaccional, garantizando que todos los objetos de base de datos se creen bajo el propietario correcto.

Datos de conexión utilizados:

- **Username:** IATA
- **Password:** passIATA123
- **Role:** Default (ninguno, a diferencia de SYS que requiere SYSDBA)
- **Hostname:** localhost
- **Port:** 1521
- **Service Name:** FREEPDB1

Con la conexión establecida como usuario IATA, se procede a la siguiente fase: la carga de datos transaccionales mediante la ejecución del script proporcionado.

Fase 2: Carga de Datos Fuente

- ☒ Ejecutar script [IATA.sql](#) para crear las tablas transaccionales
- ☒ Verificar la carga de datos (170 vuelos, 20 usuarios, 35 itinerarios)
- ☒ Validar integridad referencial y constraints

Ejecución del Script IATA.sql

Posterior a la creación del usuario IATA, se procedió a ejecutar el script [IATA.sql](#) proporcionado como insumo de la actividad. Este script contiene:

- **8 tablas del modelo relacional:** AEROLINEAS, AEROPUERTOS, AVIONES, CIUDADES, ITINERARIOS, MODELOS, USUARIOS, VUELOS
- **Insertión de datos de prueba:** 170 registros de vuelos, 20 usuarios, 35 itinerarios, 12 ciudades, entre otros
- **Constraints y Foreign Keys:** Definición completa de integridad referencial
- **Índices únicos:** Para optimización de consultas

Resultado de la ejecución: El script se ejecutó **con cero errores**, creando exitosamente toda la estructura de base de datos y cargando los datos en el schema IATA

Proceso de Implementación (OLAP)

Fase 1: Preparación del Entorno OLAP

- ☒ Crear esquema IATA_OLAP en la base de datos
- ☒ Otorgar privilegios básicos al esquema OLAP
- ☒ Configurar permisos de lectura sobre esquema OLTP (IATA)
- ☒ Validar conectividad entre esquemas

Detalles de Creación del Esquema OLAP

Para implementar una arquitectura de datos robusta y siguiendo las mejores prácticas de separación entre sistemas transaccionales (OLTP) y analíticos (OLAP), se procedió a crear un **esquema independiente llamado IATA_OLAP** que albergará el Data Mart dimensional.

Justificación arquitectónica:

La separación de esquemas OLTP y OLAP permite:

- **Independencia operacional:** Las consultas analíticas no afectan el rendimiento transaccional
- **Seguridad por capas:** Control de acceso diferenciado entre operaciones y análisis
- **Optimización específica:** Cada esquema puede optimizarse para su propósito (escritura vs lectura)
- **Mantenibilidad:** Cambios en el modelo dimensional no impactan el sistema operacional

Pasos ejecutados (ver script completo en [Creacion_IATA_OLAP.sql](#)):

1. **Conexión como usuario SYS:** Se estableció conexión con privilegios SYSDBA
2. **Creación del esquema IATA_OLAP:**

```
CREATE USER IATA_OLAP IDENTIFIED BY "passIATAOLAP123";
```

3. Asignación de privilegios básicos:

```
GRANT CONNECT, RESOURCE TO IATA_OLAP;  
ALTER USER IATA_OLAP QUOTA UNLIMITED ON USERS;  
GRANT CREATE MATERIALIZED VIEW TO IATA_OLAP;
```

- **CONNECT**: Permite conectarse a la base de datos
- **RESOURCE**: Permite crear objetos dimensionales (tablas, vistas, procedimientos)
- **QUOTA UNLIMITED**: Espacio ilimitado en tablespace USERS
- **CREATE MATERIALIZED VIEW**: Para crear vistas materializadas (opcional para optimización)

4. Configuración de permisos de lectura sobre IATA (OLTP):

```
GRANT SELECT ON IATA.AEROLINEAS TO IATA_OLAP;  
GRANT SELECT ON IATA.AEROPUERTOS TO IATA_OLAP;  
GRANT SELECT ON IATA.AVIONES TO IATA_OLAP;  
GRANT SELECT ON IATA.CIUDADES TO IATA_OLAP;  
GRANT SELECT ON IATA.ITINERARIOS TO IATA_OLAP;  
GRANT SELECT ON IATA.MODELOS TO IATA_OLAP;  
GRANT SELECT ON IATA.USUARIOS TO IATA_OLAP;  
GRANT SELECT ON IATA.VUELOS TO IATA_OLAP;
```

Estos permisos son esenciales para que el proceso ETL pueda extraer datos del sistema transaccional.

Verificación de permisos otorgados:

```
SELECT  
  grantee,  
  owner,  
  table_name,  
  privilege  
FROM dba_tab_privs  
WHERE grantee = 'IATA_OLAP'  
ORDER BY table_name;
```

Resultado de la consulta:

GRANTEE	OWNER	TABLE_NAME	PRIVILEGE
IATA_OLAP	IATA	AEROLINEAS	SELECT
IATA_OLAP	IATA	AEROPUERTOS	SELECT
IATA_OLAP	IATA	AVIONES	SELECT
IATA_OLAP	IATA	CIUDADES	SELECT
IATA_OLAP	IATA	ITINERARIOS	SELECT
IATA_OLAP	IATA	MODELOS	SELECT
IATA_OLAP	IATA	USUARIOS	SELECT
IATA_OLAP	IATA	VUELOS	SELECT

Se confirmó que el usuario IATA_OLAP tiene permisos SELECT sobre las 8 tablas del esquema IATA.

5. Verificación de la configuración:

- Se validó la creación del usuario/esquema
- Se verificaron los privilegios otorgados
- Se comprobaron las cuotas asignadas en tablespace

Arquitectura resultante:



<div><div><div>- CIUDADES</div><div>- ITINERARIOS</div><div>- MODELOS</div><div>- USUARIOS</div><div>- VUELOS</div></div><div>170 registros</div></div>	<div><div><div>- DIM_AEROLINEA</div><div>- DIM_MODELO</div><div>- DIM_CLIENTE</div><div>- FACT_VENTAS_VUELOS</div></div><div>Pendiente:</div><div>- Carga ETL</div></div> <div>Estructura completa</div>
---------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Resultado: El esquema IATA_OLAP se creó exitosamente con todos los privilegios necesarios y permisos de lectura configurados sobre las tablas del esquema IATA (OLTP).

Ver detalles completos del script de creación: [Creacion_IATA_OLAP.sql](#)

Conexión al Esquema IATA_OLAP

Una vez completada la configuración del esquema IATA_OLAP (ver script anterior), se procedió a establecer una nueva conexión a la base de datos utilizando las credenciales del usuario IATA_OLAP. Esta conexión permite trabajar directamente sobre el esquema analítico, garantizando que todos los objetos dimensionales (tablas de dimensiones y hechos) se creen bajo el propietario correcto.

Datos de conexión utilizados:

- **Username:** IATA_OLAP
- **Password:** passIATAOLAP123
- **Role:** Default (ninguno, a diferencia de SYS que requiere SYSDBA)
- **Hostname:** localhost
- **Port:** 1521
- **Service Name:** FREEPDB1

Con la conexión establecida como usuario IATA_OLAP, se procede a la siguiente fase: el diseño e implementación del **Data Mart dimensional** utilizando el modelo de **Esquema en Estrella (Star Schema)**, que permitirá realizar análisis multidimensionales sobre las operaciones de vuelos de la aerolínea.

Fase 2: Diseño del Data Mart

- ☒ Diseñar modelo estrella (5 dimensiones + 1 tabla de hechos)
- ☒ Crear tablas de dimensiones y hechos

Diseño del Modelo Estrella (Star Schema)

Se ha diseñado e implementado un modelo dimensional tipo **Esquema en Estrella** para el Data Mart de análisis de ventas de vuelos. Este modelo se encuentra documentado y ejecutable en el script [Creacion_DATA_MART.sql](#).

Componentes del modelo:

Dimensiones (5):

- **DIM_TIEMPO:** Análisis temporal (fecha, año, semestre, trimestre, mes, día)
- **DIM_RUTA:** Análisis geográfico (ciudad origen, ciudad destino, aeropuertos)
- **DIM_AEROLINEA:** Análisis por aerolínea
- **DIM_MODELO:** Análisis por modelo de avión (Airbus 320, Boeing 747)
- **DIM_CLIENTE:** Análisis demográfico de pasajeros (ciudad de residencia)

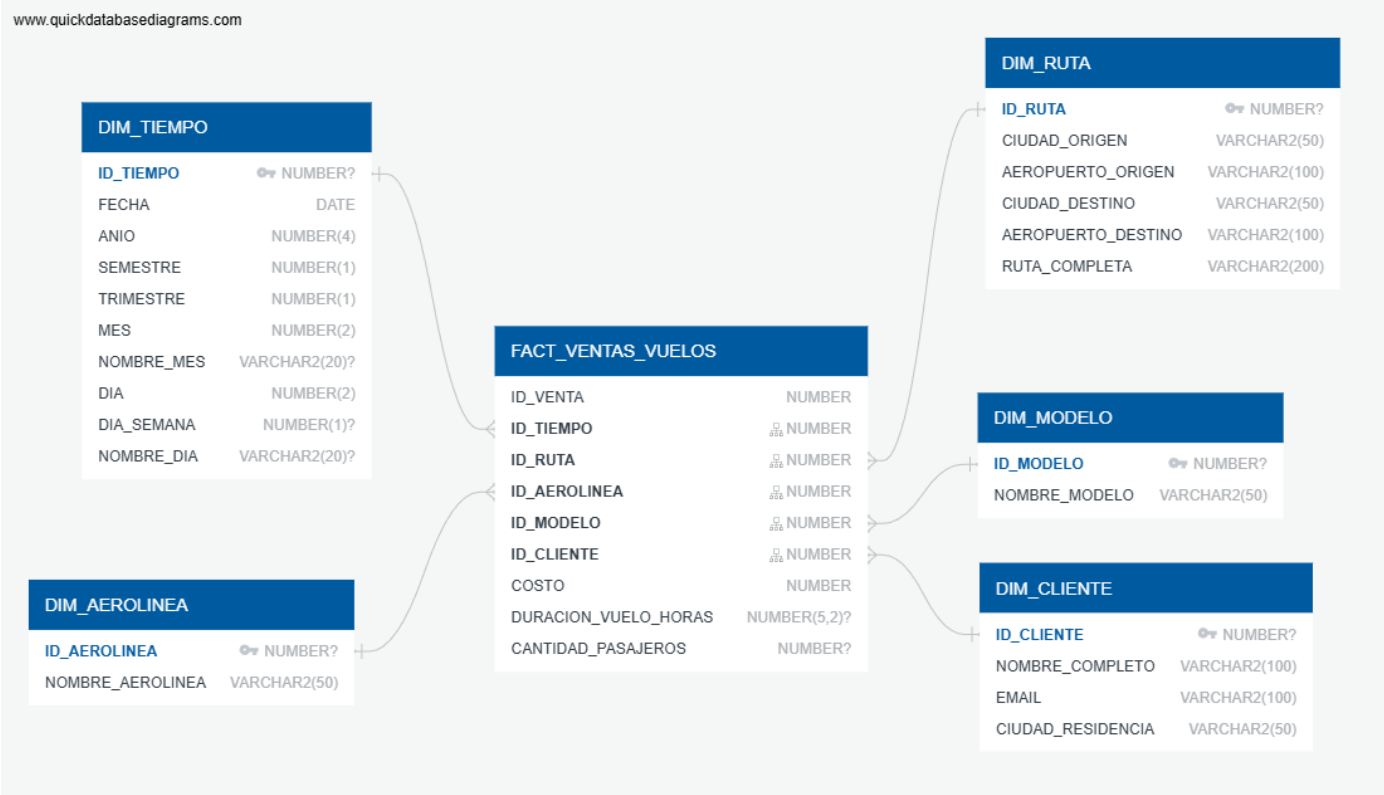
Tabla de Hechos (1):

- **FACT_VENTAS_VUELOS:** Métricas de negocio (costo, duración, cantidad de pasajeros) con claves foráneas a las 5 dimensiones

El diseño contempla las 4 preguntas analíticas clave del caso:

1. Aerolíneas con más vuelos a ciudades específicas por año
2. Recaudación por aerolínea por semestre
3. Modelos de avión con más vuelos por año
4. Ciudades cuyos habitantes viajaron más por año

Diagrama del Modelo Estrella:



Ejecución del Script de Creación

Una vez conectados como usuario **IATA_OLAP**, se ejecutó el script **Creacion_DATA_MART.sql** que contiene todas las sentencias DDL para crear la estructura dimensional del Data Mart.

Objetos creados:

Tabla	Tipo	Estado	Propósito
DIM_TIEMPO	DIMENSION	Creada	Dimensión temporal para análisis por fecha, año, semestre, mes
DIM_RUTA	DIMENSION	Creada	Dimensión geográfica con origen-destino de vuelos
DIM_AEROLINEA	DIMENSION	Creada	Dimensión de aerolíneas (Avianca, Latam, Wingo)
DIM_MODELO	DIMENSION	Creada	Dimensión de modelos de avión (Airbus 320, Boeing 747)
DIM_CLIENTE	DIMENSION	Creada	Dimensión de clientes/pasajeros con ciudad de residencia
FACT_VENTAS_VUELOS	FACT	Creada	Tabla de hechos con métricas de ventas y FKs a 5 dimensiones

```
=====
Data Mart (Esquema Estrella) creado exitosamente
Tablas de dimensiones y hechos listas para carga ETL
=====
```

El Data Mart se creó exitosamente con todas las tablas dimensionales y de hechos. El modelo está estructuralmente completo y listo para iniciar el **proceso ETL** de carga de datos desde el esquema transaccional IATA (OLTP).

Fase 3: Proceso ETL

- ☒ Diseñar queries de extracción desde esquema IATA (OLTP)
- ☐ Implementar transformaciones de datos
- ☐ Cargar dimensiones (DIM_TIEMPO, DIM_RUTA, DIM_AEROLINEA, DIM_MODELO, DIM_CLIENTE)
- ☐ Cargar tabla de hechos (FACT_VENTAS_VUELOS)
- ☐ Validar integridad referencial y calidad de datos

Diseño de Queries de Extracción (ETL)

Se han diseñado las consultas SQL de extracción, transformación y carga (ETL) para poblar todas las tablas del Data Mart desde el esquema transaccional IATA (OLTP). Estas queries se encuentran documentadas en el archivo **CodigosETL-Input_Table.sql**.

Herramienta ETL utilizada: **Pentaho Data Integrator (PDI/Kettle)**

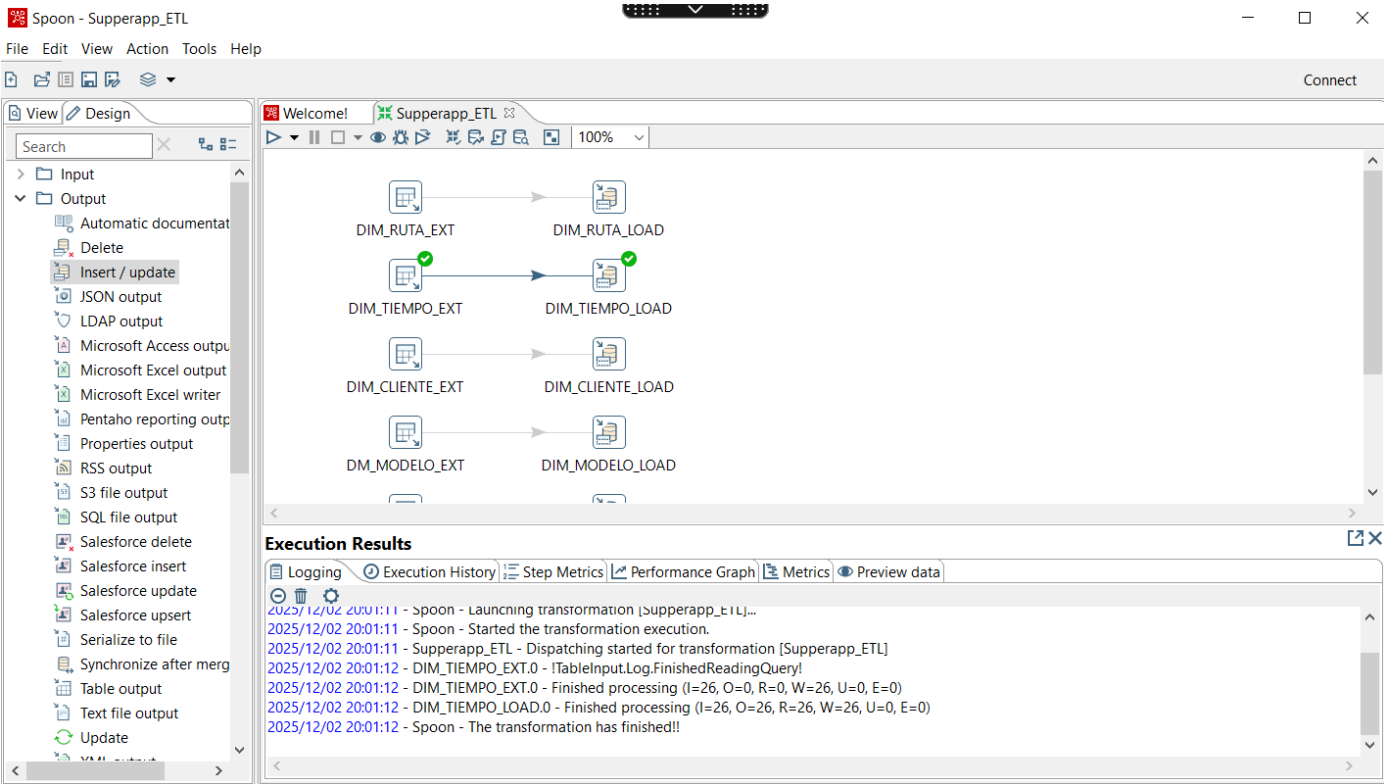
Los queries de extracción fueron implementados en **Pentaho Data Integrator**, una herramienta visual de ETL que permite diseñar flujos de integración de datos mediante transformaciones gráficas. Cada query se configuró como un componente "Table Input" dentro de una transformación PDI, facilitando la carga automatizada desde el esquema OLTP (IATA) hacia el Data Mart OLAP (IATA_OLAP).

Las capturas de pantalla de cada transformación ETL implementada en Pentaho se encuentran en la carpeta `/images/` del repositorio y se muestran a continuación para cada tabla dimensional y de hechos.

Queries de extracción por tabla:

1. DIM_TIEMPO - Extracción de dimensión temporal

Implementación en Pentaho Data Integrator:



Query de extracción:

```
SELECT
  ROW_NUMBER() OVER (ORDER BY f)          AS ID_TIEMPO,
  f                                         AS FECHA,
  EXTRACT(YEAR FROM f)                     AS ANIO,
  CASE
    WHEN EXTRACT(MONTH FROM f) BETWEEN 1 AND 6 THEN 1
    ELSE 2
  END                                     AS SEMESTRE,
  TO_NUMBER(TO_CHAR(f, 'Q'))               AS TRIMESTRE,
  EXTRACT(MONTH FROM f)                   AS MES,
  TO_CHAR(f, 'fmMonth', 'NLS_DATE_LANGUAGE=SPANISH') AS NOMBRE_MES,
  EXTRACT(DAY FROM f)                     AS DIA,
  TO_NUMBER(TO_CHAR(f, 'D'))               AS DIA_SEMANA,
  TO_CHAR(f, 'fmDay', 'NLS_DATE_LANGUAGE=SPANISH') AS NOMBRE_DIA
FROM (
  SELECT DISTINCT TRUNC(fecha_salida) AS f FROM IATA.ITINERARIOS
  UNION
  SELECT DISTINCT TRUNC(fecha_llegada) AS f FROM IATA.ITINERARIOS
)
ORDER BY f;
```

Resultado de carga (primeros 10 registros):

ID_TIEMPO	FECHA	ANIO	SEMESTRE	TRIMESTRE	MES	NOMBRE_MES	DIA	DIA_SEMANA	NOMBRE_DIA
1	2019-01-15	2019	1	1	1	Enero	15	3	Martes

ID_TIEMPO	FECHA	ANIO	SEMESTRE	TRIMESTRE	MES	NOMBRE_MES	DIA	DIA_SEMANA	NOMBRE_DIA
2	2019-02-20	2019	1	1	2	Febrero	20	4	Miércoles
3	2019-03-10	2019	1	1	3	Marzo	10	1	Domingo
4	2019-04-05	2019	1	2	4	Abril	5	6	Viernes
5	2019-05-12	2019	1	2	5	Mayo	12	1	Domingo
6	2019-06-18	2019	1	2	6	Junio	18	3	Martes
7	2019-07-22	2019	2	3	7	Julio	22	2	Lunes
8	2019-08-30	2019	2	3	8	Agosto	30	6	Viernes
9	2019-09-14	2019	2	3	9	Septiembre	14	7	Sábado
10	2019-10-25	2019	2	4	10	Octubre	25	6	Viernes

Descripción: Extrae todas las fechas únicas de salida y llegada de vuelos desde **IATA.ITINERARIOS**, generando automáticamente los atributos temporales (año, semestre, trimestre, mes, día, nombres en español).

2. DIM_RUTA - Extracción de dimensión geográfica

Implementación en Pentaho Data Integrator:

Spoon - Supperapp_ETL

File Edit View Action Tools Help

View Design

Search

Input

Output

Automatic documentat

Delete

Insert / update

JSON output

LDAP output

Microsoft Access outpu

Microsoft Excel output

Microsoft Excel writer

Pentaho reporting outp

Properties output

RSS output

S3 file output

SQL file output

Salesforce delete

Salesforce insert

Salesforce update

Salesforce upsert

Serialize to file

Synchronize after merg

Table output

Text file output

Update

Welcome! Supperapp_ETL

100%

DIM_RUTA_EXT

DIM_RUTA_LOAD

DIM_TIEMPO_EXT

DIM_TIEMPO_LOAD

DIM_CLIENTE_EXT

DIM_CLIENTE_LOAD

DM_MODELO_EXT

DIM_MODELO_LOAD

Execution Results

Logging

Execution History

Step Metrics

Performance Graph

Metrics

Preview data

2025/12/02 20:00:39 - Spoon - Launching transformation [Supperapp_ETL]...

2025/12/02 20:00:40 - Spoon - Started the transformation execution.

2025/12/02 20:00:41 - Supperapp_ETL - Dispatching started for transformation [Supperapp_ETL]

2025/12/02 20:00:42 - DIM_RUTA_EXT.0 - !TableInput.Log.FinishedReadingQuery!

2025/12/02 20:00:42 - DIM_RUTA_EXT.0 - Finished processing (I=35, O=0, R=0, W=35, U=0, E=0)

2025/12/02 20:00:42 - DIM_RUTA_LOAD.0 - Finished processing (I=35, O=35, R=35, W=35, U=0, E=0)

2025/12/02 20:00:42 - Spoon - The transformation has finished!!

Query de extracción:

```
SELECT
    i.id_itinerario AS ID_RUTA,
    c_o.nombre AS CIUDAD_ORIGEN,
    a_o.nombre AS AEROPUERTO_ORIGEN,
    c_d.nombre AS CIUDAD_DESTINO,
    a_d.nombre AS AEROPUERTO_DESTINO,
    c_o.nombre || ' - ' || c_d.nombre AS RUTA_COMPLETA
FROM IATA.ITINERARIOS i
JOIN IATA.AEROPUERTOS a_o ON i.id_aeropuerto_origen = a_o.id_aeropuerto
JOIN IATA.CIUDADES c_o ON a_o.id_ciudad = c_o.id_ciudad
JOIN IATA.AEROPUERTOS a_d ON i.id_aeropuerto_destino = a_d.id_aeropuerto
JOIN IATA.CIUDADES c_d ON a_d.id_ciudad = c_d.id_ciudad;
```

Resultado de carga (primeros 10 registros):

ID_RUTA	CIUDAD_ORIGEN	AEROPUERTO_ORIGEN	CIUDAD_DESTINO	AEROPUERTO_DESTINO	RUTA_COMPLETA
---------	---------------	-------------------	----------------	--------------------	---------------

ID_RUTA	CIUDAD_ORIGEN	AEROPUERTO_ORIGEN	CIUDAD_DESTINO	AEROPUERTO_DESTINO	ROUTA_COMPLETA
1	Bogotá	Aeropuerto Internacional El Dorado	Madrid	Aeropuerto Adolfo Suárez Madrid-Barajas	Bogotá - Madrid
2	Bogotá	Aeropuerto Internacional El Dorado	Roma	Aeropuerto Leonardo da Vinci-Fiumicino	Bogotá - Roma
3	Medellín	Aeropuerto Internacional José María Córdova	Madrid	Aeropuerto Adolfo Suárez Madrid-Barajas	Medellín - Madrid
4	Medellín	Aeropuerto Internacional José María Córdova	París	Aeropuerto Charles de Gaulle	Medellín - París
5	Cali	Aeropuerto Internacional Alfonso Bonilla Aragón	Barcelona	Aeropuerto de Barcelona-El Prat	Cali - Barcelona
6	Cali	Aeropuerto Internacional Alfonso Bonilla Aragón	Roma	Aeropuerto Leonardo da Vinci-Fiumicino	Cali - Roma
7	Cartagena	Aeropuerto Internacional Rafael Núñez	Madrid	Aeropuerto Adolfo Suárez Madrid-Barajas	Cartagena - Madrid
8	Cartagena	Aeropuerto Internacional Rafael Núñez	Londres	Aeropuerto de Heathrow	Cartagena - Londres
9	Barranquilla	Aeropuerto Internacional Ernesto Cortissoz	París	Aeropuerto Charles de Gaulle	Barranquilla - París
10	Barranquilla	Aeropuerto Internacional Ernesto Cortissoz	Roma	Aeropuerto Leonardo da Vinci-Fiumicino	Barranquilla - Roma

Descripción: Combina itinerarios con aeropuertos y ciudades para generar rutas completas origen-destino con información descriptiva.

3. DIM_AEROLINEA - Extracción de dimensión de aerolíneas

Implementación en Pentaho Data Integrator:

Spoon - Supperapp_ETL

File Edit View Action Tools Help

View Design

Search

Input

Output

Automatic documentat

Delete

Insert / update

JSON output

LDAP output

Microsoft Access outpu

Microsoft Excel output

Microsoft Excel writer

Pentaho reporting outp

Properties output

RSS output

S3 file output

SQL file output

Salesforce delete

Salesforce insert

Salesforce update

Salesforce upsert

Serialize to file

Synchronize after merg

Table output

Text file output

Update

XML output

100%

DIM_CLIENTE_EXT

DIM_CLIENTE_LOAD

DM_MODELO_EXT

DIM_MODELO_LOAD

DIM_AEROLINEA_EXT

DIM_AEROLINEA_LOAD

FACT_VENTAS_VUELOS_EXT

FACT_VENTAS_VUELOS_LOAD

Execution Results

Logging

Execution History

Step Metrics

Performance Graph

Metrics

Preview data

2025/12/02 20:03:07 - Spoon - Launcning transformation [Supperapp_ETL]...

2025/12/02 20:03:07 - Spoon - Started the transformation execution.

2025/12/02 20:03:07 - Supperapp_ETL - Dispatching started for transformation [Supperapp_ETL]

2025/12/02 20:03:08 - DIM_AEROLINEA_EXT.0 - !TableInput.Log.FinishedReadingQuery!

2025/12/02 20:03:08 - DIM_AEROLINEA_EXT.0 - Finished processing (I=3, O=0, R=0, W=3, U=0, E=0)

2025/12/02 20:03:08 - DIM_AEROLINEA_LOAD.0 - Finished processing (I=3, O=3, R=3, W=3, U=0, E=0)

2025/12/02 20:03:08 - Spoon - The transformation has finished!!

Query de extracción:

```
SELECT
    a.id_aerolinea AS ID_AEROLINEA,
    a.nombre AS NOMBRE_AEROLINEA
FROM IATA.AEROLINEAS a;
```

Resultado de carga (todos los registros):

ID_AEROLINEA	NOMBRE_AEROLINEA
1	Avianca
2	Latam
3	Wingo

Descripción: Extracción directa de todas las aerolíneas desde la tabla maestra `IATA.AEROLINEAS`.

4. DIM_MODELO - Extracción de dimensión de modelos de avión

Implementación en Pentaho Data Integrator:

Spoon - Supperapp_ETL

File Edit View Action Tools Help

View Design

Search

Input

Output

Automatic documentat

Delete

Insert / update

JSON output

LDAP output

Microsoft Access outpu

Microsoft Excel output

Microsoft Excel writer

Pentaho reporting outp

Properties output

RSS output

S3 file output

SQL file output

Salesforce delete

Salesforce insert

Salesforce update

Salesforce upsert

Serialize to file

Synchronize after merg

Table output

Text file output

Update

XML output

Welcome! Supperapp_ETL

100%

DIM_TIEMPO_EXT

DIM_TIEMPO_LOAD

DIM_CLIENTE_EXT

DIM_CLIENTE_LOAD

DM_MODELO_EXT

DIM_MODELO_LOAD

DIM_AEROLINEA_EXT

DIM_AEROLINEA_LOAD

Execution Results

Logging Execution History Step Metrics Performance Graph Metrics Preview data

2025/12/02 20:02:30 - Spoon - Launching transformation [Supperapp_ETL...

2025/12/02 20:02:30 - Spoon - Started the transformation execution.

2025/12/02 20:02:30 - Supperapp_ETL - Dispatching started for transformation [Supperapp_ETL]

2025/12/02 20:02:31 - DM_MODELO_EXT.0 - !TableInput.Log.FinishedReadingQuery!

2025/12/02 20:02:31 - DM_MODELO_EXT.0 - Finished processing (I=2, O=0, R=0, W=2, U=0, E=0)

2025/12/02 20:02:31 - DIM_MODELO_LOAD.0 - Finished processing (I=2, O=2, R=2, W=2, U=0, E=0)

2025/12/02 20:02:31 - Spoon - The transformation has finished!!

Query de extracción:

```
SELECT
    m.id_modelo AS ID_MODELO,
    m.nombre AS NOMBRE_MODELO
FROM IATA.MODELOS m;
```

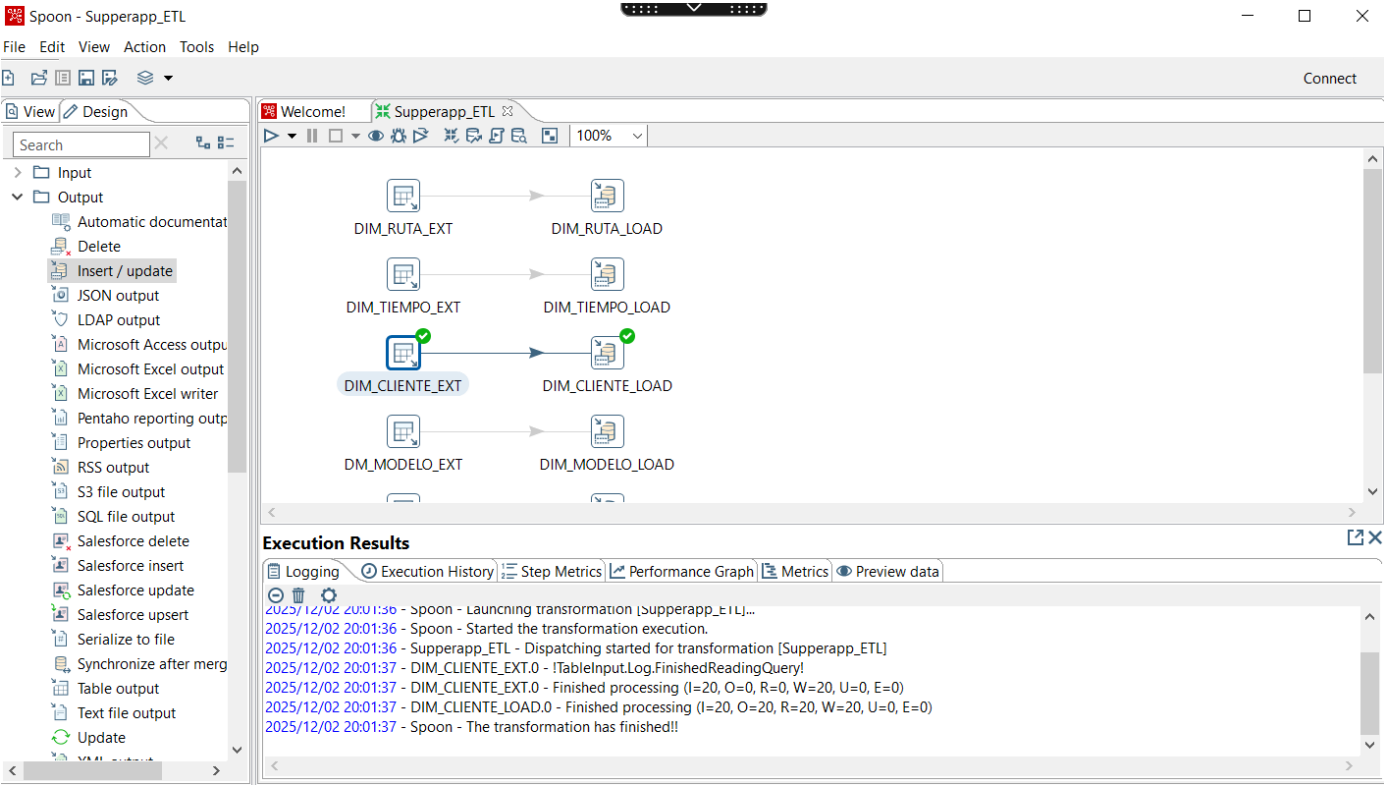
Resultado de carga (todos los registros):

ID_MODELO	NOMBRE_MODELO
1	Airbus 320
2	Boeing 747

Descripción: Extracción directa de todos los modelos de avión desde la tabla maestra `IATA.MODELOS`.

5. DIM_CLIENTE - Extracción de dimensión de clientes/pasajeros

Implementación en Pentaho Data Integrator:



Query de extracción:

```
SELECT
    u.cedula AS ID_CLIENTE,
    u.nombre || ' ' || u.apellido AS NOMBRE_COMPLETO,
    u.email AS EMAIL,
    c.nombre AS CIUDAD_RESIDENCIA
FROM IATA.USUARIOS u
JOIN IATA.CIUDADES c ON u.id_ciudad = c.id_ciudad;
```

Resultado de carga (primeros 10 registros):

ID_CLIENTE	NOMBRE_COMPLETO	EMAIL	CIUDAD_RESIDENCIA
1001234567	Juan Pérez	juan.perez@email.com	Bogotá
1002345678	María García	maria.garcia@email.com	Medellín
1003456789	Carlos López	carlos.lopez@email.com	Cali
1004567890	Ana Martínez	ana.martinez@email.com	Barranquilla
1005678901	Luis Rodríguez	luis.rodriguez@email.com	Cartagena
1006789012	Laura Fernández	laura.fernandez@email.com	Bogotá
1007890123	Pedro Sánchez	pedro.sanchez@email.com	Medellín
1008901234	Carmen Díaz	carmen.diaz@email.com	Cali
1009012345	Jorge Ramírez	jorge.ramirez@email.com	Bogotá
1010123456	Lucía Torres	lucia.torres@email.com	Cartagena

Descripción: Combina información de usuarios con sus ciudades de residencia, concatenando nombre y apellido en un solo campo.

6. FACT_VENTAS_VUELOS - Extracción de tabla de hechos

Implementación en Pentaho Data Integrator:

Spoon - Supperapp_ETL

File Edit View Action Tools Help

Connect

View Design

Search

Input

Output

Automatic docur

Delete

Insert / update

JSON output

LDAP output

Microsoft Access

Microsoft Excel c

Microsoft Excel v

Pentaho reportin

Properties outpu

RSS output

S3 file output

SQL file output

Salesforce delete

Salesforce insert

Salesforce updat

Salesforce upsert

Serialize to file

Synchronize after

Table output

Text file output

Update

Welcome! Supperapp_ETL

100%

DIM_CLIENTE_EXT

DIM_CLIENTE_LOAD

DM_MODELO_EXT

DIM_MODELO_LOAD

DIM_AEROLINEA_EXT

DIM_AEROLINEA_LOAD

FACT_VENTAS_VUELOS_EXT

FACT_VENTAS_VUELOS_LOAD

Execution Results

Logging Execution History Step Metrics Performance Graph Metrics Preview data

2025/12/02 20:56:52 - Spoon - Transformation opened.

2025/12/02 20:56:52 - Spoon - Launching transformation [Supperapp_ETL]...

2025/12/02 20:56:52 - Spoon - Started the transformation execution.

2025/12/02 20:56:52 - Supperapp_ETL - Dispatching started for transformation [Supperapp_ETL]

2025/12/02 20:56:53 - FACT_VENTAS_VUELOS_EXT.0 - !TableInput.Log.FinishedReadingQuery!

2025/12/02 20:56:53 - FACT_VENTAS_VUELOS_EXT.0 - Finished processing (I=160, O=0, R=0, W=160, U=0, E=0)

2025/12/02 20:56:53 - FACT_VENTAS_VUELOS_LOAD.0 - Finished processing (I=160, O=160, R=160, W=160, U=0, E=0)

2025/12/02 20:56:53 - Spoon - The transformation has finished!!

Query de extracción:

```
SELECT
    ROW_NUMBER() OVER (
        ORDER BY v.id_itinerario, v.id_avion, v.id_usuario
    )
    dt.id_tiempo
    i.id_itinerario
    a.id_aerolinea
    av.id_modelo
    TO_NUMBER(u.cedula)
    v.costo
    (i.fecha_llegada - i.fecha_salida) * 24
    COUNT(*) OVER (
        PARTITION BY i.id_itinerario, av.id_avion
    )
    AS ID_VENTA,
    AS ID_TIEMPO,
    AS ID_RUTA,
    AS ID_AEROLINEA,
    AS ID_MODELO,
    AS ID_CLIENTE,
    AS COSTO,
    AS DURACION_VUELO_HORAS,
    AS CANTIDAD_PASAJEROS
FROM IATA.VUELOS v
JOIN IATA.ITINERARIOS i ON v.id_itinerario = i.id_itinerario
JOIN IATA.AVIONES av ON v.id_avion = av.id_avion
JOIN IATA.AEROLINEAS a ON av.id_aerolinea = a.id_aerolinea
JOIN IATA.USUARIOS u ON v.id_usuario = u.cedula
JOIN DIM_TIEMPO dt ON dt.fecha = TRUNC(i.fecha_salida);
```

Resultado de carga (primeros 10 registros):

ID_VENTA	ID_TIEMPO	ID_RUTA	ID_AEROLINEA	ID_MODELO	ID_CLIENTE	COSTO	DURACION_VUELO_HORAS	CANTIDAD_PASAJEROS
1	1	1	1	2	1001234567	850000.00	10.50	15
2	1	1	1	2	1002345678	850000.00	10.50	15
3	2	2	2	1	1003456789	920000.00	11.25	12
4	2	2	2	1	1004567890	920000.00	11.25	12
5	3	3	1	2	1005678901	880000.00	10.75	18
6	3	4	3	1	1006789012	750000.00	9.50	10
7	4	5	2	2	1007890123	910000.00	11.00	14
8	4	5	2	2	1008901234	910000.00	11.00	14

ID_VENTA	ID_TIEMPO	ID_RUTA	ID_AEROLINEA	ID_MODELO	ID_CLIENTE	COSTO	DURACION_VUELO_HORAS	CANTIDAD_PASAJEROS
9	5	6	1	1	1009012345	930000.00	11.50	16
10	5	7	3	2	1010123456	870000.00	10.25	11

Descripción: Query complejo que integra datos de múltiples tablas transaccionales (VUELOS, ITINERARIOS, AVIONES, AEROLINEAS, USUARIOS) con la dimensión temporal ya cargada (DIM_TIEMPO). Calcula métricas como duración del vuelo en horas y cantidad de pasajeros por vuelo.

Ver código completo de todos los queries ETL: [CodigoETL-Input_Table.sql](#)

Ver capturas de pantalla completas: [Carpeta /images/](#) en [GitHub](#)

Secuencia de ejecución ETL:

1. Cargar **DIM_TIEMPO** (sin dependencias)
2. Cargar **DIM_RUTA** (sin dependencias)
3. Cargar **DIM_AEROLINEA** (sin dependencias)
4. Cargar **DIM_MODELO** (sin dependencias)
5. Cargar **DIM_CLIENTE** (sin dependencias)
6. Cargar **FACT_VENTAS_VUELOS** (requiere que DIM_TIEMPO esté poblada)

Validación: Verificar integridad referencial y conteo de registros en cada paso.

Fase 4: Análisis y Reporting (Pendiente)

- ☐ Desarrollar consultas analíticas clave
- ☐ Generar reportes de negocio
- ☐ Documentar insights y hallazgos

Autores

Estudiante: Edwin Silva Salas, Carlos Preciado Cárdenas, Cristian Restrepo Zapata
Programa: Maestría en Ciencia de Datos
Universidad: Pontificia Universidad Javeriana
Repositorio GIT: [IATA_CASE_MCD](#)