

---

# Diseño e Implementación de un Data Mart



**ASIGNATURA:** ALMACENES Y MINERÍA DE DATOS

PRÁCTICA 1

**CURSO:** 2025/26

---

Pascual Albericio, Irene (NIP 871627)

Vázquez Martín, Lucía (NIP 871886)

# Índice

|                                                                       |           |
|-----------------------------------------------------------------------|-----------|
| <b>Índice.....</b>                                                    | <b>2</b>  |
| <b>1. Introducción.....</b>                                           | <b>3</b>  |
| <b>2. Metodología.....</b>                                            | <b>3</b>  |
| <b>3. Diseño del esquema en estrella del data mart de vuelos.....</b> | <b>4</b>  |
| <b>4. Incorporación de información de pasajeros.....</b>              | <b>5</b>  |
| <b>5. Propuesta nuevo contexto y diseño alternativo.....</b>          | <b>6</b>  |
| <b>6. Implementación.....</b>                                         | <b>7</b>  |
| 6.1. Vuelos comerciales sin información de pasajeros.....             | 7         |
| 6.1.1. Script de creación de tablas.....                              | 7         |
| 6.1.2. Script de inserción de datos.....                              | 10        |
| 6.1.3. Script de consultas.....                                       | 13        |
| 6.2. Vuelos comerciales con información de pasajeros.....             | 20        |
| 6.2.1. Script de creación de tablas.....                              | 20        |
| 6.2.2. Script de inserción de datos.....                              | 21        |
| 6.2.3. Script de consultas.....                                       | 24        |
| 6.3. Nueva propuesta.....                                             | 27        |
| 6.3.1. Script de creación de tablas.....                              | 27        |
| 6.3.2. Script de inserción de datos.....                              | 31        |
| 6.3.3. Script de consultas.....                                       | 34        |
| <b>7. Distribución del trabajo.....</b>                               | <b>35</b> |
| <b>8. Conclusiones.....</b>                                           | <b>36</b> |

# 1. Introducción

El objetivo de la práctica consiste en diseñar e implementar un data mart especializado en el análisis del rendimiento de vuelos comerciales en Estados Unidos. Este sistema permitirá a los analistas y responsables de la toma de decisiones identificar patrones en los retrasos, evaluar el desempeño de aerolíneas y aeropuertos, y analizar la influencia de diversos factores operacionales en la puntualidad y eficiencia de los vuelos.

La solución propuesta se fundamenta en la metodología de Kimball, que sigue un enfoque centrado en el usuario de negocio, priorizando la accesibilidad de la información y la optimización del rendimiento de las consultas analíticas. Mediante la implementación de un esquema en estrella, se facilitará el acceso intuitivo a los datos y se proporcionará una estructura eficiente para el análisis de las consultas.

Además del análisis de datos de vuelos, la práctica requiere de la incorporación de la información sobre los pasajeros, así como el desarrollo de un segundo escenario temático, que también será resuelto mediante el diseño de su correspondiente esquema en estrella.

# 2. Metodología

Para el desarrollo del problema planteado mediante el esquema en estrella, se ha seguido la metodología de diseño dimensional de *Kimball*, aplicando sus cuatro pasos fundamentales al caso de análisis de vuelos comerciales:

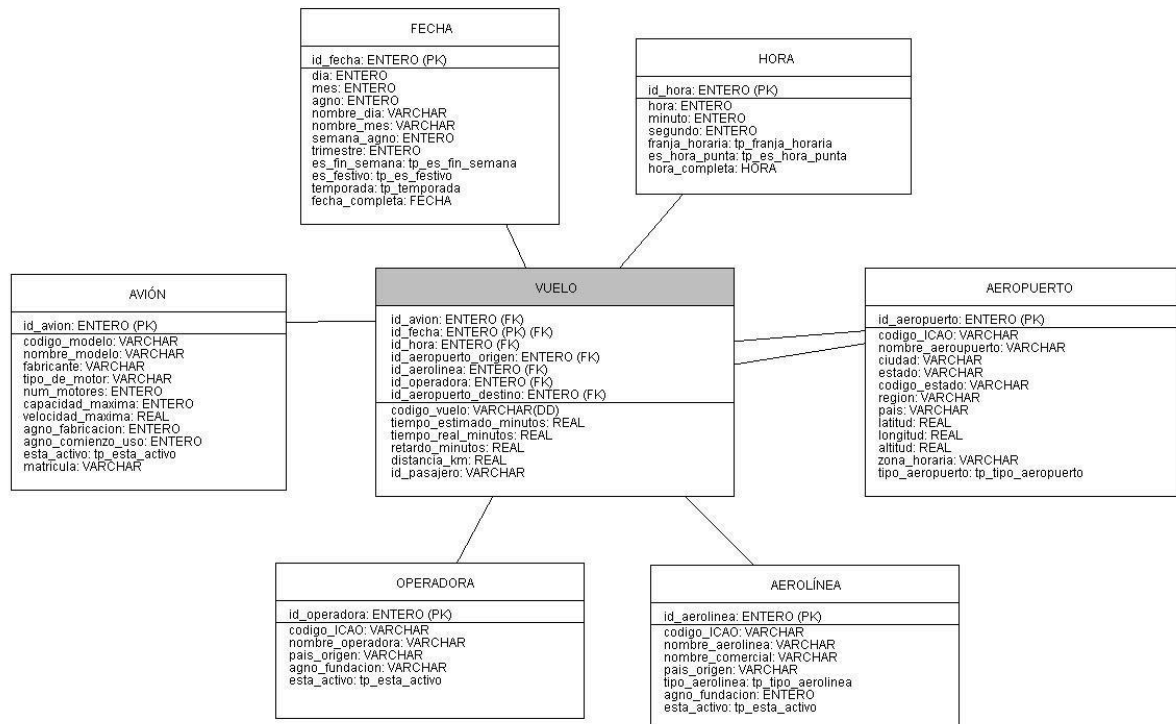
**1. Selección del proceso de negocio:** Se identificó el rendimiento operativo de los vuelos comerciales como proceso principal de negocio, dado su alto impacto en la toma de decisiones y la disponibilidad de datos que permiten estudiar aspectos como los retrasos.

**2. Declaración del grano:** En un primer planteamiento, se estableció como granularidad un registro por vuelo individual realizado en una fecha y hora específicas por un avión entre un aeropuerto de origen y otro de destino. Sin embargo, tras revisar el nivel de detalle, se observó que este grano no era el más adecuado, ya que para identificar de forma única un vuelo basta con conocer el código de vuelo y la fecha en la que se lleva a cabo. Por ello, finalmente, se definió el grano como un vuelo concreto en una fecha específica.

**3. Identificación de dimensiones:** Se seleccionaron las dimensiones que eran más representativas para el proceso de negocio con el que se estaba trabajando: *DIM\_Fecha*, *DIM\_Hora*, *DIM\_Avion*, *DIM\_Aeropuerto*, *DIM\_Aerolinea* y *DIM\_Operadora*. De todas ellas, *DIM\_Fecha* se considera una dimensión primaria, ya que forma parte del grano definido, mientras que el resto actúan como dimensiones secundarias, proporcionando información adicional.

**4. Identificación de hechos:** Se definieron como métricas cuantitativas la distancia recorrida en kilómetros y el tiempo estimado de vuelo, el tiempo real de vuelo y el retardo, medidas en minutos. Inicialmente a estos hechos numéricos aditivos se les añadió el código de vuelo, pero posteriormente se detectó que ni era aditiva ni actuaba como clave primaria o foránea. En consecuencia, se incorporó como una dimensión degenerada dentro de la tabla de hechos (**VUELO**), permitiendo mantener la trazabilidad de los registros sin requerir una tabla adicional.

### 3. Diseño del esquema en estrella del data mart de vuelos



El diseño del esquema en estrella presenta varias decisiones estructurales importantes que optimizan tanto la consistencia de los datos como la flexibilidad analítica:

#### 1. Dimensiones temporales separadas

Se ha dividido la dimensión temporal en dos tablas independientes: *DIM\_FECHA* y *DIM\_HORA*. Esta separación se debe a que la fecha forma parte del grano de la tabla de hechos, mientras que la hora aporta información complementaria. Gracias a esta decisión, es posible realizar análisis más versátiles, como puede ser la examinación de periodos amplios (trimestres), sin tener en cuenta la hora o analizar franjas horarias específicas (horas puntas), independientemente de la fecha.

#### 2. Reutilización de la dimensión AEROPUERTO

La dimensión *DIM\_AEROPUERTO* se emplea dos veces en la tabla de hechos mediante las claves foráneas *id\_aeropuerto\_origen* e *id\_aeropuerto\_destino*. De este modo, se evita la duplicación de atributos y se mantiene una única tabla coherente que describe tanto el origen como el destino de los vuelos.

#### 3. Diferenciación entre aerolínea y operadora

Se distingue claramente entre las dimensiones *DIM\_AEROLINEA* y *DIM\_OPERADORA*, siendo la primera de ellas la compañía que comercializa el vuelo (vende los billetes y bajo cuya marca viaja el pasajero), mientras que la segunda es la compañía que opera físicamente el vuelo (proporciona la tripulación y el mantenimiento de la aeronave).

#### 4. Dimensión degenerada

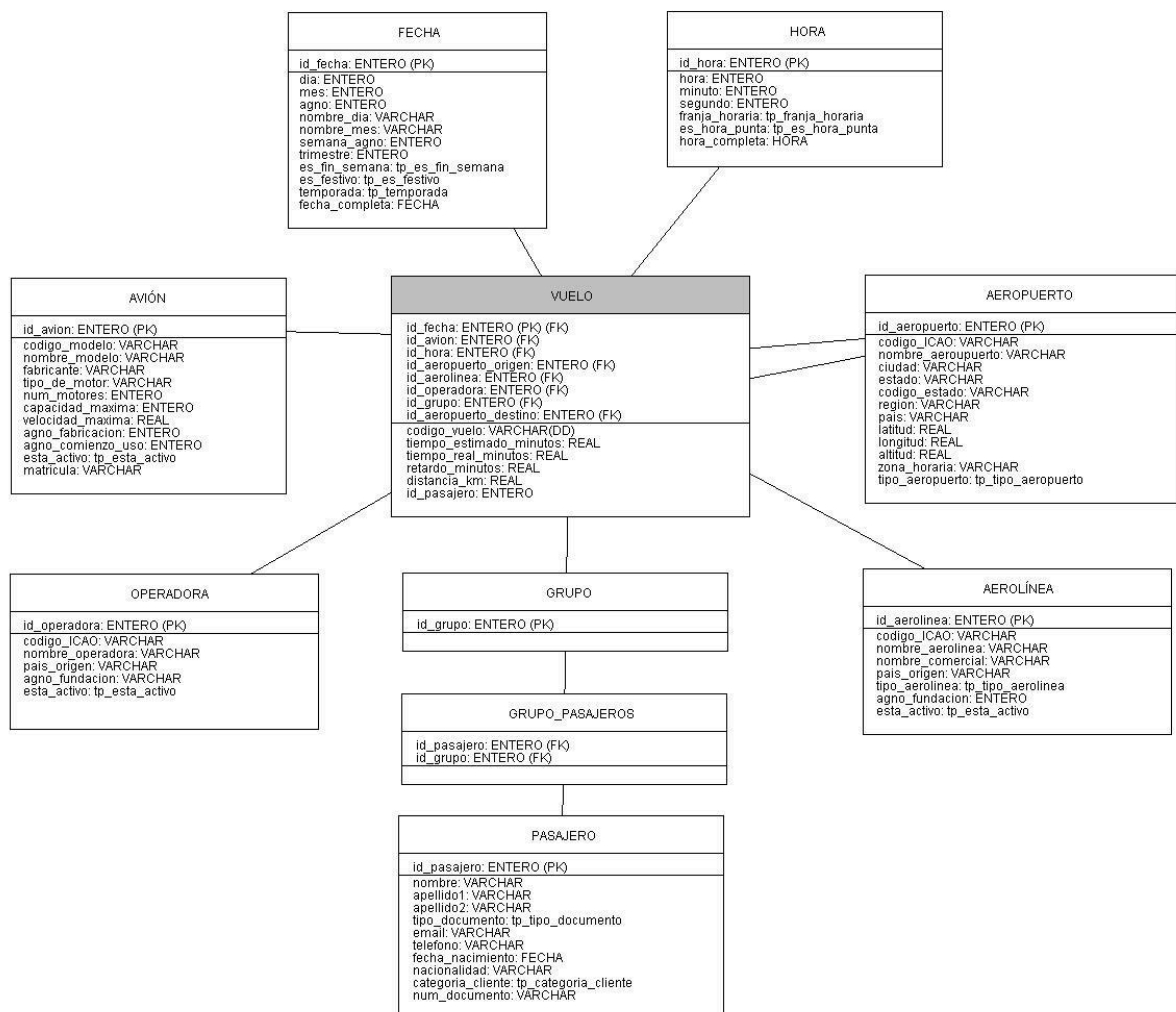
El atributo *codigo\_vuelo* se ha incorporado como dimensión degenerada dentro de la tabla de hechos. Esta decisión evita realizar joins innecesarios con una tabla de dimensión adicional y facilita la trazabilidad directa de cada registro.

#### 5. Atributos y métricas

Se han incluido tantos atributos como ha sido posible en las tablas de dimensiones, con el fin de ampliar las posibilidades de análisis dentro del contexto del modelo y mejorar la comprensión general de los datos disponibles.

Por último, todas las métricas de la tabla de hechos son medidas aditivas, lo que permite agregarlas de forma significativa a través de cualquier dimensión. Esto permite que dichas métricas puedan ser utilizadas para consultas como calcular el retardo total de una aerolínea concreta en el mes de enero o calcular la distancia total recorrida por un modelo de avión específico.

#### 4. Incorporación de información de pasajeros

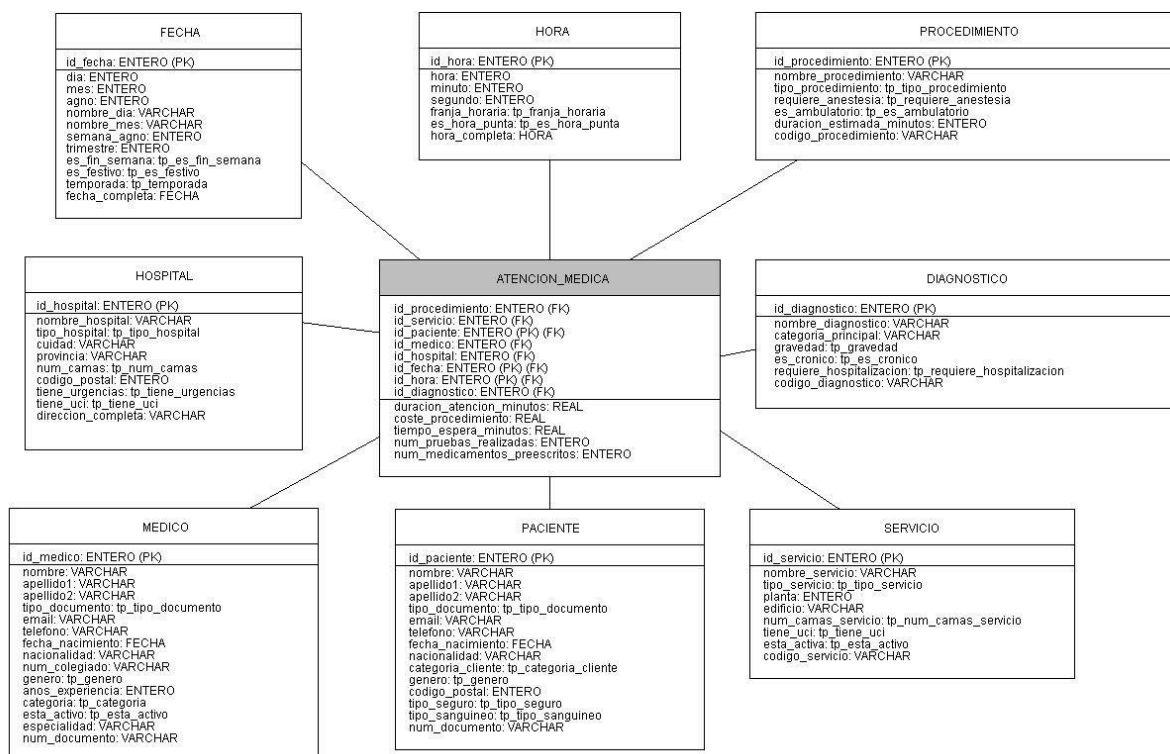


En la nueva versión del data mart se ha incorporado la gestión de pasajeros mediante la creación de las tablas **GRUPO**, **GRUPO\_PASAJEROS** y **PASAJERO**, que permiten representar correctamente la cardinalidad entre vuelos y pasajeros.

La tabla intermedia **GRUPO\_PASAJEROS** cuenta con dos claves foráneas, una de ellas hace referencia a la clave primaria de la tabla **GRUPO** (*id\_grupo*), mientras que la otra hace referencia a la clave primaria de **PASAJERO** (*id\_pasajero*). Gracias a esto se pueden asociar diferentes pasajeros a cada uno de los grupos que forman parte de un vuelo. Esto implica que, dentro de la tabla de hechos (**VUELO**), se ha incluido como clave foránea la clave primaria *id\_grupo*. Es importante destacar que esta modificación no ha alterado el grano original del data mart, que sigue definido por la combinación de *codigo\_vuelo* y *fecha*, garantizando así la consistencia y coherencia de los datos en el modelo.

Antes de optar por el diseño actual, se valoró una alternativa en la que la tabla **GRUPO\_PASAJEROS** se definía como una tabla de hechos independiente, relacionada directamente con la tabla **PASAJERO**, sin necesidad de crear la tabla **GRUPO**. Sin embargo, esta opción se descartó porque implicaba construir un data mart centrado exclusivamente en los pasajeros, desvinculado del contexto de los vuelos. Dado que el objetivo principal era mantener el análisis en torno a los vuelos y no a los pasajeros de forma aislada, se consideró más adecuado incorporar la tabla **GRUPO** que permitiese añadir una clave foránea a la tabla de hechos existentes sin necesidad de modificar el grano del data mart.

## 5. Propuesta nuevo contexto y diseño alternativo



## 6. Implementación

Las implementaciones de todos los modelos en estrella mostrados se han llevado a cabo sobre Oracle Database.

### 6.1. Vuelos comerciales sin información de pasajeros

Este primer modelo implementa un esquema en estrella básico centrado en las operaciones de vuelo, sin considerar información de pasajeros. El esquema consta de 6 tablas de dimensiones y 1 tabla de hechos central.

La implementación se ha realizado mediante tres scripts SQL:

- *crear\_tablas\_vuelos.sql*
- *insertar\_datos\_vuelos.sql*
- *consultas\_vuelos.sql*.

#### 6.1.1. Script de creación de tablas

El primer script (*crear\_tablas\_vuelos.sql*) se encarga de la creación completa del esquema.

Incluye bloques PL/SQL que eliminan las tablas existentes en orden inverso a las dependencias, para permitir su ejecución múltiple sin errores. Las tablas se eliminan y crean utilizando *EXECUTE IMMEDIATE*, capturando e ignorando las excepciones cuando una tabla no existe.

A continuación se detallan las tablas creadas:

- **DIM\_FECHA:** Dimensión temporal que proporciona contexto sobre el día en que se realiza el vuelo. Incluye atributos jerárquicos (día > mes > trimestre > año) y categóricos (día de la semana, fin de semana, festivo, temporada).

SQL

```
CREATE TABLE FECHA (  
    id_fecha INTEGER PRIMARY KEY,  
    fecha_completa DATE UNIQUE NOT NULL, -- Clave Natural  
    dia INTEGER NOT NULL,  
    mes INTEGER NOT NULL,  
    agno INTEGER NOT NULL,  
    nombre_dia VARCHAR2(20) NOT NULL,  
    nombre_mes VARCHAR2(20) NOT NULL,  
    semana_agno INTEGER NOT NULL,  
    trimestre INTEGER NOT NULL,  
    es_fin_semana VARCHAR2(20) NOT NULL CHECK (es_fin_semana IN ('es_fin_semana',  
'no_es_fin_semana')),  
    es_festivo VARCHAR2(20) NOT NULL CHECK (es_festivo IN ('es_festivo', 'no_es_festivo')),  
    temporada VARCHAR2(20) NOT NULL CHECK (temporada IN ('primavera', 'verano', 'otogno',  
'invierno'))  
);
```

- **DIM\_HORA:** Dimensión que complementa la información temporal especificando el momento exacto del vuelo.

SQL

```
CREATE TABLE HORA (
    id_hora INTEGER PRIMARY KEY,
    hora_completa DATE UNIQUE NOT NULL, -- Clave Natural
    hora INTEGER NOT NULL,
    minuto INTEGER NOT NULL,
    segundo INTEGER NOT NULL,
    franja_horaria VARCHAR2(20) NOT NULL CHECK (franja_horaria IN ('00:00 - 04:59', '05:00 - 11:59', '12:00 - 19:59', '20:00 - 23:59')),
    es_hora_punta VARCHAR2(20) NOT NULL CHECK (es_hora_punta IN ('es_hora_punta', 'no_es_hora_punta'))
);
```

- **DIM\_AEROPUERTO:** Dimensión geográfica que describe las características de los aeropuertos.

SQL

```
CREATE TABLE AEROPUERTO (
    id_aeropuerto INTEGER PRIMARY KEY,
    codigo_ICAO VARCHAR2(50) UNIQUE NOT NULL, -- Clave Natural
    nombre_aeropuerto VARCHAR2(100) NOT NULL,
    ciudad VARCHAR2(100) NOT NULL,
    estado VARCHAR2(100) NOT NULL,
    codigo_estado VARCHAR2(50) NOT NULL,
    region VARCHAR2(100) NOT NULL,
    pais VARCHAR2(100) NOT NULL,
    latitud REAL NOT NULL,
    longitud REAL NOT NULL,
    altitud REAL NOT NULL,
    zona_horaria VARCHAR2(50) NOT NULL,
    tipo_aeropuerto VARCHAR2(20) NOT NULL CHECK (tipo_aeropuerto IN ('internacional', 'nacional', 'regional'))
);
```

- **DIM\_AVION:** Dimensión con las características de los aviones utilizados en los vuelos.

SQL

```
CREATE TABLE AVION (
    id_avion INTEGER PRIMARY KEY,
    matricula VARCHAR2(50) UNIQUE NOT NULL, -- Clave Natural
    codigo_modelo VARCHAR2(50) NOT NULL,
    nombre_modelo VARCHAR2(100) NOT NULL,
    fabricante VARCHAR2(100) NOT NULL,
    tipo_de_motor VARCHAR2(50) NOT NULL,
    num_motores INTEGER NOT NULL,
    num_asientos INTEGER NOT NULL,
    velocidad_maxima REAL NOT NULL,
    agno_fabricacion INTEGER NOT NULL,
    agno_comienzo_uso INTEGER NOT NULL,
    esta_activo VARCHAR2(20) NOT NULL CHECK (esta_activo IN ('esta_activo', 'no_esta_activo'))
);
```

- **DIM\_OPERADORA:** Dimensión que identifica la compañía que opera físicamente el vuelo, proporcionando la tripulación y el mantenimiento del avión.

SQL

```
CREATE TABLE OPERADORA (
    id_operadora INTEGER PRIMARY KEY,
    codigo_ICAO VARCHAR2(50) UNIQUE NOT NULL, -- Clave Natural
    nombre_operadora VARCHAR2(100) NOT NULL,
    pais_origen VARCHAR2(100) NOT NULL,
    agno_fundacion VARCHAR2(50) NOT NULL,
    esta_activo VARCHAR2(20) NOT NULL CHECK (esta_activo IN ('esta_activo', 'no_esta_activo'))
);
```

- **DIM\_AEROLINEA:** Dimensión que identifica la aerolínea comercial bajo cuya marca se vende y comercializa el vuelo.

SQL

```
CREATE TABLE AEROLINEA (
    id_aerolinea INTEGER PRIMARY KEY,
    codigo_ICAO VARCHAR2(50) UNIQUE NOT NULL, -- Clave Natural
    nombre_aerolinea VARCHAR2(100) NOT NULL,
    nombre_comercial VARCHAR2(100) NOT NULL,
    pais_origen VARCHAR2(100) NOT NULL,
    tipo_aerolinea VARCHAR2(20) NOT NULL CHECK (tipo_aerolinea IN ('internacional', 'nacional', 'regional')),
    agno_fundacion INTEGER NOT NULL,
    esta_activo VARCHAR2(20) NOT NULL CHECK (esta_activo IN ('esta_activo', 'no_esta_activo'))
);
```

- **TABLA DE HECHOS (VUELO):** La tabla de hechos almacena las métricas operacionales de cada vuelo, relacionando todas las dimensiones mediante claves foráneas. La granularidad es un registro por vuelo y fecha concreta.

SQL

```
CREATE TABLE VUELO (
    codigo_vuelo VARCHAR2(50) NOT NULL,
    id_fecha INTEGER NOT NULL,
    id_hora INTEGER NOT NULL,
    id_avion INTEGER NOT NULL,
    id_aeropuerto_origen INTEGER NOT NULL,
    id_aeropuerto_destino INTEGER NOT NULL,
    id_aerolinea INTEGER NOT NULL,
    id_operadora INTEGER NOT NULL,
    tiempo_estimado_minutos REAL NOT NULL,
    tiempo_real_minutos REAL NOT NULL,
    retardo_minutos REAL NOT NULL,
    distancia_km REAL NOT NULL,
    PRIMARY KEY (codigo_vuelo, id_fecha),
    FOREIGN KEY (id_avion) REFERENCES AVION(id_avion) ON DELETE CASCADE,
    FOREIGN KEY (id_fecha) REFERENCES FECHA(id_fecha) ON DELETE CASCADE,
    FOREIGN KEY (id_hora) REFERENCES HORA(id_hora) ON DELETE CASCADE,
    FOREIGN KEY (id_aeropuerto_origen) REFERENCES AEROPUERTO(id_aeropuerto) ON DELETE CASCADE,
    FOREIGN KEY (id_aeropuerto_destino) REFERENCES AEROPUERTO(id_aeropuerto) ON DELETE CASCADE,
    FOREIGN KEY (id_aerolinea) REFERENCES AEROLINEA(id_aerolinea) ON DELETE CASCADE,
```

```

FOREIGN KEY (id_operadora) REFERENCES OPERADORA(id_operadora) ON DELETE CASCADE,
CONSTRAINT chk_tiempo_positivo CHECK (tiempo_estimado_minutos > 0 AND tiempo_real_minutos > 0),
CONSTRAINT chk_distancia_positiva CHECK (distancia_km > 0),
CONSTRAINT chk_retardo_no_negativo CHECK (retardo_minutos >= 0),
CONSTRAINT chk_aeropuertos_diferentes CHECK (id_aeropuerto_origen != id_aeropuerto_destino)
);

```

Las métricas incluidas son:

- ❖ tiempo\_estimado\_minutos: Duración planificada del vuelo según el plan de vuelo.
- ❖ tiempo\_real\_minutos: Duración efectiva del vuelo desde despegue hasta aterrizaje.
- ❖ retardo\_minutos: Diferencia entre el tiempo real y el estimado (métrica derivada almacenada para optimización de consultas).
- ❖ distancia\_km: Distancia en kilómetros de la ruta entre aeropuerto de origen y destino.

Todas las métricas son aditivas, permitiendo agregaciones significativas a través de cualquier combinación de dimensiones (suma de distancias, promedios de tiempos,...). Mientras que el atributo *codigo\_vuelo* forma parte de la clave primaria compuesta y actúa como dimensión degenerada, ya que no posee atributos descriptivos adicionales. Esto simplifica el modelo sin perder funcionalidad.

### 6.1.2. Script de inserción de datos

El segundo script (*insertar\_datos\_vuelos.sql*) puebla el data warehouse con datos de ejemplo para validar el modelo.

Antes de insertar los datos, se eliminan los registros existentes en el orden correcto (de tabla de hechos a dimensiones) para permitir ejecuciones repetidas sin errores.

```

SQL
DELETE FROM VUELO;
DELETE FROM AEROLINEA;
DELETE FROM OPERADORA;
DELETE FROM AVION;
DELETE FROM AEROPUERTO;
DELETE FROM HORA;
DELETE FROM FECHA;
COMMIT;

```

El script de inserción puebla el data warehouse con 13 fechas distribuidas en 2025 (todos los trimestres, 9 fines de semana, 5 festivos), 11 horas en cuatro franjas horarias, 5 aeropuertos (3 internacionales y 2 nacionales), 5 aviones (3 Airbus y 2 Boeing), 5 operadoras y aerolíneas (Iberia, Vueling, Air Europa, Ryanair e EasyJet), y 23 vuelos con rutas bidireccionales origen-destino:

## SQL

```

----- DIMENSIÓN FECHA -----
INSERT INTO FECHA VALUES (1, TO_DATE('2025-01-15', 'YYYY-MM-DD'), 15, 1, 2025, 'Miércoles',
'Enero', 3, 1, 'no_es_fin_semana', 'no_es_festivo', 'primavera');
INSERT INTO FECHA VALUES (2, TO_DATE('2025-01-16', 'YYYY-MM-DD'), 16, 1, 2025, 'Jueves', 'Enero',
3, 1, 'no_es_fin_semana', 'no_es_festivo', 'otogno');
INSERT INTO FECHA VALUES (3, TO_DATE('2025-07-20', 'YYYY-MM-DD'), 20, 7, 2025, 'Domingo', 'Julio',
29, 3, 'es_fin_semana', 'no_es_festivo', 'verano');
INSERT INTO FECHA VALUES (4, TO_DATE('2025-08-15', 'YYYY-MM-DD'), 15, 8, 2025, 'Viernes',
'Agosto', 33, 3, 'no_es_fin_semana', 'es_festivo', 'primavera');
INSERT INTO FECHA VALUES (5, TO_DATE('2025-12-25', 'YYYY-MM-DD'), 25, 12, 2025, 'Jueves',
'Diciembre', 52, 4, 'no_es_fin_semana', 'es_festivo', 'invierno');
INSERT INTO FECHA VALUES (6, TO_DATE('2025-01-18', 'YYYY-MM-DD'), 18, 1, 2025, 'Sábado', 'Enero',
3, 1, 'es_fin_semana', 'no_es_festivo', 'verano');
INSERT INTO FECHA VALUES (7, TO_DATE('2025-03-21', 'YYYY-MM-DD'), 21, 3, 2025, 'Viernes', 'Marzo',
12, 1, 'no_es_fin_semana', 'no_es_festivo', 'primavera');
INSERT INTO FECHA VALUES (8, TO_DATE('2025-04-10', 'YYYY-MM-DD'), 10, 4, 2025, 'Jueves', 'Abril',
15, 2, 'no_es_fin_semana', 'no_es_festivo', 'primavera');
INSERT INTO FECHA VALUES (9, TO_DATE('2025-06-15', 'YYYY-MM-DD'), 15, 6, 2025, 'Domingo', 'Junio',
24, 2, 'es_fin_semana', 'no_es_festivo', 'verano');
INSERT INTO FECHA VALUES (10, TO_DATE('2025-07-21', 'YYYY-MM-DD'), 21, 7, 2025, 'Lunes', 'Julio',
30, 3, 'no_es_fin_semana', 'no_es_festivo', 'verano');
INSERT INTO FECHA VALUES (11, TO_DATE('2025-09-12', 'YYYY-MM-DD'), 12, 9, 2025, 'Viernes',
'Septiembre', 37, 3, 'no_es_fin_semana', 'no_es_festivo', 'otogno');
INSERT INTO FECHA VALUES (12, TO_DATE('2025-10-12', 'YYYY-MM-DD'), 12, 10, 2025, 'Domingo',
'Octubre', 41, 4, 'es_fin_semana', 'es_festivo', 'otogno');
INSERT INTO FECHA VALUES (13, TO_DATE('2025-11-01', 'YYYY-MM-DD'), 1, 11, 2025, 'Sábado',
'Noviembre', 44, 4, 'es_fin_semana', 'es_festivo', 'otogno');

----- DIMENSIÓN HORA -----
INSERT INTO HORA VALUES (1, TO_DATE('2025-01-01 08:30:00', 'YYYY-MM-DD HH24:MI:SS'), 8, 30, 0,
'05:00 - 11:59', 'es_hora_punta');
INSERT INTO HORA VALUES (2, TO_DATE('2025-01-01 14:45:00', 'YYYY-MM-DD HH24:MI:SS'), 14, 45, 0,
'12:00 - 19:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (3, TO_DATE('2025-01-01 18:15:00', 'YYYY-MM-DD HH24:MI:SS'), 18, 15, 0,
'12:00 - 19:59', 'es_hora_punta');
INSERT INTO HORA VALUES (4, TO_DATE('2025-01-01 21:00:00', 'YYYY-MM-DD HH24:MI:SS'), 21, 0, 0,
'20:00 - 23:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (5, TO_DATE('2025-01-01 06:00:00', 'YYYY-MM-DD HH24:MI:SS'), 6, 0, 0,
'05:00 - 11:59', 'es_hora_punta');
INSERT INTO HORA VALUES (6, TO_DATE('2025-01-01 07:15:00', 'YYYY-MM-DD HH24:MI:SS'), 7, 15, 0,
'05:00 - 11:59', 'es_hora_punta');
INSERT INTO HORA VALUES (7, TO_DATE('2025-01-01 10:30:00', 'YYYY-MM-DD HH24:MI:SS'), 10, 30, 0,
'05:00 - 11:59', 'es_hora_punta');
INSERT INTO HORA VALUES (8, TO_DATE('2025-01-01 13:00:00', 'YYYY-MM-DD HH24:MI:SS'), 13, 0, 0,
'12:00 - 19:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (9, TO_DATE('2025-01-01 16:45:00', 'YYYY-MM-DD HH24:MI:SS'), 16, 45, 0,
'12:00 - 19:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (10, TO_DATE('2025-01-01 22:30:00', 'YYYY-MM-DD HH24:MI:SS'), 22, 30, 0,
'20:00 - 23:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (11, TO_DATE('2025-01-01 03:15:00', 'YYYY-MM-DD HH24:MI:SS'), 3, 15, 0,
'00:00 - 04:59', 'no_es_hora_punta');

----- DIMENSIÓN AEROPUERTO -----
INSERT INTO AEROPUERTO VALUES (1, 'LEMD', 'Aeropuerto Adolfo Suárez Madrid-Barajas', 'Madrid',
'Comunidad de Madrid', 'MD', 'Centro', 'España', 40.4983, -3.5676, 610.0, 'Europe/Madrid',
'internacional');
INSERT INTO AEROPUERTO VALUES (2, 'LEBL', 'Aeropuerto de Barcelona-El Prat', 'Barcelona',
'Cataluña', 'CT', 'Noreste', 'España', 41.2974, 2.0833, 4.0, 'Europe/Madrid', 'internacional');
INSERT INTO AEROPUERTO VALUES (3, 'LEZL', 'Aeropuerto de Sevilla-San Pablo', 'Sevilla',
'Andalucía', 'AN', 'Sur', 'España', 37.4180, -5.8931, 34.0, 'Europe/Madrid', 'nacional');

```

```

INSERT INTO AEROPUERTO VALUES (4, 'LEVC', 'Aeropuerto de Valencia', 'Valencia', 'Comunidad Valenciana', 'VC', 'Este', 'España', 39.4893, -0.4817, 69.0, 'Europe/Madrid', 'nacional');
INSERT INTO AEROPUERTO VALUES (5, 'LEAL', 'Aeropuerto de Alicante-Elche', 'Alicante', 'Comunidad Valenciana', 'VC', 'Este', 'España', 38.2822, -0.5581, 43.0, 'Europe/Madrid', 'internacional');

----- DIMENSIÓN AVION -----
INSERT INTO AVION VALUES (1, 'A320-214', 'EC-MFZ', 'Airbus A320-200', 'Airbus', 'Turbofan', 2, 180, 871.0, 2010, 2011, 'esta_activo');
INSERT INTO AVION VALUES (2, 'B737-800', 'EC-JFN', 'Boeing 737-800', 'Boeing', 'Turbofan', 2, 189, 842.0, 2012, 2013, 'esta_activo');
INSERT INTO AVION VALUES (3, 'A321-211', 'EC-LUN', 'Airbus A321-200', 'Airbus', 'Turbofan', 2, 220, 903.0, 2015, 2016, 'esta_activo');
INSERT INTO AVION VALUES (4, 'B787-8', 'EC-MIG', 'Boeing 787-8 Dreamliner', 'Boeing', 'Turbofan', 2, 296, 954.0, 2018, 2019, 'esta_activo');
INSERT INTO AVION VALUES (5, 'A330-300', 'EC-LXR', 'Airbus A330-300', 'Airbus', 'Turbofan', 2, 335, 913.0, 2008, 2009, 'no_esta_activo');

----- DIMENSIÓN OPERADORA -----
INSERT INTO OPERADORA VALUES (1, 'IBE', 'Iberia', 'España', 1927, 'esta_activo');
INSERT INTO OPERADORA VALUES (2, 'VLG', 'Vueling', 'España', 2004, 'esta_activo');
INSERT INTO OPERADORA VALUES (3, 'AEA', 'Air Europa', 'España', 1986, 'esta_activo');
INSERT INTO OPERADORA VALUES (4, 'RZR', 'Ryanair', 'Irlanda', 1984, 'esta_activo');
INSERT INTO OPERADORA VALUES (5, 'EZY', 'easyJet', 'Reino Unido', 1995, 'esta_activo');

----- DIMENSIÓN AEROLINEA -----
INSERT INTO AEROLINEA VALUES (1, 'IBE', 'Iberia Líneas Aéreas de España', 'Iberia', 'España', 'internacional', 1927, 'esta_activo');
INSERT INTO AEROLINEA VALUES (2, 'VLG', 'Vueling Airlines', 'Vueling', 'España', 'internacional', 2004, 'esta_activo');
INSERT INTO AEROLINEA VALUES (3, 'AEA', 'Air Europa Líneas Aéreas', 'Air Europa', 'España', 'internacional', 1986, 'esta_activo');
INSERT INTO AEROLINEA VALUES (4, 'RZR', 'Ryanair DAC', 'Ryanair', 'Irlanda', 'internacional', 1984, 'esta_activo');
INSERT INTO AEROLINEA VALUES (5, 'EZY', 'easyJet Airline Company Limited', 'easyJet', 'Reino Unido', 'internacional', 1995, 'esta_activo');

----- TABLA DE HECHOS VUELO -----
INSERT INTO VUELO VALUES ('IB3425', 1, 1, 1, 1, 2, 1, 1, 75.0, 78.5, 3.5, 505.0);
INSERT INTO VUELO VALUES ('VY2105', 2, 1, 2, 2, 1, 2, 2, 85.0, 82.0, 0, 620.0);
INSERT INTO VUELO VALUES ('UX5032', 3, 2, 3, 3, 1, 3, 3, 125.0, 130.0, 5.0, 920.0);
INSERT INTO VUELO VALUES ('IB6721', 1, 3, 4, 4, 1, 1, 1, 95.0, 98.0, 3.0, 680.0);
INSERT INTO VUELO VALUES ('FR4582', 4, 4, 5, 5, 2, 4, 4, 110.0, 105.0, 0, 780.0);
INSERT INTO VUELO VALUES ('VY8934', 2, 5, 1, 1, 3, 2, 2, 90.0, 95.0, 5.0, 625.0);
INSERT INTO VUELO VALUES ('U28217', 3, 2, 2, 2, 4, 5, 5, 80.0, 85.0, 5.0, 600.0);
INSERT INTO VUELO VALUES ('UX7654', 5, 3, 3, 3, 1, 3, 3, 140.0, 145.0, 5.0, 1050.0);
INSERT INTO VUELO VALUES ('IB2301', 6, 6, 1, 2, 3, 1, 1, 80.0, 82.0, 2.0, 620.0);
INSERT INTO VUELO VALUES ('IB5512', 7, 7, 4, 1, 5, 1, 1, 95.0, 100.0, 5.0, 680.0);
INSERT INTO VUELO VALUES ('IB9200', 8, 8, 1, 3, 2, 1, 2, 120.0, 118.0, 0, 920.0);
INSERT INTO VUELO VALUES ('VY1234', 9, 3, 2, 2, 5, 2, 2, 85.0, 90.0, 5.0, 620.0);
INSERT INTO VUELO VALUES ('VY7788', 10, 6, 2, 4, 1, 2, 1, 70.0, 68.0, 0, 505.0);
INSERT INTO VUELO VALUES ('VY3456', 11, 9, 1, 5, 3, 2, 2, 100.0, 105.0, 5.0, 780.0);
INSERT INTO VUELO VALUES ('UX1111', 12, 3, 3, 1, 4, 3, 3, 140.0, 138.0, 0, 1050.0);
INSERT INTO VUELO VALUES ('UX2222', 13, 10, 5, 2, 1, 3, 3, 90.0, 95.0, 5.0, 625.0);
INSERT INTO VUELO VALUES ('FR7890', 7, 11, 2, 5, 1, 4, 4, 110.0, 115.0, 5.0, 780.0);
INSERT INTO VUELO VALUES ('FR1122', 11, 8, 2, 3, 2, 4, 4, 125.0, 122.0, 0, 920.0);
INSERT INTO VUELO VALUES ('U23344', 8, 2, 1, 4, 2, 5, 5, 95.0, 98.0, 3.0, 680.0);
INSERT INTO VUELO VALUES ('U25566', 9, 7, 2, 1, 5, 5, 5, 82.0, 85.0, 3.0, 600.0);
INSERT INTO VUELO VALUES ('IB7001', 6, 1, 2, 3, 1, 1, 1, 125.0, 130.0, 5.0, 920.0);
INSERT INTO VUELO VALUES ('VY8002', 10, 4, 2, 1, 2, 2, 2, 80.0, 75.0, 0, 505.0);
INSERT INTO VUELO VALUES ('UX9003', 12, 9, 4, 5, 3, 3, 3, 110.0, 108.0, 0, 780.0);

```

### 6.1.3. Script de consultas

**Consulta 1 (Aerolíneas con más de 1 vuelo):** Evalúa la actividad y puntualidad de las aerolíneas para aquellas que hayan hecho más de 1 vuelo.

```
SQL
SELECT
    a.nombre_comercial,
    COUNT(*) as total_vuelos,
    ROUND(AVG(v.retrazo_minutos), 2) as retraso_promedio,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_promedio,
    ROUND(SUM(v.distancia_km), 2) as distancia_total_km
FROM VUELO v
JOIN AEROLINEA a ON v.id_aerolinea = a.id_aerolinea
GROUP BY a.nombre_comercial
HAVING COUNT(*) > 1
ORDER BY total_vuelos DESC;
```

| Nombre Comercial | Total Vuelos | Retraso Promedio | Duración Promedio | Distancia Total |
|------------------|--------------|------------------|-------------------|-----------------|
| Iberica          | 6            | 3.08             | 101.08            | 4325            |
| Vueling          | 6            | 2.5              | 85.83             | 3655            |
| Air Europa       | 5            | 3.0              | 123.2             | 4425            |
| Ryanair          | 3            | 1.67             | 114               | 2480            |
| EasyJet          | 3            | 3.67             | 89.33             | 1880            |

Ryanair es la aerolínea más puntual (1.67 min) mientras que easyJet tiene el mayor retraso (3.67 min). Iberia y Vueling tienen más vuelos (6 cada una) y Air Europa opera los vuelos más largos en duración (123.2 min promedio).

**Consulta 2 (Análisis jerárquico por aerolínea y operadora):** Muestra quién vende el billete (aerolínea) y quién realmente hace volar el avión (operadora), utilizando *ROLLUP* para generar subtotales automáticos.

```
SQL
SELECT
    a.nombre_comercial as aerolinea,
    o.nombre_operadora as operadora,
    COUNT(*) as num_vuelos,
    ROUND(SUM(v.distancia_km), 2) as distancia_total_km,
    ROUND(AVG(v.retrazo_minutos), 2) as retraso_promedio,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_promedio
FROM VUELO v
JOIN AEROLINEA a ON v.id_aerolinea = a.id_aerolinea
JOIN OPERADORA o ON v.id_operadora = o.id_operadora
GROUP BY ROLLUP(a.nombre_comercial, o.nombre_operadora)
ORDER BY a.nombre_comercial NULLS LAST, o.nombre_operadora NULLS LAST;
```

| Aerolínea  | Operadora  | Vuelos | Dist. Total | Retraso Prom. | Duración Prom. |
|------------|------------|--------|-------------|---------------|----------------|
| Air Europa | Air Europa | 5      | 1425        | 3.0           | 123.2          |
| Air Europa | SUBTOTAL   | 5      | 4425        | 3.0           | 123.2          |
| Iberia     | Iberia     | 5      | 3405        | 3.7           | 97.7           |
| Iberia     | Vueling    | 1      | 920         | 0             | 118            |
| Iberia     | SUBTOTAL   | 6      | 4325        | 3.08          | 101.08         |
| Vueling    | Iberia     | 1      | 505         | 0             | 68             |
| Vueling    | Vueling    | 5      | 3150        | 3.0           | 89.4           |
| Vueling    | SUBTOTAL   | 6      | 3655        | 2.5           | 85.83          |
| Ryanair    | Ryanair    | 3      | 2480        | 1.67          | 114            |
| Ryanair    | SUBTOTAL   | 3      | 2480        | 1.67          | 114            |
| EasyJet    | EasyJet    | 3      | 1880        | 3.67          | 89.33          |
| EasyJet    | SUBTOTAL   | 3      | 1880        | 3.67          | 89.33          |

En la mayoría de casos, quienes venden el billete son quienes operan también el avión, excepto en el caso de Iberia y Vueling. Esto pasa cuando dos aerolíneas colaboran, y los subtotales muestran el total de cada aerolínea sumando todos sus vuelos.

**Consulta 3 (Análisis por temporada y fin de semana):** Analiza la puntualidad según la época del año y si es fin de semana o no, usando *CUBE* para mostrar todas las combinaciones.

SQL

SELECT

```

    f.temporada,
    f.es_fin_semana,
    COUNT(*) as total_vuelos,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_promedio,
    ROUND(AVG(v.retraso_minutos), 2) as retraso_promedio,
    ROUND(SUM(v.distancia_km), 2) as distancia_total,
    ROUND(AVG(v.tiempo_real_minutos - v.tiempo_estimado_minutos), 2) as diferencia_tiempo
FROM VUELO v
JOIN FECHA f ON v.id_fecha = f.id_fecha
GROUP BY CUBE(f.temporada, f.es_fin_semana)
ORDER BY f.temporada NULLS LAST, f.es_fin_semana NULLS LAST;
```

| Temporada | Fin de Semana    | Vuelos | Duración | Retraso | Dist. Total | Dif. Tiempo |
|-----------|------------------|--------|----------|---------|-------------|-------------|
| invierno  | no_es_fin_semana | 1      | 145      | 5       | 1050        | 5           |
| invierno  | SUBTOTAL         | 1      | 145      | 5       | 1050        | 5           |
| otogno    | es_fin_semana    | 3      | 113.67   | 1.67    | 2455        | 0.33        |
| otogno    | no_es_fin_semana | 4      | 101      | 2.5     | 2945        | 1           |
| otogno    | SUBTOTAL         | 7      | 106.43   | 2.14    | 5400        | 0.71        |
| primavera | no_es_fin_semana | 7      | 101.79   | 2.79    | 5025        | 1.79        |

|           |                  |    |        |      |       |      |
|-----------|------------------|----|--------|------|-------|------|
| primavera | SUBTOTAL         | 7  | 101.79 | 2.79 | 5025  | 1.79 |
| verano    | es_fin_semana    | 6  | 100.33 | 4.17 | 4280  | 4.17 |
| verano    | no_es_fin_semana | 2  | 71.5   | 0    | 1010  | -3.5 |
| TODAS     | es_fin_semana    | 9  | 104.78 | 3.33 | 4735  | 2.89 |
| TODAS     | no_es_fin_semana | 14 | 100.32 | 2.46 | 10030 | 1.04 |
| TOTAL     | TOTAL            | 23 | 102.07 | 2.8  | 16765 | 1.76 |

Los fines de semana (y sobre todo en verano) tienen más retraso que entre semana. Otoño es la temporada más puntual.

**Consulta 4 (Aeropuertos con distancia promedio mayor a 600km):** Identifica los aeropuertos que operan rutas con una distancia media superior a 600km.

SQL

```
SELECT
    ap.ciudad,
    ap.nombre_aeropuerto,
    COUNT(*) as num_vuelos,
    ROUND(AVG(v.distancia_km), 2) as distancia_promedio_km,
    ROUND(MIN(v.distancia_km), 2) as distancia_minima,
    ROUND(MAX(v.distancia_km), 2) as distancia_maxima,
    ROUND(AVG(v.retraso_minutos), 2) as retraso_promedio
FROM VUELO v
JOIN AEROPUERTO ap ON v.id_aeropuerto_origen = ap.id_aeropuerto
GROUP BY ap.ciudad, ap.nombre_aeropuerto
HAVING AVG(v.distancia_km) > 600
ORDER BY distancia_promedio_km DESC;
```

| Ciudad    | Aeropuerto        | Vuelos | Dist. Prom. | Dist. Min | Dist. Max | Retraso |
|-----------|-------------------|--------|-------------|-----------|-----------|---------|
| Sevilla   | Sevilla-San Pablo | 5      | 946         | 920       | 1050      | 3       |
| Alicante  | Alicante-Elche    | 4      | 780         | 780       | 780       | 2.5     |
| Madrid    | Madrid-Barajas    | 6      | 660.83      | 505       | 1050      | 2.75    |
| Valencia  | Valencia          | 3      | 621.67      | 505       | 680       | 2       |
| Barcelona | Barcelona-El Prat | 5      | 617         | 600       | 625       | 3.4     |

Los resultados evidencian que los cinco aeropuertos analizados cumplen con el criterio de rutas de más de 600 km de media, siendo el aeropuerto de Sevilla-San Pablo el que presenta los trayectos más largos (946 km de promedio) y el aeropuerto de Valencia el más puntual (2 minutos de retraso medio).

**Consulta 5, 6 y 7 (Análisis por niveles geográficos):** Estas tres consultas implementan un análisis jerárquico (*DRILL DOWN*), que permite ver la información a diferentes niveles de detalle geográfico: desde regiones de origen hasta los aeropuertos específicos.

SQL

```
-- 5. Análisis por región de ORIGEN (DRILL DOWN Nivel 1)
SELECT
    ap.region,
    COUNT(*) as total_vuelos,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio,
    ROUND(SUM(v.distancia_km), 2) as distancia_total,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_promedio
FROM VUELO v
JOIN AEROPUERTO ap ON v.id_aeropuerto_origen = ap.id_aeropuerto
GROUP BY ap.region
ORDER BY total_vuelos DESC;

-- 6. Análisis por región → ciudad de ORIGEN (DRILL DOWN Nivel 2)
SELECT
    ap.region,
    ap.ciudad,
    COUNT(*) as total_vuelos,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio,
    ROUND(SUM(v.distancia_km), 2) as distancia_total,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_promedio
FROM VUELO v
JOIN AEROPUERTO ap ON v.id_aeropuerto_origen = ap.id_aeropuerto
GROUP BY ap.region, ap.ciudad
ORDER BY ap.region, total_vuelos DESC;

-- 7. Análisis por región → ciudad → aeropuerto de ORIGEN (DRILL DOWN Nivel 3)
SELECT
    ap.region,
    ap.ciudad,
    ap.nombre_aeropuerto,
    ap.tipo_aeropuerto,
    COUNT(*) as total_vuelos,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio,
    ROUND(SUM(v.distancia_km), 2) as distancia_total
FROM VUELO v
JOIN AEROPUERTO ap ON v.id_aeropuerto_origen = ap.id_aeropuerto
GROUP BY ap.region, ap.ciudad, ap.nombre_aeropuerto, ap.tipo_aeropuerto
ORDER BY ap.region, ap.ciudad, total_vuelos DESC;
```

| Región  | Vuelos | Retraso | Dist. Total | Duración |
|---------|--------|---------|-------------|----------|
| Este    | 7      | 2.29    | 4955        | 99.57    |
| Centro  | 6      | 2.75    | 3965        | 95.25    |
| Sur     | 5      | 3       | 4730        | 129      |
| Noreste | 5      | 3.4     | 3085        | 86.8     |

Los resultados indican que la región Este concentra más número de vuelos y además presenta la mayor puntualidad. Por el contrario, la región Sur tiene los vuelos más largos en tiempo, mientras que el Noreste muestra la menor duración media pero también mayor retraso.

| Región  | Ciudad    | Vuelos | Retraso | Dist. Total | Duración |
|---------|-----------|--------|---------|-------------|----------|
| Centro  | Madrid    | 6      | 2.75    | 3965        | 95.25    |
| Este    | Alicante  | 4      | 2.5     | 3120        | 108.25   |
| Este    | Valencia  | 3      | 2       | 1865        | 88       |
| Noreste | Barcelona | 5      | 3.4     | 3085        | 86.8     |
| Sur     | Sevilla   | 5      | 3       | 4730        | 129      |

El segundo nivel profundiza el análisis incorporando la dimensión ciudad, dentro de cada región. Se observa que el Este agrupa dos ciudades (Alicante y Valencia), siendo Valencia la más puntual de todas.

| Región  | Ciudad    | Aeropuerto        | Tipo          | Vuelos | Retraso | Dist. Total |
|---------|-----------|-------------------|---------------|--------|---------|-------------|
| Centro  | Madrid    | Madrid-Barajas    | Internacional | 6      | 2.75    | 3965        |
| Este    | Alicante  | Alicante-Elche    | Internacional | 4      | 2.5     | 3120        |
| Este    | Valencia  | Valencia          | Nacional      | 3      | 2       | 1865        |
| Noreste | Barcelona | Barcelona-El Prat | Internacional | 5      | 3.4     | 3085        |
| Sur     | Sevilla   | Sevilla-San Pablo | Nacional      | 5      | 3       | 4730        |

El tercer nivel desciende al máximo detalle geográfico, mostrando cada aeropuerto junto con su tipo (Nacional o Internacional). El aeropuerto más puntual es el de Valencia, mientras que el menos puntual es el de Barcelona.

**Consulta 8 (Análisis por año, trimestre y mes):** Muestra cómo se distribuyen los vuelos a lo largo del año, con totales automáticos por trimestre y año completo usando *ROLLUP*.

```
SQL
SELECT
    f.agno,
    f.trimestre,
    f.nombre_mes,
    COUNT(*) as num_vuelos,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio,
    ROUND(SUM(v.distancia_km), 2) as km_totales,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_promedio
FROM VUELO v
JOIN FECHA f ON v.id_fecha = f.id_fecha
GROUP BY ROLLUP(f.agno, f.trimestre, f.nombre_mes)
ORDER BY f.agno NULLS LAST, f.trimestre NULLS LAST, f.nombre_mes NULLS LAST;
```

| Año  | Trim. | Mes   | Vuelos | Retraso | Km total | Duración |
|------|-------|-------|--------|---------|----------|----------|
| 2025 | 1     | Enero | 6      | 3.08    | 3970     | 94.25    |
| 2025 | 1     | Marzo | 2      | 5       | 1460     | 107.5    |
| 2025 | 1     | (Q1)  | 8      | 3.56    | 5430     | 97.56    |
| 2025 | 2     | Abril | 2      | 1.5     | 1600     | 108      |
| 2025 | 2     | Junio | 2      | 4       | 1220     | 87.5     |

|      |       |            |    |      |       |        |
|------|-------|------------|----|------|-------|--------|
| 2025 | 2     | (Q2)       | 4  | 2.75 | 2530  | 97.75  |
| 2025 | 3     | Agosto     | 1  | 1    | 780   | 105    |
| 2025 | 3     | Julio      | 4  | 2.5  | 2530  | 89.5   |
| 2025 | 3     | Septiembre | 2  | 2.5  | 1700  | 113.5  |
| 2025 | 3     | (Q3)       | 7  | 2.14 | 5010  | 98.57  |
| 2025 | 4     | Octubre    | 2  | 0    | 1830  | 123    |
| 2025 | 4     | Noviembre  | 1  | 5    | 625   | 95     |
| 2025 | 4     | Diciembre  | 1  | 5    | 1050  | 145    |
| 2025 | 4     | (Q4)       | 4  | 2.5  | 3505  | 121.5  |
| 2025 | (AÑO) | (AÑO)      | 23 | 2.8  | 16765 | 102.07 |

El primer trimestre (Q1) tiene más vuelos, pero el tercer trimestre (Q3) es el más puntual. Agosto y Octubre tienen un total de 0 minutos de retraso. En cambio, los meses de Marzo, Noviembre y Diciembre son los meses más problemáticos en cuanto al tiempo de retraso. Los totales aparecen automáticamente donde pone Q1, Q2, Q3, Q4 y AÑO.

**Consulta 9 (Análisis por hora del día y festivos):** Compara la puntualidad según la hora del día (madrugada, mañana, tarde y noche) y si es festivo o no, usando *CUBE* para obtener todas las combinaciones.

SQL

SELECT

```
h.franja_horaria,
f.es_festivo,
COUNT(*) as total_vuelos,
ROUND(AVG(v.retraso_minutos), 2) as retraso_promedio,
ROUND(AVG(v.tiempo_real_minutos - v.tiempo_estimado_minutos), 2) as diferencia_tiempo,
ROUND(SUM(v.distancia_km), 2) as distancia_total,
ROUND(AVG(v.distancia_km), 2) as distancia_promedio
```

FROM VUELO v

JOIN HORA h ON v.id\_hora = h.id\_hora

JOIN FECHA f ON v.id\_fecha = f.id\_fecha

GROUP BY CUBE(h.franja\_horaria, f.es\_festivo)

ORDER BY h.franja\_horaria NULLS LAST, f.es\_festivo NULLS LAST;

| Franja        | Festivo    | Vuelos | Retraso | Dif. Tiempo | Dist. Total |
|---------------|------------|--------|---------|-------------|-------------|
| 00:00 - 04:59 | no_festivo | 1      | 5       | 5           | 780         |
| 00:00 - 04:59 | (SUBTOTAL) | 1      | 5       | 5           | 780         |
| 05:00 - 11:59 | no_festivo | 8      | 2.94    | 2.31        | 5075        |
| 05:00 - 11:59 | (SUBTOTAL) | 8      | 2.94    | 2.31        | 5075        |
| 12:00 - 19:59 | es_festivo | 3      | 1.67    | 0.33        | 2880        |
| 12:00 - 19:59 | no_festivo | 8      | 3.25    | 2.63        | 6120        |
| 12:00 - 19:59 | (SUBTOTAL) | 11     | 2.82    | 2           | 9000        |
| 20:00 - 23:59 | es_festivo | 2      | 2.5     | 0           | 1405        |

|               |            |    |      |       |       |
|---------------|------------|----|------|-------|-------|
| 20:00 - 23:59 | no_festivo | 1  | 0    | -5    | 505   |
| 20:00 - 23:59 | (SUBTOTAL) | 3  | 1.67 | -1.67 | 1910  |
| (TODAS)       | es_festivo | 5  | 2    | 0.2   | 4285  |
| (TODAS)       | no_festivo | 18 | 3.03 | 2.19  | 12480 |

Las tardes, de 12:00 - 19:59, son la franja más usada, mientras que las noches, de 20:00 - 23:59, son la franja más puntual y algunos vuelos llegan antes de lo previsto.

Los días festivos son más puntuales que los normales.

**Consulta 10 (Modelos de avión modernos y usados):** Muestra solo los aviones más rápidos (más de 850 km/h) que se usan con más frecuencia (más de 1 vuelo), usando *HAVING* para filtrar.

SQL

```
SELECT
    av.fabricante,
    av.nombre_modelo,
    av.velocidad_maxima,
    av.num_asientos,
    COUNT(*) as vuelos_realizados,
    ROUND(AVG(v.distancia_km), 2) as distancia_promedio,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_real_promedio,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio,
    ROUND(AVG(v.distancia_km / v.tiempo_real_minutos * 60), 2) as velocidad_efectiva_kmh,
    ROUND(SUM(v.distancia_km), 2) as distancia_total_recorrida
FROM VUELO v
JOIN AVION av ON v.id_avion = av.id_avion
GROUP BY av.fabricante, av.nombre_modelo, av.velocidad_maxima, av.num_asientos
HAVING COUNT(*) > 1 AND av.velocidad_maxima > 850
ORDER BY vuelos_realizados DESC, distancia_promedio DESC;
```

| Fabricante | Modelo           | Vel. Max | Asientos | Vuelos | Dist. Prom | Retraso | Vel. Efectiva | Dist. Total |
|------------|------------------|----------|----------|--------|------------|---------|---------------|-------------|
| Airbus     | A320-200         | 871      | 180      | 6      | 688.33     | 3.08    | 427.37        | 4130        |
| Airbus     | A321-200         | 903      | 220      | 3      | 1006.67    | 3.33    | 438.54        | 3020        |
| Boeing     | 787-8 Dreamliner | 954      | 296      | 3      | 713.33     | 2.67    | 419.22        | 2140        |
| Airbus     | A330 - 300       | 913      | 335      | 2      | 702.5      | 2.5     | 420.23        | 1405        |

Solo 4 modelos cumplen los requisitos. El A320-200 es el más usado, el A321-200 vuela más rápido en la práctica y el A330-300 es el más puntual y el que más pasajeros lleva.

## 6.2. Vuelos comerciales con información de pasajeros

Este segundo modelo, extiende el esquema en estrella anterior incorporando información detallada sobre los pasajeros que viajan en cada vuelo.

El modelo añade 3 nuevas tablas (**PASAJERO**, **GRUPO** y **GRUPO\_PASAJEROS**) al esquema base, manteniendo las 6 dimensiones originales y modificando la tabla de hechos, **VUELO**, para que incluya la nueva clave foránea (*id\_grupo*).

La implementación se ha realizado mediante tres scripts SQL:

- *crear\_tablas\_vuelos\_pasajero.sql*
- *insertar\_datos\_vuelos\_pasajero.sql*
- *consultas\_vuelos\_pasajero.sql*

### 6.2.1. Script de creación de tablas

El script de creación extiende el modelo anterior añadiendo las siguientes tablas:

- **DIM\_PASAJERO**: Almacena la información personal de los pasajeros.

SQL

```
CREATE TABLE PASAJERO (
    id_pasajero INTEGER PRIMARY KEY,
    nombre VARCHAR2(100) NOT NULL,
    apellido1 VARCHAR2(100) NOT NULL,
    apellido2 VARCHAR2(100),
    tipo_documento VARCHAR2(20) NOT NULL CHECK (tipo_documento IN ('DNI', 'NIE', 'pasaporte',
'otro')),
    num_documento VARCHAR2(50) UNIQUE NOT NULL, -- Clave Natural
    telefono VARCHAR2(20),
    fecha_nacimiento DATE NOT NULL,
    nacionalidad VARCHAR2(100) NOT NULL,
    categoria_cliente VARCHAR2(20) NOT NULL CHECK (categoria_cliente IN ('vip', 'estandar'))
);
```

- **DIM\_GRUPO (tabla auxiliar)**: Tabla que representa grupos de pasajeros que viajan juntos (grupos grandes o pequeños, parejas, viajeros individuales...)

SQL

```
CREATE TABLE GRUPO (
    id_grupo INTEGER PRIMARY KEY
);
```

- **GRUPO\_PASAJEROS (tabla puente)**: Implementa la relación muchos a muchos entre pasajeros y grupos, permitiendo que un pasajero pueda pertenecer a múltiples grupos (en múltiples vuelos) y un grupo pueda contener múltiples pasajeros.

SQL

```
CREATE TABLE GRUPO_PASAJEROS (
    id_grupo INTEGER NOT NULL,
    id_pasajero INTEGER NOT NULL,
    PRIMARY KEY (id_grupo, id_pasajero),
    FOREIGN KEY (id_grupo) REFERENCES GRUPO(id_grupo) ON DELETE CASCADE,
    FOREIGN KEY (id_pasajero) REFERENCES PASAJERO(id_pasajero) ON DELETE CASCADE
);
```

- **TABLA DE HECHOS (VUELO):** La tabla **VUELO** se modifica para incluir la dimensión grupo, pero sin modificar el grano.

SQL

```
CREATE TABLE VUELO (
    codigo_vuelo VARCHAR2(50) NOT NULL,
    id_fecha INTEGER NOT NULL,
    id_hora INTEGER NOT NULL,
    id_avion INTEGER NOT NULL,
    id_aeropuerto_origen INTEGER NOT NULL,
    id_aeropuerto_destino INTEGER NOT NULL,
    id_aerolinea INTEGER NOT NULL,
    id_operadora INTEGER NOT NULL,
    id_grupo INTEGER NOT NULL,
    tiempo_estimado_minutos REAL NOT NULL,
    tiempo_real_minutos REAL NOT NULL,
    retardo_minutos REAL NOT NULL,
    distancia_km REAL NOT NULL,
    PRIMARY KEY (codigo_vuelo, id_fecha, id_hora),
    FOREIGN KEY (id_avion) REFERENCES AVION(id_avion) ON DELETE CASCADE,
    FOREIGN KEY (id_fecha) REFERENCES FECHA(id_fecha) ON DELETE CASCADE,
    FOREIGN KEY (id_hora) REFERENCES HORA(id_hora) ON DELETE CASCADE,
    FOREIGN KEY (id_aeropuerto_origen) REFERENCES AEROPUERTO(id_aeropuerto) ON DELETE CASCADE,
    FOREIGN KEY (id_aeropuerto_destino) REFERENCES AEROPUERTO(id_aeropuerto) ON DELETE CASCADE,
    FOREIGN KEY (id_aerolinea) REFERENCES AEROLINEA(id_aerolinea) ON DELETE CASCADE,
    FOREIGN KEY (id_operadora) REFERENCES OPERADORA(id_operadora) ON DELETE CASCADE,
    FOREIGN KEY (id_grupo) REFERENCES GRUPO(id_grupo) ON DELETE CASCADE,
    CONSTRAINT chk_tiempo_positivo CHECK (tiempo_estimado_minutos > 0 AND tiempo_real_minutos > 0),
    CONSTRAINT chk_distancia_positiva CHECK (distancia_km > 0),
    CONSTRAINT chk_retardo_no_negativo CHECK (retardo_minutos >= 0),
    CONSTRAINT chk_aeropuertos_diferentes CHECK (id_aeropuerto_origen != id_aeropuerto_destino)
);
```

### 6.2.2. Script de inserción de datos

El script de inserción puebla las tablas con datos, manteniendo los del modelo anterior.

SQL

```
----- DIMENSIÓN PASAJERO -----
INSERT INTO PASAJERO VALUES (1, 'Carlos', 'García', 'López', 'DNI', '12345678A', '+34600111222',
TO_DATE('1980-05-15', 'YYYY-MM-DD'), 'España', 'vip');
INSERT INTO PASAJERO VALUES (2, 'María', 'Rodríguez', 'Martín', 'DNI', '23456789B',
'+34600222333', TO_DATE('1985-08-20', 'YYYY-MM-DD'), 'España', 'vip');
INSERT INTO PASAJERO VALUES (3, 'John', 'Smith', NULL, 'pasaporte', 'AB123456', '+44700111222',
TO_DATE('1978-03-10', 'YYYY-MM-DD'), 'Reino Unido', 'vip');
```

```

INSERT INTO PASAJERO VALUES (4, 'Sophie', 'Dubois', NULL, 'pasaporte', 'FR987654', '+33600111222',
TO_DATE('1990-11-25', 'YYYY-MM-DD'), 'Francia', 'vip');
INSERT INTO PASAJERO VALUES (5, 'Hans', 'Müller', NULL, 'pasaporte', 'DE456789', '+49170111222',
TO_DATE('1975-07-08', 'YYYY-MM-DD'), 'Alemania', 'vip');
INSERT INTO PASAJERO VALUES (6, 'Ana', 'López', 'Fernández', 'DNI', '34567890C', '+34600333444',
TO_DATE('1992-02-14', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (7, 'Pedro', 'Sánchez', 'García', 'DNI', '45678901D', '+34600444555',
TO_DATE('1988-06-22', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (8, 'Laura', 'Martínez', 'Ruiz', 'DNI', '56789012E', '+34600555666',
TO_DATE('1995-09-30', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (9, 'David', 'González', 'Pérez', 'DNI', '67890123F', '+34600666777',
TO_DATE('1982-12-05', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (10, 'Elena', 'Hernández', 'Díaz', 'DNI', '78901234G', '+34600777888',
TO_DATE('1991-04-18', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (11, 'Miguel', 'Jiménez', 'Moreno', 'DNI', '89012345H',
'+34600888999', TO_DATE('1987-01-27', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (12, 'Isabel', 'Álvarez', 'Romero', 'DNI', '90123456I',
'+34600999000', TO_DATE('1993-08-13', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (13, 'Javier', 'Torres', 'Navarro', 'DNI', '01234567J',
'+34601000111', TO_DATE('1984-10-09', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (14, 'Carmen', 'Ramírez', 'Gil', 'DNI', '11234567K', '+34601111222',
TO_DATE('1996-03-21', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (15, 'Antonio', 'Vázquez', 'Serrano', 'DNI', '21234567L',
'+34601222333', TO_DATE('1989-07-16', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (16, 'Lucía', 'Ramos', 'Blanco', 'DNI', '31234567M', '+34601333444',
TO_DATE('1994-11-02', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (17, 'Francisco', 'Castro', 'Suárez', 'DNI', '41234567N',
'+34601444555', TO_DATE('1981-05-28', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (18, 'Marta', 'Ortega', 'Rubio', 'DNI', '51234567O', '+34601555666',
TO_DATE('1997-09-14', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (19, 'Raúl', 'Delgado', 'Molina', 'DNI', '61234567P', '+34601666777',
TO_DATE('1986-02-07', 'YYYY-MM-DD'), 'España', 'estandar');
INSERT INTO PASAJERO VALUES (20, 'Beatriz', 'Morales', 'Ortiz', 'DNI', '71234567Q',
'+34601777888', TO_DATE('1992-06-19', 'YYYY-MM-DD'), 'España', 'estandar');

```

----- TABLA GRUPO -----

```

INSERT INTO GRUPO VALUES (1);
INSERT INTO GRUPO VALUES (2);
INSERT INTO GRUPO VALUES (3);
INSERT INTO GRUPO VALUES (4);
INSERT INTO GRUPO VALUES (5);
INSERT INTO GRUPO VALUES (6);
INSERT INTO GRUPO VALUES (7);
INSERT INTO GRUPO VALUES (8);
INSERT INTO GRUPO VALUES (9);
INSERT INTO GRUPO VALUES (10);
INSERT INTO GRUPO VALUES (11);
INSERT INTO GRUPO VALUES (12);
INSERT INTO GRUPO VALUES (13);
INSERT INTO GRUPO VALUES (14);
INSERT INTO GRUPO VALUES (15);
INSERT INTO GRUPO VALUES (16);
INSERT INTO GRUPO VALUES (17);
INSERT INTO GRUPO VALUES (18);
INSERT INTO GRUPO VALUES (19);
INSERT INTO GRUPO VALUES (20);
INSERT INTO GRUPO VALUES (21);
INSERT INTO GRUPO VALUES (22);
INSERT INTO GRUPO VALUES (23);

```

## ----- TABLA PUENTE GRUPO\_PASAJEROS -----

```

INSERT INTO GRUPO_PASAJEROS VALUES (1, 1);
INSERT INTO GRUPO_PASAJEROS VALUES (2, 2);
INSERT INTO GRUPO_PASAJEROS VALUES (2, 6);
INSERT INTO GRUPO_PASAJEROS VALUES (3, 7);
INSERT INTO GRUPO_PASAJEROS VALUES (3, 8);
INSERT INTO GRUPO_PASAJEROS VALUES (3, 14);
INSERT INTO GRUPO_PASAJEROS VALUES (3, 18);
INSERT INTO GRUPO_PASAJEROS VALUES (4, 3);
INSERT INTO GRUPO_PASAJEROS VALUES (5, 9);
INSERT INTO GRUPO_PASAJEROS VALUES (5, 10);
INSERT INTO GRUPO_PASAJEROS VALUES (6, 11);
INSERT INTO GRUPO_PASAJEROS VALUES (6, 12);
INSERT INTO GRUPO_PASAJEROS VALUES (6, 13);
INSERT INTO GRUPO_PASAJEROS VALUES (7, 4);
INSERT INTO GRUPO_PASAJEROS VALUES (8, 15);
INSERT INTO GRUPO_PASAJEROS VALUES (8, 16);
INSERT INTO GRUPO_PASAJEROS VALUES (8, 17);
INSERT INTO GRUPO_PASAJEROS VALUES (8, 19);
INSERT INTO GRUPO_PASAJEROS VALUES (8, 20);
INSERT INTO GRUPO_PASAJEROS VALUES (9, 5);
INSERT INTO GRUPO_PASAJEROS VALUES (10, 6);
INSERT INTO GRUPO_PASAJEROS VALUES (10, 7);
INSERT INTO GRUPO_PASAJEROS VALUES (11, 8);
INSERT INTO GRUPO_PASAJEROS VALUES (12, 9);
INSERT INTO GRUPO_PASAJEROS VALUES (12, 10);
INSERT INTO GRUPO_PASAJEROS VALUES (12, 11);
INSERT INTO GRUPO_PASAJEROS VALUES (13, 1);
INSERT INTO GRUPO_PASAJEROS VALUES (13, 2);
INSERT INTO GRUPO_PASAJEROS VALUES (14, 12);
INSERT INTO GRUPO_PASAJEROS VALUES (15, 13);
INSERT INTO GRUPO_PASAJEROS VALUES (15, 14);
INSERT INTO GRUPO_PASAJEROS VALUES (15, 15);
INSERT INTO GRUPO_PASAJEROS VALUES (15, 16);
INSERT INTO GRUPO_PASAJEROS VALUES (16, 3);
INSERT INTO GRUPO_PASAJEROS VALUES (17, 17);
INSERT INTO GRUPO_PASAJEROS VALUES (17, 18);
INSERT INTO GRUPO_PASAJEROS VALUES (18, 19);
INSERT INTO GRUPO_PASAJEROS VALUES (19, 1);
INSERT INTO GRUPO_PASAJEROS VALUES (19, 3);
INSERT INTO GRUPO_PASAJEROS VALUES (19, 4);
INSERT INTO GRUPO_PASAJEROS VALUES (20, 20);
INSERT INTO GRUPO_PASAJEROS VALUES (20, 6);
INSERT INTO GRUPO_PASAJEROS VALUES (21, 5);
INSERT INTO GRUPO_PASAJEROS VALUES (22, 7);
INSERT INTO GRUPO_PASAJEROS VALUES (22, 8);
INSERT INTO GRUPO_PASAJEROS VALUES (22, 9);
INSERT INTO GRUPO_PASAJEROS VALUES (22, 10);
INSERT INTO GRUPO_PASAJEROS VALUES (23, 11);
INSERT INTO GRUPO_PASAJEROS VALUES (23, 12);
INSERT INTO GRUPO_PASAJEROS VALUES (23, 13);

```

## ----- TABLA DE HECHOS VUELO -----

```

INSERT INTO VUELO VALUES ('IB3425', 1, 1, 1, 1, 2, 1, 1, 1, 75.0, 78.5, 3.5, 505.0);
INSERT INTO VUELO VALUES ('VY2105', 2, 1, 2, 2, 1, 2, 2, 2, 85.0, 82.0, 0, 620.0);
INSERT INTO VUELO VALUES ('UX5032', 3, 2, 3, 3, 1, 3, 3, 3, 125.0, 130.0, 5.0, 920.0);
INSERT INTO VUELO VALUES ('IB6721', 1, 3, 4, 4, 1, 1, 1, 4, 95.0, 98.0, 3.0, 680.0);
INSERT INTO VUELO VALUES ('FR4582', 4, 4, 5, 5, 2, 4, 4, 5, 110.0, 105.0, 0, 780.0);
INSERT INTO VUELO VALUES ('VY8934', 2, 5, 1, 1, 3, 2, 2, 6, 90.0, 95.0, 5.0, 625.0);
INSERT INTO VUELO VALUES ('U28217', 3, 2, 2, 2, 4, 5, 5, 7, 80.0, 85.0, 5.0, 600.0);
INSERT INTO VUELO VALUES ('UX7654', 5, 3, 3, 3, 1, 3, 3, 8, 140.0, 145.0, 5.0, 1050.0);

```

```

INSERT INTO VUELO VALUES ('IB2301', 6, 6, 1, 2, 3, 1, 1, 9, 80.0, 82.0, 2.0, 620.0);
INSERT INTO VUELO VALUES ('IB5512', 7, 7, 4, 1, 5, 1, 1, 10, 95.0, 100.0, 5.0, 680.0);
INSERT INTO VUELO VALUES ('IB9200', 8, 8, 1, 3, 2, 1, 2, 11, 120.0, 118.0, 0, 920.0);
INSERT INTO VUELO VALUES ('VY1234', 9, 3, 2, 2, 5, 2, 2, 12, 85.0, 90.0, 5.0, 620.0);
INSERT INTO VUELO VALUES ('VY7788', 10, 6, 2, 4, 1, 2, 1, 13, 70.0, 68.0, 0, 505.0);
INSERT INTO VUELO VALUES ('VY3456', 11, 9, 1, 5, 3, 2, 2, 14, 100.0, 105.0, 5.0, 780.0);
INSERT INTO VUELO VALUES ('UX1111', 12, 3, 3, 1, 4, 3, 3, 15, 140.0, 138.0, 0, 1050.0);
INSERT INTO VUELO VALUES ('UX2222', 13, 10, 5, 2, 1, 3, 3, 16, 90.0, 95.0, 5.0, 625.0);
INSERT INTO VUELO VALUES ('FR7890', 7, 11, 2, 5, 1, 4, 4, 17, 110.0, 115.0, 5.0, 780.0);
INSERT INTO VUELO VALUES ('FR1122', 11, 8, 2, 3, 2, 4, 4, 18, 125.0, 122.0, 0, 920.0);
INSERT INTO VUELO VALUES ('U23344', 8, 2, 1, 4, 2, 5, 5, 19, 95.0, 98.0, 3.0, 680.0);
INSERT INTO VUELO VALUES ('U25566', 9, 7, 2, 1, 5, 5, 5, 20, 82.0, 85.0, 3.0, 600.0);
INSERT INTO VUELO VALUES ('IB7001', 6, 1, 2, 3, 1, 1, 1, 21, 125.0, 130.0, 5.0, 920.0);
INSERT INTO VUELO VALUES ('VY8002', 10, 4, 2, 1, 2, 2, 2, 22, 80.0, 75.0, 0, 505.0);
INSERT INTO VUELO VALUES ('UX9003', 12, 9, 4, 5, 3, 3, 3, 23, 110.0, 108.0, 0, 780.0);

```

### 6.2.3. Script de consultas

Se han desarrollado 4 consultas basadas únicamente en la dimensión pasajeros.

**Consulta 1 (Análisis de pasajeros por categoría, VIP vs Estándar):** Compara el comportamiento de viaje entre pasajeros VIP y Estándar.

```

SQL
SELECT
    p.categoria_cliente,
    COUNT(DISTINCT p.id_pasajero) as total_pasajeros,
    COUNT(DISTINCT v.codigo_vuelo) as total_vuelos,
    ROUND(AVG(v.distancia_km), 2) as distancia_promedio,
    ROUND(AVG(v.retraso_minutos), 2) as retraso_promedio,
    ROUND(AVG(v.tiempo_real_minutos), 2) as duracion_promedio
FROM PASAJERO p
JOIN GRUPO_PASAJEROS gp ON p.id_pasajero = gp.id_pasajero
JOIN VUELO v ON gp.id_grupo = v.id_grupo
GROUP BY p.categoria_cliente
ORDER BY total_vuelos DESC;

```

| Categoría | Pasajeros | Vuelos | Dist.Prom. | Retraso | Duración |
|-----------|-----------|--------|------------|---------|----------|
| Estándar  | 15        | 15     | 793.29     | 2.79    | 110.87   |
| VIP       | 5         | 9      | 635        | 2.71    | 90.04    |

Los pasajeros estándar realizan más vuelos en promedio y recorren distancias más largas.

**Consulta 2 (Tamaño de grupos y su relación con los vuelos):** Analiza cómo el tamaño del grupo afecta a los patrones de viaje.

```

SQL
SELECT
    CASE
        WHEN num_pasajeros = 1 THEN 'Individual'

```

```

        WHEN num_pasajeros = 2 THEN 'Pareja'
        WHEN num_pasajeros BETWEEN 3 AND 4 THEN 'Familia pequeña'
        WHEN num_pasajeros >= 5 THEN 'Familia grande'
    END as tipo_grupo,
    COUNT(*) as cantidad_grupos,
    ROUND(AVG(v.distancia_km), 2) as distancia_promedio,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio
FROM (
    SELECT g.id_grupo, COUNT(gp.id_pasajero) as num_pasajeros
    FROM GRUPO g
    JOIN GRUPO_PASAJEROS gp ON g.id_grupo = gp.id_grupo
    GROUP BY g.id_grupo
) grupos
JOIN VUELO v ON grupos.id_grupo = v.id_grupo
GROUP BY CASE
    WHEN num_pasajeros = 1 THEN 'Individual'
    WHEN num_pasajeros = 2 THEN 'Pareja'
    WHEN num_pasajeros BETWEEN 3 AND 4 THEN 'Grupo pequeño'
    WHEN num_pasajeros >= 5 THEN 'Grupo grande'
END
ORDER BY cantidad_grupos DESC;

```

| Tipo Grupo    | Cantidad | Dist. Prom | Retraso |
|---------------|----------|------------|---------|
| Individual    | 9        | 730        | 3.17    |
| Grupo pequeño | 7        | 740        | 2.57    |
| Pareja        | 6        | 660.83     | 2.17    |
| Grupo grande  | 1        | 1050       | 5       |

Los viajeros individuales son el grupo más numeroso, mientras que los grupos grandes tienden a realizar vuelos más largos y suelen tener mayores retrasos. Las parejas son el tipo de grupo más puntual.

**Consulta 3 (Pasajeros frecuentes y sus patrones de viaje):** Identifica a los pasajeros que han realizado múltiples vuelos.

```

SQL
SELECT
    p.nombre,
    p.apellido1,
    p.categoria_cliente,
    p.nacionalidad,
    COUNT(DISTINCT v.codigo_vuelo) as vuelos_realizados,
    ROUND(SUM(v.distancia_km), 2) as distancia_total,
    ROUND(AVG(v.distancia_km), 2) as distancia_promedio,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio_experimentado
FROM PASAJERO p
JOIN GRUPO_PASAJEROS gp ON p.id_pasajero = gp.id_pasajero
JOIN VUELO v ON gp.id_grupo = v.id_grupo
GROUP BY p.id_pasajero, p.nombre, p.apellido1, p.categoria_cliente, p.nacionalidad
HAVING COUNT(DISTINCT v.codigo_vuelo) > 1
ORDER BY vuelos_realizados DESC, distancia_total DESC;

```

| Nombre | Apellido | Categoría | Nacionalidad | Vuelos | Dist. Total | Dist. Prom. | Retraso |
|--------|----------|-----------|--------------|--------|-------------|-------------|---------|
| Javier | Torres   | estándar  | España       | 3      | 2455        | 818.33      | 1.67    |
| Laura  | Martínez | estándar  | España       | 3      | 2455        | 781.67      | 1.67    |
| Isabel | Álvarez  | estándar  | España       | 3      | 2185        | 728.33      | 3.33    |
| Pedro  | Sánchez  | estándar  | España       | 3      | 2105        | 701.67      | 3.33    |
| Miguel | Jiménez  | estándar  | España       | 3      | 2025        | 675         | 3.33    |
| John   | Smith    | estándar  | Reino Unido  | 3      | 1985        | 661.67      | 3.67    |
| ...    | ...      | ...       | ...          | ...    | ...         | ...         | ...     |

Los pasajeros con más frecuencia tienen 3 vuelos. Además, los pasajeros estándar españoles dominan la lista de viajeros frecuentes.

**Consulta 4: Pasajeros frecuentes y sus patrones de viaje:** Identifica a los pasajeros que han realizado múltiples vuelos.

SQL

```
SELECT
    p.nacionalidad,
    COUNT(DISTINCT p.id_pasajero) as total_pasajeros,
    COUNT(DISTINCT v.codigo_vuelo) as vuelos_totales,
    ao.region as region_origen_mas_comun,
    COUNT(*) as vuelos_desde_region,
    ROUND(AVG(v.distancia_km), 2) as distancia_promedio,
    ROUND(AVG(v.retardo_minutos), 2) as retraso_promedio
FROM PASAJERO p
JOIN GRUPO_PASAJEROS gp ON p.id_pasajero = gp.id_pasajero
JOIN VUELO v ON gp.id_grupo = v.id_grupo
JOIN AEROPUERTO ao ON v.id_aeropuerto_origen = ao.id_aeropuerto
GROUP BY p.nacionalidad, ao.region
HAVING COUNT(DISTINCT p.id_pasajero) >= 1
ORDER BY p.nacionalidad, vuelos_desde_region DESC;
```

| Nacionalidad | Pasajeros | Vuelos | Región  | Vuelos Región | Dist. Prom. | Retraso |
|--------------|-----------|--------|---------|---------------|-------------|---------|
| Alemania     | 1         | 1      | Sur     | 1             | 920         | 5       |
| Alemania     | 1         | 1      | Noreste | 1             | 620         | 2       |
| España       | 13        | 6      | Centro  | 16            | 697.5       | 2.16    |
| España       | 9         | 4      | Sur     | 11            | 979.09      | 4.09    |
| España       | 9         | 6      | Este    | 11            | 720.91      | 1.64    |
| España       | 5         | 2      | Noreste | 5             | 620         | 3       |
| Francia      | 1         | 1      | Noreste | 1             | 600         | 5       |
| Francia      | 1         | 1      | Este    | 1             | 680         | 3       |
| Reino Unido  | 1         | 2      | Este    | 2             | 980         | 3       |
| Reino Unido  | 1         | 1      | Noreste | 1             | 625         | 3       |

Los pasajeros españoles dominan en los vuelos, prefiriendo volar desde la región Centro. La región Este es la más puntual para viajeros españoles.

Los pasajeros internacionales, realizan vuelos pero con patrones variados de origen.

### 6.3. Nueva propuesta

Este modelo implementa un esquema en estrella para el análisis de atenciones médicas en un entorno hospitalario. El esquema consta de 8 tablas de dimensiones y 1 tabla de hechos central.

La implementación se ha realizado mediante tres scripts SQL:

- *crear\_tablas\_hospital.sql*
- *insertar\_datos\_hospital.sql*
- *consultas\_hospital.sql*

#### 6.3.1. Script de creación de tablas

El script de creación establece el esquema completo del data mart hospitalario. Al igual que en el modelo de vuelos, incluye bloques PL/SQL que eliminan las tablas existentes en orden inverso a las dependencias, permitiendo ejecuciones múltiples sin errores.

- **DIM\_FECHA:** Proporciona información temporal sobre el día de la atención médica, con atributos jerárquicos (día>mes>trimestre>año) y categóricos (temporada, fin de semana y festivo).

SQL

```
CREATE TABLE FECHA (  
    id_fecha INTEGER PRIMARY KEY,  
    fecha_completa DATE UNIQUE NOT NULL,  
    dia INTEGER NOT NULL,  
    mes INTEGER NOT NULL,  
    agno INTEGER NOT NULL,  
    nombre_dia VARCHAR2(20) NOT NULL,  
    nombre_mes VARCHAR2(20) NOT NULL,  
    semana_agno INTEGER NOT NULL,  
    trimestre INTEGER NOT NULL,  
    es_fin_semana VARCHAR2(20) NOT NULL CHECK (es_fin_semana IN ('es_fin_semana',  
'no_es_fin_semana')),  
    es_festivo VARCHAR2(20) NOT NULL CHECK (es_festivo IN ('es_festivo', 'no_es_festivo')),  
    temporada VARCHAR2(20) NOT NULL CHECK (temporada IN ('primavera', 'verano', 'otono',  
'invierno'))  
);
```

- **DIM\_HORA:** Complementa la información temporal especificando el momento exacto de la atención. De esta forma se permite analizar por franjas horarias y horas punta.

SQL

```
CREATE TABLE HORA (  
    id_hora INTEGER PRIMARY KEY,  
    hora_completa DATE UNIQUE NOT NULL,  
    hora INTEGER NOT NULL,
```

```

minuto INTEGER NOT NULL,
segundo INTEGER NOT NULL,
franja_horaria VARCHAR2(20) NOT NULL CHECK (franja_horaria IN ('00:00-05:59', '06:00-11:59',
'12:00-17:59', '18:00-23:59')),
es_hora_punta VARCHAR2(20) NOT NULL CHECK (es_hora_punta IN ('es_hora_punta',
'no_es_hora_punta'))
);

```

- **DIM\_HOSPITAL:** Describe las características de los hospitales, incluyendo ubicación geográfica, capacidad y servicios disponibles.

SQL

```

CREATE TABLE HOSPITAL (
  id_hospital INTEGER PRIMARY KEY,
  direccion_completa VARCHAR2(50) UNIQUE NOT NULL,
  nombre_hospital VARCHAR2(100) NOT NULL,
  tipo_hospital VARCHAR2(30) CHECK (tipo_hospital IN ('publico', 'privado')),
  ciudad VARCHAR2(100) NOT NULL,
  provincia VARCHAR2(100) NOT NULL,
  codigo_postal INTEGER NOT NULL,
  num_camas VARCHAR2(20) CHECK (num_camas IN ('0-99', '100-299', '300-599', '600-999',
'1000+')),
  tiene_urgencias VARCHAR2(20) CHECK (tiene_urgencias IN ('si', 'no')),
  tiene_uci VARCHAR2(20) CHECK (tiene_uci IN ('si', 'no'))
);

```

- **DIM\_SERVICIO:** Representa los diferentes departamentos o servicios hospitalarios (consultas externas, urgencias, quirófano, hospitalización...).

SQL

```

CREATE TABLE SERVICIO (
  id_servicio INTEGER PRIMARY KEY,
  codigo_servicio VARCHAR2(50) UNIQUE NOT NULL,
  nombre_servicio VARCHAR2(100) NOT NULL,
  tipo_servicio VARCHAR2(50) CHECK (tipo_servicio IN ('consulta_externa', 'hospitalizacion',
'urgencias', 'quirofano', 'diagnostico', 'rehabilitacion')),
  planta INTEGER,
  edificio VARCHAR2(50),
  num_camas_servicio VARCHAR2(20) CHECK (num_camas_servicio IN ('0-9', '10-19', '20-49',
'50-99', '100+')),
  tiene_uci VARCHAR2(20) CHECK (tiene_uci IN ('tiene_uci', 'no_tiene_uci')),
  esta_activo VARCHAR2(20) CHECK (esta_activo IN ('esta_activo', 'no_esta_activo'))
);

```

- **DIM\_MEDICO:** Tiene información sobre los profesionales médicos, especificando su especialidad, experiencia, categoría, si sigue activo...

SQL

```
CREATE TABLE MEDICO (
    id_medico INTEGER PRIMARY KEY,
    nombre VARCHAR2(100) NOT NULL,
    apellido1 VARCHAR2(150) NOT NULL,
    apellido2 VARCHAR2(150) NOT NULL,
    tipo_documento VARCHAR2(20) NOT NULL CHECK (tipo_documento IN ('DNI', 'NIE', 'pasaporte')),
    num_documento VARCHAR2(50) UNIQUE NOT NULL,
    email VARCHAR2(100),
    telefono VARCHAR2(20),
    fecha_nacimiento DATE NOT NULL,
    nacionalidad VARCHAR2(100) NOT NULL,
    num_colegiado VARCHAR2(50) NOT NULL UNIQUE,
    genero VARCHAR2(20) CHECK (genero IN ('masculino', 'femenino', 'otro')),
    especialidad VARCHAR2(100) NOT NULL,
    anos_experiencia INTEGER,
    categoria VARCHAR2(50) CHECK (categoria IN ('residente', 'adjunto', 'jefe_servicio',
    'jefe_seccion')),
    esta_activo VARCHAR2(20) CHECK (esta_activo IN ('activo', 'inactivo'))
);
```

- **DIM\_PACIENTE:** Tiene información sobre los pacientes, guardando tanto sus datos personales como sus datos médicos.

SQL

```
CREATE TABLE PACIENTE (
    id_paciente INTEGER PRIMARY KEY,
    nombre VARCHAR2(100) NOT NULL,
    apellido1 VARCHAR2(150) NOT NULL,
    apellido2 VARCHAR2(150) NOT NULL,
    tipo_documento VARCHAR2(20) NOT NULL CHECK (tipo_documento IN ('DNI', 'NIE', 'pasaporte',
    'otro')),
    num_documento VARCHAR2(50) UNIQUE NOT NULL,
    email VARCHAR2(100),
    telefono VARCHAR2(20),
    fecha_nacimiento DATE NOT NULL,
    nacionalidad VARCHAR2(100) NOT NULL,
    genero VARCHAR2(20) CHECK (genero IN ('masculino', 'femenino', 'otro')),
    codigo_postal INTEGER NOT NULL,
    tipo_seguro VARCHAR2(30) CHECK (tipo_seguro IN ('publico', 'privado', 'mutua', 'sin_seguro')),
    categoria_cliente VARCHAR2(30) CHECK (categoria_cliente IN ('estandar', 'vip')),
    tipo_sanguineo VARCHAR2(10) CHECK (tipo_sanguineo IN ('A+', 'A-', 'B+', 'B-', 'AB+', 'AB-',
    'O+', 'O-', 'desconocido'))
);
```

- **DIM\_DIAGNOSTICO:** Guarda diagnósticos basados en la clasificación “CIE-10”.

SQL

```
CREATE TABLE DIAGNOSTICO (
    id_diagnostico INTEGER PRIMARY KEY,
    codigo_diagnostico VARCHAR2(50) UNIQUE NOT NULL,
```

```

nombre_diagnostico VARCHAR2(300) NOT NULL,
categoria_principal VARCHAR2(100) NOT NULL,
gravedad VARCHAR2(20) CHECK (gravedad IN ('leve', 'moderada', 'grave', 'critica')),
es_cronico VARCHAR2(20) CHECK (es_cronico IN ('es_cronico', 'no_es_cronico')),
requiere_hospitalizacion VARCHAR2(30) CHECK (requiere_hospitalizacion IN
('requiere_hospitalizacion', 'no_requiere_hospitalizacion'))
);

```

- **DIM\_PROCEDIMIENTO:** Describe los procedimientos médicos realizados durante la cita médica.

SQL

```

CREATE TABLE PROCEDIMIENTO (
    id_procedimiento INTEGER PRIMARY KEY,
    codigo_procedimiento VARCHAR2(50) UNIQUE NOT NULL,
    nombre_procedimiento VARCHAR2(300) NOT NULL,
    tipo_procedimiento VARCHAR2(50) CHECK (tipo_procedimiento IN ('consulta', 'cirugia_menor',
'cirugia_mayor', 'prueba_diagnostica', 'tratamiento', 'vacunacion', 'rehabilitacion')),
    duracion_estimada_minutos INTEGER,
    requiere_anestesia VARCHAR2(25) CHECK (requiere_anestesia IN ('requiere_anestesia',
'no_requiere_anestesia')),
    es_ambulatorio VARCHAR2(20) CHECK (es_ambulatorio IN ('es_ambulatorio', 'no_es_ambulatorio'))
);

```

- **TABLA DE HECHOS ATENCIÓN MÉDICA:** Registra cada atención médica, relacionando todas las dimensiones mediante claves foráneas. El grano es un registro por atención médica (paciente, fecha y hora específica).

SQL

```

CREATE TABLE ATENCION_MEDICA (
    id_paciente INTEGER NOT NULL,
    id_fecha INTEGER NOT NULL,
    id_hora INTEGER NOT NULL,
    id_hospital INTEGER NOT NULL,
    id_servicio INTEGER NOT NULL,
    id_medico INTEGER NOT NULL,
    id_diagnostico INTEGER NOT NULL,
    id_procedimiento INTEGER NOT NULL,
    duracion_atencion_minutos REAL NOT NULL,
    coste_procedimiento REAL NOT NULL,
    tiempo_espera_minutos REAL DEFAULT 0,
    num_pruebas_realizadas INTEGER DEFAULT 0,
    num_medicamentos_prescritos INTEGER DEFAULT 0,
    PRIMARY KEY (id_paciente, id_fecha, id_hora),
    FOREIGN KEY (id_paciente) REFERENCES PACIENTE(id_paciente) ON DELETE CASCADE,
    FOREIGN KEY (id_fecha) REFERENCES FECHA(id_fecha) ON DELETE CASCADE,
    FOREIGN KEY (id_hora) REFERENCES HORA(id_hora) ON DELETE CASCADE,
    FOREIGN KEY (id_hospital) REFERENCES HOSPITAL(id_hospital) ON DELETE CASCADE,
    FOREIGN KEY (id_servicio) REFERENCES SERVICIO(id_servicio) ON DELETE CASCADE,
    FOREIGN KEY (id_medico) REFERENCES MEDICO(id_medico) ON DELETE CASCADE,
    FOREIGN KEY (id_diagnostico) REFERENCES DIAGNOSTICO(id_diagnostico) ON DELETE CASCADE,
    FOREIGN KEY (id_procedimiento) REFERENCES PROCEDIMIENTO(id_procedimiento) ON DELETE CASCADE,
    CONSTRAINT chk_duracion_positiva CHECK (duracion_atencion_minutos > 0),
    CONSTRAINT chk_coste_positivo CHECK (coste_procedimiento >= 0),
);

```

```
CONSTRAINT chk_espera_positiva CHECK (tiempo_espera_minutos >= 0)
);
```

Las métricas incluidas son:

- ❖ duracion\_atencion\_minutos: Duración de la atención médica.
- ❖ coste\_procedimiento: Coste económico del procedimiento realizado.
- ❖ tiempo\_espera\_minutos: Tiempo que el paciente esperó desde la fecha y hora fijada hasta que recibió la atención médica.
- ❖ num\_pruebas\_realizadas: Cantidad de pruebas diagnósticas realizadas.
- ❖ num\_medicamentos\_prescritos: Cantidad de medicamentos recetados.

Todas las métricas son aditivas, excepto las métricas de tiempo *duracion\_atencion\_minutos* y *tiempo\_espera\_minutos*, que son semi-aditivas en la dimensión temporal. Esto es porque, aunque su suma proporciona el tiempo total acumulado, el promedio resulta más útil para comparar periodos (ya que sumar duraciones de eventos independientes puede generar interpretaciones erróneas como que 30 minutos en enero y 40 minutos en febrero, no significan 70 minutos de atención continua).

### 6.3.2. Script de inserción de datos

```
SQL

-- =====
-- SCRIPT DE INSERCIÓN DE DATOS - MODELO HOSPITALARIO
-- =====

-- Limpiar datos existentes
DELETE FROM ATENCION_MEDICA;
DELETE FROM PROCEDIMIENTO;
DELETE FROM DIAGNOSTICO;
DELETE FROM SERVICIO;
DELETE FROM PACIENTE;
DELETE FROM MEDICO;
DELETE FROM HOSPITAL;
DELETE FROM HORA;
DELETE FROM FECHA;
COMMIT;

----- DIMENSIÓN FECHA -----
INSERT INTO FECHA VALUES (1, TO_DATE('2025-01-15', 'YYYY-MM-DD'), 15, 1, 2025, 'Miércoles',
'Enero', 3, 1, 'no_es_fin_semana', 'no_es_festivo', 'invierno');
INSERT INTO FECHA VALUES (2, TO_DATE('2025-02-20', 'YYYY-MM-DD'), 20, 2, 2025, 'Jueves',
'Febrero', 8, 1, 'no_es_fin_semana', 'no_es_festivo', 'invierno');
INSERT INTO FECHA VALUES (3, TO_DATE('2025-03-25', 'YYYY-MM-DD'), 25, 3, 2025, 'Martes', 'Marzo',
13, 1, 'no_es_fin_semana', 'no_es_festivo', 'primavera');
INSERT INTO FECHA VALUES (4, TO_DATE('2025-04-12', 'YYYY-MM-DD'), 12, 4, 2025, 'Sábado', 'Abril',
15, 2, 'es_fin_semana', 'no_es_festivo', 'primavera');
INSERT INTO FECHA VALUES (5, TO_DATE('2025-05-01', 'YYYY-MM-DD'), 1, 5, 2025, 'Jueves', 'Mayo',
18, 2, 'no_es_fin_semana', 'es_festivo', 'primavera');
```

```

INSERT INTO FECHA VALUES (6, TO_DATE('2025-06-15', 'YYYY-MM-DD'), 15, 6, 2025, 'Domingo', 'Junio',
24, 2, 'es_fin_semana', 'no_es_festivo', 'verano');
INSERT INTO FECHA VALUES (7, TO_DATE('2025-07-20', 'YYYY-MM-DD'), 20, 7, 2025, 'Domingo', 'Julio',
29, 3, 'es_fin_semana', 'no_es_festivo', 'verano');
INSERT INTO FECHA VALUES (8, TO_DATE('2025-08-15', 'YYYY-MM-DD'), 15, 8, 2025, 'Viernes',
'Agosto', 33, 3, 'no_es_fin_semana', 'es_festivo', 'verano');
INSERT INTO FECHA VALUES (9, TO_DATE('2025-09-10', 'YYYY-MM-DD'), 10, 9, 2025, 'Miércoles',
'Septiembre', 37, 3, 'no_es_fin_semana', 'no_es_festivo', 'otono');
INSERT INTO FECHA VALUES (10, TO_DATE('2025-10-12', 'YYYY-MM-DD'), 12, 10, 2025, 'Domingo',
'Octubre', 41, 4, 'es_fin_semana', 'es_festivo', 'otono');

```

#### ----- DIMENSIÓN HORA -----

```

INSERT INTO HORA VALUES (1, TO_DATE('2025-01-01 08:30:00', 'YYYY-MM-DD HH24:MI:SS'), 8, 30, 0,
'06:00-11:59', 'es_hora_punta');
INSERT INTO HORA VALUES (2, TO_DATE('2025-01-01 09:15:00', 'YYYY-MM-DD HH24:MI:SS'), 9, 15, 0,
'06:00-11:59', 'es_hora_punta');
INSERT INTO HORA VALUES (3, TO_DATE('2025-01-01 10:45:00', 'YYYY-MM-DD HH24:MI:SS'), 10, 45, 0,
'06:00-11:59', 'es_hora_punta');
INSERT INTO HORA VALUES (4, TO_DATE('2025-01-01 14:00:00', 'YYYY-MM-DD HH24:MI:SS'), 14, 0, 0,
'12:00-17:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (5, TO_DATE('2025-01-01 15:30:00', 'YYYY-MM-DD HH24:MI:SS'), 15, 30, 0,
'12:00-17:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (6, TO_DATE('2025-01-01 16:45:00', 'YYYY-MM-DD HH24:MI:SS'), 16, 45, 0,
'12:00-17:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (7, TO_DATE('2025-01-01 19:00:00', 'YYYY-MM-DD HH24:MI:SS'), 19, 0, 0,
'18:00-23:59', 'no_es_hora_punta');
INSERT INTO HORA VALUES (8, TO_DATE('2025-01-01 20:30:00', 'YYYY-MM-DD HH24:MI:SS'), 20, 30, 0,
'18:00-23:59', 'no_es_hora_punta');

```

#### ----- DIMENSIÓN HOSPITAL -----

```

INSERT INTO HOSPITAL VALUES (1, 'Calle Mayor 1, Madrid', 'Hospital Clínico San Carlos', 'publico',
'Madrid', 'Madrid', 28040, '1000+', 'si', 'si');
INSERT INTO HOSPITAL VALUES (2, 'Av. Diagonal 100, Barcelona', 'Hospital Clínic de Barcelona',
'publico', 'Barcelona', 'Barcelona', 8036, '1000+', 'si', 'si');
INSERT INTO HOSPITAL VALUES (3, 'Calle Sevilla 50, Sevilla', 'Hospital Virgen del Rocío',
'publico', 'Sevilla', 'Sevilla', 41013, '600-999', 'si', 'si');
INSERT INTO HOSPITAL VALUES (4, 'Av. Europa 20, Valencia', 'Clínica Quirón Valencia', 'privado',
'Valencia', 'Valencia', 46010, '100-299', 'si', 'no');
INSERT INTO HOSPITAL VALUES (5, 'Calle Principal 5, Zaragoza', 'Hospital Miguel Servet',
'publico', 'Zaragoza', 'Zaragoza', 50009, '600-999', 'si', 'si');

```

#### ----- DIMENSIÓN MÉDICO -----

```

INSERT INTO MEDICO VALUES (1, 'Ana', 'García', 'López', 'DNI', '12345678A',
'ana.garcia@hospital.es', '+34600111222', TO_DATE('1975-05-15', 'YYYY-MM-DD'), 'España', 'COL001',
'femenino', 'Cardiología', 20, 'jefe_servicio', 'activo');
INSERT INTO MEDICO VALUES (2, 'Carlos', 'Martínez', 'Ruiz', 'DNI', '23456789B',
'carlos.martinez@hospital.es', '+34600222333', TO_DATE('1980-08-20', 'YYYY-MM-DD'), 'España',
'COL002', 'masculino', 'Traumatología', 15, 'adjunto', 'activo');
INSERT INTO MEDICO VALUES (3, 'María', 'Fernández', 'Sánchez', 'DNI', '34567890C',
'maria.fernandez@hospital.es', '+34600333444', TO_DATE('1985-03-10', 'YYYY-MM-DD'), 'España',
'COL003', 'femenino', 'Pediatría', 10, 'adjunto', 'activo');
INSERT INTO MEDICO VALUES (4, 'Juan', 'López', 'González', 'DNI', '45678901D',
'juan.lopez@hospital.es', '+34600444555', TO_DATE('1990-11-25', 'YYYY-MM-DD'), 'España', 'COL004',
'masculino', 'Medicina Interna', 5, 'residente', 'activo');
INSERT INTO MEDICO VALUES (5, 'Laura', 'Rodríguez', 'Pérez', 'DNI', '56789012E',
'laura.rodriguez@hospital.es', '+34600555666', TO_DATE('1982-07-08', 'YYYY-MM-DD'), 'España',
'COL005', 'femenino', 'Neurología', 12, 'jefe_seccion', 'activo');

```

#### ----- DIMENSIÓN PACIENTE -----

```

INSERT INTO PACIENTE VALUES (1, 'Pedro', 'Jiménez', 'Díaz', 'DNI', '11111111A',
'pedro.jimenez@email.com', '+3461111111', TO_DATE('1960-01-15', 'YYYY-MM-DD'), 'España',
'masculino', 28001, 'publico', 'estandar', 'A+');
INSERT INTO PACIENTE VALUES (2, 'Isabel', 'Moreno', 'Vázquez', 'DNI', '22222222B',
'isabel.moreno@email.com', '+3462222222', TO_DATE('1975-06-20', 'YYYY-MM-DD'), 'España',
'femenino', 8001, 'privado', 'vip', 'B+');
INSERT INTO PACIENTE VALUES (3, 'Miguel', 'Torres', 'Ramírez', 'DNI', '33333333C',
'miguel.torres@email.com', '+3463333333', TO_DATE('1988-09-10', 'YYYY-MM-DD'), 'España',
'masculino', 41001, 'publico', 'estandar', 'O+');
INSERT INTO PACIENTE VALUES (4, 'Carmen', 'Navarro', 'Gil', 'DNI', '44444444D',
'carmen.navarro@email.com', '+3464444444', TO_DATE('1995-03-25', 'YYYY-MM-DD'), 'España',
'femenino', 46001, 'mutua', 'estandar', 'AB+');
INSERT INTO PACIENTE VALUES (5, 'Antonio', 'Serrano', 'Castro', 'DNI', '55555555E',
'antonio.serrano@email.com', '+3465555555', TO_DATE('1970-12-05', 'YYYY-MM-DD'), 'España',
'masculino', 50001, 'publico', 'estandar', 'A-');
INSERT INTO PACIENTE VALUES (6, 'Rosa', 'Blanco', 'Ortega', 'DNI', '66666666F',
'rosa.blanco@email.com', '+3466666666', TO_DATE('1982-04-18', 'YYYY-MM-DD'), 'España',
'femenino', 28002, 'privado', 'vip', 'O-');
INSERT INTO PACIENTE VALUES (7, 'Francisco', 'Rubio', 'Suárez', 'DNI', '77777777G',
'francisco.rubio@email.com', '+3467777777', TO_DATE('1968-08-30', 'YYYY-MM-DD'), 'España',
'masculino', 8002, 'publico', 'estandar', 'B-');
INSERT INTO PACIENTE VALUES (8, 'Elena', 'Ramos', 'Delgado', 'DNI', '88888888H',
'elena.ramos@email.com', '+3468888888', TO_DATE('1992-11-12', 'YYYY-MM-DD'), 'España',
'femenino', 41002, 'publico', 'estandar', 'A+');

----- DIMENSIÓN SERVICIO -----
INSERT INTO SERVICIO VALUES (1, 'SRV-CARD-01', 'Cardiología', 'consulta_externa', 3, 'Edificio A',
'10-19', 'no_tiene_uci', 'esta_activo');
INSERT INTO SERVICIO VALUES (2, 'SRV-TRAU-01', 'Traumatología', 'hospitalizacion', 2, 'Edificio
B', '20-49', 'no_tiene_uci', 'esta_activo');
INSERT INTO SERVICIO VALUES (3, 'SRV-PEDI-01', 'Pediatría', 'consulta_externa', 1, 'Edificio C',
'10-19', 'no_tiene_uci', 'esta_activo');
INSERT INTO SERVICIO VALUES (4, 'SRV-URG-01', 'Urgencias Generales', 'urgencias', 0, 'Edificio
Central', '20-49', 'tiene_uci', 'esta_activo');
INSERT INTO SERVICIO VALUES (5, 'SRV-NEUR-01', 'Neurología', 'hospitalizacion', 4, 'Edificio A',
'10-19', 'tiene_uci', 'esta_activo');

----- DIMENSIÓN DIAGNOSTICO -----
INSERT INTO DIAGNOSTICO VALUES (1, 'I10', 'Hipertensión arterial esencial', 'Enfermedades del
sistema circulatorio', 'moderada', 'es_cronico', 'no_requiere_hospitalizacion');
INSERT INTO DIAGNOSTICO VALUES (2, 'M25.5', 'Dolor en articulación', 'Enfermedades del sistema
osteomuscular', 'leve', 'no_es_cronico', 'no_requiere_hospitalizacion');
INSERT INTO DIAGNOSTICO VALUES (3, 'J06.9', 'Infección aguda de vías respiratorias superiores',
'Enfermedades del sistema respiratorio', 'leve', 'no_es_cronico', 'no_requiere_hospitalizacion');
INSERT INTO DIAGNOSTICO VALUES (4, 'S72.0', 'Fractura del cuello del fémur', 'Traumatismos y
envenenamientos', 'grave', 'no_es_cronico', 'requiere_hospitalizacion');
INSERT INTO DIAGNOSTICO VALUES (5, 'G40.9', 'Epilepsia', 'Enfermedades del sistema nervioso',
'grave', 'es_cronico', 'requiere_hospitalizacion');
INSERT INTO DIAGNOSTICO VALUES (6, 'E11.9', 'Diabetes mellitus tipo 2', 'Enfermedades endocrinas',
'moderada', 'es_cronico', 'no_requiere_hospitalizacion');

----- DIMENSIÓN PROCEDIMIENTO -----
INSERT INTO PROCEDIMIENTO VALUES (1, 'PROC-001', 'Consulta de cardiología', 'consulta', 30,
'no_requiere_anestesia', 'es_ambulatorio');
INSERT INTO PROCEDIMIENTO VALUES (2, 'PROC-002', 'Reducción de fractura con osteosíntesis',
'cirugia_mayor', 180, 'requiere_anestesia', 'no_es_ambulatorio');
INSERT INTO PROCEDIMIENTO VALUES (3, 'PROC-003', 'Consulta pediátrica', 'consulta', 20,
'no_requiere_anestesia', 'es_ambulatorio');
INSERT INTO PROCEDIMIENTO VALUES (4, 'PROC-004', 'Radiografía de tórax', 'prueba_diagnostica', 15,
'no_requiere_anestesia', 'es_ambulatorio');

```

```
INSERT INTO PROCEDIMIENTO VALUES (5, 'PROC-005', 'Electroencefalograma', 'prueba_diagnostica', 60,
'no_requiere_anestesia', 'es_ambulatorio');
INSERT INTO PROCEDIMIENTO VALUES (6, 'PROC-006', 'Control de diabetes', 'tratamiento', 25,
'no_requiere_anestesia', 'es_ambulatorio');
```

```
----- TABLA DE HECHOS ATENCION_MEDICA -----
INSERT INTO ATENCION_MEDICA VALUES (1, 1, 1, 1, 1, 1, 1, 1, 35, 50.0, 15, 1, 2);
INSERT INTO ATENCION_MEDICA VALUES (2, 2, 2, 2, 4, 2, 4, 2, 200, 3500.0, 45, 3, 5);
INSERT INTO ATENCION_MEDICA VALUES (3, 3, 3, 3, 3, 3, 3, 3, 25, 30.0, 10, 0, 1);
INSERT INTO ATENCION_MEDICA VALUES (4, 4, 4, 4, 2, 2, 2, 4, 20, 25.0, 5, 1, 0);
INSERT INTO ATENCION_MEDICA VALUES (5, 5, 5, 5, 5, 5, 5, 5, 70, 150.0, 30, 2, 3);
INSERT INTO ATENCION_MEDICA VALUES (6, 6, 6, 1, 1, 1, 6, 6, 30, 45.0, 20, 1, 2);
INSERT INTO ATENCION_MEDICA VALUES (7, 7, 7, 2, 4, 4, 3, 4, 18, 20.0, 60, 1, 1);
INSERT INTO ATENCION_MEDICA VALUES (8, 8, 8, 3, 2, 2, 2, 4, 22, 28.0, 25, 1, 0);
INSERT INTO ATENCION_MEDICA VALUES (1, 9, 1, 1, 1, 1, 1, 1, 32, 50.0, 10, 1, 2);
INSERT INTO ATENCION_MEDICA VALUES (2, 10, 2, 2, 2, 2, 2, 2, 185, 3200.0, 30, 2, 4);

COMMIT;
```

### 6.3.3. Script de consultas

SQL

```
-- =====
-- SCRIPT DE CONSULTAS - MODELO HOSPITALARIO
-- =====

-- 1. Hospitales con más de 2 atenciones: rendimiento y costes
SELECT
    h.nombre_hospital,
    h.tipo_hospital,
    h.ciudad,
    COUNT(*) as total_atenciones,
    ROUND(AVG(am.duracion_atencion_minutos), 2) as duracion_promedio,
    ROUND(AVG(am.tiempo_espera_minutos), 2) as espera_promedio,
    ROUND(SUM(am.coste_procedimiento), 2) as coste_total,
    ROUND(AVG(am.coste_procedimiento), 2) as coste_promedio
FROM ATENCION_MEDICA am
JOIN HOSPITAL h ON am.id_hospital = h.id_hospital
GROUP BY h.nombre_hospital, h.tipo_hospital, h.ciudad
HAVING COUNT(*) > 2
ORDER BY total_atenciones DESC, coste_total DESC;

-- 2. Análisis jerárquico por especialidad y categoría médica (ROLLUP)
SELECT
    m.especialidad,
    m.categoria,
    COUNT(*) as num_atenciones,
    ROUND(AVG(am.duracion_atencion_minutos), 2) as duracion_promedio,
    ROUND(AVG(am.coste_procedimiento), 2) as coste_promedio,
    ROUND(SUM(am.coste_procedimiento), 2) as facturacion_total,
    ROUND(AVG(am.tiempo_espera_minutos), 2) as espera_promedio
FROM ATENCION_MEDICA am
JOIN MEDICO m ON am.id_medico = m.id_medico
GROUP BY ROLLUP(m.especialidad, m.categoria)
ORDER BY m.especialidad NULLS LAST, m.categoria NULLS LAST;

-- 3. Análisis multidimensional por tipo de seguro y gravedad (CUBE)
SELECT
    p.tipo_seguro,
    d.gravedad,
```

```

COUNT(*) as total_casos,
ROUND(AVG(am.coste_procedimiento), 2) as coste_promedio,
ROUND(AVG(am.duracion_atencion_minutos), 2) as duracion_promedio,
ROUND(AVG(am.tiempo_espera_minutos), 2) as espera_promedio,
ROUND(SUM(am.num_medicamentos_prescritos), 0) as medicamentos_totales
FROM ATENCION_MEDICA am
JOIN PACIENTE p ON am.id_paciente = p.id_paciente
JOIN DIAGNOSTICO d ON am.id_diagnostico = d.id_diagnostico
GROUP BY CUBE(p.tipo_seguro, d.gravedad)
ORDER BY p.tipo_seguro NULLS LAST, d.gravedad NULLS LAST;

-- 4. Servicios con coste promedio superior a 100€
SELECT
    s.nombre_servicio,
    s.tipo_servicio,
    COUNT(*) as num_atenciones,
    ROUND(AVG(am.coste_procedimiento), 2) as coste_promedio,
    ROUND(MIN(am.coste_procedimiento), 2) as coste_minimo,
    ROUND(MAX(am.coste_procedimiento), 2) as coste_maximo,
    ROUND(AVG(am.duracion_atencion_minutos), 2) as duracion_promedio,
    ROUND(AVG(am.tiempo_espera_minutos), 2) as espera_promedio
FROM ATENCION_MEDICA am
JOIN SERVICIO s ON am.id_servicio = s.id_servicio
GROUP BY s.nombre_servicio, s.tipo_servicio
HAVING AVG(am.coste_procedimiento) > 100
ORDER BY coste_promedio DESC;

-- 5. Análisis temporal: temporada y franja horaria (CUBE)
SELECT
    f.temporada,
    h.franja_horaria,
    COUNT(*) as total_atenciones,
    ROUND(AVG(am.tiempo_espera_minutos), 2) as espera_promedio,
    ROUND(AVG(am.duracion_atencion_minutos), 2) as duracion_promedio,
    ROUND(AVG(am.coste_procedimiento), 2) as coste_promedio,
    ROUND(SUM(am.num_medicamentos_prescritos), 0) as medicamentos_totales
FROM ATENCION_MEDICA am
JOIN FECHA f ON am.id_fecha = f.id_fecha
JOIN HORA h ON am.id_hora = h.id_hora
GROUP BY CUBE(f.temporada, h.franja_horaria)
ORDER BY f.temporada NULLS LAST, h.franja_horaria NULLS LAST;

COMMIT;

```

## 7. Distribución del trabajo

En el desarrollo de esta práctica, todas las tareas se llevaron a cabo de forma conjunta por los integrantes del equipo. La colaboración fue continua en todas las fases del proyecto, combinando esfuerzos en el diseño conceptual, el modelado, la validación del esquema y la elaboración de los diferentes ficheros asociados.

Las actividades desarrolladas conjuntamente incluyen:

- **Definición del grano** para el modelo de **vuelos**, tanto con pasajeros como sin pasajeros.
- **Decisión de la idea de negocio y definición del nuevo contexto** orientado al análisis de la actividad hospitalaria.

- **Identificación del proceso de negocio, el grano y las dimensiones**, tanto primarias como secundarias.
- **Diseño de los modelos en estrella.**
- **Elaboración de todos los ficheros asociados a estos contextos** (hospitalario y de vuelos), incluyendo los scripts de creación, inserción de datos y consultas.
- **Redacción de los apartados correspondientes en la memoria**, integrando de manera coherente los resultados obtenidos y las decisiones de diseño adoptadas.

En conjunto, el proyecto fue resultado de una **colaboración equitativa y coordinada** entre todos los miembros del grupo, sin asignaciones individuales diferenciadas.

## 8. Conclusiones

En esta práctica se ha podido aplicar de forma práctica la metodología de Kimball, lo que ha permitido comprender la importancia de definir correctamente las dimensiones, la tabla de hechos y el grano, asegurándonos de que es lo menor posible, dentro de un data mart. El diseño del esquema en estrella para el análisis de vuelos comerciales ha facilitado entender cómo una estructura bien definida, a pesar de poder contener atributos que sean redundantes, puede mejorar la eficiencia de las consultas y la interpretación de los resultados.

La posterior incorporación de la información de pasajeros supuso un desafío que ayudó a profundizar en el tratamiento de relaciones muchos a muchos y en la ampliación de un modelo sin alterar su granularidad. Asimismo, el desarrollo del nuevo contexto hospitalario permitió consolidar los conocimientos adquiridos y poner en práctica la capacidad de adaptación del diseño dimensional a distintos dominios de análisis.

| Apartados del trabajo realizados                          | Lucía | Irene |
|-----------------------------------------------------------|-------|-------|
| Diseño esquema en estrella vuelos sin pasajeros           | 3,5   | 3     |
| Diseño esquema en estrella vuelos con pasajeros           | 1,5   | 1     |
| Diseño esquema en estrella propuesta hospital             | 2     | 2     |
| Creación, implementación y consultas vuelos sin pasajeros | 3     | 3     |
| Creación, implementación y consultas vuelos con pasajeros | 2     | 4     |
| Creación, implementación y consultas hospital             | 0     | 2     |
| Documentación                                             | 5     | 1,5   |
|                                                           | 17    | 16,5  |