BASES DE DATOS

2º Grado Ingeniería Informática

Curso 2021-2022

PRÁCTICA 3: Base de datos de aviones

Seral Gracia, Álvaro	819425@unizar.es
Selivanov Dobrisan, Cristian Andrei	816456@unizar.es
Dorian Boleslaw Wozniak	817570@unizar.es

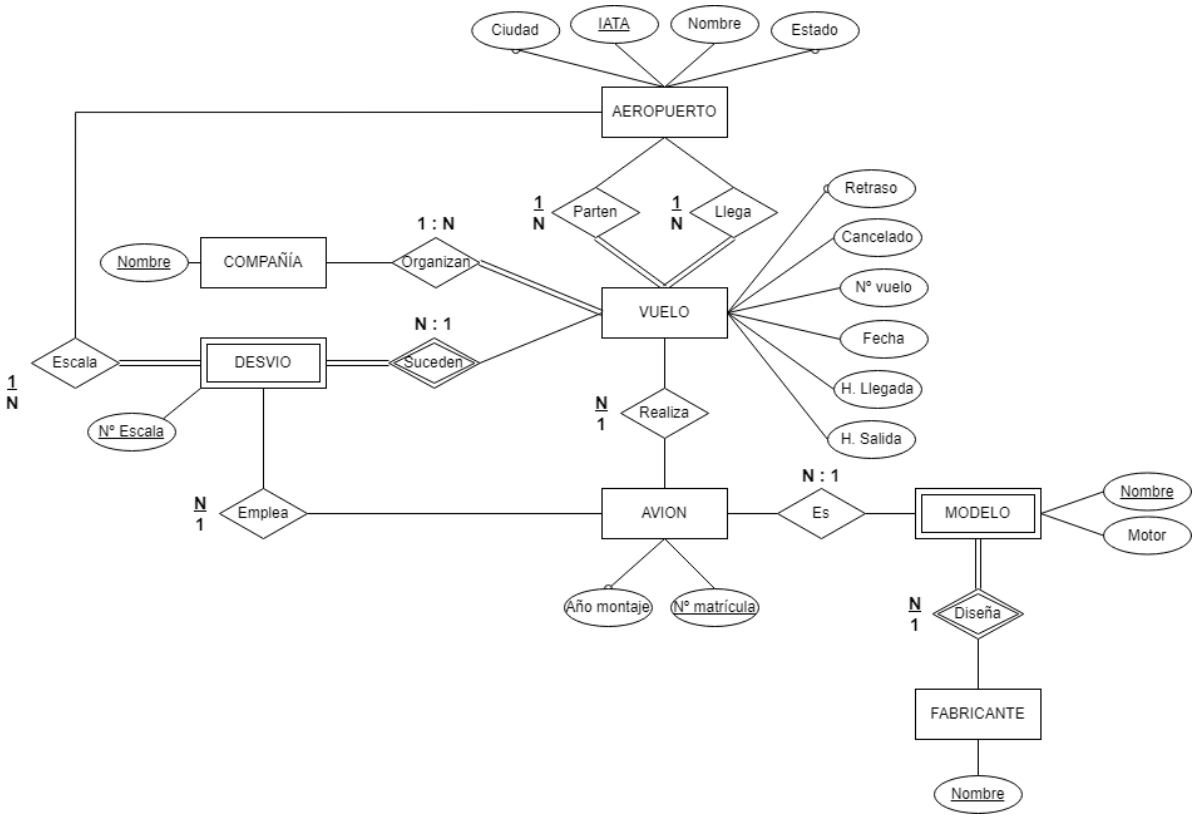
Entregado el 30/05/2022

Índice:

1.- Creación de la base de datos

1.1.- Esquema Entidad-Relación

1.1.1.- Diagrama



1.1.2.- Restricciones

- La hora de salida de un vuelo tiene que ser anterior a su hora de llegada.
- No pueden ocurrir dos vuelos con el mismo avión en las mismas fechas.
- Un vuelo no puede ocurrir en un avión que todavía no se ha creado.
- El aeropuerto de salida debe ser distinto del aeropuerto de llegada.
- El aeropuerto de llegada de un vuelo debe ser distinto del aeropuerto relativo a un desvío.
- Las horas de llegada y de salida y el tiempo de retraso deben ser positivos.
- El valor del atributo “Cancelado” sólo puede ser “0” o “1”.
- El valor del atributo “Escala” sólo puede puede ser “1” o “2”.
- Si un vuelo está cancelado, no tiene retrasos ni desvíos.

1.1.3.- Consideraciones

En un primer modelo, Ciudad era débil respecto a un tipo de entidad Estado, comprendiendo que un una ciudad se encuentra físicamente dentro de un estado. Sin embargo, ya que los tipos de entidades no disponen de ningún atributo a excepción de su clave primaria, y como al ser transitiva dependientes supone que el tipo de entidad Aeropuerto siga disponiendo de ambas claves primarias de Ciudad y Estado, se ha decidido interpretarlos únicamente como atributos de Aeropuerto.

Además, como se observó que un mismo modelo tenía distintos motores dependiendo del fabricante en cuestión, se deliberó que el tipo de entidad Modelo dependiera de un tipo de entidad Fabricante, de modo que el atributo Motor de Modelo dependa indirectamente de Modelo 'X' creado por el Fabricante 'Y'.

A la hora de tratar los distintos tipos de incidencias que ocurren en un vuelo, en un principio se consideró mantener un tipo de entidad dependiente de Vuelo para cada tipo de incidencia, de modo que hubiera una para desvíos, otra para retrasos y una tercera para vuelos cancelados. Sin embargo, como los tipos de entidad Retraso y Cancelado solo tenían un único atributo (en Retraso se guarda el tiempo total que se ha retrasado sin distinguir los distintos tipos de retrasos), se dispusieron como atributos en el tipo de entidad Vuelo. De este modo, Cancelado siempre tiene un valor 0 o 1, según si el vuelo ha sido o no cancelado. En los casos en los que valga 1 (vuelo cancelado), como no puede haber ni retrasos ni desvíos, el valor de Retraso será NULL y la tabla Desvío no podrá contener una entrada que dependa de dicho vuelo.

Al investigar la base de datos ofrecida como referencia para la práctica, se observó que había vuelos que no tenían asociados ningún avión en concreto, por lo que la relación entre Vuelo y Avión no debía ser de participación total. Esto mismo también ocurre con el avión al que puede ser redirigido un vuelo cuando ocurre un desvío, por lo que la relación entre entre Desvío y Vuelo tampoco podía ser de participación total.

1.2.- Esquema Relacional

1.2.1.- Esquema Relacional inicial

A partir del esquema E-R realizado se ha desarrollado el esquema relacional correspondiente. Hay que tener en consideración que para la clave primaria de Vuelo se ha utilizado una clave artificial.

Dominios:

tpIATA = cadena(3);
tpEstado = cadena(2);
tpCompañia = cadena(7);
tpNombreLargo = cadena(100);
tpNombre = cadena(50);
tpVuelo = number(4);
tpFecha = number(10);
tpAvion = cadena(7);
tpAnyo = number(4);
tpBool = number(2);
tpMinutos = number(3);

tpEscala = number(1);

Esquema Relacional:

Aeropuerto (
IATA : tpIATA;
NombreA : tpNombre, NO NULO;
Ciudad : tpNombre;
Estado : tpEstado);

Avion (
Matricula : tpAvion;
Montaje : tpAnyo;
clvModelo, clvFabricante : tpNombre;
(clvModelo, clvFabricante) clave ajena de **Modelo**);

Modelo (
NombreM : tpNombre;
clvFabricante : tpNombre, clave ajena de **Fabricante**;
Motor : tpNombre, NO NULO);

Fabricante (
NombreF : tpNombre);

Compañia (
idCompañia : tpCompañia;
NombreC : tpNombreLargo, UNICO, NO NULO);

Vuelo (
idVuelo : tpNatural;
numVuelo, Salida, Llegada : tpVuelo, NO NULO;
Fecha : tpFecha, NO NULO;
Cancelado : tpBool, NO NULO;
Retraso : tpMinutos;
clvOrigen, clvDestino : tpIATA, NO NULO, clave ajena de **Aeropuerto**;
clvCompañia : tpCompañia, NO NULO, clave ajena de **Compañia**;
clvAvion : tpAvion, clave ajena de **Avion**);

Desvio (
Escala : tpEscala;
clvVuelo : tpNatural, NO NULO, clave ajena de **Vuelo**;
clvAeropuerto : tpIATA, NO NULO, clave ajena de **Aeropuerto**;
clvAvion : tpAvion, clave ajena de **Avion**);

1.2.2.- Normalización y Esquema Relacional final

Con el objetivo de que el esquema relacional anterior se encuentre en Tercera Forma Normal Boyce-Codd, se han comprobado una por una todas las formas normales hasta llegar a la deseada, de modo que para cuando se verifique la Forma Normal Boyce-Codd, las relaciones del modelo se hayan transformado el número de veces requeridas para validar la correcta normalización.

Puesto que no existe ningún atributo multivaluado, todos los atributos tienen un único valor, lo que implica que el modelo cumpla la **Primera Forma Normal**.

Como todos los atributos no clave dependen en su totalidad de la clave primaria, y no de una combinación de parte de la clave primaria, se encuentra en **Segunda Forma Normal**.

Además, como dichos atributos no clave no dependen de otros conjuntos de atributos no clave, de modo que dependieran de la clave primaria de forma transitiva, también verifica la **Tercera Forma Normal**.

Finalmente, ya que ningún atributo que forma parte de las claves dependen de otros atributos no clave, las relaciones también se encuentran en **Forma Normal Boyce-Codd**.

Como no ha sido necesario transformar el modelo anterior, el esquema relacional resultante de haberlo normalizado es idéntico al inicial. Con el fin de obtener una mejor visibilidad en la memoria, y para evitar información redundante, se ha optado por no repetir el esquema relacional anterior. Esto se debe a que, al no haber sido necesario transformar el primer modelo, el esquema relacional resultante de haberlo normalizado es idéntico al inicial.

1.3.- Sentencias SQL de creación de tablas

Se han definido las sentencias SQL para la creación de las tablas necesarias que implementen en su completitud el modelo relacional anteriormente descrito y normalizado.

```
CREATE TABLE Aeropuerto (
    IATA            VARCHAR(4)      PRIMARY KEY,
    NombreA         VARCHAR(50)     NOT NULL,
    Ciudad          VARCHAR(50),
    Estado          VARCHAR(2)
);

CREATE TABLE Fabricante (
    NombreF         VARCHAR(50)     PRIMARY KEY
);

CREATE TABLE Modelo (
    nombreM         VARCHAR(50),
    clvFabricante   VARCHAR(50)     NOT NULL REFERENCES Fabricante(NombreF),
    Motor           VARCHAR(50)     NOT NULL,
    PRIMARY KEY (nombreM, clvFabricante)
);
```

```
CREATE TABLE Avion (
    Matricula       VARCHAR(7)       PRIMARY KEY,
    Montaje         NUMBER(4),
    clvModelo       VARCHAR(50),
    clvFabricante   VARCHAR(50),
    FOREIGN KEY (clvModelo, clvFabricante) REFERENCES Modelo(NombreM,clvFabricante)
);

CREATE TABLE Compania (
    idCompania      VARCHAR(7)      PRIMARY KEY,
    NombreC         VARCHAR(100)    UNIQUE NOT NULL
);

CREATE TABLE Vuelo (
    idVuelo         NUMBER          PRIMARY KEY,
    numVuelo        NUMBER(4)       NOT NULL,
    Salida          NUMBER(4)       NOT NULL CHECK (Salida >=0),
    Llegada         NUMBER(4)       NOT NULL CHECK (Llegada >=0),
    Fecha           VARCHAR(10)     NOT NULL,
    Cancelado       NUMBER(1)       NOT NULL CHECK (Cancelado='0' OR Cancelado='1'),
    Retraso         NUMBER(3)       CHECK (Retraso >0),
    clvOrigen       VARCHAR(4)      NOT NULL REFERENCES Aeropuerto(IATA),
    clvDestino      VARCHAR(4)      NOT NULL REFERENCES Aeropuerto(IATA),
    clvCompania     VARCHAR(7)      NOT NULL REFERENCES Compania(idCompania),
    clvAvion        VARCHAR(7)      REFERENCES Avion(Matricula),
    CHECK (clvOrigen <> clvDestino),
    CHECK ((Cancelado=1 AND Retraso=NULL) OR Cancelado=0)
);

CREATE TABLE Desvio (
    Escala          NUMBER(1)       CHECK (Escala='1' OR Escala='2'),
    clvVuelo        NUMBER          NOT NULL REFERENCES Vuelo(idVuelo),
    clvAeropuerto   VARCHAR(4)      NOT NULL REFERENCES Aeropuerto(IATA),
    clvAvion        VARCHAR(7)      REFERENCES Avion(Matricula),
    PRIMARY KEY (Escala, clvVuelo)
);
```

2.- Introducción de datos y ejecución de consultas

2.1.- Procedimiento de población

Para conseguir las tablas completas derivadas del esquema relacional realizado, en una primera instancia se realizaron las consultas necesarias para hallar, a partir de la base de datos ofrecida para la práctica como referencia, dichas tablas. Cabe recalcar que, para conseguir las claves artificiales del atributo idVuelo, se utilizó una variable @rownum iniciada a 0 (... `from (select @rownum:=0) as fila,` ...) que por cada fila se suma 1 a sí misma (`select @rownum:=@rownum+1,` ...). Una vez obtenidas las tablas, con el fin de poder insertar los valores de estas a la base de datos, se utilizó la herramienta MySQL Workbench para exportar dichas tablas a ficheros .sql donde cada fila corresponde a una instrucción de inserción de datos.

Después de conseguir todos los ficheros necesarios para poblar la base de datos, para facilitar la carga de los datos se creó un nuevo fichero poblar.sql de modo que ejecutase, por el orden correspondiente, todos los ficheros de inserción de datos. Además, al final de este fichero se añadió la instrucción `commit;` para que la base de datos no se borrase al cerrar sesión de Oracle.

Entre las dificultades encontradas al realizar la inserción de los datos en la base, hay que destacar varios puntos que se tuvieron que depurar. Por un lado, en los atributos que eran cadenas de texto se eliminaron los caracteres <\'> y se sustituyeron los caracteres <&> por <AND>. Además, había valores que se referían a atributos nulos pero no tenían el formato adecuado. Para ello se sustituyeron las cadenas de texto <''> y <'na'> por NULL. El último cambio que se realizó fue, en la columna "Retraso" correspondiente a la tabla "Vuelo", sustituir los valores <0> por <NULL>.

2.2.- Consultas

2.2.1.- Consulta 1

El **objetivo** de la consulta:
"Calcular el número de compañías que operan en al menos cinco aeropuertos de Alaska".

El **álgebra relacional** que describe el enunciado de la consulta:

$$R1 = \sigma_{(clvDestino=IATA \vee clvOrigen=IATA) \wedge Estado='AK'} (Vuelo \bowtie Aeropuerto)$$

$$R2 = \pi_{IATA} (\sigma_{5 \leq numIATAS} (AGRUPAR_{CONTAR(IATA)\ numIATAS} (R1, clvCompania)))$$

$$\pi_{numCompanias} (AGRUPAR_{CONTAR(IATA)\ numCcompanias} (R2))$$

La **sentencia SQL** que cumple con el objetivo indicado:

```
-- Consulta 1.
-- Número de 1's que aparecen, es decir, número de compañías que cumplen las condiciones
SELECT COUNT(*) Num_Companias FROM (
  -- '1' por cada compañía que cumplen las condiciones
  SELECT 1
  FROM Vuelo, Aeropuerto
  -- El aeropuerto de salida o de llegada del vuelo
  -- se encuentra en el estado de Alaska ('AK')
  WHERE (clvDestino=IATA OR clvOrigen=IATA) AND Estado='AK'
  -- Agrupar por compañías
  GROUP BY clvCompania
  -- El número de aeropuertos de Alaska donde opera la compañía debe ser >= 5
  HAVING 5<=(COUNT(DISTINCT IATA))
);
```

La **respuesta** obtenida por la base de datos tras ejecutar la sentencia SQL:

```
NUM_COMPANIAS
-------------
            1
```

Para poder comprobar los resultados de la consulta, se ha creado una consulta extra que, en vez de contar el número de compañías que cumplen el objetivo, muestra cuáles son estas compañías.

```
-- Extra: Nombres de las compañías que verifican la consulta 1.
SELECT NombreC Companias
FROM Vuelo, Aeropuerto, Compania
-- El aeropuerto de salida o de llegada del vuelo
-- se encuentra en el estado de Alaska ('AK')
WHERE (clvDestino=IATA OR clvOrigen=IATA) AND Estado='AK' AND clvCompania=idCompania
-- Agrupar por nombres de compañías
GROUP BY NombreC
-- El número de aeropuertos de Alaska donde opera la compañía debe ser >= 5
HAVING 5<=(COUNT(DISTINCT IATA));
```

El resultado alcanzado con esta consulta extra corresponde a:

```
COMPANIAS
--------------------------------------------------------------------------------
Alaska Airlines Inc.
```

2.2.2.- Consulta 2

El **objetivo** de la consulta:
"Obtener el nombre, IATA y media de edad del aeropuerto en el que operan los aviones más modernos (es decir, con menor media de edad)".

El **álgebra relacional** que describe el enunciado de la consulta:

$R1 = \pi_{NombreA,\ IATA,\ Matricula,\ Montaje}\ (\sigma_{(clvDestino=IATA\ \vee\ clvOrigen=IATA)\ \wedge\ clvAvion=Matricula}\ (Aeropuerto \bowtie Avion \bowtie Vuelo))$

$R2 = \pi_{NombreA,\ IATA,\ total/cantidad\ Media}\ (AGRUPAR_{SUMAR(montaje)\ total,\ CONTAR(Montaje)\ cantidad}\ (R1,\ NombreA,\ IATA)$

$\pi_{NombreA,\ IATA,\ 2022-Media}\ (\sigma_{Media=m}\ (AGRUPAR_{MAX(Media)\ m}\ (R2)))$

La **sentencia SQL** que cumple con el objetivo indicado:

```sql
-- Vista con nombre, IATA y media de edad (en años) de cada aeropuerto.
CREATE VIEW Medias AS
  -- Para calcular la media de edad de cada aeropuerto:
  -- Se suman todas las fechas de montaje de cada avión y el resultado
  -- se divide entre el número de aviones que disponen de fecha de montaje.
  SELECT NombreA, IATA, SUM(Montaje)/COUNT(Montaje) Media
  FROM (
    -- Tabla con los distintos aviones y sus fechas de montaje de cada aeropuerto
    SELECT DISTINCT NombreA, IATA, Matricula, Montaje
    FROM Aeropuerto, Avion, Vuelo
    -- "Relacionar" el avión de cada vuelo con el aeropuerto en cuestión
    WHERE (clvDestino=IATA OR clvOrigen=IATA) AND clvAvion=Matricula
  )
  -- Agrupar por nombre y IATA de aeropuertos
  GROUP BY NombreA, IATA;


-- Consulta 2.
-- Para conseguir la media de edad real a partir de la media de los años
-- se hace la resta entre 2022 y la media calculada
SELECT NombreA Aeropuerto, IATA, 2022-Media Media
FROM Medias
-- La media de edad (en años) tiene que ser la mayor entre las medias calculadas
WHERE Media=(SELECT MAX(Media) FROM Medias);
```

La **respuesta** obtenida por la base de datos tras ejecutar la sentencia SQL:

```
AEROPUERTO                                          IATA      MEDIA
--------------------------------------------------- ---- ----------
Yampa Valley                                        HDN  17,8888889
```

2.2.3.- Consulta 3

El **objetivo** de la consulta:
"Obtener la(s) compañía(s) con el mayor porcentaje de vuelos que despegan y aterrizan en un mismo estado".

El **álgebra relacional** que describe el enunciado de la consulta:

```
R1 = π clvCompania, numT (AGRUPAR CONTAR(idVuelo) numT (Vuelo, clvCompania))
R2 = σ A1.IATA=clvOrigen ∧ A2.IATA=clvDestino ∧ A1.Estado=A2.Estado (R1 ⋈ Vuelo ⋈ Aeropuerto A1 ⋈ Aeropuerto A2)
π clvCompania, (numSt/numT)*100 (AGRUPAR CONTAR(idVuelo) numSt (R2, clvCompania, numT))
```

La **sentencia SQL** que cumple con el objetivo indicado:

```
-- Vista con clvCompania y porcentaje de vuelos de cada compañía
-- que despegan y aterrizan en un mismo estado
CREATE VIEW Porcentajes AS
  -- Para calcular el porcentaje de vuelos de cada compañía:
  -- Se cuenta el número de vuelos que cumplen las condiciones y se divide entre
  -- el número total de vuelos de la compañía. Luego, se multiplica por 100
  SELECT VC.clvCompania, (COUNT(*)/VC.num)*100 p
  FROM Vuelo V, Aeropuerto A1, Aeropuerto A2, (
    -- Tabla con el número total de vuelos de cada compañía
    SELECT clvCompania, COUNT(*) num
    FROM Vuelo
    -- Agrupar por compañía
    GROUP BY clvCompania)
    VC
  -- Los aeropuertos de salida y de llegada del vuelo ocurren en el mismo estado
  WHERE A1.IATA=V.clvOrigen AND A2.IATA=V.clvDestino AND A1.Estado=A2.Estado
    -- El número de vuelos totales y el número de vuelos condicionados
    -- pertenecen a la misma compañía
    AND VC.clvCompania=V.clvCompania
  -- Agrupar por compañía (y número de vuelos totales de esta)
  GROUP BY VC.clvCompania, VC.num;


-- Consulta 3.
SELECT NombreC Compania, p Porcentaje
FROM porcentajes, Compania
-- El porcentaje de vuelos tiene que ser el mayor
WHERE clvCompania=idCompania AND p=(SELECT MAX(p) FROM porcentajes);
```

La **respuesta** obtenida por la base de datos tras ejecutar la sentencia SQL:

```
COMPANIA                                           PORCENTAJE
-------------------------------------------------- ----------
Hawaiian Airlines Inc.                             85,0661626
```

3.- Diseño físico

3.1.- Rendimiento de las consultas

3.1.1.- Sin optimizar

Los resultados de rendimiento obtenidos sin optimizar las consultas son:

Consulta 1: coste de 222

```
-------------------------------------------------------------------------------------------
| Id  | Operation                  | Name            | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT           |                 |     1 |       |   222   (2)| 00:00:01 |
|   1 |  SORT AGGREGATE            |                 |     1 |       |            |          |
|   2 |   VIEW                     |                 |   100K|       |   222   (2)| 00:00:01 |
|*  3 |    HASH GROUP BY           |                 |   100K|   887K|   222   (2)| 00:00:01 |
|   4 |     VIEW                   | VM_NWVW_1       |   100K|   887K|   222   (2)| 00:00:01 |
|   5 |      HASH GROUP BY         |                 |   100K|   887K|   222   (2)| 00:00:01 |
|   6 |       VIEW                 | VW_ORE_EC848D2F |   100K|   887K|   219   (1)| 00:00:01 |
|   7 |        UNION-ALL           |                 |       |       |            |          |
|*  8 |         HASH JOIN          |                 | 50473 |   788K|   110   (1)| 00:00:01 |
|*  9 |          TABLE ACCESS FULL | AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  10 |          TABLE ACCESS FULL | VUELO           | 50473 |   443K|   102   (0)| 00:00:01 |
|* 11 |         HASH JOIN          |                 | 50458 |   985K|   110   (1)| 00:00:01 |
|* 12 |          TABLE ACCESS FULL | AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  13 |          TABLE ACCESS FULL | VUELO           | 50473 |   640K|   102   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

Consulta 2: coste de 232

```
-------------------------------------------------------------------------------------------------------
| Id  | Operation                  | Name                | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT           |                     |   100K|  4336K|   232   (3)| 00:00:01 |
|*  1 |  FILTER                    |                     |       |       |            |          |
|   2 |   HASH GROUP BY            |                     |   100K|  4336K|   232   (3)| 00:00:01 |
|   3 |    VIEW                    | VM_NWVW_0           |   100K|  4336K|   232   (3)| 00:00:01 |
|   4 |     HASH GROUP BY          |                     |   100K|  5322K|   232   (3)| 00:00:01 |
|*  5 |      HASH JOIN             |                     |   100K|  5322K|   229   (1)| 00:00:01 |
|   6 |       TABLE ACCESS FULL    | AVION               |  5333 | 95994 |     9   (0)| 00:00:01 |
|   7 |       VIEW                 | VW_JF_SET$5C077DE6  |   100K|  3548K|   219   (1)| 00:00:01 |
|   8 |        UNION-ALL           |                     |       |       |            |          |
|*  9 |         HASH JOIN          |                     | 50458 |  2168K|   110   (1)| 00:00:01 |
|  10 |          TABLE ACCESS FULL | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  11 |          TABLE ACCESS FULL | VUELO               | 50473 |   640K|   103   (1)| 00:00:01 |
|* 12 |         HASH JOIN          |                     | 50473 |  1971K|   110   (1)| 00:00:01 |
|  13 |          TABLE ACCESS FULL | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  14 |          TABLE ACCESS FULL | VUELO               | 50473 |   443K|   103   (1)| 00:00:01 |
|  15 |   SORT AGGREGATE           |                     |     1 |    13 |            |          |
|  16 |    VIEW                    | MEDIAS              |   100K|  1281K|   232   (3)| 00:00:01 |
|  17 |     SORT GROUP BY          |                     |   100K|  4336K|   232   (3)| 00:00:01 |
|  18 |      VIEW                  | VM_NWVW_1           |   100K|  4336K|   232   (3)| 00:00:01 |
|  19 |       SORT GROUP BY        |                     |   100K|  5322K|   232   (3)| 00:00:01 |
|* 20 |        HASH JOIN           |                     |   100K|  5322K|   229   (1)| 00:00:01 |
|  21 |         TABLE ACCESS FULL  | AVION               |  5333 | 95994 |     9   (0)| 00:00:01 |
|  22 |         VIEW               | VW_JF_SET$7711B281  |   100K|  3548K|   219   (1)| 00:00:01 |
|  23 |          UNION-ALL         |                     |       |       |            |          |
|* 24 |           HASH JOIN        |                     | 50458 |  2168K|   110   (1)| 00:00:01 |
|  25 |            TABLE ACCESS FULL| AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  26 |            TABLE ACCESS FULL| VUELO               | 50473 |   640K|   103   (1)| 00:00:01 |
|* 27 |           HASH JOIN        |                     | 50473 |  1971K|   110   (1)| 00:00:01 |
|  28 |            TABLE ACCESS FULL| AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  29 |            TABLE ACCESS FULL| VUELO               | 50473 |   443K|   103   (1)| 00:00:01 |
-------------------------------------------------------------------------------------------------------
```

```
--------------------------------------------------------------------------------------------
```

Consulta 3: coste de 296

```
-------------------------------------------------------------------------------------------
| Id  | Operation                    | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |             |  2509K|   179M|   296  (25)| 00:00:01 |
|   1 |  MERGE JOIN                  |             |  2509K|   179M|   296  (25)| 00:00:01 |
|   2 |   SORT JOIN                  |             |  2509K|    43M|   290  (25)| 00:00:01 |
|   3 |    VIEW                      | PORCENTAJES |  2509K|    43M|   290  (25)| 00:00:01 |
|*  4 |     FILTER                   |             |       |       |            |          |
|   5 |      HASH GROUP BY           |             |  2509K|   107M|   290  (25)| 00:00:01 |
|*  6 |       HASH JOIN              |             |  2509K|   107M|   228   (5)| 00:00:01 |
|*  7 |        HASH JOIN             |             |   895 | 24165 |   117   (1)| 00:00:01 |
|   8 |         TABLE ACCESS FULL    | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|*  9 |         HASH JOIN            |             | 50473 |   985K|   110   (1)| 00:00:01 |
|  10 |          TABLE ACCESS FULL   | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|  11 |          TABLE ACCESS FULL   | VUELO       | 50473 |   640K|   102   (0)| 00:00:01 |
|  12 |        VIEW                  |             | 50473 |   887K|   104   (2)| 00:00:01 |
|  13 |         HASH GROUP BY        |             | 50473 |   246K|   104   (2)| 00:00:01 |
|  14 |          TABLE ACCESS FULL   | VUELO       | 50473 |   246K|   102   (0)| 00:00:01 |
|  15 |      SORT AGGREGATE          |             |     1 |    13 |            |          |
|  16 |       VIEW                   | PORCENTAJES |  2509K|    31M|   290  (25)| 00:00:01 |
|  17 |        SORT GROUP BY         |             |  2509K|   107M|   290  (25)| 00:00:01 |
|* 18 |         HASH JOIN            |             |  2509K|   107M|   228   (5)| 00:00:01 |
|* 19 |          HASH JOIN           |             |   895 | 24165 |   117   (1)| 00:00:01 |
|  20 |           TABLE ACCESS FULL  | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|* 21 |           HASH JOIN          |             | 50473 |   985K|   110   (1)| 00:00:01 |
|  22 |            TABLE ACCESS FULL | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|  23 |            TABLE ACCESS FULL | VUELO       | 50473 |   640K|   102   (0)| 00:00:01 |
|  24 |          VIEW                |             | 50473 |   887K|   104   (2)| 00:00:01 |
|  25 |           SORT GROUP BY      |             | 50473 |   246K|   104   (2)| 00:00:01 |
|  26 |            TABLE ACCESS FULL | VUELO       | 50473 |   246K|   102   (0)| 00:00:01 |
|* 27 |   SORT JOIN                  |             |  1491 | 84987 |     6  (17)| 00:00:01 |
|  28 |    TABLE ACCESS FULL         | COMPANIA    |  1491 | 84987 |     5   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

3.1.2.- Partición vertical

A la hora de optimizar las consultas, se consideró probar a realizar una partición vertical con la tabla Vuelo, ya que más de la mitad de las columnas de esta no se utilizaban en ningún momento durante las consultas. Sin embargo, Oracle no dispone de un método predefinido para efectuar dichas particiones verticales, por lo que se realizó manualmente. Este proceso consistió en, primeramente, copiar el fichero crear.sql que contenía todas las sentencias de creación de tablas base. Dicha copia se modificó separando la tabla Vuelo en dos tablas distintas (VueloPrincipal y VueloSecundario, donde VueloSecundario depende de VueloPrincipal). Al eliminar la tabla Vuelo de esta forma, se creó una vista Vuelo con las mismas columnas que componían la tabla Vuelo original. La función de esta vista únicamente era, mediante un trigger auxiliar, poder insertar vuelos de la forma original. Así, el trigger, que se mostrará posteriormente en el apartado dedicado a ello, consiste en, cuando se intenta insertar una fila en Vuelo, en vez de eso, insertarla de forma particionada en VueloPrincipal y VueloSecundario, dejando Vuelo vacía.

Los resultados de rendimiento obtenidos utilizando particiones verticales en cada una de las consultas:

Consulta 1: coste de 154

```
-------------------------------------------------------------------------------------------
| Id  | Operation                | Name            | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
```

```
|   0 | SELECT STATEMENT          |                 |     1 |       |   154   (3)| 00:00:01 |
|   1 |  SORT AGGREGATE           |                 |     1 |       |            |          |
|   2 |   VIEW                    |                 |   107K|       |   154   (3)| 00:00:01 |
|*  3 |    HASH GROUP BY          |                 |   107K|   941K|   154   (3)| 00:00:01 |
|   4 |     VIEW                  | VM_NWVW_1       |   107K|   941K|   154   (3)| 00:00:01 |
|   5 |      HASH GROUP BY        |                 |   107K|   941K|   154   (3)| 00:00:01 |
|   6 |       VIEW                | VW_ORE_C7C2CD8F |   107K|   941K|   151   (1)| 00:00:01 |
|   7 |        UNION-ALL          |                 |       |       |            |          |
|*  8 |         HASH JOIN         |                 | 53583 |   837K|    76   (2)| 00:00:01 |
|*  9 |          TABLE ACCESS FULL| AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  10 |          TABLE ACCESS FULL| VUELOPRINCIPAL  | 53583 |   470K|    68   (0)| 00:00:01 |
|* 11 |         HASH JOIN         |                 | 53567 |  1046K|    76   (2)| 00:00:01 |
|* 12 |          TABLE ACCESS FULL| AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  13 |          TABLE ACCESS FULL| VUELOPRINCIPAL  | 53583 |   680K|    68   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

Consulta 2: coste de 163

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name                | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |                     |   107K|  4604K|   163   (3)| 00:00:01 |
|*  1 |  FILTER                      |                     |       |       |            |          |
|   2 |   HASH GROUP BY              |                     |   107K|  4604K|   163   (3)| 00:00:01 |
|   3 |    VIEW                      | VM_NWVW_0           |   107K|  4604K|   163   (3)| 00:00:01 |
|   4 |     HASH GROUP BY            |                     |   107K|  5650K|   163   (3)| 00:00:01 |
|*  5 |      HASH JOIN               |                     |   107K|  5650K|   160   (1)| 00:00:01 |
|   6 |       TABLE ACCESS FULL      | AVION               |  5333 | 95994 |     9   (0)| 00:00:01 |
|   7 |       VIEW                   | VW_JF_SET$47F9BF4C  |   107K|  3766K|   151   (1)| 00:00:01 |
|   8 |        UNION-ALL             |                     |       |       |            |          |
|*  9 |         HASH JOIN            |                     | 53567 |  2301K|    76   (2)| 00:00:01 |
|  10 |          TABLE ACCESS FULL   | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  11 |          TABLE ACCESS FULL   | VUELOPRINCIPAL      | 53583 |   680K|    68   (0)| 00:00:01 |
|* 12 |         HASH JOIN            |                     | 53583 |  2093K|    76   (2)| 00:00:01 |
|  13 |          TABLE ACCESS FULL   | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  14 |          TABLE ACCESS FULL   | VUELOPRINCIPAL      | 53583 |   470K|    68   (0)| 00:00:01 |
|  15 |   SORT AGGREGATE             |                     |     1 |    13 |            |          |
|  16 |    VIEW                      | MEDIAS              |   107K|  1360K|   163   (3)| 00:00:01 |
|  17 |     SORT GROUP BY            |                     |   107K|  4604K|   163   (3)| 00:00:01 |
|  18 |      VIEW                    | VM_NWVW_1           |   107K|  4604K|   163   (3)| 00:00:01 |
|  19 |       SORT GROUP BY          |                     |   107K|  5650K|   163   (3)| 00:00:01 |
|* 20 |        HASH JOIN             |                     |   107K|  5650K|   160   (1)| 00:00:01 |
|  21 |         TABLE ACCESS FULL    | AVION               |  5333 | 95994 |     9   (0)| 00:00:01 |
|  22 |         VIEW                 | VW_JF_SET$F94C21C4  |   107K|  3766K|   151   (1)| 00:00:01 |
|  23 |          UNION-ALL           |                     |       |       |            |          |
|* 24 |           HASH JOIN          |                     | 53567 |  2301K|    76   (2)| 00:00:01 |
|  25 |            TABLE ACCESS FULL | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  26 |            TABLE ACCESS FULL | VUELOPRINCIPAL      | 53583 |   680K|    68   (0)| 00:00:01 |
|* 27 |           HASH JOIN          |                     | 53583 |  2093K|    76   (2)| 00:00:01 |
|  28 |            TABLE ACCESS FULL | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01 |
|  29 |            TABLE ACCESS FULL | VUELOPRINCIPAL      | 53583 |   470K|    68   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

Consulta 3: coste de 237

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |                |  2828K|   202M|   237  (35)| 00:00:01 |
|   1 |  MERGE JOIN                  |                |  2828K|   202M|   237  (35)| 00:00:01 |
|   2 |   SORT JOIN                  |                |  2828K|    48M|   231  (36)| 00:00:01 |
|   3 |    VIEW                      | PORCENTAJES    |  2828K|    48M|   231  (36)| 00:00:01 |
|*  4 |     FILTER                   |                |       |       |            |          |
|   5 |      HASH GROUP BY           |                |  2828K|   121M|   231  (36)| 00:00:01 |
|*  6 |       HASH JOIN              |                |  2828K|   121M|   160   (7)| 00:00:01 |
|*  7 |        HASH JOIN             |                |   950 | 25650 |    83   (2)| 00:00:01 |
|   8 |         TABLE ACCESS FULL    | AEROPUERTO     |  3376 | 23632 |     7   (0)| 00:00:01 |
|*  9 |         HASH JOIN            |                | 53583 |  1046K|    76   (2)| 00:00:01 |
|  10 |          TABLE ACCESS FULL   | AEROPUERTO     |  3376 | 23632 |     7   (0)| 00:00:01 |
```

```
|  11 |          TABLE ACCESS FULL  | VUELOPRINCIPAL | 53583 |   680K|    68   (0)| 00:00:01 |
|  12 |        VIEW                 |                | 53583 |   941K|    70   (3)| 00:00:01 |
|  13 |         HASH GROUP BY       |                | 53583 |   261K|    70   (3)| 00:00:01 |
|  14 |          TABLE ACCESS FULL  | VUELOPRINCIPAL | 53583 |   261K|    68   (0)| 00:00:01 |
|  15 |      SORT AGGREGATE         |                |     1 |    13 |             |          |
|  16 |       VIEW                  | PORCENTAJES    |  2828K|    35M|   231  (36)| 00:00:01 |
|  17 |        SORT GROUP BY        |                |  2828K|   121M|   231  (36)| 00:00:01 |
|* 18 |         HASH JOIN           |                |  2828K|   121M|   160   (7)| 00:00:01 |
|* 19 |          HASH JOIN          |                |   950 | 25650 |    83   (2)| 00:00:01 |
|  20 |           TABLE ACCESS FULL | AEROPUERTO     |  3376 | 23632 |     7   (0)| 00:00:01 |
|* 21 |           HASH JOIN         |                | 53583 |  1046K|    76   (2)| 00:00:01 |
|  22 |            TABLE ACCESS FULL| AEROPUERTO     |  3376 | 23632 |     7   (0)| 00:00:01 |
|  23 |            TABLE ACCESS FULL| VUELOPRINCIPAL | 53583 |   680K|    68   (0)| 00:00:01 |
|  24 |          VIEW               |                | 53583 |   941K|    70   (3)| 00:00:01 |
|  25 |           SORT GROUP BY     |                | 53583 |   261K|    70   (3)| 00:00:01 |
|  26 |            TABLE ACCESS FULL| VUELOPRINCIPAL | 53583 |   261K|    68   (0)| 00:00:01 |
|* 27 |   SORT JOIN                 |                |  1491 | 84987 |     6  (17)| 00:00:01 |
|  28 |    TABLE ACCESS FULL        | COMPANIA       |  1491 | 84987 |     5   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

3.1.3.- Índice

También se creyó conveniente al momento de optimizar probar varios índices, de los cuales los que mejor funcionaron fueron dos distintos, ambos aplicados a tuplas de la tabla Vuelo.

El índice que se aplicó para la primera y tercera consulta es:

```
CREATE INDEX ind_vuelo1 ON Vuelo(clvOrigen, clvDestino, clvCompania);
```

El índice que se aplicó para la segundas consulta es:

```
CREATE INDEX ind_vuelo2 ON Vuelo(clvOrigen, clvDestino, clvAvion);
```

Los resultados de rendimiento obtenidos utilizando índices en cada una de las consultas:

Consulta 1: coste de 110

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name            | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |                 |     1 |       |   110   (4)| 00:00:01 |
|   1 |  SORT AGGREGATE              |                 |     1 |       |            |          |
|   2 |   VIEW                       |                 |   100K|       |   110   (4)| 00:00:01 |
|*  3 |    HASH GROUP BY             |                 |   100K|   887K|   110   (4)| 00:00:01 |
|   4 |     VIEW                     | VM_NWVW_1       |   100K|   887K|   110   (4)| 00:00:01 |
|   5 |      HASH GROUP BY           |                 |   100K|   887K|   110   (4)| 00:00:01 |
|   6 |       VIEW                   | VW_ORE_EC848D2F |   100K|   887K|   107   (1)| 00:00:01 |
|   7 |        UNION-ALL             |                 |       |       |            |          |
|*  8 |         HASH JOIN            |                 | 50473 |   788K|    53   (0)| 00:00:01 |
|*  9 |          TABLE ACCESS FULL   | AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  10 |          INDEX FAST FULL SCAN| IND_VUELO2      | 50473 |   443K|    46   (0)| 00:00:01 |
|* 11 |         HASH JOIN            |                 | 50458 |   985K|    53   (0)| 00:00:01 |
|* 12 |          TABLE ACCESS FULL   | AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  13 |          INDEX FAST FULL SCAN| IND_VUELO2      | 50473 |   640K|    46   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

Consulta 2: coste de 133

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name              | Rows  | Bytes | Cost (%CPU)| Time
|
```

```
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT               |                       |   100K|  4336K|   133   (4)| 00:00:01
|*  1 |  FILTER                        |                       |       |       |            |
|   2 |   HASH GROUP BY                |                       |   100K|  4336K|   133   (4)| 00:00:01
|   3 |    VIEW                        | VM_NWVW_0             |   100K|  4336K|   133   (4)| 00:00:01
|   4 |     HASH GROUP BY              |                       |   100K|  5322K|   133   (4)| 00:00:01
|*  5 |      HASH JOIN                 |                       |   100K|  5322K|   130   (1)| 00:00:01
|   6 |       TABLE ACCESS FULL        | AVION                 |  5333 | 95994 |     9   (0)| 00:00:01
|
|   7 |       VIEW                     | VW_JF_SET$5C077DE6    |   100K|  3548K|   121   (1)| 00:00:01|
|   8 |        UNION-ALL               |                       |       |       |            |
|*  9 |         HASH JOIN              |                       | 50458 |  2168K|    60   (0)| 00:00:01
|  10 |          TABLE ACCESS FULL     | AEROPUERTO            |  3376 |   102K|     7   (0)| 00:00:01
|  11 |          INDEX FAST FULL SCAN  | IND_VUELO1            | 50473 |   640K|    53   (0)| 00:00:01
|* 12 |         HASH JOIN              |                       | 50473 |  1971K|    60   (0)| 00:00:01
|  13 |          TABLE ACCESS FULL     | AEROPUERTO            |  3376 |   102K|     7   (0)| 00:00:01
|  14 |          INDEX FAST FULL SCAN  | IND_VUELO1            | 50473 |   443K|    53   (0)| 00:00:01
|  15 |   SORT AGGREGATE               |                       |     1 |    13 |            |
|  16 |    VIEW                        | MEDIAS                |   100K|  1281K|   133   (4)| 00:00:01
|  17 |     SORT GROUP BY              |                       |   100K|  4336K|   133   (4)| 00:00:01
|  18 |      VIEW                      | VM_NWVW_1             |   100K|  4336K|   133   (4)| 00:00:01
|  19 |       SORT GROUP BY            |                       |   100K|  5322K|   133   (4)| 00:00:01
|* 20 |        HASH JOIN               |                       |   100K|  5322K|   130   (1)| 00:00:01
|  21 |         TABLE ACCESS FULL      | AVION                 |  5333 | 95994 |     9   (0)| 00:00:01
|  22 |         VIEW                   | VW_JF_SET$7711B281    |   100K|  3548K|   121   (1)| 00:00:01
|  23 |          UNION-ALL             |                       |       |       |            |
|* 24 |           HASH JOIN            |                       | 50458 |  2168K|    60   (0)| 00:00:01
|  25 |            TABLE ACCESS FULL   | AEROPUERTO            |  3376 |   102K|     7   (0)| 00:00:01
|  26 |            INDEX FAST FULL SCAN| IND_VUELO1            | 50473 |   640K|    53   (0)| 00:00:01
|* 27 |           HASH JOIN            |                       | 50473 |  1971K|    60   (0)| 00:00:01
|  28 |            TABLE ACCESS FULL   | AEROPUERTO            |  3376 |   102K|     7   (0)| 00:00:01
|  29 |            INDEX FAST FULL SCAN| IND_VUELO1            | 50473 |   443K|    53   (0)| 00:00:01
-----------------------------------------------------------------------------------------------
```

Consulta 3: coste de 184

```
-------------------------------------------------------------------------------------------------
| Id  | Operation                       | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                |              |  2509K|   179M|   184  (40)| 00:00:01 |
|   1 |  MERGE JOIN                     |              |  2509K|   179M|   184  (40)| 00:00:01 |
|   2 |   SORT JOIN                     |              |  2509K|    43M|   178  (41)| 00:00:01 |
|   3 |    VIEW                         | PORCENTAJES  |  2509K|    43M|   178  (41)| 00:00:01 |
|*  4 |     FILTER                      |              |       |       |            |          |
|   5 |      HASH GROUP BY              |              |  2509K|   107M|   178  (41)| 00:00:01 |
|*  6 |       HASH JOIN                 |              |  2509K|   107M|   115   (8)| 00:00:01 |
|*  7 |        HASH JOIN                |              |   895 | 24165 |    61   (2)| 00:00:01 |
|   8 |         TABLE ACCESS FULL       | AEROPUERTO   |  3376 | 23632 |     7   (0)| 00:00:01 |
|*  9 |         HASH JOIN               |              | 50473 |   985K|    53   (0)| 00:00:01 |
|  10 |          TABLE ACCESS FULL      | AEROPUERTO   |  3376 | 23632 |     7   (0)| 00:00:01 |
|  11 |          INDEX FAST FULL SCAN   | IND_VUELO2   | 50473 |   640K|    46   (0)| 00:00:01 |
|  12 |        VIEW                     |              | 50473 |   887K|    48   (5)| 00:00:01 |
|  13 |         HASH GROUP BY           |              | 50473 |   246K|    48   (5)| 00:00:01 |
|  14 |          INDEX FAST FULL SCAN   | IND_VUELO2   | 50473 |   246K|    46   (0)| 00:00:01 |
|  15 |      SORT AGGREGATE             |              |     1 |    13 |            |          |
|  16 |       VIEW                      | PORCENTAJES  |  2509K|    31M|   178  (41)| 00:00:01 |
|  17 |        SORT GROUP BY            |              |  2509K|   107M|   178  (41)| 00:00:01 |
|* 18 |         HASH JOIN               |              |  2509K|   107M|   115   (8)| 00:00:01 |
|* 19 |          HASH JOIN              |              |   895 | 24165 |    61   (2)| 00:00:01 |
|  20 |           TABLE ACCESS FULL     | AEROPUERTO   |  3376 | 23632 |     7   (0)| 00:00:01 |
|* 21 |           HASH JOIN             |              | 50473 |   985K|    53   (0)| 00:00:01 |
|  22 |            TABLE ACCESS FULL    | AEROPUERTO   |  3376 | 23632 |     7   (0)| 00:00:01 |
|  23 |            INDEX FAST FULL SCAN | IND_VUELO2   | 50473 |   640K|    46   (0)| 00:00:01 |
|  24 |          VIEW                   |              | 50473 |   887K|    48   (5)| 00:00:01 |
|  25 |           SORT GROUP BY         |              | 50473 |   246K|    48   (5)| 00:00:01 |
|  26 |            INDEX FAST FULL SCAN | IND_VUELO2   | 50473 |   246K|    46   (0)| 00:00:01 |
|* 27 |   SORT JOIN                     |              |  1491 | 84987 |     6  (17)| 00:00:01 |
|  28 |    TABLE ACCESS FULL            | COMPANIA     |  1491 | 84987 |     5   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------
```

3.1.4.- Partición vertical e índice conjuntos

Al observar que tanto mediante particiones verticales como mediante índices se detectaron mejoras significativas en el rendimiento de las consultas, se trató de fusionarlos esperando una mejoría, de forma que se aplicaron los índices anteriores a la tabla VueloPrincipal.

El índice que se aplicó para la primera y tercera consulta es:

```
CREATE INDEX ind_vuelo1 on VueloPrincipal(clvOrigen, clvDestino, clvCompania);
```

El índice que se aplicó para la segundas consulta es:

```
CREATE INDEX ind_vuelo2 on VueloPrincipal(clvOrigen, clvDestino, clvAvion);
```

Los resultados de rendimiento obtenidos utilizando particiones verticales e índices en cada una de las consultas:

Consulta 1: coste de 110

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                     | Name            | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                 |     1 |       |   110   (4)| 00:00:01 |
|   1 |  SORT AGGREGATE               |                 |     1 |       |            |          |
|   2 |   VIEW                        |                 |   107K|       |   110   (4)| 00:00:01 |
|*  3 |    HASH GROUP BY              |                 |   107K|   941K|   110   (4)| 00:00:01 |
|   4 |     VIEW                      | VM_NWVW_1       |   107K|   941K|   110   (4)| 00:00:01 |
|   5 |      HASH GROUP BY            |                 |   107K|   941K|   110   (4)| 00:00:01 |
|   6 |       VIEW                    | VW_ORE_C7C2CD8F |   107K|   941K|   107   (1)| 00:00:01 |
|   7 |        UNION-ALL              |                 |       |       |            |          |
|*  8 |         HASH JOIN             |                 | 53583 |   837K|    53   (0)| 00:00:01 |
|*  9 |          TABLE ACCESS FULL    | AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  10 |          INDEX FAST FULL SCAN | IND_VUELO2      | 53583 |   470K|    46   (0)| 00:00:01 |
|* 11 |         HASH JOIN             |                 | 53567 |  1046K|    53   (0)| 00:00:01 |
|* 12 |          TABLE ACCESS FULL    | AEROPUERTO      |   263 |  1841 |     7   (0)| 00:00:01 |
|  13 |          INDEX FAST FULL SCAN | IND_VUELO2      | 53583 |   680K|    46   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

Consulta 2: coste de 133

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                     | Name               | Rows  | Bytes | Cost (%CPU)| Time
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                    |   107K|  4604K|   133   (4)| 00:00:01
|*  1 |  FI|
|   2 |   HASH GROUP BY               |                    |   107K|  4604K|   133   (4)| 00:00:01
|   3 |    VIEW                       | VM_NWVW_0          |   107K|  4604K|   133   (4)| 00:00:01
|   4 |     HASH GROUP BY             |                    |   107K|  5650K|   133   (4)| 00:00:01
|*  5 |      HASH JOIN                |                    |   107K|  5650K|   130   (1)| 00:00:01
|   6 |       TABLE ACCESS FULL       | AVION              |  5333 | 95994 |     9   (0)| 00:00:01
|   7 |       VIEW                    | VW_JF_SET$47F9BF4C |   107K|  3766K|   121   (1)| 00:00:01
|   8 |        UNION-ALL              |                    |       |       |            |
|*  9 |         HASH JOIN             |                    | 53567 |  2301K|    60   (0)| 00:00:01
|  10 |          TABLE ACCESS FULL    | AEROPUERTO         |  3376 |   102K|     7   (0)| 00:00:01
|  11 |          INDEX FAST FULL SCAN | IND_VUELO1         | 53583 |   680K|    53   (0)| 00:00:01
|* 12 |         HASH JOIN             |                    | 53583 |  2093K|    60   (0)| 00:00:01
|  13 |          TABLE ACCESS FULL    | AEROPUERTO         |  3376 |   102K|     7   (0)| 00:00:01
|  14 |          INDEX FAST FULL SCAN | IND_VUELO1         | 53583 |   470K|    53   (0)| 00:00:01
|  15 |   SORT AGGREGATE              |                    |     1 |    13 |            |
|  16 |    VIEW                       | MEDIAS             |   107K|  1360K|   133   (4)| 00:00:01
|  17 |     SORT GROUP BY             |                    |   107K|  4604K|   133   (4)| 00:00:01
|  18 |      VIEW                     | VM_NWVW_1          |   107K|  4604K|   133   (4)| 00:00:01
```

```
|  19 |       SORT GROUP BY           |                     |   107K|  5650K|   133   (4)| 00:00:01
|* 20 |        HASH JOIN              |                     |   107K|  5650K|   130   (1)| 00:00:01
|  21 |         TABLE ACCESS FULL     | AVION               |  5333 | 95994 |     9   (0)| 00:00:01
|  22 |         VIEW                  | VW_JF_SET$F94C21C4  |   107K|  3766K|   121   (1)| 00:00:01
|  23 |          UNION-ALL            |                     |       |       |            |
|* 24 |           HASH JOIN           |                     | 53567 |  2301K|    60   (0)| 00:00:01
|  25 |            TABLE ACCESS FULL  | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01
|  26 |            INDEX FAST FULL SCAN| IND_VUELO1          | 53583 |   680K|    53   (0)| 00:00:01
|* 27 |           HASH JOIN           |                     | 53583 |  2093K|    60   (0)| 00:00:01
|  28 |            TABLE ACCESS FULL  | AEROPUERTO          |  3376 |   102K|     7   (0)| 00:00:01
|  29 |            INDEX FAST FULL SCAN| IND_VUELO1          | 53583 |   470K|    53   (0)| 00:00:01
-------------------------------------------------------------------------------------------------
```

Consulta 3: coste de 193

```
-------------------------------------------------------------------------------------------------
| Id  | Operation                      | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT               |             |  2828K|   202M|   193  (43)| 00:00:01 |
|   1 |  MERGE JOIN                    |             |  2828K|   202M|   193  (43)| 00:00:01 |
|   2 |   SORT JOIN                    |             |  2828K|    48M|   187  (44)| 00:00:01 |
|   3 |    VIEW                        | PORCENTAJES |  2828K|    48M|   187  (44)| 00:00:01 |
|*  4 |     FILTER                     |             |       |       |            |          |
|   5 |      HASH GROUP BY             |             |  2828K|   121M|   187  (44)| 00:00:01 |
|*  6 |       HASH JOIN                |             |  2828K|   121M|   116   (9)| 00:00:01 |
|*  7 |        HASH JOIN               |             |   950 | 25650 |    61   (2)| 00:00:01 |
|   8 |         TABLE ACCESS FULL      | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|*  9 |         HASH JOIN              |             | 53583 |  1046K|    53   (0)| 00:00:01 |
|  10 |          TABLE ACCESS FULL     | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|  11 |          INDEX FAST FULL SCAN  | IND_VUELO2  | 53583 |   680K|    46   (0)| 00:00:01 |
|  12 |        VIEW                    |             | 53583 |   941K|    48   (5)| 00:00:01 |
|  13 |         HASH GROUP BY          |             | 53583 |   261K|    48   (5)| 00:00:01 |
|  14 |          INDEX FAST FULL SCAN  | IND_VUELO2  | 53583 |   261K|    46   (0)| 00:00:01 |
|  15 |       SORT AGGREGATE           |             |     1 |    13 |            |          |
|  16 |        VIEW                    | PORCENTAJES |  2828K|    35M|   187  (44)| 00:00:01 |
|  17 |         SORT GROUP BY          |             |  2828K|   121M|   187  (44)| 00:00:01 |
|* 18 |          HASH JOIN             |             |  2828K|   121M|   116   (9)| 00:00:01 |
|* 19 |           HASH JOIN            |             |   950 | 25650 |    61   (2)| 00:00:01 |
|  20 |            TABLE ACCESS FULL   | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|* 21 |            HASH JOIN           |             | 53583 |  1046K|    53   (0)| 00:00:01 |
|  22 |             TABLE ACCESS FULL  | AEROPUERTO  |  3376 | 23632 |     7   (0)| 00:00:01 |
|  23 |             INDEX FAST FULL SCAN| IND_VUELO2  | 53583 |   680K|    46   (0)| 00:00:01 |
|  24 |           VIEW                 |             | 53583 |   941K|    48   (5)| 00:00:01 |
|  25 |            SORT GROUP BY       |             | 53583 |   261K|    48   (5)| 00:00:01 |
|  26 |             INDEX FAST FULL SCAN| IND_VUELO2  | 53583 |   261K|    46   (0)| 00:00:01 |
|* 27 |    SORT JOIN                   |             |  1491 | 84987 |     6  (17)| 00:00:01 |
|  28 |     TABLE ACCESS FULL          | COMPANIA    |  1491 | 84987 |     5   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------
```

3.1.5.- Vistas materializadas

Finalmente, como en dos de las consultas se utilizan vistas, se creyó apropiado probar a materializar dichas vistas para evaluar la mejora que conllevaría.

La vistas materializadas que se han aplicado:

```
CREATE MATERIALIZED VIEW Medias ...;
```

```
CREATE MATERIALIZED VIEW Porcentajes ...;
```

Los resultados de rendimiento obtenidos utilizando vistas materializadas en las dos últimas consultas:

Consulta 2: coste de 6

```
-------------------------------------------------------------------------------
| Id  | Operation             | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |        |     1 |    44 |     6   (0)| 00:00:01 |
|*  1 |  MAT_VIEW ACCESS FULL | MEDIAS |     1 |    44 |            |          |
|   2 |   SORT AGGREGATE      |        |     1 |    13 |     3   (0)| 00:00:01 |
|   3 |    MAT_VIEW ACCESS FULL| MEDIAS |   272 |  3536 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

Consulta 3: coste de 7

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                     | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                |     1 |    75 |     7   (0)| 00:00:01 |
|   1 |  NESTED LOOPS                 |                |     1 |    75 |     4   (0)| 00:00:01 |
|   2 |   NESTED LOOPS                |                |     1 |    75 |     4   (0)| 00:00:01 |
|*  3 |    MAT_VIEW ACCESS FULL       | PORCENTAJES    |     1 |    18 |     3   (0)| 00:00:01 |
|   4 |     SORT AGGREGATE            |                |     1 |    13 |            |          |
|   5 |      MAT_VIEW ACCESS FULL     | PORCENTAJES    |    17 |   221 |     3  (40)| 00:00:01 |
|*  6 |     INDEX UNIQUE SCAN         | SYS_C003403309 |     1 |       |     0   (0)| 00:00:01 |
|   7 |    TABLE ACCESS BY INDEX ROWID| COMPANIA       |     1 |    57 |     1   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

3.1.6.- Conclusión

Después de haber obtenido todos estos valores, excluyendo los datos de las vistas materializadas que son muy condicionales de determinadas situaciones, se observa que la alternativa más óptima sería utilizar únicamente índices. Esta conclusión difiere de la esperada inicialmente, donde se sostenía que la más eficaz sería aplicar índices con particiones verticales al mismo tiempo. Sin embargo, después de volver a evaluarlo se ha deducido que, al aplicar una partición sobre la tabla Vuelo, la tabla VueloPrincipal resultante ocupa menos bloques de disco, por lo que la aplicación de los índices, aunque sigue mejorando el rendimiento considerablemente, no es tan eficaz como al aplicarlos sobre una tabla que ocupa más bloques de disco.

En lo relativo a las vistas materializadas, la mejora en coste tan alta que se observa se debe a que dichas vistas precalculan todos los valores necesarios para las consultas, de modo que estas solo tengan que buscar aquellas filas con un valor máximo de entre los que hay en la vista materializada. Sin embargo, el uso óptimo de vistas materializadas depende de la frecuencia con que los datos de la base se vayan a actualizar. Cuanto mayor sea la frecuencia, más implicaría crear la vista materializada cada menos tiempo, lo que conllevaría un coste constante extra. Por otro lado, cuanto menor sea la frecuencia, al tener un tiempo más distante entre las actualizaciones de las vistas materializadas, rentabilizaría utilizarlas.

3.2.- Restricciones no verificables

Las restricciones que Oracle no puede verificar con la estructura de tablas definidas ni con las restricciones de integridad de los `CREATE TABLE` son:

- No pueden ocurrir dos vuelos con el mismo avión en las mismas fechas.
- Un vuelo no puede ocurrir en un avión que todavía no se ha creado.
- El aeropuerto de llegada de un vuelo debe ser distinto del aeropuerto relativo a un desvío.
- Si un vuelo está cancelado, no tiene desvíos.
- Cuando se usan particiones verticales, al intentar insertar elementos en la tabla (o vista) Vuelo, en vez de eso se han de insertar en las tablas VueloPrincipal y VueloSecundario.

3.3.- Triggers

3.3.1.- Trigger 1

El **objetivo** del trigger es:
"No pueden existir desvíos de vuelos cancelados y el destino del desvío no puede ser el mismo que el destino del vuelo".

La **sentencia SQL** que cumple con el objetivo indicado:

```
-- Trigger 1.
CREATE OR REPLACE TRIGGER trigger1
BEFORE INSERT ON Desvio
FOR EACH ROW
DECLARE
  n NUMBER;
  m NUMBER;
BEGIN
  -- No pueden existir desvios de vuelos cancelados
  -- m=1 si Cancelado=1
  SELECT Cancelado
  INTO m
  FROM Vuelo
  WHERE idVuelo=:NEW.clvVuelo;
  IF m=1 THEN RAISE_APPLICATION_ERROR(-20002, 'No pueden existir desvios de vuelos
cancelados.');
  END IF;
  -- El destino del desvío no puede ser el mismo que el destino del vuelo
  -- n=1 si hay el vuelo al que hace referencia tiene el mismo destino
  SELECT COUNT(*)
  INTO n
  FROM Vuelo
  WHERE idVuelo=:NEW.clvVuelo AND clvDestino=:NEW.clvAeropuerto;
  IF n=1 THEN RAISE_APPLICATION_ERROR(-20001, 'El destino del desvio no puede ser el
mismo que el destino del vuelo.');
  END IF;
END trigger1;
/
```

3.3.2.- Trigger 2

El **objetivo** del trigger es:
"Un vuelo no puede ocurrir en un avión que todavía no se ha creado".

La **sentencia SQL** que cumple con el objetivo indicado:

```
-- Trigger 2.
CREATE OR REPLACE TRIGGER trigger2
BEFORE INSERT ON Vuelo
FOR EACH ROW
DECLARE
  n NUMBER;
BEGIN
```

```
  -- n=1 si el año de creación del avión es posterior al año del vuelo
  SELECT COUNT(*)
  INTO n
  FROM (
    -- Se utiliza SUBSTR para extraer el año de la fecha de vuelo
    -- (en formato aaaa-mm-dd)
    SELECT Montaje, SUBSTR(:NEW.Fecha,1,4) Anyo
    FROM Avion
    WHERE Matricula=:NEW.clvAvion )
  WHERE Montaje>Anyo;
  IF n=1 THEN RAISE_APPLICATION_ERROR(-20004, 'Un vuelo no puede ocurrir en un avion que
todavia no se ha creado.');
  END IF;
END trigger2;
/
```

3.3.3.- Trigger 3

El **objetivo** del trigger es:
"Cuando se usan particiones verticales, al intentar insertar elementos en la tabla (o vista) Vuelo, en vez de eso se han de insertar en las tablas VueloPrincipal y VueloSecundario".

La **sentencia SQL** que cumple con el objetivo indicado:

```
-- Trigger 3.
CREATE OR REPLACE TRIGGER trigger3
-- Evitar que se inserte en Vuelo
INSTEAD OF INSERT ON Vuelo
FOR EACH ROW
BEGIN
  -- Insertar en VueloPrincipal
  INSERT INTO VueloPrincipal (idVuelo, clvOrigen, clvDestino, clvCompania, clvAvion)
    VALUES (:NEW.idVuelo, :NEW.clvOrigen, :NEW.clvDestino, :NEW.clvCompania,
:NEW.clvAvion);
  -- Insertar en VueloSecundario
  INSERT INTO VueloSecundario (idVuelo, numVuelo, Salida, Llegada, Fecha, Cancelado,
Retraso)
    VALUES (:NEW.idVuelo, :NEW.numVuelo, :NEW.Salida, :NEW.Llegada, :NEW.Fecha,
:NEW.Cancelado, :NEW.Retraso);
END trigger3;
/
```

4.- Anexo: organización

Tiempo (h)	Álvaro Seral	Cristian Selivanov	Dorian Wozniak	TOTAL
Creación modelo	3	7	2	12
Población	5	3	2	10
Consultas	6	4	3	13
Optimización	6	4	5	15
Triggers	4	3	4	11
Memoria	7	5	4	16
TOTAL	31	26	20	76