

UAM EPS

# Memoria Practica1

Bases de Datos

Luis Fernández Fernández y Nicolás Ruiz Martínez  
10-25-2023

## Índice

Claves primarias y extranjeras de cada tabla: .....	1
Diagrama del modelo relacional .....	2
Query 1: .....	3
Query 2: .....	4
Query 3: .....	5
Query 4: .....	6
Query 5: .....	7
Query 6: .....	7

## Claves primarias y extranjeras de cada tabla:

aircrafts\_data (**aircraft\_code**, model)

airports\_data (**airport\_code**, airport\_name, city, coordinates, timezone)

boarding\_passes (**ticket\_no** → ticket\_flights.ticket\_no, **flight\_id** → ticket\_flights.flight\_id, boarding\_no, seat\_no)

bookings (**book\_ref**, book\_date, total\_amount)

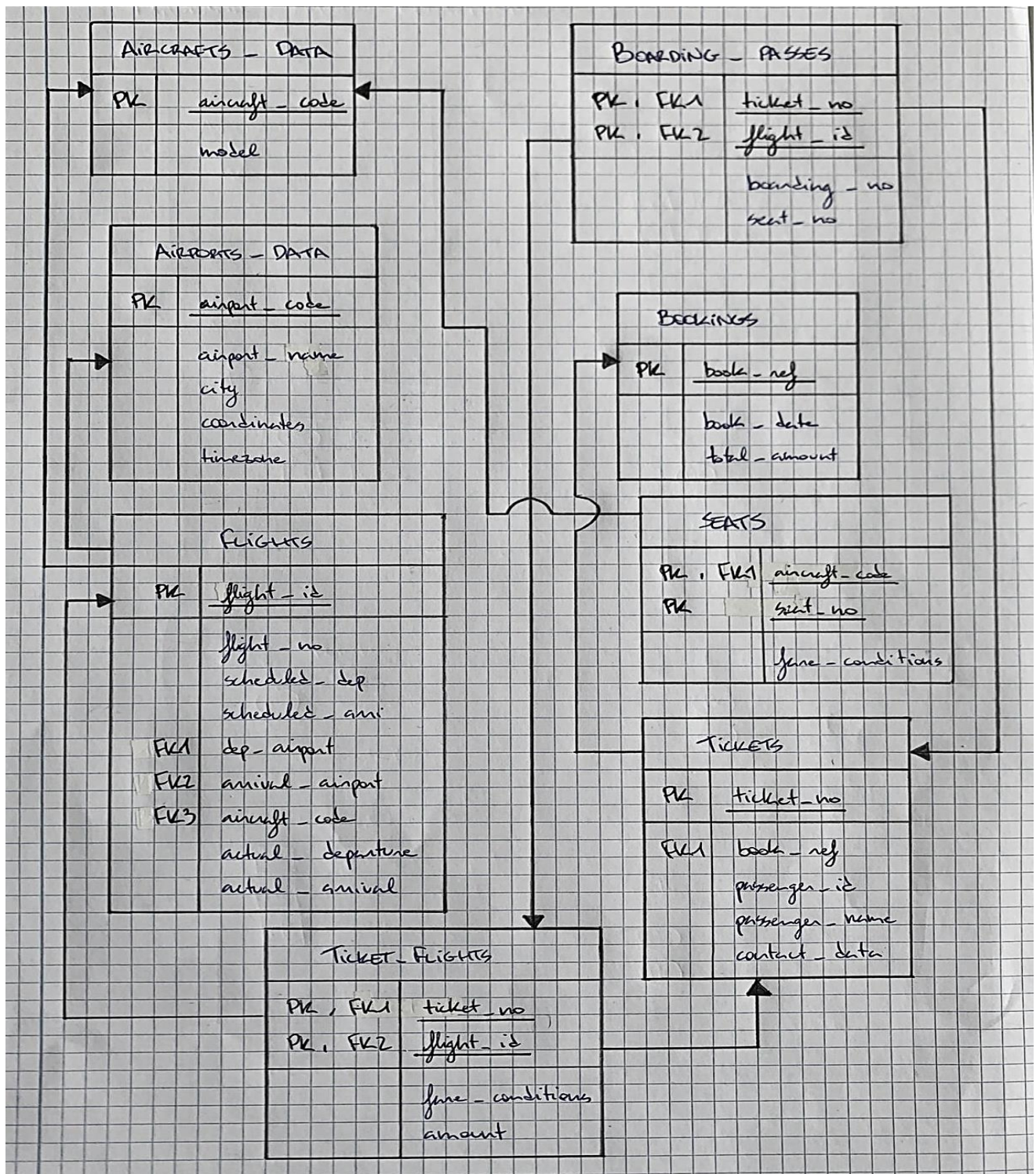
flights (**flight\_id**, flight\_no, scheduled\_departure, scheduled\_arrival, departure\_airport → airports\_data.airport\_code, arrival\_airport → airports\_data.airport\_code, aircraft\_code → aircrafts\_data.aircraft\_code, actual\_departure, actual\_arrival)

seats (**seat\_no**, **aircraft\_code** → aircrafts\_data.aircraft\_code, fare\_conditions)

ticket\_flights (**ticket\_no** → tickets.ticket\_no, **flight\_id** → flights.flight\_id, fare\_conditions, amount)

tickets (**ticket\_no**, book\_ref → bookings.book\_ref, passenger\_id, passenger\_name, contact\_data)

## Diagrama del modelo relacional



Todas las queries se han obtenido en formato html (y texto) para una mejor visualización de la información usando el comando \H en psql para cambiar la salida a dicho formato. El makefile realiza la query de manera normal (cargandola en formato texto en el log ) y después la vuelve a realizar pero la guarda en formato html y no la muestra en la terminal.

### Query 1:

Lista el número de reservas (bookings) que contengan un billete de ida y vuelta agrupadas por aeropuerto. Un billete es de ida y vuelta si el aeropuerto de origen del primer vuelo es idéntico al aeropuerto de destino del último vuelo. El resultado mostrará dos columnas con el código del aeropuerto de salida, así como el número de reservas que contienen billetes de ida y vuelta para ese aeropuerto. Igualmente, el resultado se ordenará de forma ascendente usando el atributo (departure airport).

No se ha hecho

## Query 2:

El precio de una reserva (booking.total amount) puede ser calculado a partir del precio de cada vuelo (ticket flights.amount). Crea una consulta que para cada reserva muestre el precio de la misma calculado a partir del valor almacenado en ticket flights.amount. La salida debe mostrar tres columnas conteniendo los atributos booking.book ref, booking.total amount y el valor calculado por vuestra consulta. Ordena la consulta de forma ascendente usando el atributo bookigs.book ref.

```
SELECT bookings.book_ref,
       bookings.total_amount,
       Sum(amounts.amount) AS "Coste calculado"
FROM   (SELECT tickets.book_ref,
               ticket_flights.amount
        FROM   tickets
              JOIN ticket_flights
                ON tickets.ticket_no = ticket_flights.ticket_no) AS amounts
JOIN   bookings
      ON bookings.book_ref = amounts.book_ref
GROUP BY bookings.book_ref
ORDER BY bookings.book_ref ASC;
```

Se ha implementado de manera anidada, el select del from permite obtener los atributos book\_ref y el coste de cada ticket de la unión de las tablas tickets y ticket\_flights. Esta tabla se une con bookings, donde se encuentra el coste total de la reserva. Todo esto se agrupa y ordena de forma ascendente por book\_ref. En el select principal se obtiene el book\_ref, el coste por reserva y la suma de los costes de cada ticket agrupados por reserva(book\_ref).

query-2: Muestra el book\_ref de la reserva, su precio y el precio calculado en base a la suma los precios de los tickets en la reserva en orden ascendente.

book_ref	total_amount	Coste calculado
00000F	265700.00	265700.00
000012	37900.00	37900.00
000068	18100.00	18100.00
000181	131800.00	131800.00
0002D8	23600.00	23600.00
0002DB	101500.00	101500.00
0002E0	89600.00	89600.00
0002F3	69600.00	69600.00
00034E	73300.00	73300.00
000352	109500.00	109500.00
000374	136200.00	136200.00
00044D	6000.00	6000.00

### Query 3:

Crea una consulta que muestre por pantalla el código de aeropuerto y el número de pasajeros recibidos, y ordena la misma de forma ascendente por el número de pasajeros recibidos. Se considera que un pasajero ha volado al aeropuerto si se ha emitido una tarjeta de embarque. Una tarjeta de embarque para un vuelo del aeropuerto X al Y incrementa en 1 el número de pasajeros recibidos en el aeropuerto Y.

```
SELECT f.arrival_airport AS "Código del Aeropuerto",  
       Count(bp.seat_no) AS "Número de Pasajeros Recibidos"  
FROM   flights AS f  
       JOIN boarding_passes AS bp  
         ON f.flight_id = bp.flight_id  
GROUP BY f.arrival_airport  
ORDER BY Count(bp.seat_no) ASC;
```

Se ha implementado haciendo una unión de flights y boarding\_passes, agrupándolo por el aeropuerto de llegada, y haciendo un count en cada grupo del número de asientos(seat\_no).

Esto se ha ordenado por la cuenta de los asientos de forma ascendente.

Al final muestra el código del aeropuerto de llegada y el número de asientos(llenos) de los aviones que llegan a dicho aeropuerto, lo cual equivale a decir el número de pasajeros recibidos.

```
e459630@6B-11-18-11:~/bases_de_datos-main$ make query3  
query-3: Obtiene el número de pasajeros recibidos totales por aeropuerto, lo muestra con el código del aeropuerto al que llegan y el número de pasajeros calculado en orden ascendente
```

Código del Aeropuerto	Número de Pasajeros Recibidos
SWT	8
RGK	51
CEE	80
NYA	82
USK	131
KXK	158
KLF	168
UCT	169
GDX	253
KGP	257
NFG	279
DYR	283
CNN	340
TBW	341
PEZ	356
GRV	364



## Query 4:

Vuelo con más asientos vacíos. Muestra en la salida el atributo flight id y el número de asientos vacíos. En caso de empate muestra todos los vuelos que hayan empatado.

```
SELECT f.flight_id AS "vuelo",
       Count(*) AS "número de asientos vacíos"
FROM   flights AS f
       JOIN seats AS s
         ON f.aircraft_code = s.aircraft_code
       LEFT JOIN boarding_passes AS bp
         ON f.flight_id = bp.flight_id
         AND s.seat_no = bp.seat_no
GROUP BY f.flight_id
HAVING Count(s.seat_no) - Count(bp.seat_no) = (SELECT Max(cs - cbp)
                                              FROM
                                              (SELECT Count(s.seat_no) AS cs,
                                                       Count(bp.seat_no) AS cbp
                                              FROM   flights AS f
                                                       JOIN seats AS s
                                                         ON f.aircraft_code =
                                                            s.aircraft_code
                                              LEFT JOIN boarding_passes
                                                         AS bp
                                                         ON f.flight_id =
                                                            bp.flight_id
                                                         AND
                                                            s.seat_no = bp.seat_no
                                              GROUP BY f.flight_id) AS sitios_vacios);
```

Se ha implementado haciendo un left join de manera que la tabla boarding\_passes no tenga la prioridad y por tanto en los casos que el seat\_no de esta tabla no pueda añadirse al join se pondrá null y al contarla contribuirá como 0 asientos. Al unir flights y seats se busca obtener los asientos que se pueden llenar de cada avión, al restarle a esto el número de asientos no nulos de boarding\_passes se están contando los asientos que no han sido ocupados en cada avión (ya que se agrupa por flight\_id).

En la parte inferior al Having se da la condición de que el número de asientos no ocupados de cada avión(Count(s.seat\_no)-Count(bp.seat\_no)) sea igual al máximo número de asientos vacíos contado(SELECT Max(cs-cbp) FROM) en el select de abajo(que hace exactamente lo mismo que el FROM y GROUP BY de arriba).

Se muestra con el select principal el flight\_id de cada vuelo y la cuenta de asientos vacíos de este. Mostrando solo los que tengan el máximo número de asientos vacíos por la condición del GROUP BY HAVING.

query-4: Muestra el número de asientos vacíos para cada vuelo representado por su flight_id	
vuelo	número de asientos vacíos
9924	402
5291	402
5257	402
9822	402
7722	402
7743	402
30587	402
256	402

## Query 5:

Muestra reservas para los cuales no se haya emitido alguna de las tarjetas de embarque. La salida debe mostrar los atributos book ref y flight id. Ordenar el resultado de forma ascendente por la pareja (book ref, flight id).

No se ha hecho

## Query 6:

Por definición dos vuelos (flights) realizan el mismo trayecto si su número de vuelo (flight no) es el mismo. Se solicita el trayecto donde la media de los retrasos acumulados sea mayor. Se define retraso como la diferencia entre los valores actual arrival - scheduled arrival. Mostrar en la línea de salida flight no y el retraso medio. En caso de empate deben aparecer todos los trayectos con mayor retraso.

```
SELECT flight_no,
       Avg(Extract(epoch FROM ( actual_arrival - scheduled_arrival ))) AS
       "retraso medio"
FROM   flights
GROUP BY flight_no
HAVING Avg(Extract(epoch FROM ( actual_arrival - scheduled_arrival ))) >= ALL (
  SELECT
    Avg(Extract(epoch FROM ( actual_arrival - scheduled_arrival )))
  FROM
    flights
  GROUP
    BY flight_no);
```

Se ha implementado haciendo uso de Extract epoch FROM con lo que se ha obtenido la diferencia actual arrival - scheduled arrival en segundos; a eso se le ha realizado un Avg para obtener la media acumulada.

Se han agrupado los valores de las medias en función de flight\_no y se muestran aquellas tuplas, de flight\_no y la media de sus retrasos, que tengan una media mayor o igual a los demás trayectos(flight\_no).

De esta manera se obtienen las que mayor media de retrasos tienen.

```
e459630@6B-11-18-11:~/bases_de_datos-main$ make query6
query-6: Calcula y muestra el o los trayectos con la mayor media de retrasos acumulados
, el trayecto lo muestra por su flight_no y muestra a su derecha el valor de la media
acumulada del trayecto

 flight_no |      retraso medio
-----+-----
 PG0450   | 6435.0000000000000000
(1 row)

query-6: The query-6 result has been successfully saved inside folder solutions in html
format and in the main folder as log
```