



UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Estructuras de Datos – Práctica 1

JORGE ÁLVAREZ FERNÁNDEZ

CRISTINA RODRÍGUEZ DE LOS RÍOS MEDINA



Índice

1. Introducción	3
2. Esquema BD	3
3. Diagrama Relacional	4
4. Consultas	5
5. Evidencias	9

Introducción

Durante esta primera práctica de Estructuras de Datos nos hemos enfrentado por primera vez al estudio en profundidad de una base de datos sencilla. Hemos trabajado a través de Ubuntu, que ofrece muchas facilidades a la hora de ordenar y compilar proyectos; además de contar con un comando *make* más intuitivo que Windows.

Con la guía disponible en Moodle hemos importado la base de datos *flights* a la herramienta de visualización DBeaver, y desde ahí hemos comenzado a escribir el código que nos permitirá cumplir con las consultas. Para ello nos apoyaremos en los ejemplos del libro de Ramez Elmasri y en las diapositivas de teoría. DBeaver nos permitirá además acceder con facilidad al esquema de la base de datos, mostrándonos los nombres de cada tabla con sus respectivas 'claves primarias'. También podremos consultar el diagrama relacional de la base, mostrado más adelante.

Esquema BD

```
nombreTabla(attrbA, attrbB, ...)  
otraTabla(attrb1, attrb2 → nombreTabla.attrA, attr3, ...)  
...
```

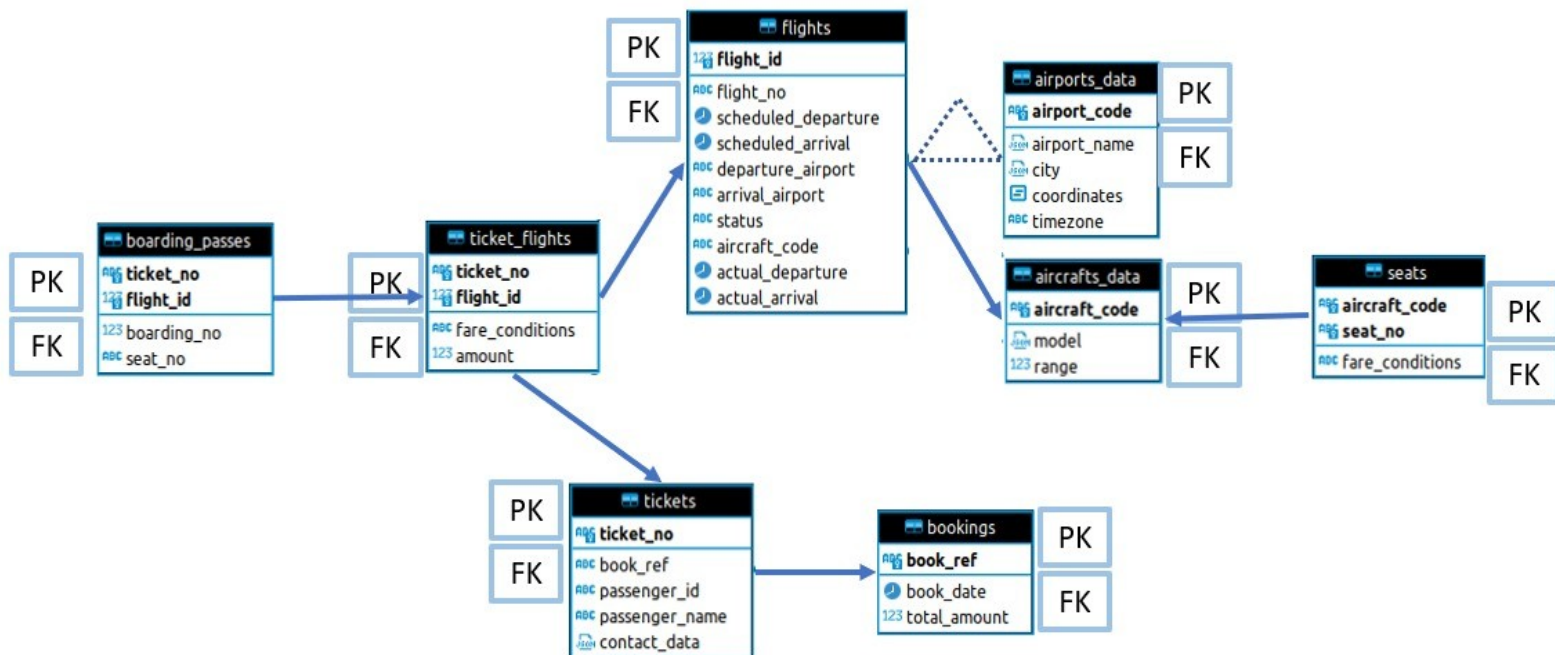
A continuacións se muestran las tablas con cada uno de sus apartados, destacando las claves primarias con **negrita**.

- boarding_passes (**ticket_no**, **flight_id**, boarding_no, seat_no)
- ticket_flights (**ticket_no**, **flight_id**, fare_conditions, amount)
- flights (**flight_id**, flight_no, schedule_departure, schedule_arrival, departure_airport, arrival_airport, status, aircraft_code, actual_departure, actual_arrival)
- airports_data (**airport_code**, airport_name, city, coordinates, timezone)
- aircrafts_data (**aircraft.code**, model, range)
- seats (**aircraft_code**, **seat_no**, fare_conditions)
- tickets (**ticket_no**, book_ref, passenger_id, passenger_name, contact_data)
- bookings (**book_ref**, book_date, total_amount)

A continuación, las tablas declarando las claves extranjeras que hacen referencia a otras tablas por su clave primaria:

- boarding_passes (**ticket_no** -> ticket_flights.ticket_no, **flight_id** -> tickets_flights.flight_id, boarding_no, seat_no)
- ticket_flights (**ticket_no** -> tickets.ticket_no, **flight_id** -> flights.flight_id, fare_conditions, amount)
- flights (**flight_id**, flight_no, schedule_departure, schedule_arrival, departure_airport -> airports_data.airport_code, arrival_airport -> airports_data.airport_code, status, aircraft_code -> aircrafts_data.aircraft_code, actual_departure, actual_arrival)
- airports_data (**airport_code**, airport_name, city, coordinates, timezone)
- aircrafts_data (**aircraft.code**, model, range)
- seats (**aircraft_code** -> aircraft_data.aircraft_code, **seat_no**, fare_conditions)
- tickets (**ticket_no**, book_ref -> bookings.book_ref, passenger_id, passenger_name, contact_data)
- bookings (**book_ref**, book_date, total_amount)

DIAGRAMA RELACIONAL



Consultas

- Query 1:

```
SELECT DISTINCT aux1.departure_airport,
                Count(*) AS tickets
FROM    (SELECT bookings.book_ref,
                flights.ticket_no,
                bookings.passenger_name,
                flights.arrival_airport,
                flights.departure_airport
        FROM    (SELECT bookings.book_ref,
                tickets.ticket_no,
                tickets.passenger_name
                FROM    bookings
                    natural JOIN tickets) AS bookings,
                (SELECT ticket_flights.ticket_no,
                flights.arrival_airport,
                flights.departure_airport
                FROM    ticket_flights
                    natural JOIN flights) AS flights
        WHERE   flights.ticket_no = bookings.ticket_no
        ORDER BY bookings.book_ref) AS aux1
JOIN (SELECT bookings.book_ref,
            flights.ticket_no,
            bookings.passenger_name,
            flights.arrival_airport,
            flights.departure_airport
    FROM    (SELECT bookings.book_ref,
                tickets.ticket_no,
                tickets.passenger_name
                FROM    bookings
                    natural JOIN tickets) AS bookings,
            (SELECT ticket_flights.ticket_no,
                flights.arrival_airport,
                flights.departure_airport
                FROM    ticket_flights)
```

```

                                natural JOIN flights) AS flights
        WHERE flights.ticket_no = bookings.ticket_no
        ORDER BY bookings.book_ref) AS aux2
    ON aux1.book_ref = aux2.book_ref
    AND aux1.departure_airport = aux2.arrival_airport
GROUP BY aux1.departure_airport
ORDER BY aux1.departure_airport ASC

```

- Query 2:

```

SELECT bookings.total_amount,
       bookings.book_ref,
       Sum(ticket_flights.amount) AS precio_booking
FROM   ticket_flights
       natural JOIN tickets
       natural JOIN bookings
GROUP BY bookings.book_ref
ORDER BY bookings.book_ref ASC

```

- Query 3:

```

SELECT airports_data.airport_code,
       Count(flights.arrival_airport) AS passengers
FROM   boarding_passes
       natural JOIN ticket_flights
       natural JOIN flights
       natural JOIN airports_data
WHERE  flights.arrival_airport = airports_data.airport_code
GROUP BY airports_data.airport_code
ORDER BY passengers DESC

```

- Query 4:

```

WITH empty_total
    AS (SELECT occupied.flight_id,
               total_seats.seats,
               occupied.purchased,
               ( total_seats.seats - occupied.purchased ) AS empty

```

```

FROM    (SELECT flights.flight_id,
              Count(*) AS purchased
        FROM    ticket_flights
              natural JOIN flights
        GROUP BY flights.flight_id
        ORDER BY Count(*) ASC) AS occupied,
        (SELECT flights.flight_id,
              Count(*) AS seats
        FROM    seats
              natural JOIN aircrafts_data
              natural JOIN flights
        GROUP BY flights.flight_id) AS total_seats
WHERE   occupied.flight_id = total_seats.flight_id)
SELECT occupied.flight_id,
       total_seats.seats,
       occupied.purchased,
       ( total_seats.seats - occupied.purchased ) AS empty
FROM   (SELECT flights.flight_id,
              Count(*) AS purchased
        FROM    ticket_flights
              natural JOIN flights
        GROUP BY flights.flight_id
        ORDER BY Count(*) ASC) AS occupied,
        (SELECT flights.flight_id,
              Count(*) AS seats
        FROM    seats
              natural JOIN aircrafts_data
              natural JOIN flights
        GROUP BY flights.flight_id) AS total_seats
WHERE  occupied.flight_id = total_seats.flight_id
AND    total_seats.seats - occupied.purchased = (SELECT( Max(empty) )
                                                FROM    empty_total)

```

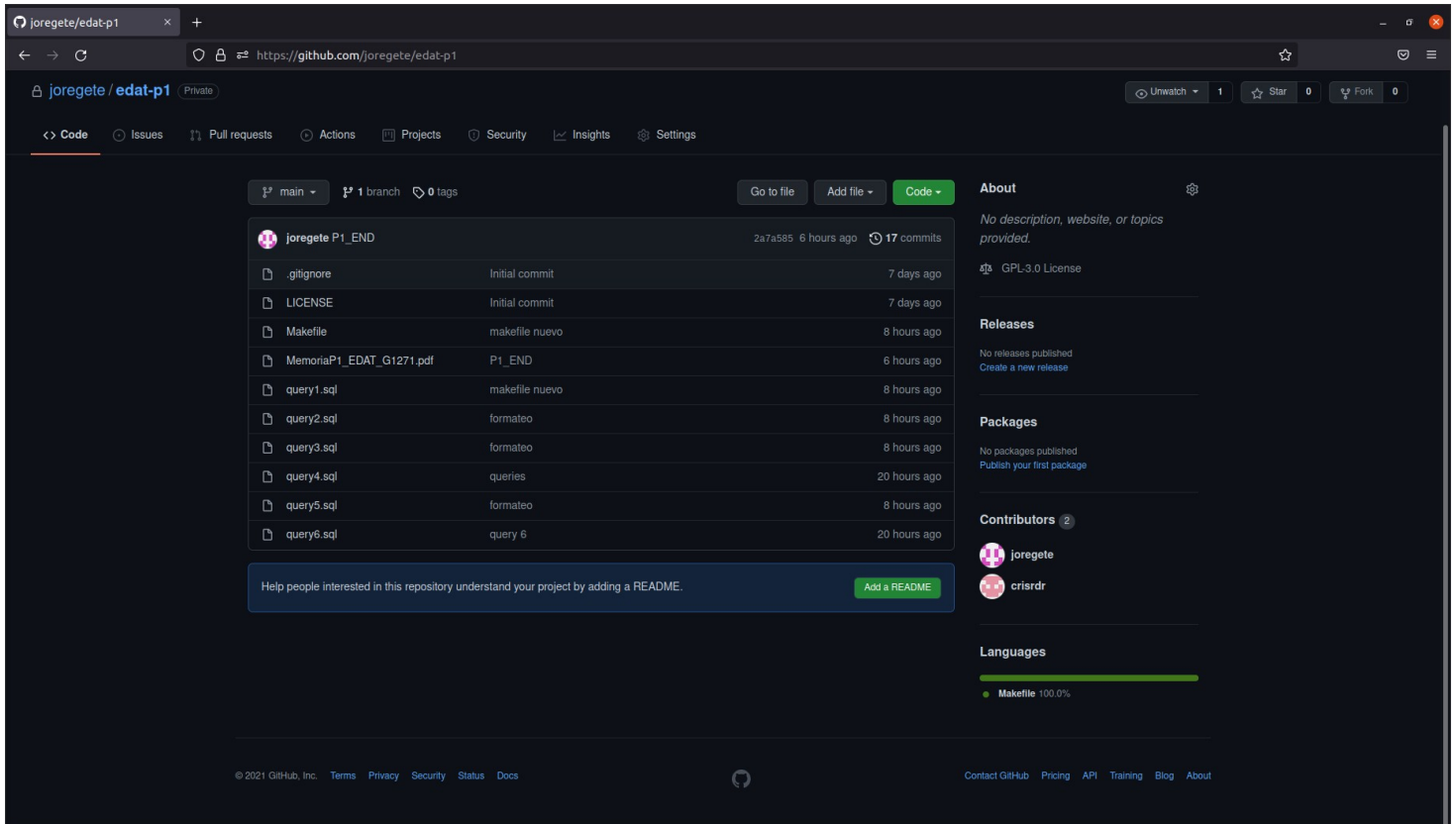
-Query 5:

-- yet to be implemented

-Query 6:

```
WITH delay
  AS (SELECT flights.flight_no,
            Avg(flights.actual_arrival - flights.scheduled_arrival) AS
            delay_avg
      FROM flights
      GROUP BY flights.flight_no
      ORDER BY delay_avg DESC)
SELECT flights.flight_no,
       Avg(flights.actual_arrival - flights.scheduled_arrival) AS delay_avg
FROM   flights
       natural JOIN delay
WHERE  delay_avg = (SELECT ( Max(delay_avg) )
                   FROM   delay)
GROUP BY flights.flight_no
ORDER BY delay_avg DESC
```


Evidencias



Arriba: repositorio de la entrega, con 17 commits realizados por ambos miembros del grupo. Para más información sobre los commits se puede ejecutar *git log* en el directorio *tmpDir*.

flights 1

SELECT DISTINCT

	departure_airport	tickets
1	AAQ	15,174
2	ABA	3,096
3	AER	69,068
4	ARH	8,507
5	ASF	4,252
6	BAX	7,069
7	BQS	3,421
8	BTK	4,463
9	BZK	45,858
10	CEE	530
11	CEK	17,785
12	CNN	1,227
13	CSY	21,015
14	DME	237,869
15	DYR	1,205
16	EGO	30,821
17	ESL	21,619
18	GDX	627
19	GDZ	2,762
20	GOJ	23,164
21	GRV	978
22	HMA	15,975
23	HTA	2,570
24	IJK	1,227
25	IKT	22,015
26	KEJ	4,524
27	KGD	13,038
28	KGP	705
29	KHV	55,980
30	KJA	16,360

Save Cancel 94 row(s) fetched - 10.329s (+1ms)

Arriba: Output de la primera query.

bookings 1

SELECT bookings. Enter a SQL exp

	total_amount	book_ref	precio_booking
2	37,900	000012	37,900
3	18,100	000068	18,100
4	131,800	000181	131,800
5	23,600	0002D8	23,600
6	101,500	0002DB	101,500
7	89,600	0002E0	89,600
8	69,600	0002F3	69,600
9	73,300	00034E	73,300
10	109,500	000352	109,500
11	136,200	000374	136,200
12	6,000	00044D	6,000
13	140,100	00044E	140,100
14	12,000	0004B0	12,000
15	139,300	0004E1	139,300
16	26,700	000511	26,700
17	6,000	00053F	6,000
18	50,700	00054E	50,700
19	28,800	0005E7	28,800
20	95,400	0005F4	95,400
21	56,300	0005FF	56,300
22	296,700	00067B	296,700
23	59,400	0006C3	59,400
24	80,200	0006F5	80,200
25	51,900	000735	51,900
26	32,800	000769	32,800
27	28,800	000784	28,800
28	43,200	0007A9	43,200
29	137,700	0007ED	137,700
30	75,600	0007FC	75,600
31	23,400	000836	23,400

Save Cancel 200 row(s) fetched - 3.439s (+4ms)

Arriba: Output de la segunda query.

airports_data 1

SELECT airports_c

	airport_code	passengers
1	SVO	77,909
2	DME	76,137
3	LED	35,867
4	VKO	29,701
5	OVV	27,477
6	SVX	20,628
7	AER	19,182
8	PEE	18,122
9	KHV	12,956
10	KZN	10,474
11	ULV	10,153
12	KVX	9,545
13	BZK	9,316
14	VOG	8,526
15	ROV	8,412
16	KRR	8,382
17	UFA	7,852
18	KUF	7,632
19	IKT	7,518
20	SCW	7,285
21	MRV	7,015
22	OVS	6,916
23	MCX	6,632
24	TJM	6,374
25	EGO	6,071
26	CSY	6,021
27	AAQ	5,990
28	CEK	5,847
29	KJA	5,806
30	KGD	5,470

Save Cancel 94 row(s) fetched - 1.215s

Arriba: Output de la tercera query.

WITH empty_tota

	flight_id	seats	purchased	empty
1	3,494	402	2	400

Save Cancel 1 row(s) fetched - 2.275s

Arriba: Output de la cuarta query.

flights 1

WITH delay AS (SE

	flight_no	delay_avg
1	PG0450	01:47:15

Save Cancel 1 row(s) fetched - 68ms

Arriba: Output de la sexta query.