"BASES DE DATOS Y SISTEMAS DE INFORMACIÓN" EXAM 27/11/2017

UNITS 2 AND 3

Consider the following relational schema about a travel agency:

COUNTRY (count_cod: char(10), name: char(15), info: char(150))
PK:{count_cod}
NNV:{name}

CITY (city_cod: char(10), name:char(15), info: char(150), count_cod: char(10))
PK:{city_cod}
FK:{count_cod}→COUNTRY
NNV:{name}

TOUR (tour_cod: char(5), description: char(50), dep_date: date, arrival_date: date, price: number, capacity: number)
PK:{tour_cod}
NNV:{description, dep_date, price}

HOTEL (hotel_cod: char(10), name: char(20), category: number, city_cod: char(10))
PK:{hotel_cod}
FK:{city_cod}→CITY
NNV:{name, category, city_cod}
IC: category in (1, 2, 3, 4, 5)

VISIT(tour_cod: char(5), city_cod: char(10), arrival_date: date, dep_date: date, hotel_cod: char(10))
PK:{tour_cod, city_cod}
FK:{city_cod}→CITY
FK:{tour_cod}→TOUR
FK:{hotel_cod}→HOTEL
NNV:{arrival_date, dep_date}

CLIENT (SSN: char(10), name: char(50), age: number, address: char(30), city_cod: char(10))
PK:{SSN}
FK:{city_cod}→CITY
NNV:{name, age, address, city_cod}

PARTICIPATE (tour_cod: char(5), client_SSN: char(10))
PK: {tour_cod, client_SSN}
FK:{tour_cod}→TOUR
FK:{client_SSN}→CLIENT(SSN)

Where the meaning of the relations is the following:

- **Country**:
  - *count_cod*: Country code
  - *name*: Country name
  - *info*: Description of the country
- **City**:
  - *city_cod*: City code
  - *name*: City name
  - *info*: Description of the city
  - *count_cod*: Code of the country where the city is
- **Tour**:
  - *tour_cod:* Tour code
  - *description*: Description of the tour
  - *dep_date*: Departure date of the tour
  - *arrival_date*: Scheduled arrival date of the tour
  - *price*: Price of the tour
  - *capacity:* Maximum number of clients that can participate in the tour
- **Hotel**
  - *hotel_cod*: Hotel code
  - *name*: Name of the hotel
  - *category*: Category of the hotel
  - *city_cod*: Code of the city where the hotel is
- **Visit**: The tour ***tour_cod*** visits the city ***city_cod.*** The visit is starting on ***dep_date*** and it is ending on ***arrival_dat.*** The clients will stay at the hotel ***hotel_cod*.**
- **Client:**
  - *SSN*: SSN of the client
  - *name*: Name of the client
  - *age*: Age of the client
  - *address*: Client address
  - *city_cod*: Code of the city where the client lives
- **Participate**: The client with SSN ***client_SSN*** will participate in the tour ***tour_cod***

1) Describe what is a journal file and the information that it contains. Describe the database recovery process when a secondary memory failure appears. (0.5 points)

The journal file is a file used to recover the database when a main or secondary memory failure appears. In the journal file the DBMS saves all operations included in the transactions.

In order to recover the database when a secondary memory failures appears, the DBMS restores the database using the most recent backup and then, all the transactions which appears confirmed in the journal file since the backup moment will be redone. Note that it is not necessary to undo transactions that are cancelled in the journal file.

2) Write the following SQL queries:

a) List the description of the tours with a price lower than 500€, containing the word 'luxury' in its description, and with no scheduled arrival date. (0.5 points)

```
SELECT description
FROM Tour
WHERE price < 500 AND description LIKE '%luxury%' AND arrival_date IS NULL
```

b) List the SSN, name, and age of the youngest client living in the city called 'València' (0.5 points)

```
SELECT SSN, C.name, age
FROM Client C, City Ci
WHERE C.city_cod = Ci.city_cod
   AND Ci.name = 'València'
    AND age = (SELECT MIN(age)
                    FROM Client C, City Ci
                    WHERE C.city_cod = Ci.city_cod AND Ci.name = 'València')
```

c) List the description, the price, and the number of visited cities of the tour/s that visit most cities (tour/s visiting the greatest number of cities). (0.75 points)

```
SELECT T.description, price, COUNT(*)
FROM Tour T, Visit V
WHERE T.tour_cod = V.tour_cod
GROUP BY T.tour_cod, T.description, T.price
HAVING COUNT(*) = (SELECT MAX(COUNT(*)) FROM Visit GROUP BY tour_cod)
```

d) List, for all the cities in the database, its name, number of hotels in the city, and the average category of the hotels in that city. (0.75 points)

```
SELECT C.name, COUNT(H.hotel_cod), AVG(H.category)
FROM City C LEFT JOIN Hotel H ON C.city_cod = H.city_cod
GROUP BY C.city_cod, C.name

  --- ALTERNATIVE -------

(SELECT C.name, COUNT(H.hotel_cod), AVG(H.category)
FROM City C, Hotel H
WHERE C.city_cod = H.city_cod
GROUP BY C.city_cod, C.name)
UNION
(SELECT C.name, 0, NULL
FROM City C
WHERE C.city_cod NOT IN (SELECT city_cod FROM Hotel)
```

e) List the SSN and name of the clients such that all the tours in which the client has participated have visited the city with code 'AJR1'. (1 point)

```
SELECT SSN, name
FROM Client C
WHERE NOT EXISTS (SELECT *
                  FROM Participate P
                   WHERE P.client_SSN = C.SSN
                     AND NOT EXISTS (SELECT * Visit Vi
                                     WHERE Vi.tour_cod = P.tour_cod
                                     AND Vi.city_cod = 'AJR1'))
  AND EXISTS (SELECT *
           FROM Participate P
           WHERE P.client_SSN = C.SSN)

--- ALTERNATIVE -------

SELECT SSN, client
FROM Client C
WHERE (SELECT COUNT(*)
       FROM Participate P
       WHERE P.client_SSN = C.SSN )
       =
       (SELECT COUNT(*)
       FROM Participate P, Visit Vi
       WHERE P.client_SSN = C.SSN
          AND Vi.tour_cod = P.tour_cod
          AND Vi.city_cod =  'ARJ1'))
AND (SELECT COUNT(*)
     FROM Participate P
     WHERE P.client_SSN = C.SSN )  > 0
```