"BASES DE DATOS Y SISTEMAS DE INFORMACIÓN" EXAM 23/01/2017

UNITS 2 AND 3

Consider the following relational schema about the "Escola d'Estiu" (Summer School) for children, where children are assigned to groups based on their age:

GROUP(group_cod: d_gro, max: d_max, color : d_color)
PK: {group_cod}

CHILD(child_code: d_chil, name: d_nam, age: d_age, group_cod : d_gro)
PK: {child_code}
FK: {group_cod} →GROUP
NNV: {group_cod}

INSTRUCTOR(ssn: d_ssn, name: d_nam, age: d_age, speciality : d_sp)
PK: {ssn}

ACTIVITY(act_code: d_act, objectives: d_obj, resp: d_ssn, length: d_len)
PK: {act_code}
FK: {resp} → INSTRUCTOR(ssn)
NNV: {resp}

PARTICIPATE (group_cod: d_gru, act_code: d_act, date: d_date)
PK: {group_cod, act_code}
FK: {group_cod} → GROUP
FK: {act_code} →ACTIVITY
NNV: {date}

ASSIGNED (ssn: d_ssn, group_cod: d_gro)
PK: {ssn, group_cod}
FK: {group_cod} → GROUP
FK: {ssn} →INSTRUCTOR

RI: Every group has at least one assigned instructor who is responsible of some activity.

==========

Where the meaning of the relations is the following:

- **GROUP:**
  - *group_cod*: group identifier
  - *color*: of the shirt of the group
  - *max*: Maximun number of children in the group
- **CHILD**:
  - *child_code*: child's identifier
  - *age*: child's age
  - *name*: child's name
  - *group_cod*: child's group id
- **INSTRUCTOR**:
  - *ssn*: instructor's SSN
  - *age*: instructor's age
  - *name*: instructor's name
  - speciality: instructor's speciality
- **ACTIVITY**:
  - *act_code:* activity code
  - *resp*: SSN of the responsible instructor of the activity
  - *objectives*: objectives of the activity
  - *length*: length of the activity

- **PARTICIPATE**: The group identified by *group_cod* is going to participate in the activity *act_code* the day *date***.**
- **ASSIGNED**: The instructor with SSN *ssn* is assigned to the group identified by *group_cod***.**

1) Define briefly the properties of a transaction. (0.6 points)

2) Write the following queries in SQL:

   a) List the SSN, name, age, and speciality of the instructors who are not assigned to any group and are not responsible of any activity. (0.6 points)

   b) List the code, objectives and name of the responsible person, of those activities with more than one group participating the same day. (0.6 points)

   c) List for all the groups in the database, the code, color of its shirt, the number of activities in which the group is participating, and the number of instructors assignned to the group. (0.6 points)

   d) List the code, objectives, and length of the activities with more than one group in which the responsible instructor of the activity is more than 15 years older than the oldest child participating in that activity. (0.8 points)

   e) List the code and shirt color of the groups that have participated in all the activities whose responsible instructor is one of the instructors assigned to that group. (0.8 points)

1.-

- **Atomicity:** A transaction is an indivisible unit that is either performed in its entirety or is not performed at all ("All or nothing").
- **Consistency:** the transaction must transform the DB from one consistent state to another consistent state (all integrity constraints must be met).
- **Isolation:** Transactions execute independently of one another: All the modifications introduced by a non-confirmed transaction are not visible to other transactions.
- **Durability:** The effects of a successfully completed (committed) transaction are permanently recorded in the DB and must not be lost because of a subsequent system or other transaction failure.

2 a)

```
SELECT *
FROM instructor M
WHERE  NOT EXISTS (SELECT * FROM activity A WHERE A.resp = M.ssn) AND
       NOT EXISTS (SELECT * FROM assigned S WHERE S.ssn = M.ssn);
--alternative
SELECT *
FROM instructor M
WHERE  M.ssn NOT IN  (SELECT A.resp FROM activity A) AND
       M.ssn NOT IN  (SELECT S.ssn FROM assigned S);
--alternative
SELECT *
FROM instructor M
WHERE M.ssn NOT IN  (SELECT A.resp FROM activity A
                            UNION
                     SELECT S.ssn FROM assigned S);
```

2 b)

```
SELECT a.act_code,a.objectives,m.name
FROM activity a, instructor m
WHERE a.resp = m.ssn AND
    EXISTS (SELECT *
            FROM participate p1, participate p2
            WHERE  p1.act_code = a.act_code AND P2.act_code = a.act_code AND
                   p1.date  = p2.date AND p1.group_cod <> p2.group_cod);
--alternative
SELECT a.act_code,a.objectives,m.name
FROM activity a, instructor m
WHERE  a.resp = m.ssn AND
       a.act_code IN (SELECT p.act_code
```

```
                    FROM participate p
                    GROUP BY p.act_code, p.date
                    HAVING COUNT(*)>1);
2 c)
SELECT g.group_cod, g.color,
       COUNT(distinct p.act_code),COUNT(distinct s.ssn)
FROM group g LEFT JOIN participate p ON g.group_cod = p.group_cod
                    LEFT JOIN assigned s ON g.group_cod = s.group_cod
GROUP BY g.group_cod, g.color;

2 d)
SELECT a.act_code,a.objectives,a.length
FROM activity a
WHERE (SELECT COUNT(*) FROM participate p WHERE p.act_code=a.act_code)>1 AND
       (SELECT max(age)+15
       FROM participate p1, child c
       WHERE  p1.act_code=a.act_code AND
              p1.group_cod=c.group_cod)<(SELECT age
                                         FROM instructor m WHERE m.ssn=a.resp);
--alternative
SELECT a.act_code,a.objectives,a.length
FROM activity a, instructor m
WHERE  a.resp=m.ssn AND
       a.act_code IN (SELECT p.act_code
                      FROM participate p
                      GROUP BY p.act_code
                      HAVING COUNT(*)>1) AND
       (SELECT max(age)+15
       FROM participate p1, child c
       WHERE  p1.act_code=a.act_code AND
              p1.group_cod=c.group_cod)< m.age;

2 e)
SELECT g.group_cod, g.color
FROM group g
WHERE NOT EXISTS(SELECT * FROM activity a
                 WHERE a.resp IN (SELECT a1.ssn FROM assigned a1
                                  WHERE a1.group_cod = g.group_cod)
                  AND NOT EXISTS (SELECT * FROM participate p
                                  WHERE p.group_cod= g.group_cod
                                    AND p.act_code=a.act_code));
```

```
--alternative

SELECT g.group_code, g.color
FROM group g
WHERE NOT EXISTS(SELECT * FROM activity a, assigned a1
                 WHERE a.resp=a1.ssn and a1.group_code = g.group_code AND
                       NOT EXISTS (SELECT * FROM participate p
                                   WHERE p.group_code= g.group_code AND
                                         p.act_code=a.act_code));
--alternative

SELECT g.group_code, g.color
FROM group g
WHERE (SELECT COUNT(*) FROM activity a
       WHERE a.resp IN (SELECT a1.ssn FROM assigned a1
                                   WHERE a1.group_code = g.group_code))
         =
       (SELECT COUNT(*) FROM activity a
        WHERE a.resp IN (SELECT a1.ssn FROM assigned a1
                                   WHERE a1.group_code = g.group_code)
           AND a.act_code IN (SELECT p.act_code FROM participate p
                              WHERE p.group_code= g.group_code));
```