Test Pearl 010 — Databases
(201300070) Pearls of Computer Science
(Answer Key)

17 September 2015, 08:45-09:45

- You are allowed to use one A4 with notes. Calculators, laptops, mobile phones and similar devices are not allowed ***Put them in your bag!***
- The amount of achievable points are noted for each exercise.

☞ **1** *SQL (20 points)* For this question we will use the tables belonging to a 'music' database. A description of the table layout can be found in Figure 2. The database maintains records of songs, which are composed by artists, and performed by (possibly different) artists on albums.

To support covers, live performances, and different performances of the same classical piece, this table layout was chosen with a separate table "performed_by". In Figure 3 you can find an example that displays this scenario. Take the artist "Led Zeppelin" (artist_id=1). In addition to writing the song "Stairway to Heaven" (song_id=10), they performed it on the album "Led Zeppeling [Disc 2]" (album_id=20). And this song is also on the album "Top 2000 editie 2010" (album_id=21).
It is also visible in the example data that the song "Sing Our Own Song" by "UB 40" is covered by an unknown artist (artist_id=*NULL*) on the album "A Tribute to UB 40".

(a) Write an SQL-query that supplies the names of all songs composed by the artist 'Led Zeppelin'.
(b) Write an SQL-query that, for the artist 'Led Zeppelin', shows the names of all songs performed and the title and year of the album which the song belongs to.
(c) Write an SQL-query that, for every song, shows the id and name of the song, and the amount of performances.
(d) Write an SQL-query that, for the artist 'UB 40', shows all albums (id and title) that contain songs composed by them, including the amount of performances (of songs composed by them).
(e) There are concerns for the data integrity, we want to know if there exist songs with a track_nr that exceeds the nr_of_tracks of the album. Write an SQL-query that finds these songs and shows the id and name of the song, in addition to the name and id belonging to the album and the track_nr and nr_of_tracks.
(f) The artist that performed "Sing Our Own Song" by UB 40 on the album "A Tribute to UB 40" is still unknown in the database (artist_id is still *NULL*). Write an SQL-statement or the SQL-statements that register that the artist 'The Pop Hit Crew' performed this song. Make up the unique numbers for ids if you need them. You are allowed to use the correct ids in the statements (you don't have to use subqueries).

□

Answer to 1.

(a) Simple join-query:

```
SELECT song.name
FROM artist, song
WHERE artist_id = composed_by
  AND artist.name = "Led Zeppelin"
```

(b) Now the join happens in the table performed_by:

```
SELECT song.name, title, year
FROM artist, song, album, performed_by
WHERE performed_by.artist_id = artist.artist_id
  AND performed_by.song_id = song.song_id
  AND performed_by.album_id = album.album_id
  AND artist.name = "Led Zeppelin"
```

(c) A default GroupBy-query

```
SELECT song.song_id, song.name, COUNT(*)
FROM song, performed_by
WHERE performed_by.song_id = song.song_id
GROUP BY song.song_id, song.name
```

(d) This is a complicated GroupBy-query, due to the fact that the join should work with song.composed_by instead of performed_by.artist_id.

```
SELECT album.album_id, album.title, COUNT(*)
FROM artist, song, album, performed_by
WHERE performed_by.song_id = song.song_id
  AND performed_by.album_id = album.album_id
  AND song.composed_by = artist.artist_id
  AND artist.name = "UB˜40"
GROUP BY album.album_id, album.title
```

(e) This is a default join query:

```
SELECT song.song_id, song.name, album.album_id, album.name,
    song.track_nr, album.mr_of_tracks
FROM song, album, performed_by
WHERE performed_by.song_id = song.song_id
  AND performed_by.album_id = album.album_id
  AND track_nr > nr_of_tracks
```

(f) Create a new artist before updating the specific row.

```
INSERT INTO artist(artist_id,name) VALUES (3,"The Pop Hit Crew");
UPDATE performed_by
SET artist_id=3
WHERE song_id=11 AND album_id=22
```

☞ **2** *Databases (10 points)*

(a) Imagine that you are developing an application that sends SQL-queries to a database server. One of the queries is to slow for the application to behave according to the specification. Who needs to solve this, which *role* does the person have that needs to speed up the query? Explain your answer.
(b) Explain clearly why you would define an *external schema*. Explain your answer.
(c) Explain what a *foreign key* is.
(d) One of the *ACID*-properties is *Consistency*. What does a dbms do when it is asked to execute a query, whose result violates the integrity rules, i.e. which endangers the consistency of the data. Explain your answer.

□

Answer to 2.

(a) The *database administrator*. He can tune things. SQL is declarative, which means that a different formulation of the query wont help. This means that the application programmer can't do anything. The databasedesigner is worth half the points (a different database design can speed up queries).
(b) Security (you can give users explicit access to a limited scheme). Ease of use (schemes can be built to be simpler and specific for use).
(c) One or more attributes in a table that reference the key of a different table, i.e. exactly a row in a different table.
(d) It will rollback the transaction, i.e. it will be as if it never happened.

☞ **3** *Databaseontwerp (10 points)*

Figure 1 contains part of the datamodel belonging to the registration of external internship and graduation assignments. External, meaning at an external organisation like a business. The external assignments have a title, description, main supervisor and they belong to a Company. Companies have a name and address. Each company has a contact. There are two kinds of external assignments: internships (with differing durations) and graduation assignments. The latter has multiple supervisors. supervisors have a name, email-address and a phone number.

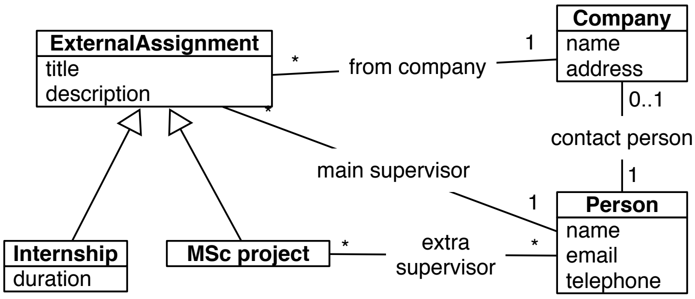

Figure 1: ER-model containing part of the data model belonging to the registration of external internship and graduation assignments.

(a) At the relation 'contact' you see the notation "0..1", meaning "0 or 1". Why is this important? Why doesn't it say "1"? Explain your answer.
(b) Supply a table structure for the ER-model in Figure 1. Do this using a list of tables and attribute names. The attribute types may be omitted.

□

Answer to 3.

(a) Contacts are not the only persons, supervisors are people too. And supervisors are not contacts for a company. So a person is a contact for 0 or 1 companies, 0 if he is not and 1 if he is (0 if he or she is a supervisor; and 1 if he or she is a contact).

(b) A possible solution is displayed below. Variants with separate internship and graduation assignment tables are allowed as well.

Table	**Attributes**
External Assignment	eoid, title, description, kind, duration, from_company, main_supervisor
Company	cid, name, address, contact_person
Person	pid, name, email, telephone
ExtraSupervisor	eoid, pid

Table	Attributes / Description
artist	**artist_id**, name
	For every artist a unique number (artist_id; also primary key) and the name of the artist.
song	**song_id**, name, composed_by
	For every song a unique number (song_id; also primary key), the name and composer of the song (composed_by contains an artist_id).
album	**album_id**, title, year, nr_of_tracks
	For every album a unique number (album_id; also primary key), the title, release year and amount of tracks on the album.
performed_by	**artist_id, song_id, album_id**, track_nr, duration.
	A song (song_id) is performed by an artist (artist_id), not necessarily the composer. The performance belongs to an album (album_id). This way a song can have multiple performances by multiple artists (for example, covers or performances by orchestras of the same piece of music). Besides the primary key(artist_id, song_id, album_id), the (track_nr) and duration of the performance is stored.

Figure 2: Description of the table structure belonging to "music"-database (Primary keys in **bold**).

artist

artist_id	name
1	Led Zeppelin
2	UB 40

album

album_id	title	year	nr_of_tracks
20	Led Zeppelin [Disc 2]	1990	15
21	Top 2000 editie 2010	2010	2000
22	A Tribute to UB 40	1993	14
23	Rat in the Kitchen	1986	9

song

song_id	name	composed_by
10	Stairway to Heaven	1
11	Sing Our Own Song	2

performed_by

artist_id	song_id	album_id	track_nr	duration
1	10	20	15	8:00
1	10	21	5	8:01
2	11	23	9	7:21
2	11	21	1957	4:04
NULL	11	22	8	2:54

Figure 3: Example data for the "music"-database

In the informal syntax, we use the following notations

- $A|B$ to indicate a choice between A and B
- $[A]$ to indicate that A is optional
- $A*$ to indicate that A appears 0 or more times
- $A+$ to indicate that A appears 1 or more times
- 'A' to indicate that the symbol A is literally that symbol

We are not precise in punctuation in the syntax, but this is irrelevant in this exam anyway.

SQL

```
createtable : CREATE TABLE tablename '(' columndef+ constraint* ')'
columndef: colname type [NOT NULL] [UNIQUE] [PRIMARY KEY] [REFERENCES tablename (colname+)]
constraint : PRIMARY KEY (colname, ... ) | CHECK ( condition )
             | FOREIGN KEY(colname, ... ) REFERENCES tablename(colname, ... )
query : SELECT ( column [ AS colname ] )+
        FROM ( tablename [ AS colname ] )+
        WHERE condition [ GROUP BY column+ ] [ ORDER BY column+ ]
column : [ tablename '.' ] colname | '*'
Examples of condition: column = value [ (OR | AND) [NOT] column <> value ]
                       | column IS [NOT] NULL
                       | column [NOT] IN (value, ... ) ...
```

Figure 4: Informal syntax of SQL