Introduction to Databases

Homework no. 2: SQL

1. The following relations are given (primary keys are underlined):

```
AUTHOR(AuthorCode, Name, Surname, Department, University)
ARTICLE(ArticleCode, Title, Topic)
AUTHORS_OF_ARTICLE(ArticleCode, AuthorCode)
EDITIONS_OF_CONFERENCE(Conference, Edition, EditionName, StartDate, EndDate, Editor)
AUTHOR_PRESENTS_ARTICLE(AuthorCode, Date, StartTime, EndTime, Room,
                        ArticleCode, Conference, Edition)
```

   Write the following query in SQL

   (a) For the authors who have *exclusively* presented articles with topic 'Data Mining', show the code of the author, the surname of the author, her/his university, and the total number of articles presented by the author in each edition of every conference.

2. The following relations are given (primary keys are underlined):

```
STUDENT(StudentID, Name, Surname, DegreeProgramme)
ASSIGNMENT_TO_BE_DELIVERED(ACode, Title, Topic, ScheduledExpirationDate)
TEACHER(TeacherID, Name, Surname, Department)
EVALUATION_OF_DELIVERED_ASSIGNMENT(StudentID, ACode, TeacherID,
                                   DeliveryDate, EvaluationDate, Score)
```

   Write the following query in SQL

   (a) For each student who has delivered at least 3 assignments with score greater than 4, show the surname of the student, the total number of assignments delivered by the student, the average score of all delivered assignments, and the number of different teachers who evaluated their delivered assignments.

3. The following relations are given (primary keys are underlined):

```
AUTHOR(AuthorCode, Name, Surname, Department, University)
ARTICLE(ArticleCode, Title, Topic)
AUTHORS_OF_ARTICLE(ArticleCode, AuthorCode)
EDITIONS_OF_CONFERENCE(Conference, Edition, EditionName, StartDate, EndDate, Editor)
AUTHOR_PRESENTS_ARTICLE(AuthorCode, Date, StartTime, EndTime, Room,
                        ArticleCode, Conference, Edition)
```

   Write the following queries in SQL

   (a) Considering the conferences with at least 10 editions, for each edition of the conference show the name of the edition and the code of the author who presented the highest number of articles in the edition.

4. The following relations are given (primary keys are underlined):

```
SEMINAR(SCode, STitle, Topic, Duration)
SPEAKER(S-SSN, SName, BirthDate)
SEMINAR-CALENDAR(SCode, Date, StartTime, S-SSN, Room)
EXPERTISE(S-SSN, Topic)
```

Write the following query in SQL

(a) Show the code of the seminars for which at least one scheduled presentation was held by the speaker with the highest number of topics of expertise.

5. The following relations are given (primary keys are underlined):

```
TEACHER(TCode, TName, TSurname, Department, ResearchGroupName, ResearchArea)
COURSE(CCode, CName, EnrollingStudent#, TCode, Topic)
CLASSROOM(RoomID, Floor#, Video_Kit, Seat#)
LECTURE(RoomID, Date, StartHour, EndHour, CCode, AttendingStudent#)
Video_Kit = {yes, no}
```

Write the following query in SQL

(a) For each teacher who has taught *exclusively* courses whose topic is databases, select the code of the teacher and, among her courses, the code of the course for which the average number of students attending the course lectures is the highest.

6. The following relations are given (primary keys are underlined):

```
STUDENT(StudentID, Name, Surname, DegreeProgramme)
ASSIGNMENT_TO_BE_DELIVERED(ACode, Title, Topic, ScheduledExpirationDate)
TEACHER(TeacherID, Name, Surname, Department)
EVALUATION_OF_DELIVERED_ASSIGNMENT(StudentID, ACode, TeacherID,
                                   DeliveryDate, EvaluationDate, Score)
```

Write the following query in SQL

(a) Show the identifier, surname and degree programme of the students who have *never* delivered an assignment after the scheduled expiration date, and who have delivered *all* the assignments due always getting the highest score.

Ex. no. 1

```
SELECT A.AuthorCode, Surname, University, COUNT(*)
FROM AUTHOR A, AUTROR_PRESENTS_ARTICLE APA
WHERE A.AuthorCode NOT IN
                (SELECT AuthorCode
                 FROM AUTHOR_PRESENT_ARTICLE AP1,
                      ARTICLE AR
                 WHERE AR.ArticleCode = AP1.
                       ArticleCode
                 AND Topic <> 'Data Mining' )
      AND APA.AuthorCode = A.AuthorCode
GROUP BY A.AuthorCode, Edition, Conference
         Surname, University;
```

Exercise no. 2

```
SELECT Surname, COUNT(*), AVG(Score),
                          COUNT(DISTINCT TeacherId)
FROM STUDENT S, EVALUATION_OF_ASSIGNMENT_TO... EA1
WHERE StudentId IN (SELECT StudentId
                    FROM EVALUATION_OF_DELIVERED_ASS EA
                    WHERE Score > 4
                    GROUP BY StudentId
                    HAVING COUNT(*) >= 3)
      AND S.StudentId = EA1.StudentId
GROUP BY S.StudentId, Surname
```

Exercise no. 3

```
SELECT EditionName, APA.AuthorCode
FROM AUTHOR_PRESENTS_ARTICLE APA, EDITION_OF_
                                  CONFERENCE E
WHERE Conference IN (SELECT Conference
                     FROM EDITIONS_OF_
                          CONFERENCE EC
                     GROUP BY Conference
                     HAVING COUNT(*) >= 10)
      AND E.Edition = APA.Edition AND E.Conference = APA.Conf.
GROUP BY APA.AuthorCode, APA.Edition, APA.Conference
         EditionName
HAVING COUNT(*) =

                    ||

                    (SELECT MAX(TotPA)
                     FROM (SELECT AuthorCode, Edition
                                  Conference,
                                  COUNT(*) AS TotPA
                           FROM AUTHOR_PRESENTS_AR
                                AA
                           GROUP BY AuthorCode,
                                    Edition, Conference,
                                    AS TFA
                     WHERE TFA.Edition = APA.Edition
                           AND TFA.Conference = APA.Conf
```

Exercise no. 4

```
SELECT DISTINCT SCode
FROM SEMINAR-CALENDAR
WHERE S-SSN IN
    (SELECT S-SSN
     FROM EXPERTISE
     GROUP BY S-SSN
     HAVING COUNT(*) =
         (SELECT MAX(TotExp)
          FROM (SELECT S-SSN, COUNT(*)
                       AS TotExp
                FROM EXPERTISE
                GROUP BY S-SSN)))
```

Exercise no. 5

```
SELECT C.TCode, C.CCode
FROM COURSE C, LECTURE L
WHERE C.TCode NOT IN ( SELECT TCode
                       FROM COURSE C2
                       WHERE Topic <> 'Database'
      AND C.CCode = L.CCode
GROUP BY C.TCode, C.CCode
HAVING AVG(AttendingStudent#) =
       (SELECT MAX(AVGS)
        FROM (SELECT AVG(AttendingStudent#) AS AVGS,
                     C1.TCode
              FROM LECTURE L1, COURSE C1
              WHERE L1.CCode = C1.CCode
              GROUP BY C1.CCode, C1.TCode)
              AS TPA
        WHERE TPA.TCode = C.TCode )
```

c)

Ex. nr. 6

```
SELECT T.StudentId, Surname, DegreeProgramme
FROM   EODA, STUDENT S
WHERE  EODA.StudentId = T.StudentId AND
       S.StudentId NOT IN (SELECT StudentId
                           FROM   EODA1, ASSIGN A
                           WHERE  EODA1.ACode =
                                  A.ACode AND
                           DeliveryDate > ScheduledExpDate)
AND    Score = (SELECT MAX (Score)
                FROM   EODA2
                WHERE  EODA2.ACode = EODA.ACode)
GROUP BY S.StudentId, Surname, DegreeProgramme
HAVING COUNT (*) = (SELECT COUNT (*)
                    FROM   ASSIGNMENT_TO_BE_DELIVERED)
```

(2)

(2) Alternative solution

```
(A.Code, Score) IN (SELECT ACode, MAX(Score)
                    FROM   EODA2
                    GROUP BY ACode)
```