

Introduction to Databases

DBDMG - Politecnico di Torino

SQL (IV) - Solutions

1. Given the following relations (primary keys are underlined):

AUTHOR (AuthorCode, Name, Surname, Department, University)

ARTICLE (ArticleCode, Title, Topic)

AUTHORS OF ARTICLE (AuthorCode, ArticleCode)

EDITIONS OF CONFERENCE (Conference, Edition, EditionName, StartDate, EndDate, Editor)

AUTHOR PRESENTS ARTICLE (AuthorCode, Date, StartTime, EndTime, Room, ArticleCode, Conference, Edition)

express the following queries in SQL language.

- (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.

```
SELECT A.AuthorCode, Surname, University, COUNT(*)
FROM Author A, Author_Presents_Article APA
WHERE A.AuthorCode NOT IN (SELECT AuthorCode
                           FROM Article AR,
                           AUTHOR_PRESENTS_ARTICLE APA
                           WHERE APA.ArticleCode=AR.ArticleCode
                              AND Topic<>'Data_Mining'
                              AND APA.AuthorCode=A.AuthorCode)
GROUP BY A.AuthorCode, Conference, Edition,
         Conference, Surname, University)
```

- (b) 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

```
SELECT EditionName, APA.AuthorCode
FROM Author_Presents_Article APA, Edition_Of_Conference EOC
WHERE Conference IN (SELECT Conference
                    FROM Editions_Of_Conference EOC1
                    GROUP BY Conference
                    HAVING COUNT(*) >=10)
AND EOC.Edition=APA.Edition AND E.Conference=APA.Conference
GROUP BY APA.AuthorCode, APA.Edition, APA.Conference,
         EOC.EditionName
HAVING COUNT(*)=(SELECT MAX(TotPa)
                  FROM (SELECT AuthorCode, Edition, Conference,
                              Count(*) AS TotPa)
                  FROM Author_Presents_Article AA)
```

```

GROUP BY AuthorCode, Edition, Conference)
AS TFA
WHERE TFA.Edition=APA.Edition AND
TFA.Conference=APA.Conference

```

2. Given the following relations (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)

express the following queries in SQL language.

- (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.

```

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

```

- (b) 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.

```

SELECT S.StudentId, Surname, DegreeProgramme
FROM EVAL_OF_DEL_ASSIGN 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>SchedeledExpDate)
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

```

or

```

SELECT S.StudentId, Surname, DegreeProgramme
FROM EVAL_OF_DEL_ASSIGN 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>SchedeledExpDate)
(A.Code, Score) IN (SELECT Acode, MAX(Score)
FROM EODA 2
GROUP BY Acode)
GROUP BY S.StudentId, Surname, DegreeProgramme
HAVING COUNT(*) = SELECT (COUNT(*)
FROM ASSIGNMENT_TO_BE_DELIVERED)

```

3. Given the following relations (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)**

express the following queries in SQL language.

(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

```

SELECT DISTINCT Scode
FROM Seminare_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 TptExp
FROM Expertise
GROUP BY S-SSN)))

```

4. Given the following relations (primary keys are underlined):

TEACHER(TCode, TName, TSurname, Department, ResearchGroupName, ResearchArea)
COURSE(CCode, CName, EnrollingStudent, TCode, Topic)
CLASSROOM(RoomID, Floor, VideoKit, Seat)
LECTURE(RoomID, Date, StartHour, EndHour, CCode, AttendingStudent, VideoKit=[yes, no])

express the following queries in SQL language.

(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.

```

SELECT C.Tcode, C.Ccode
FROM Course C, Lecture L
WHERE C.Tcode NOT IN (SELECT Tcode
FROM Course C2
WHERE Topic <> Databases )
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)
```