資料庫管理 Midterm HW 1 0613216 紀侑岑

Q1. a) entity integrity constraint: Primary key attributes PK of each relation schema R cannot have null value in any tuple. Because PK values are used to identify the individual tuple.

b) foreign key: will exist in the referencing relation. It specify which tuple of referenced relation the referencing relation want to reference.

c) referential integrity constraint: supposed relation R1 reference R2.

① The attribute in FK of R1 have the same domain as the primary key attribute pk of R2.

② A value of FK in a tuple t_1 of the current state $r_1(R_1)$ either occurs as a value of PK for some tuple t_2 in the current state $r_2(R_2)$ or is NULL.

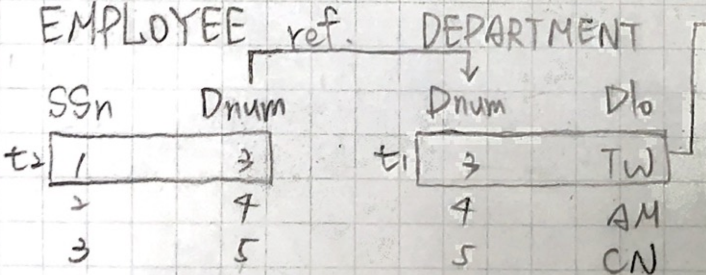
Q2: a) 假設有一個 table Student <ID, NAME, SCORE> 其~~中~~ ID attribute 中無相同值, NAME, SCORE 中均有相同值。

⇒ Super Key: 任何 attribute 的 subset 在 relation 中沒有相同的 tuple 都可以稱作 Superkey, 如: <ID, NAME, SCORE>, <ID, NAME>

Key: 只要再拿掉任一 attribute, 在該 relation 中就會出現相同的 tuple. 如 <ID>

b) (i) TRUE (ii) UNKNOWN (iii) UNKNOWN (iv) TRUE

Q3: EMPLOYEE ref. DEPARTMENT



	SSn	Dnum
t_2	1	3
	2	4
	3	5

	Dnum	Dlo
t_1	3	TW
	4	AM
	5	CN

→ Delete this tuple.

a) set NULL: t_2[Dnum] would be set to NULL.

b) set default: t_2[Dnum] would be set to the default value of Dnum attribute

c) cascade: t_2 would be deleted.

Q4: a) It would be eliminated before calculation. b) 0 (Zero)

c) ① The tuple would be counted in because COUNT(*) count the number of tuple not of attribute.

② If there're identical salary in different tuple, COUNT(DISPINCT Salary) only count once while COUNT(Salary) count as many as the number of these tuples.

Q5:

```
a) UPDATE COMPETETION
   SET Score = 'A'
   WHERE Song_id IN (Select S2.Song_id
                     FROM STUDENT S1, SONG S2, COMPETETION C
                     WHERE type='pop' AND name='Tony'
                           AND C.student_number = S1.student_number
                           AND C.song_id = S2.song_id );

b) INSERT INTO STUDENT
   VALUES      (13, 'Alice', 'female', 'IMF');

c) DELETE FROM SONG
   WHERE Type = 'pop' AND Producer = 'BinMusic';

d) SELECT Name
   FROM STUDENT S1, SONG S2, COMPETETION C
   WHERE Sex='male' AND
         S1.student_number IN (SELECT DISDINCT C.student_number
                               FROM COMPETETION C, SONG S2
                               WHERE Score = 'B', AND Producer = 'SonyMusic'
                                     AND C.song_id = S2.song_id
                               GROUP BY C.student_number
                               HAVING COUNT(*) >= 2);
```

Q6:

```
1) SELECT FNAME, LNAME, PNAME
   FROM EMPLOYEE E, DEPARTMENT D, PROJECT P, WORKS_ON W
   WHERE DNAME = 'Research' AND E.DNO=D.PNUMBER AND
         E.SSN = W.ESSN   AND W.PNO = P.PNUMBER ;

2) SELECT FNAME, LNAME, SALARY, DNAME
   FROM EMPLOYEE E, DEPARTMEMT D1
   WHERE E.SSN IN (SELECT D2.ESSN
                   FROM DEPENDENT D2
                   GROUP BY D2.ESSN
                   HAVING COUNT(*)>=2)
         AND D1.DNUMBER = E.DNO;
```

```
3) SELECT DNAME, SUM(Salary), COUNT(*)
   FROM EMPLOYEE JOIN DEPARTMENT ON DNO=DNUMBER
   GROUP BY DNAME
   HAVING COUNT(*)>5;
```

```
4) SELECT SSN, SUPERSSN
   FROM EMPLOYEE E1, WORKS_ON, PROJECT
   WHERE ESSN=SSN AND PNO=PNUMBER AND PNAME='Mountain Travel'
           AND SALARY > (SELECT SALARY
                         FROM EMPLOYEE E2
                         WHERE E2.SSN = E1.SUPERSSN);
```

```
5) SELECT PNUMBER, PNAME, COUNT(*)
   FROM (EMPLOYEE JOIN WORK_ON ON SSN=ESSN) JOIN PROJECT ON PNO=PNUMBER
   WHERE PLOCATION='Hsinchu'
   GROUP BY PNUMBER, PNAME
   HAVING COUNT(*)>10;
```

```
6) SELECT E1.FNAME, E2.FNAME
   FROM EMPLOYEE E1, EMPLOYEE E2,
   WHERE NOT EXISTS (SELECT *
                     FROM DEPENDENT
                     WHERE ESSN=SSN)
         AND NOT EXISTS (SELECT *
                         FROM WORK_ON
                         WHERE ESSN=SSN)
         AND E2.SSN=E1.SUPERSSN;
```

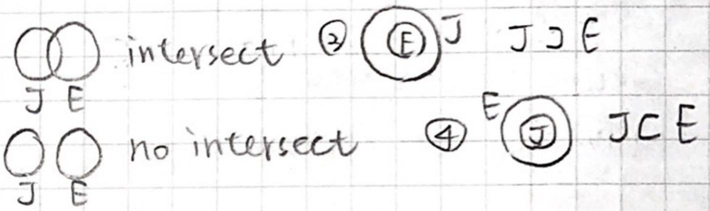
```
7) SELECT FNAME, DNAME
   FROM (EMPLOYEE JOIN DEPARTMENT ON DNO=DNUMBER) JOIN WORKS_ON ON ESSN=SSN
   WHERE NOT EXIST (SELECT * FROM DEPENDENT WHERE ESSN=SSN)
   GROUP BY SSN
   HAVING COUNT(*) > ALL (SELECT COUNT(*)
                          FROM EMPLOYEE JOIN WORKS_ON ON SSN=ESSN
                          WHERE DNO=5
                          GROUP BY SSN)
```

```
8) SELECT FNAME, SALARY
   FROM EMPLOYEE E, DEPARTMENT D,
   WHERE DNAME='R&D' AND SSN=MGRSSN
         AND SSN IN (SELECT SUPERSSN
                     FROM EMPLOYEE
                     GROUP BY SUPERSSN
                     HAVING COUNT(*)>5)
```

```
9) a) SELECT FNAME
      FROM EMPLOYEE E, WORKS_ON W, PROJECT P
      WHERE SSN=ESSN AND PNO IN (SELECT PNO FROM WORKS_ON, EMPLOYEE
                                 WHERE SSN=ESSN AND FNAME='John'
                                 AND LNAME='Smith)
```

b) i) JSmithPNOs → Set J ⇒ 4 set relation :

EmpPNOs → Set E



ii) NOT EXIST (J EXCEPT E) ⇒ { true ⇒ implies J ⊂ E

{ false ⇒ implies J ⊃ E

```
10) WITH RECURSIVE E_S(ESSN, SSSN) AS
      (SELECT SSN, SUPERSSN
       FROM EMPLOYEE, PROJECT, WORKS_ON
       WHERE ESSN=SSN  AND PNAME='AI' AND PNO=PNUMBER
       UNION
       SELECT E2.ESSN, E1.SUPERSSN
       FROM EMPLOYEE E1, E_S E2
       WHERE E2.SSSN=E1.SSN)
    SELECT E1.FNAME, E2.FNAME
    FROM EMPLOYEE E1, EMPLOYEE E2, E_S E3
    WHERE E1.SSN=E3.ESSN AND E2.SSN=E3.SSSN
          AND E3.ESSN IN (SELECT ESSN FROM E_S
                          GROUP BY ESSN HAVING COUNT(*)>2)
```