**Dokumentation databaslabbar**

**Inledning**

Syftet med laborationen var att skapa en databas för ett universitet med data som t.ex. information om studenter, lärare och kurser. Datan tilldelades inför labben men vissa datatyper samt definition av nycklar saknades. Uppgiften var att fylla i korrekta datatyper och nycklar för de olika tabellerna för att få en fungerande databas. Efter detta skulle ett antal sql-frågor samt triggers skrivas.

**Relationsdatamodell**

**Student**

*Beskrivning*: Innehåller studenter som är inskrivna på minst en kurs, antingen på Faculty of Arts eller Faculty of Science.

*Attribut*

|  |  |  |
| --- | --- | --- |
| **spid** | **beskrivning:** | personligt id-nummer som varje student har |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:** | > 0 |
|  | **obligatorisk:** | ja |
|  |  |  |
| **name** | **beskrivning:** | studentens efternamn |
|  | **datatyp:** | character(20) |
|  | **verifieringsuttryck:** | - |
|  | **obligatorisk:** | ja |
|  |  |  |
| **year** | **beskrivning:** | kod som indikerar vilket år studenten går |
|  | **datatyp:** | character(1) |
|  | **verifieringsuttryck:** | "F" ELLER "S" ELLER "J" ELLER "E" |
|  | **obligatorisk:** | ja |
|  |  |  |
| **sup** | **beskrivning:** | personligt id-nummer för stud. handledare |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:** | > 0 |
|  | **obligatorisk:** | nej |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | spid |
| **Primärnyckel:** | spid |
| **Främmandenycklar:** | sup(refererar till tpid i teacher) |

**Teacher**

*Beskrivning:* Innehåller lärare som håller i minst en kurs vid ett av flera ”departments”.

|  |  |  |
| --- | --- | --- |
| **tpid** | **beskrivning:** | personligt id-nummer som varje lärare har |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:** | > 0 |
|  | **obligatorisk:** | Ja |
| **forename** | **beskrivning:** | förnamn |
|  | **datatyp:** | Character(20) |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | - |
| surname | **beskrivning:** | efternamn |
|  | **datatyp:** | Character(20) |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | - |
| **dept** | **beskrivning:** | Kod som indikerar vilket ”department” läraren jobbar för. |
|  | **datatyp:** | Character(2) |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | - |
| **rank** | **beskrivning:** | personligt id-nummer som varje student har |
|  | **datatyp:** | Character(10) |
|  | **verifieringsuttryck:** | ”assis prof” ELLER ”professor” ELLER ”assoc prof” |
|  | **obligatorisk:** | ja |

*Nycklar*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kandidatnycklar:** | | tpid | | |
| **Primärnyckel:** | | tpid | | |
| **Främmandenycklar:**  **Course**  *Beskrivning:* Innehåller de kurser som lärs ut av en lärare. | | dept (refererar till dcode i department) | | |
| ccode | **beskrivning:** | | Unik kod för varje kurs |
|  | **datatyp:** | | Character(5) |
|  | **verifieringsuttryck:** | | - |
|  | **obligatorisk:** | | ja |
| **name** | **beskrivning:** | | Namn på kursen |
|  | **datatyp:** | | Character(20) |
|  | **verifieringsuttryck:** | | - |
|  | **obligatorisk:** | | ja |
| **Does teach** | **beskrivning:** | | tpid för läraren som lär ut kursen |
|  | **datatyp:** | | integer |
|  | **verifieringsuttryck:** | | >0 |
|  | **obligatorisk:** | | ja |
| **responsible** | **beskrivning:** | | Tpid för kursansvarig lärare (kan vara samma som lär ut) |
|  | **datatyp:** | | integer |
|  | **verifieringsuttryck:** | | >0 |
|  | **obligatorisk:** | | ja |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | Ccode |
| **Primärnyckel:** | Ccode |
| **Främmandenycklar:** | Does teach, responsible (refererar båda till tpid i teacher) |

**Grade**

*Beskrivning:* Innehåller värden för att kunna bestämma en students betyg för en viss kurs.

|  |  |  |
| --- | --- | --- |
| letter | **beskrivning:** | En students betyg för en kurs |
|  | **datatyp:** | Character(1) |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | ”A” ELLER ”B” ELLER ”C” ELLER ”D” ELLER ”F” |
| **min** | **beskrivning:** | Minvärde för att komma upp till ett visst betyg |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | >=0 |
| **max** | **beskrivning:** | Maximalt värde för ett betyg |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | <=100 |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | Letter |
| **Primärnyckel:** | letter |
| **Främmandenycklar:** | - |

**Faculty**

*Beskrivning:* Innehåller unika koder för de olika ”faculties” science och arts.

|  |  |  |
| --- | --- | --- |
| fcode | **beskrivning:** | Unik kod för varje faculty |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | >0 |
| **name** | **beskrivning:** | Namn för faculty |
|  | **datatyp:** | Character(20) |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | - |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | fcode |
| **Primärnyckel:** | fcode |
| **Främmandenycklar:** | - |

**Department**

*Beskrivning:* Innehåller namn/koder för alla departments samt vilket faculty de tillhör.

|  |  |  |
| --- | --- | --- |
| **dcode** | **beskrivning:** | Unik kod för varje department |
|  | **datatyp:** | Character(2) |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | - |
| **Name** | **beskrivning:** | Namn för department |
|  | **datatyp:** | Character(20) |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | - |
| fcode | **beskrivning:** | Unik kod för varje faculty |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | >0 |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | dcode |
| **Primärnyckel:** | dcode |
| **Främmandenycklar:** | Fcode (refererar till fcode i faculty) |

**Can\_teach**

*Beskrivning:* Innehåller vilka kurser varje lärare kan lära ut.

|  |  |  |
| --- | --- | --- |
| **tpid** | **beskrivning:** | personligt id-nummer som varje lärare har |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:** | > 0 |
| ccode | **beskrivning:** | Unik kod för varje kurs |
|  | **datatyp:** | Character(5) |
|  | **verifieringsuttryck:** | - |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | Tpid, ccode |
| **Primärnyckel:** | Tpid + ccode |
| **Främmandenycklar:** | Tpid, ccode (refererar till tpid i teacher och ccode i course) |

**Committee**

*Beskrivning:* Innehåller unik kod och namn för de olika kommittéerna.

|  |  |  |
| --- | --- | --- |
| **xcode** | **beskrivning:** | Unik kod för varje kommitte |
|  | **datatyp:** | Character(4) |
|  | **verifieringsuttryck:**  **obligatorisk:** ja | - |
| **name** | **beskrivning:** | Namn på kommitte |
|  | **datatyp:** | Character(20) |
|  | **verifieringsuttryck:**  **obligatorisk:** ja | - |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | Xcode |
| **Primärnyckel:** | xcode |
| **Främmandenycklar:** | - |

**Stud\_course**

*Beskrivning:* Innehåller betyg för varje kurs en student har gått.

|  |  |  |
| --- | --- | --- |
| **spid** | **beskrivning:** | personligt id-nummer som varje student har |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:**  obligatoriskt: ja | > 0 |
| **ccode** | **beskrivning:** | Unik kod för varje kurs |
|  | **datatyp:**  obligatoriskt: ja | Character(5) |
| grade | **beskrivning:** | Poäng för kurs |
|  | **datatyp:** | integer |
|  | **verifieringsuttryck:**  **obligatoriskt: ja** | 0 =< grade <= 100 |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | Spid, ccode |
| **Primärnyckel:** | Spid + ccode |
| **Främmandenycklar:** | Spid, ccode (referrerar till student och course) |

**Teach\_comm**

*Beskrivning:* Innehåller de lärare som är med i en kommitte

|  |  |  |
| --- | --- | --- |
| xcode | **beskrivning:** | Unik kod för varje kommitte |
|  | **datatyp:** | Character(4) |
|  | **verifieringsuttryck:**  obligatoriskt: ja | - |
| **tpid** | **beskrivning:** | Unik kod för varje lärare |
|  | **datatyp:**  **verifieringsuttryck:**  obligatoriskt: ja | Integer  >0 |

*Nycklar*

|  |  |
| --- | --- |
| **Kandidatnycklar:** | Xcode, tpid |
| **Primärnyckel:** | Xcode + tpid |
| **Främmandenycklar:** | tpid, xcode (referrerar till teacher och committee) |

**Slutsummering**

Tidsåtgången för labben var ca 8 timmar för sql-frågor, triggers samt framtagning av relationsmodellen. Rapporten tog ytterligare ett par timmar. Labben har gett mig ökad förståelse och gjort mig mer bekant med mySQL. Jag har också lärt mig hur man kan använda triggers och skriva egna funktioner.

**Bilagor**

SQL-Frågor:

1. How many courses does each department gives? Include departments (if

any) that does not give any courses

SELECT

d.dcode, COUNT(c.ccode)

FROM

course c

RIGHT JOIN

department d ON d.dcode = SUBSTR(c.ccode, 1, 2)

GROUP BY d.dcode

2. List the teachers that does not teach this semester

SELECT

t.tpid

FROM

teacher t

LEFT JOIN

course c ON c.does\_teach = t.tpid

WHERE

c.does\_teach IS NULL;

3. List the members in the committee "Course Planning" (name,

department)

SELECT

t.forename, t.surname, t.dept

FROM

committee c

INNER JOIN

teach\_comm tc ON tc.xcode = c.xcode

INNER JOIN

teacher t ON t.tpid = tc.tpid\_teach\_comm

WHERE

c.name = 'Course Planning'

4. List the teachers that are responsible for more than two courses

SELECT

t.forename, t.surname, COUNT(\*) AS 'Number of Classes'

FROM

course c

INNER JOIN

teacher t ON t.tpid = c.responsible

GROUP BY t.forename , t.surname

HAVING COUNT(\*) > 2

5. List the teachers that can teach the course "Linear Algebra"

SELECT

t.forename, t.surname

FROM

course c

INNER JOIN

can\_teach ct ON ct.ccode\_can\_teach = c.ccode

INNER JOIN

teacher t ON t.tpid = ct.tpid\_can\_teach

WHERE

c.name = 'LINEAR ALGEBRA'

6. List, for each student, the number of course the student takes.

SELECT

s.spid, s.name, COUNT(\*) AS 'Number of Courses'

FROM

student s

INNER JOIN

stud\_course sc ON sc.spid = s.spid

GROUP BY s.spid

7. List for each course in Computer Science that has more than four

students: the name of the course, the highest grade, the lowest grade,

and the average grade on the course.

SELECT

c.ccode,

MAX(sc.grade) AS 'Highest Grade',

MIN(sc.grade) AS 'Lowest Grade',

ROUND(AVG(sc.grade)) AS 'Average Grade'

FROM

course c

INNER JOIN

department d ON d.dcode = SUBSTR(c.ccode, 1, 2)

INNER JOIN

stud\_course sc ON c.ccode = sc.ccode

WHERE

d.name = 'Computer Science'

GROUP BY c.ccode

HAVING COUNT(\*) > 4

8. List name and (numeric) grade for the students on the course "CS

Introduction" that have received the grade 'C'.

SELECT

s.name, sc.grade

FROM

stud\_course sc

INNER JOIN

course c ON c.ccode = sc.ccode

INNER JOIN

student s ON s.spid = sc.spid

INNER JOIN

grade g ON g.min <= sc.grade AND g.max >= sc.grade

WHERE

c.name = 'CS INTRODUCTION'

AND g.letter = 'C'

9. List for each course in Mathematics: the name of the course, the

number of students and the teacher that does teach the course

SELECT

c.name, COUNT(sc.spid) AS 'Number of students', t.forename, t.surname

FROM

course c

INNER JOIN

department d ON d.dcode = SUBSTR(c.ccode, 1, 2)

INNER JOIN

stud\_course sc ON sc.ccode = c.ccode

Inner join teacher t ON t.tpid = c.does\_teach

WHERE

d.name = 'Mathematics'

GROUP BY c.ccode

10.List for each student that has failed a course: the student’s name, the

name of the course and the grade

SELECT

s.name, c.name, sc.grade

FROM

stud\_course sc

INNER JOIN

grade g ON g.min <= sc.grade AND g.max >= sc.grade

INNER JOIN

student s ON s.spid = sc.spid

INNER JOIN

course c ON c.ccode = sc.ccode

WHERE

g.letter = 'F'

--------------------TRIGGERS----------------------

CREATE DEFINER=`root`@`localhost` TRIGGER `grade\_BEFORE\_UPDATE` BEFORE UPDATE ON `grade` FOR EACH ROW BEGIN

delimiter $$

CREATE TRIGGER update\_grade

BEFORE UPDATE ON grade

FOR EACH ROW

BEGIN

IF ((NEW.max > 100) or (NEW.min > 100)) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Value too large';

end if;

END$$

delimiter;

-----------------------------------------------------

DELIMITER $$

USE `lab2`$$

CREATE DEFINER=`root`@`localhost` TRIGGER `before\_insert` AFTER INSERT ON `course` FOR EACH ROW BEGIN

DECLARE X INT;

select COUNT(\*) INTO X

from course c

inner join teacher t\_does ON t\_does.tpid = c.does\_teach

inner join teacher t\_resp ON t\_resp.tpid = c.responsible

where t\_does.dept != t\_resp.dept;

IF(X > 0) THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Department not matching';

END IF;

END$$

DELIMITER ;

-------------------------------------------------------

USE `lab2`;

DELIMITER $$

USE `lab2`$$

DROP TRIGGER IF EXISTS `lab2`.`student\_BEFORE\_DELETE` $$

DELIMITER ;

DROP TRIGGER IF EXISTS `lab2`.`student\_BEFORE\_DELET`;

DELIMITER $$

USE `lab2`$$

CREATE DEFINER=`root`@`localhost` TRIGGER `student\_BEFORE\_DELETE` BEFORE DELETE ON `student` FOR EACH ROW BEGIN

INSERT INTO alumni\_students SELECT OLD.spid, OLD.name;

END$$

DELIMITER ;