SELECT statement (6)

Sample database

```
CREATE TABLE Department(
name                VARCHAR2(50),
code                CHAR(5),
total_staff_number  NUMBER(2)              NOT NULL,
Chair               VARCHAR2(50),
budget              NUMBER(9,1)            NULL,
CONSTRAINT dept_pkey PRIMARY KEY(name),
CONSTRAINT dept_ckey1 UNIQUE(code),
CONSTRAINT dept_ckey2 UNIQUE(chair),
CONSTRAINT dept_check1
CHECK (total_staff_number BETWEEN 1 AND 50)  );
```

```
CREATE TABLE Course(
c#                  CHAR(7),
title               VARCHAR2(200)          NOT NULL,
credits             NUMBER(1)              NOT NULL,
offered_by          VARCHAR2(50)           NULL,
CONSTRAINT course_pkey PRIMARY KEY(c#),
CONSTRAINT course_check1
        CHECK (credits IN (6, 12) ),
CONSTRAINT course_fkey1 FOREIGN KEY(offered_by)
        REFERENCES Department(name) );
```


Queries with existential quantifiers

(Correlated nested queries)

Find the codes of departments that offer at least one course

Department

name | code | total staff number | chair | budget

Course

c# | title | credits | offered by

```
SELECT code
FROM Department JOIN Course
ON name = offered_by;
```


Queries with existential quantifiers

(Correlated nested queries)

Find the codes of departments such that there exists at least one course offered by a department we are looking for

```
SELECT code
FROM Department D
WHERE
      EXISTS
  ( SELECT title
    FROM Course
    WHERE Course.offered_by = D.name);
```


Queries with existential quantifiers

(Correlated nested queries)

Find the codes of departments such that there exists at least one course offered by a department we are looking for

```
SELECT code
FROM Department D
WHERE EXISTS
  ( SELECT title
    FROM Course
    WHERE Course.offered_by = D.name);
```


Computational model (queries with existential quantifiers)

```
SELECT <ATTRIBUTES_1>
FROM <TABLE_1> T
WHERE EXISTS
            (SELECT <ATTRIBUTES_2>
             FROM <TABLE_2>
             WHERE <CONDITION_2>(T));
```

```
forall rows s in <TABLE_1>
   forall rows t in <TABLE_2>
      if evaluate(<CONDITION_2>, t, s) then
         TEMP ← append(t.<ATTRIBUTES_2>)
      endif;
   endforall;
      if TEMP is not empty then
         output(s.<ATTRIBUTES_1>)
endforall;
```

Queries with negated existential quantifiers

Find the codes of departments that offer no courses

Find the codes of departments such that does not exist a course offered a department we are looking for

```
SELECT code
FROM Department D
WHERE
      NOT EXISTS
( SELECT title
  FROM Course
  WHERE Course.offered_by = D.name);
```


Computational model (queries with negated existential quantifiers)

```
SELECT <ATTRIBUTES_1>
FROM <TABLE_1> T
WHERE NOT EXISTS
           (SELECT <ATTRIBUTES_2>
            FROM <TABLE_2>
            WHERE <CONDITION_2>(T));
```

```
forall rows s in <TABLE_1>
   forall rows t in <TABLE_2>
      if evaluate(<CONDITION_2>, t, s) then
         TEMP ← append(t.<ATTRIBUTES_2>)
      endif;
   endforall;
      if TEMP is empty then
         output(s.<ATTRIBUTES_1>)
endforall;
```


Nested queries revisited

Find the codes of departments that offer no courses

```
SELECT code
FROM Department
WHERE name NOT IN
                  ( SELECT offered_by
                    FROM Courses );
```

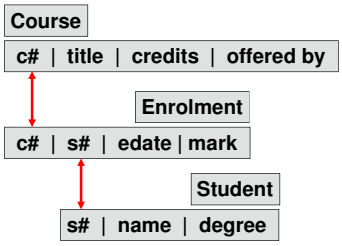

Set algebra revisited

Find the codes of departments that offer no courses

```
SELECT code
FROM DEPARTMENT
WHERE dname IN ( SELECT dname
                 FROM Department
                  MINUS
                 SELECT offered_by
                 FROM Courses);
```


Queries with universal quantifiers

Find the titles of all courses enrolled by each student



Queries with universal quantifiers

Find the titles of courses enrolled by each student

If a course is enrolled by each student then does no exist a student such that he/she is not enrolled in the course

Find the titles of all courses such that does not exist a student who is not enrolled in a course we are looking for

For example a fact that "all students are enrolled in CSCI235" can be expressed as the following formula of First Order Logic:

$$\forall x \in Students \;\; Enrolled(x, CSCI235)$$

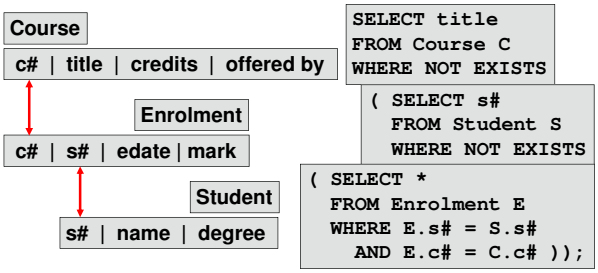
It is logically equivalent to

$$not(not(\forall x \in Students \;\; Enrolled(x, CSCI235)))$$

$$not\ (\exists x \in Students \;\; not\ Enrolled(x, CSCI235)))$$

Queries with universal quantifiers

Find the titles of courses such that does not exist a student who is not enrolled in a course we are looking for




```
SELECT title
FROM Course C
WHERE NOT EXISTS
    ( SELECT s#
      FROM Student S
      WHERE NOT EXISTS
          ( SELECT *
            FROM Enrolment E
            WHERE E.s# = S.s#
              AND E.c# = C.c# ));
```

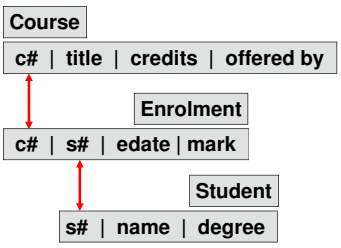

Queries with universal quantifiers

Find the titles of courses such that does not exist a student who is not enrolled in a course we are looking for

```
SELECT title
FROM Course C
WHERE NOT EXISTS
          ( SELECT s#
            FROM Student S
            WHERE NOT EXISTS
                      ( SELECT *
                        FROM Enrolment E
                        WHERE E.s# = S.s#
                              AND
                              E.c# = C.c# ) );
```


Grouping revisited

Find the titles of courses enrolled by each student




```
SELECT title
FROM Course
WHERE c# IN
    ( SELECT c#
      FROM Enrolment
      GROUP BY c#
      HAVING count(*) =
          ( SELECT count(*)
            FROM Student ) );
```


References

Elmasri R., Navathe S. B., *Database Systems*, chapter 6

Ramakrishnan R., Gehrke J., *Database Management Systems*, chapter 5.4.2, 5.4.4

https://sai.uow.edu.au/oradocs/
SQL Reference, SELECT statement