QUERIES WITH QUANTIFIERS (PART 1)

Venn Diagram and SQL Templates

EXAMPLE DATABASE SCHEMA

Student

sid	sname	major	byear

Course

cno	cname	dept

Enroll

sid	cno	grade

Department

dept	location

QUERIES WITH QUANTIFIERS

- Find the sid of each student who takes some CS courses
- Find the sid of each student who takes no CS courses
- Find the sid of each student who takes not only CS courses
- Find the sid of each student who takes only CS courses
- Find the sid of each student who takes not all CS courses
- Find the sid of each student who takes all CS courses

QUERIES WITH QUANTIFIERS

- Find the cno of each course that enrolls some Math majors
- Find the cno of each course that enrolls no Math majors
- Find the cno of each course that enrolls only Math majors
- Find the cno of each course that enrolls all Math majors
- Find the cno of each course that enrolls more than half Math majors

QUERIES WITH QUANTIFIERS

- Find each (s,d) pair such that student s takes some courses offered by department d
- Find each (s,d) pair such that student s takes all courses offered by department d
- Find each (s,d) pair such that student s takes fewer than 5 courses offered by department d
- …

EXPRESSING QUERIES WITH QUANTIFIERS IN SQL

- We will provide a general technique using Venn-diagrams with conditions to express queries with quantifiers
- We will use views and parameterized views to represents the sets in these Venn diagrams
- We will use the set predicates [NOT] EXISTS and [NOT] IN, and the set operations INTERSECT and EXCEPT to translate Venn diagrams with conditions into SQL queries that express queries with quantifiers

QUERIES WITH QUANTIFIERS

- Find the sid of each student who takes some CS courses
- Find the sid of each student who takes no CS courses
- Find the sid of each student who takes not only CS courses
- Find the sid of each student who takes only CS courses
- Find the sid of each student who takes not all CS courses
- Find the sid of each student who takes all CS courses

TEMPLATE FOR QUERY WITH QUANTIFIER

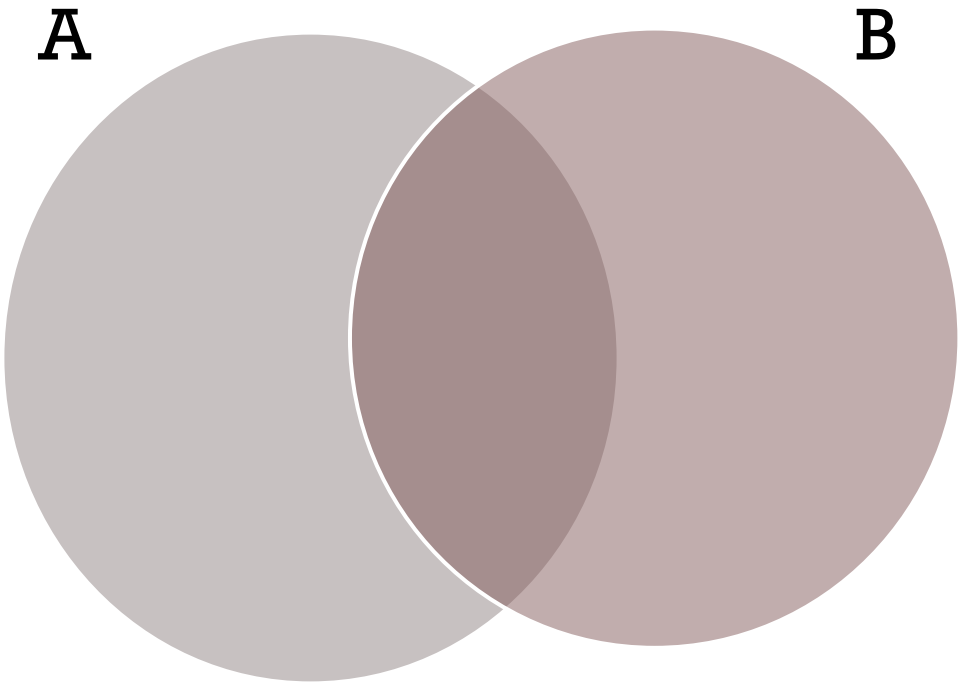
- Find the sid of each student who takes "quantifier" CS courses

quantifier
some
no
not only
only
not all
all
all and only
at least 2
…

VENN DIAGRAMS AND SQL TEMPLATES

- There is a Venn diagram with conditions to express a query with a quantifier
- There is a corresponding SQL statement to express this conditioned Venn diagram

VENN DIAGRAM OF 2 SETS

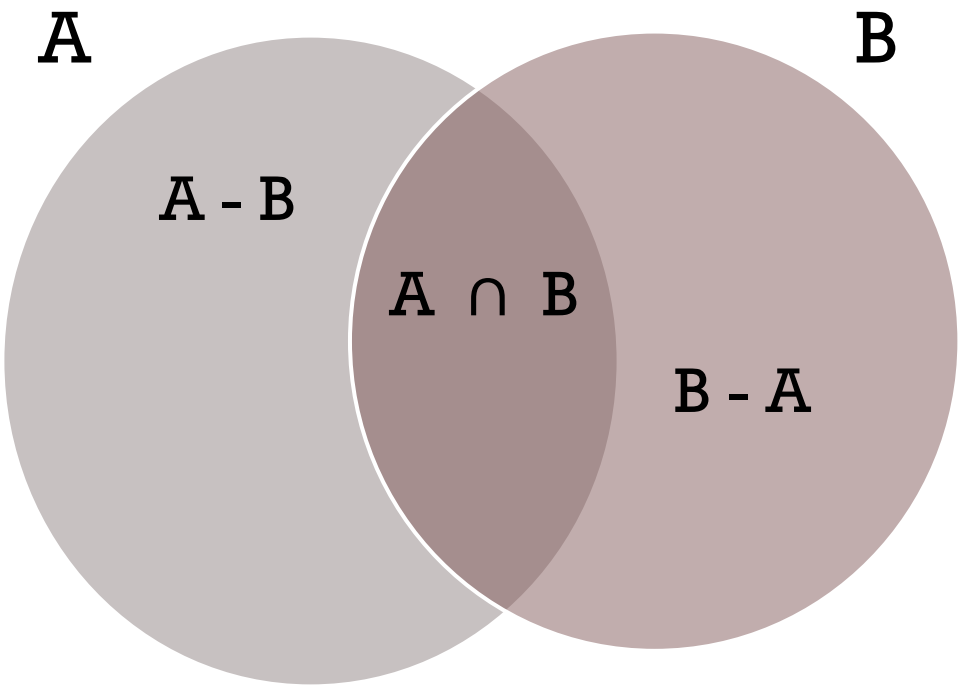


A-B Left Ear

B-A Right Ear

$A \cap B$ Lens

VENN DIAGRAM OF 2 SETS WITH CONDITIONS



Condition
$A \cap B \neq \emptyset$
$A \cap B = \emptyset$
$A - B \neq \emptyset$
$A - B = \emptyset$
$B - A \neq \emptyset$
$B - A = \emptyset$
$A - B = \emptyset$ and $B - A = \emptyset$
$\lvert A \cap B \rvert \geq 2$
…

CONDITIONS EXPRESSED IN LOGIC

$A \cap B \neq \emptyset$ $\quad \exists x (x \in A \cap B)$

$\exists x (x \in A \wedge x \in B)$

$A \cap B = \emptyset$ $\quad \neg \exists x (x \in A \cap B)$

$\neg \exists x (x \in A \wedge x \in B)$

CONDITIONS EXPRESSED IN LOGIC

$A - B \neq \emptyset$ $\quad \exists x\,(x \in A - B)$

$\exists x\,(x \in A \ \wedge x \notin B)$

$A - B = \emptyset$ $\quad \neg\,\exists x\,(x \in A - B)$

$\neg\,\exists x\,(x \in A \ \wedge x \notin B)$

$\forall x(\,x \in A \rightarrow x \in B)$

$A \subseteq B$

CONDITIONS EXPRESSED IN LOGIC

$B - A \neq \emptyset$ $\quad \exists\, x\,(x \in B - A)$

$\exists\, x\,(x \in B \ \wedge\ x \notin A)$

$B - A = \emptyset$ $\quad \neg\, \exists\, x\,(x \in B - A)$

$\neg\, \exists\, x\,(x \in B \ \wedge\ x \notin A)$

$\forall\, x(\, x \in B \rightarrow x \in A)$

$B \subseteq A$

CONDITIONS EXPRESSED IN LOGIC

$A - B = \emptyset$ and $B - A = \emptyset$

$A \subseteq B$ and $B \subseteq A$

$A = B$

CONDITIONS EXPRESSED IN LOGIC

$$|A \cap B| \geq 2$$

$$\exists x \exists y (x \neq y \wedge x \in A \cap B \wedge y \in A \cap B)$$

$$|A \cap B| < 2$$

$$\neg \exists x \exists y (x \neq y \wedge x \in A \cap B \wedge y \in A \cap B)$$

VENN DIAGRAM FOR OUR QUERIES

- For a student with key sid, CoursesEnrolledIn(sid) denotes the set of courses taken by that student
- CS_Courses denotes the set of courses offered by the 'CS' department
- Linking this to a Venn diagram, we set

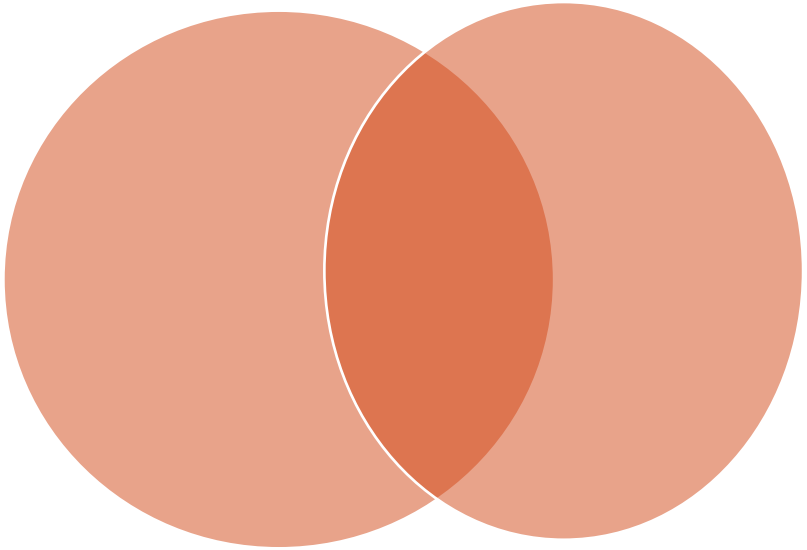
A = CoursesEnrolledIn(sid)

B = CS_Courses

- Note that for different values of sid, CoursesEnrolledIn(sid) denote different sets

VENN DIAGRAM FOR OUR QUERIES

A = CoursesEnrolledIn(sid)



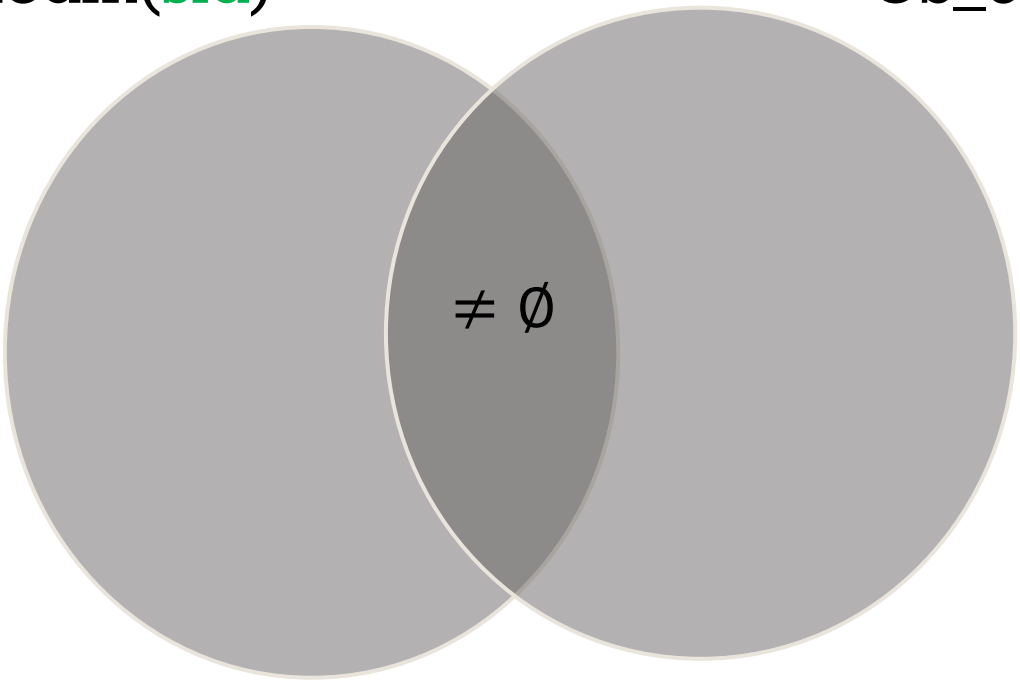
B = CS_Courses

SOME

Find the sid of each student who takes some CS courses

CoursesEnrolledIn(sid)

CS_courses



SOME

$$A \cap B \neq \emptyset \qquad \exists x (x \in A \cap B)$$

CoursesEnrolledIn(sid)

```
SELECT  S.sid
FROM    Student S
WHERE EXISTS (  SELECT E.cno
                FROM    Enroll E
                WHERE E.sid = S.sid
                    INTERSECT
                 SELECT  C.cno
                 FROM    Course C
                 WHERE C.dept = 'CS')
```

CS_Courses

SOME

```
SELECT  S.sid
FROM    Student S
WHERE EXISTS (SELECT  E.cno
                FROM    Enroll E
                WHERE E.sid = S.sid AND
                        E.cno IN (SELECT C.cno
                                  FROM    Course C
                                  WHERE C.dept = 'CS'))
```

SOME

- In SQL, the SOME quantifier can be expressed using the

EXISTS (A INTERSECT B) template

or using the

EXISTS (A IN B) template

SOME (IS A VERY SPECIAL CASE)

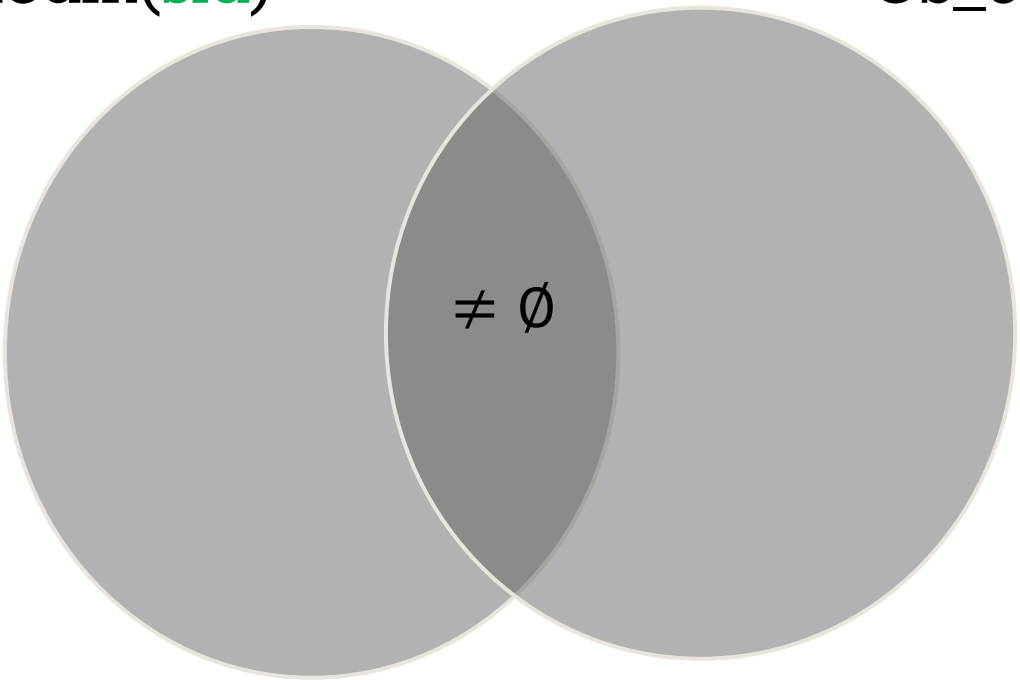
```
SELECT  DISTINCT E.Sid
FROM    Enroll E, Course C
WHERE E.Cno  =  C.Cno  AND C.Dept = 'CS'
```

SOME

Find the sid of each student who takes some CS courses

CoursesEnrolledIn(sid)

CS_courses



DEFINING RELEVANT SETS WITH VIEWS

- Definition of CoursesEnrolledIn(sid)

```
CREATE FUNCTION CoursesEnrolledIn(sid INTEGER)
    RETURNS TABLE (cno INTEGER) AS
    $$
      SELECT  E.cno
      FROM    Enroll E
      WHERE  E.sid = CoursesEnrolledIn.sid;
    $$  LANGUAGE SQL
```

- Definition CS_Courses

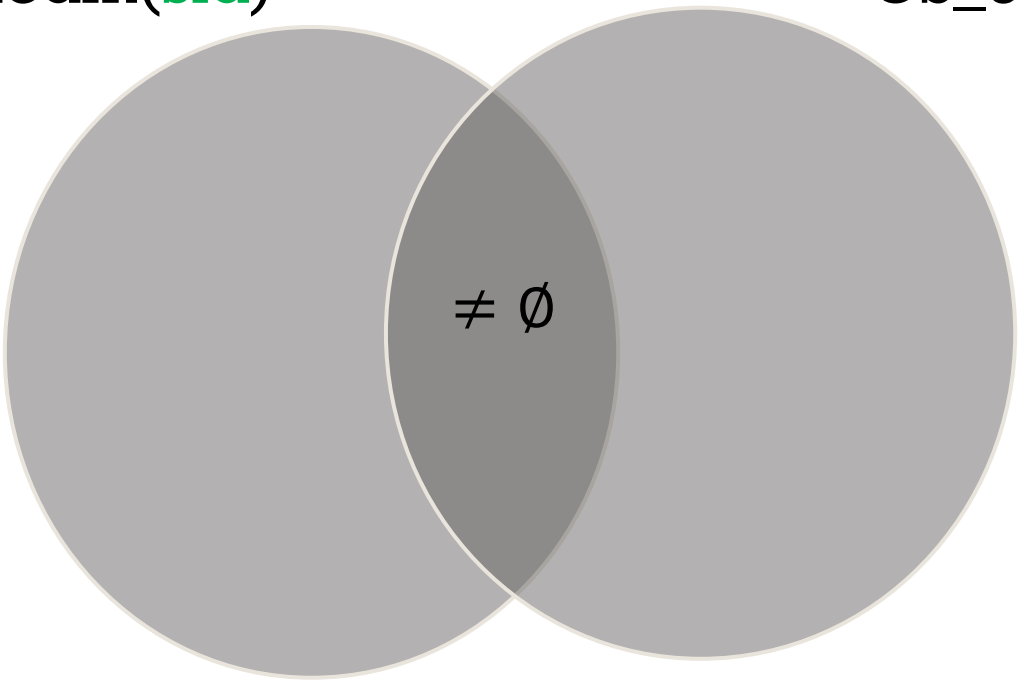
```
CREATE VIEW CS_Courses AS (SELECT  C.cno
                           FROM    Course C
                           WHERE  C.Dept = 'CS')
```

SOME

Find the sid of each student who takes some CS courses

CoursesEnrolledIn(sid)

CS_courses



SOME

```
SELECT  sid
FROM    Student
WHERE EXISTS ( SELECT  cno
                 FROM    CoursesEnrolledIn(sid)
                   INTERSECT
                 SELECT  cno
                 FROM    CS_Courses )
```

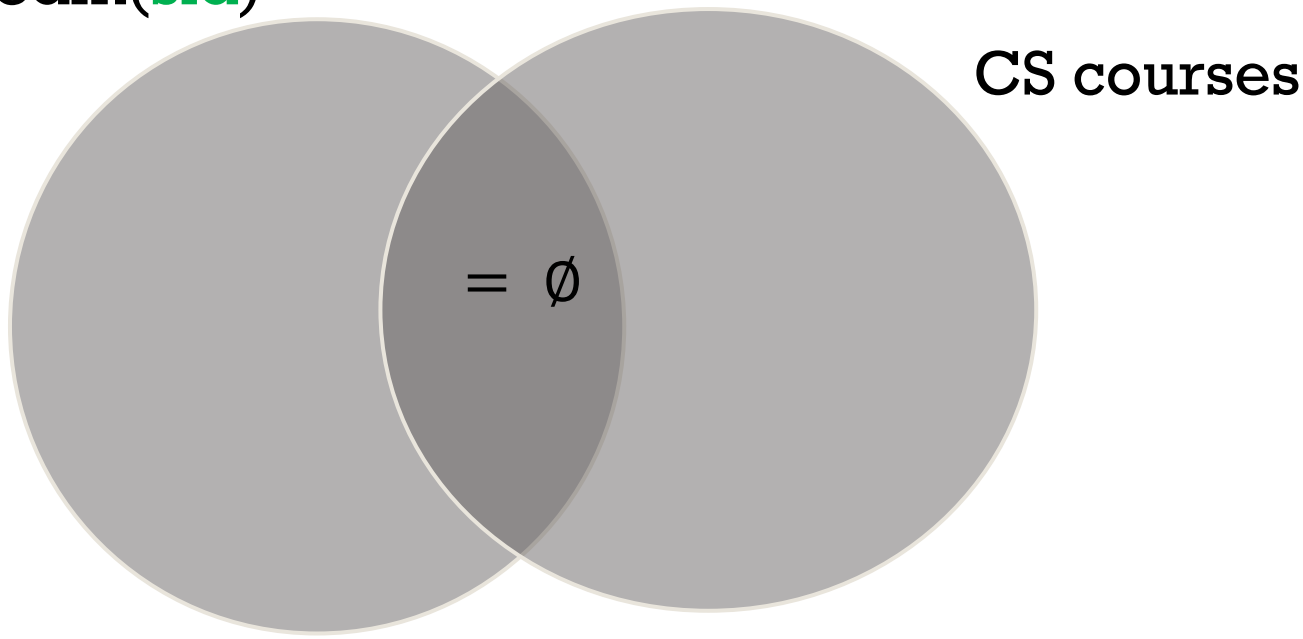
SOME

```
SELECT  sid
FROM    Student
WHERE EXISTS (SELECT  cno
                    FROM    CoursesEnrolledIn(sid)
                    WHERE cno IN (SELECT cno
                                        FROM    CS_Courses))
```

NO

Find the sid of each student who takes no CS courses

CoursesEnrolledIn(sid)

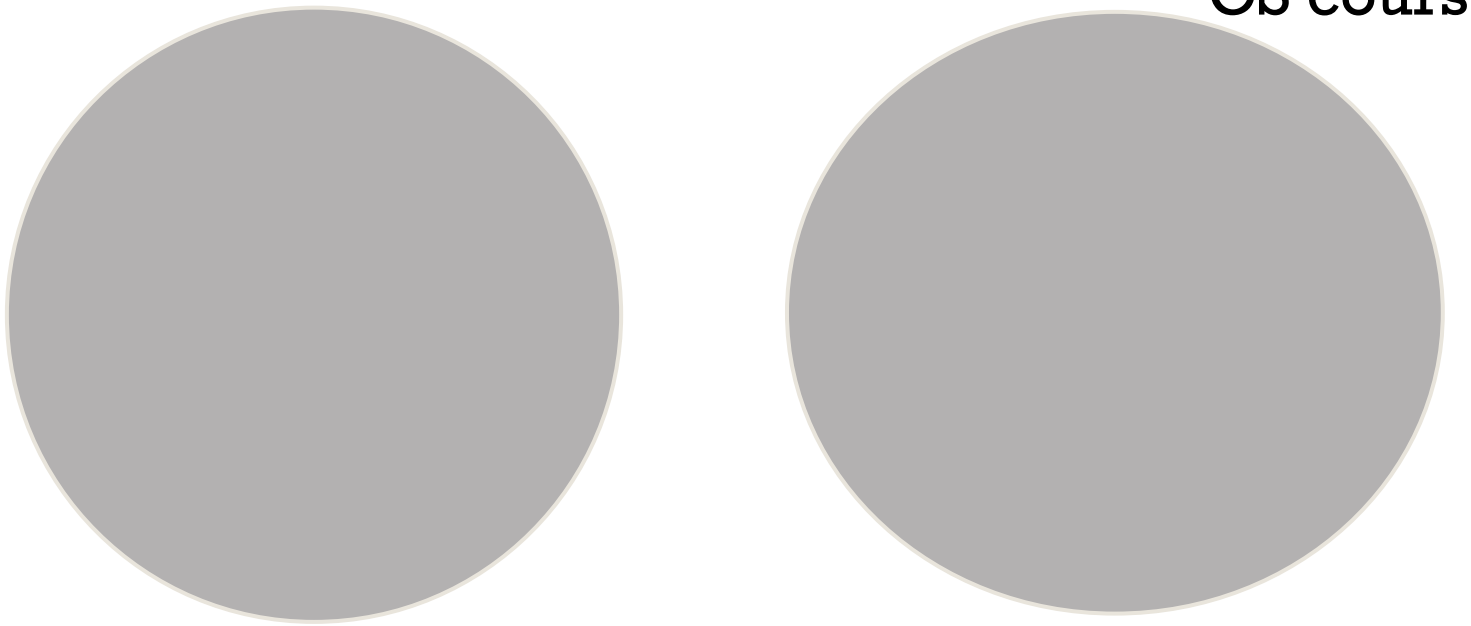


NO

Find the sid of each student who takes no CS courses

CoursesEnrolledIn(sid)

CS courses



NO

$$A \cap B = \emptyset \qquad \neg \exists x (x \in A \cap B)$$

```
SELECT  sid
FROM    Student
WHERE NOT EXISTS ( SELECT  cno
                   FROM    CoursesEnrolledIn(sid)
                      INTERSECT
                   SELECT  cno
                   FROM    CS_Courses )
```

NO

```
SELECT  sid
FROM    Student
WHERE NOT EXISTS (SELECT  cno
                  FROM    CoursesEnrolledIn(sid)
                  WHERE cno IN (SELECT cno
                                FROM    CS_Courses))
```

NO

- In SQL, the NO quantifier can be expressed using the

NOT EXISTS (A INTERSECT B) template

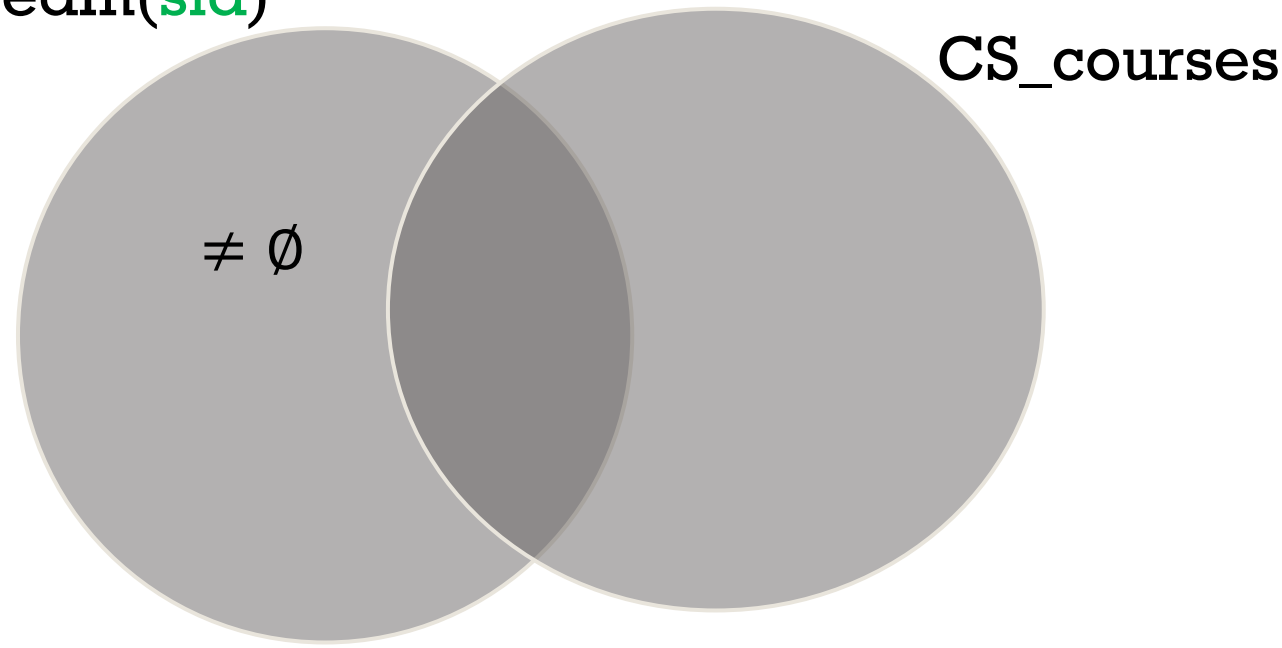
or using the

NOT EXISTS (A IN B) template

NOT ONLY

Find the sid of each student who takes not only CS cou

CoursesEnrolledIn(sid)



NOT ONLY

$$A - B \neq \emptyset \qquad \exists x (x \in A - B)$$

```
SELECT  sid
FROM    Student
WHERE EXISTS ( SELECT  cno
                 FROM    CoursesEnrolledIn(sid)
                   EXCEPT
                 SELECT  cno
                 FROM    CS_Courses )
```

NOT ONLY

```
SELECT  sid
FROM    Student
WHERE EXISTS (SELECT  cno
                    FROM    CoursesEnrolledIn(sid)
                    WHERE cno NOT IN (SELECT cno
                                                FROM    CS_Courses))
```

NOT ONLY

- In SQL, the NOT ONLY quantifier can be expressed using the

EXISTS (A EXCEPT B) template

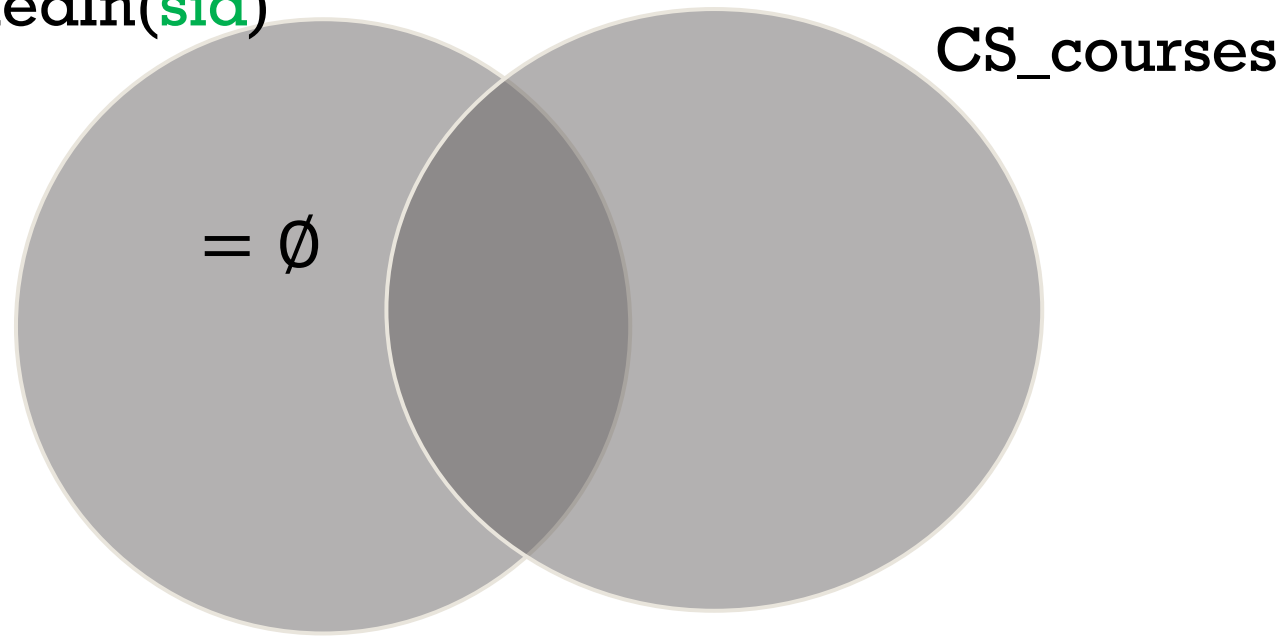
or using the

EXISTS (A NOT IN B) template

ONLY

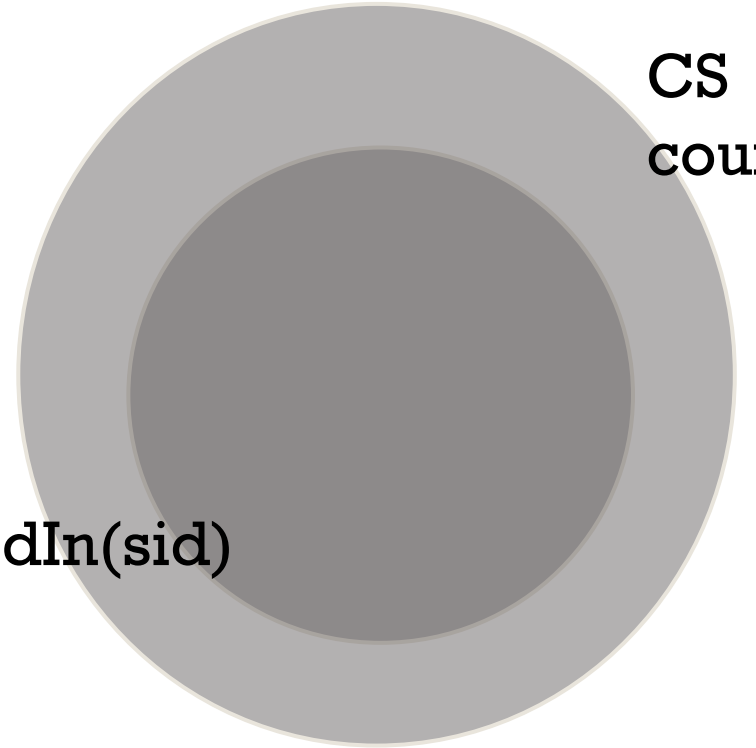
Find the sid of each student who takes only CS courses

CoursesEnrolledIn(sid)



ONLY

Find the sid of each student who takes only CS courses



ONLY

$$A - B = \emptyset \qquad \neg \exists x (x \in A - B)$$

```
SELECT  sid
FROM    Student
WHERE NOT EXISTS ( SELECT  cno
                   FROM    CoursesEnrolledIn(sid)
                    EXCEPT
                   SELECT  cno
                   FROM    CS_Courses )
```

ONLY

```
SELECT  sid
FROM    Student
WHERE NOT EXISTS (SELECT  cno
                  FROM    CoursesEnrolledIn(sid)
                  WHERE cno NOT IN (SELECT cno
                                    FROM CS_Courses))
```

ONLY

- In SQL, the ONLY quantifier can be expressed using the

NOT EXISTS (A EXCEPT B) template

or using the

NOT EXISTS (A NOT IN B) template

CAUTION WITH ONLY QUANTIFIER!

- Observe that if a student with sid "s" takes no courses then CourseEnrolledIn("s") = ∅
- In that case, the condition

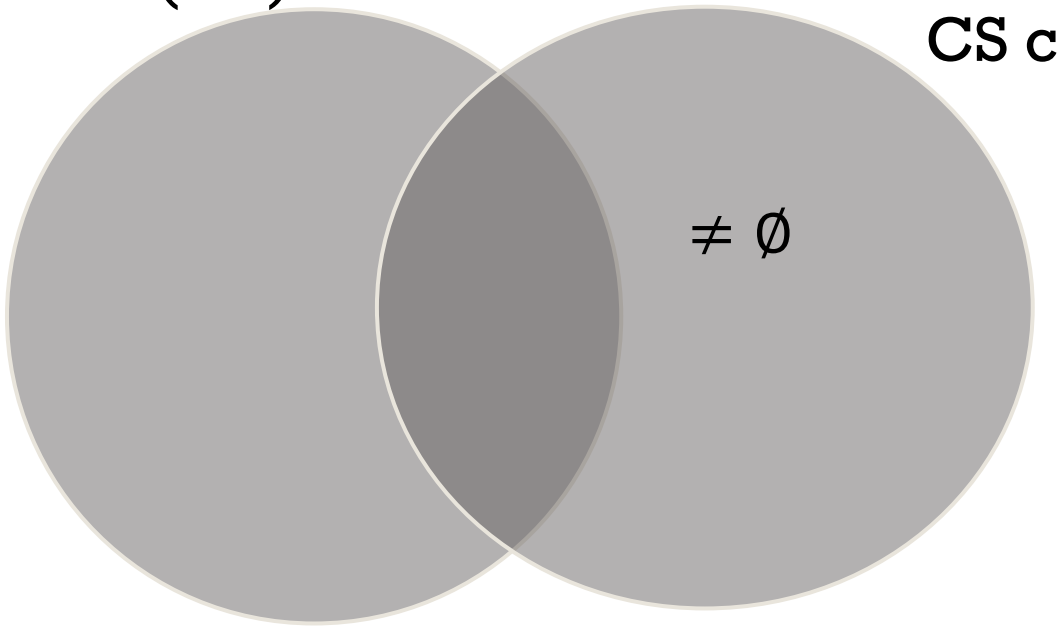
```
NOT EXISTS ( SELECT cno
             FROM   CoursesEnrolledIN(s)
               EXCEPT
             SELECT cno
             FROM   CS_Course)
```

is true and therefore that student sid "s" is part of the solution even though that student takes no courses

NOT ALL

Find the sid of each student who takes not all CS course

CoursesEnrolledIn(sid)



NOT ALL

$$B - A \neq \emptyset \qquad \exists x\,(x \in B - A)$$

```
SELECT  sid
FROM    Student
WHERE EXISTS (SELECT cno
                    FROM   CS_courses
                        EXCEPT
                    SELECT E.cno
                    FROM    CoursesEnrolledIn(sid))
```

NOT ALL

```
SELECT  sid
FROM    Student
WHERE EXISTS (SELECT cno
                    FROM   CS_Courses
                    WHERE
                        cno  NOT IN(SELECT cno
                                           FROM   CoursesEnrolledIn(sid))
```

NOT ALL

- In SQL, the NOT ALL quantifier can be expressed using the

EXISTS (B EXCEPT A) template

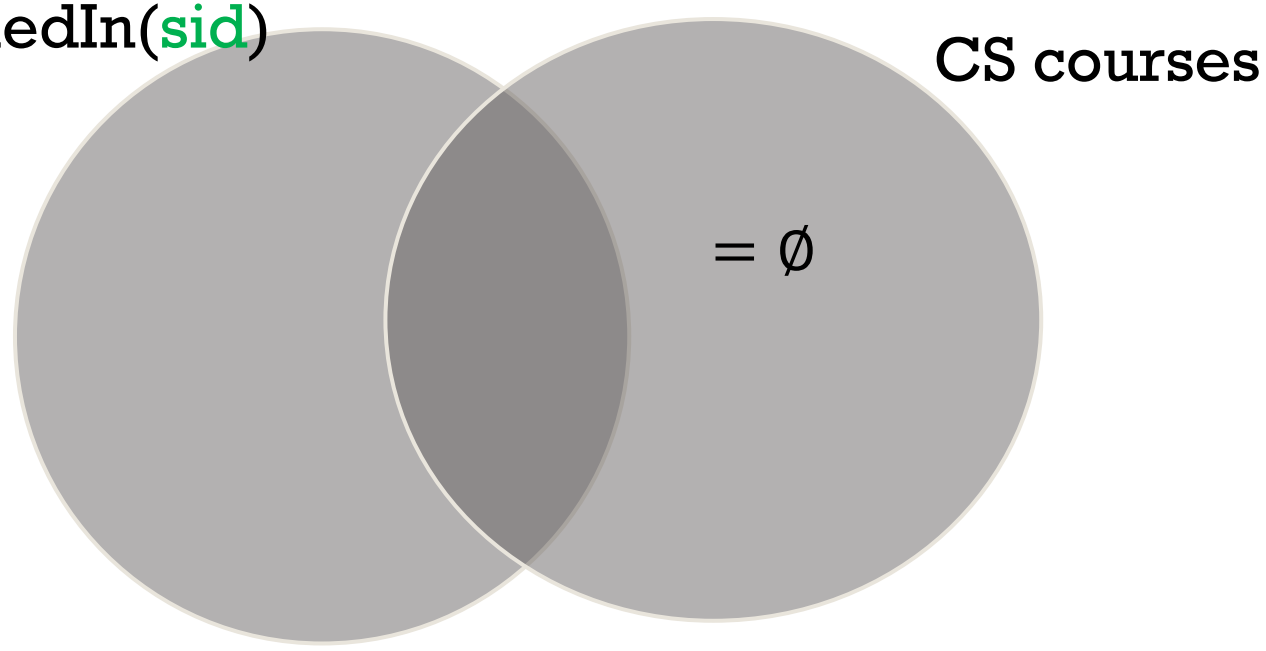
or using the

EXISTS (B NOT IN A) template

ALL

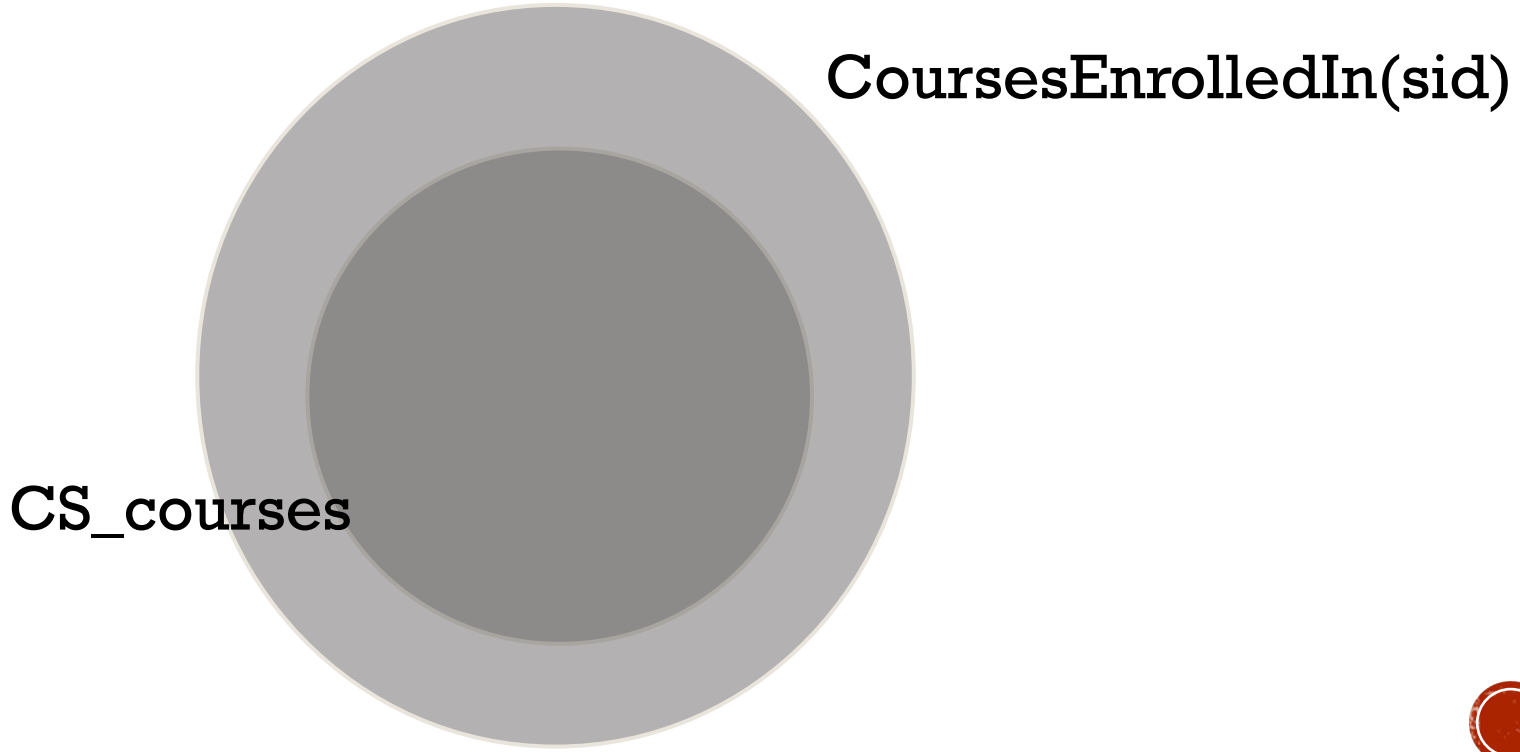
Find the sid of each student who takes all CS courses

CoursesEnrolledIn(sid)



ALL

Find sid of each student who takes all CS courses



ALL

$$B - A = \emptyset \qquad \neg \exists x \, (x \in B - A)$$

```
SELECT  sid
FROM    Student
WHERE  NOT EXISTS (SELECT cno
                   FROM   CS_courses
                      EXCEPT
                   SELECT E.cno
                   FROM    CoursesEnrolledIn(sid))
```

ALL

```
SELECT  sid
FROM    Student
WHERE NOT EXISTS (SELECT cno
                  FROM   CS_Courses
                  WHERE
                cno  NOT IN(SELECT cno
                            FROM   CoursesEnrolledIn(sid))
```

ALL

- In SQL, the ALL quantifier can be expressed using the

NOT EXISTS (B EXCEPT A) template

or using the

NOT EXISTS (B NOT IN A) template

CAUTION WITH ALL QUANTIFIER

- Observe that if there are no CS courses, then a student with sid "s" will satisfy the ALL condition
- E.g, the condition

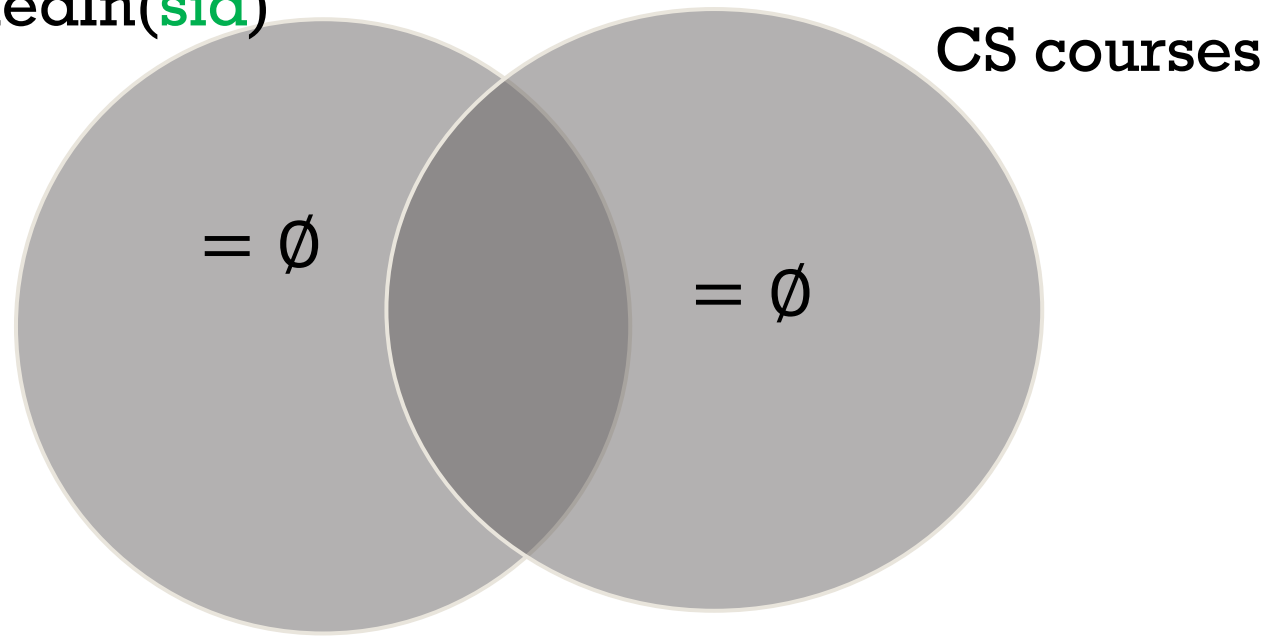
```
NOT EXISTS (SELECT cno
            FROM   CS_Courses
             EXCEPT
            SELECT cno
            FROM   CoursesEnrolledIn(s))
```

is true and therefore that student sid "s" is part of the solution even though there are no CS courses

ALL AND ONLY

Find the sid of each student who takes all and only CS courses

CoursesEnrolledIn(sid)



ALL AND ONLY

We have multiple quantifiers: ALL and ONLY

These must both be specified in the WHERE clause using the ALL and ONLY quantifier templates:

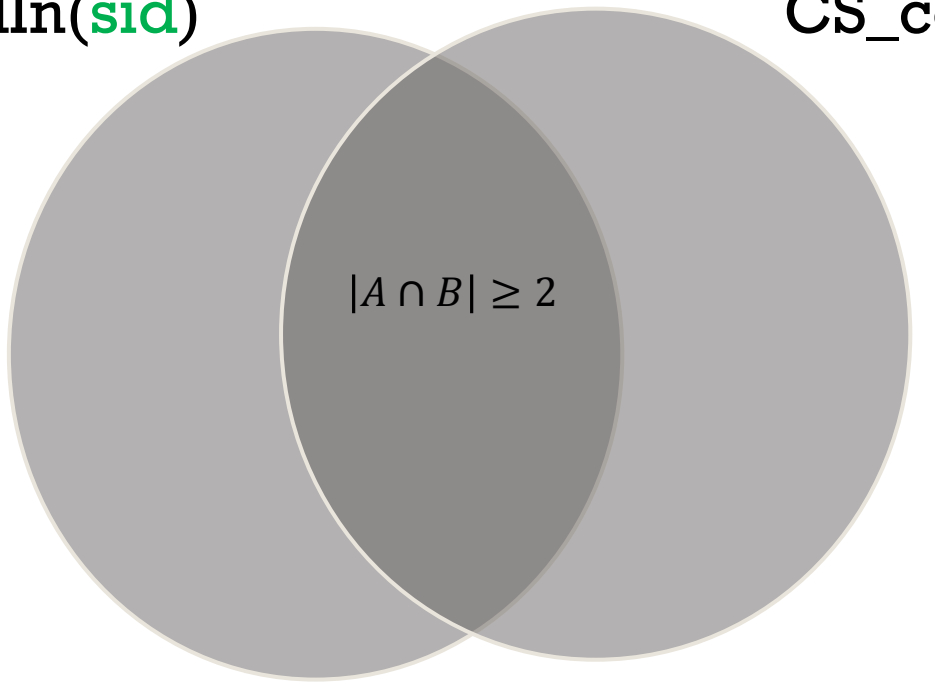
```
SELECT  sid
FROM    Student S
WHERE ALL template     AND
          ONLY template
```

AT LEAST TWO

Find the sid of each student who takes at least two CS courses

CoursesEnrolledIn(sid)

CS_courses



AT LEAST 2

$|A \cap B| \geq 2. \quad \exists x \exists y (x \neq y \wedge x \in A \cap B \wedge y \in A \cap B)$

```
SELECT  sid
FROM    Student
WHERE EXISTS (SELECT 1
                FROM    CoursesEnrolledIn(sid) c1,
                        CoursesEnrolledIn(sid) c2
                WHERE  c1.cno <> c2.cno AND
                       c1.cno IN (SELECT cno
                                  FROM   CS_Courses) AND
                       c2.cno IN (SELECT cno
                                  FROM  CS_Courses))
```

At least two

QUERIES WITH QUANTIFIERS

- Find each (s,d) pair such that student s takes some courses offered by department d
- Find each (s,d) pair such that student s takes all courses offered by department d
- Find each (s,d) pair such that student s takes fewer than 5 courses offered by department d
- …

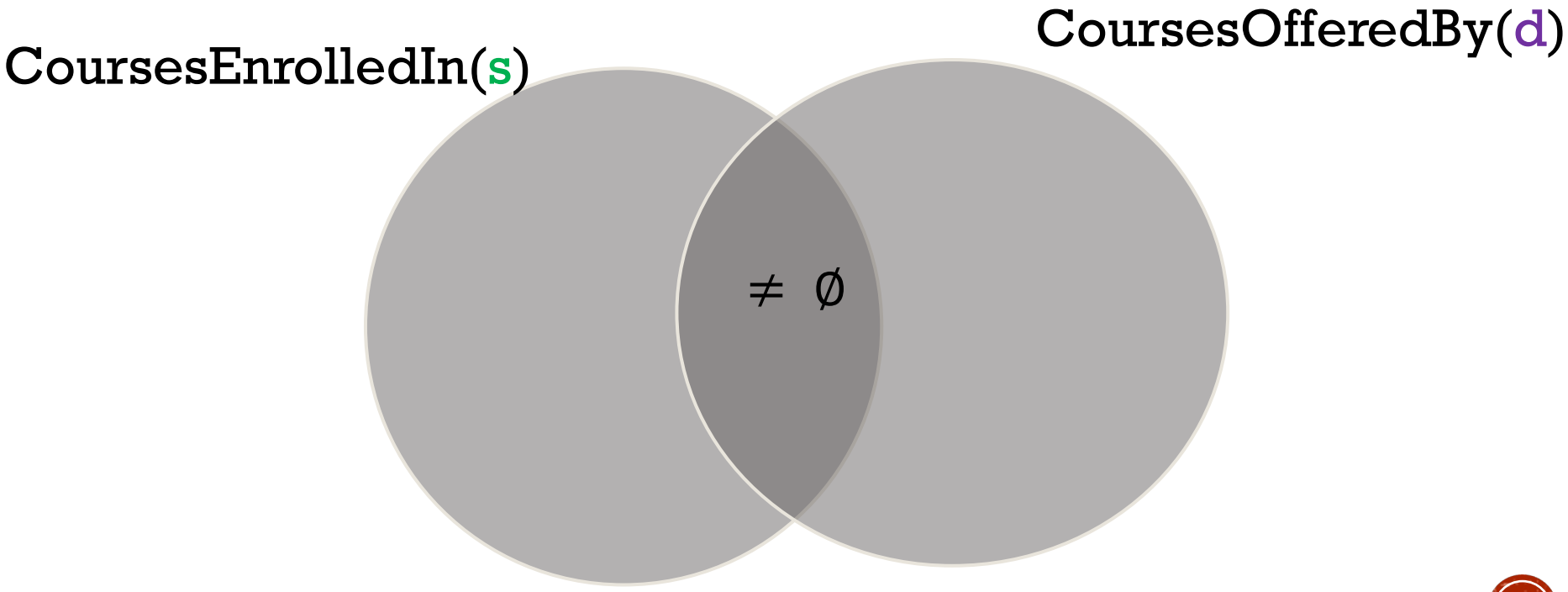
QUERIES WITH QUANTIFIERS RETURNING PAIRS

Find each (s,d) pair such that student s takes 'quantifier' courses offered by department d.

quantifier
some
no
not only
only
not all
all
all but only
at least 2
…

SOME

Find each (s,d) pair such that student s takes some courses offered by department d



DEFINING RELEVANT SETS WITH VIEWS

- Definition of CoursesEnrolledIn(sid)

```
CREATE FUNCTION CoursesEnrolledIn(sid TEXT)
    RETURNS TABLE (cno TEXT) AS
    $$
      SELECT  E.cno
      FROM    Enroll E
      WHERE  E.sid = CoursesEnrolledIn.sid;
    $$  LANGUAGE SQL
```

DEFINING RELEVANT SETS WITH VIEWS

- Definition of CoursesOfferedBy(dept)

```
CREATE FUNCTION Courses OfferedBy(dept TEXT)
    RETURNS TABLE (cno TEXT) AS
    $$
      SELECT   C.cno
      FROM     Course C
      WHERE    C.dept = CoursesOfferedBy.dept;
    $$  LANGUAGE SQL
```

SOME FOR PAIRS

$$A \cap B \neq \emptyset \qquad \exists x\,(x \in A \cap B)$$

```
SELECT S.sid, D.dept
FROM   Student S, Department D
WHERE EXISTS (SELECT cno
                    FROM   CoursesEnrolledIn(S.sid)
                     INTERSECT
                    SELECT cno
                    FROM   CoursesOfferedBy(D.dept))
```

SOME FOR PAIRS

```
SELECT  S.sid, D.dept
FROM    Student S, Department D
WHERE EXISTS (SELECT  cno
                    FROM    CoursesEnrolledIn(S.sid)
                    WHERE
                        cno IN (SELECT cno
                                FROM    CoursesOfferedBy(D.dept)
```

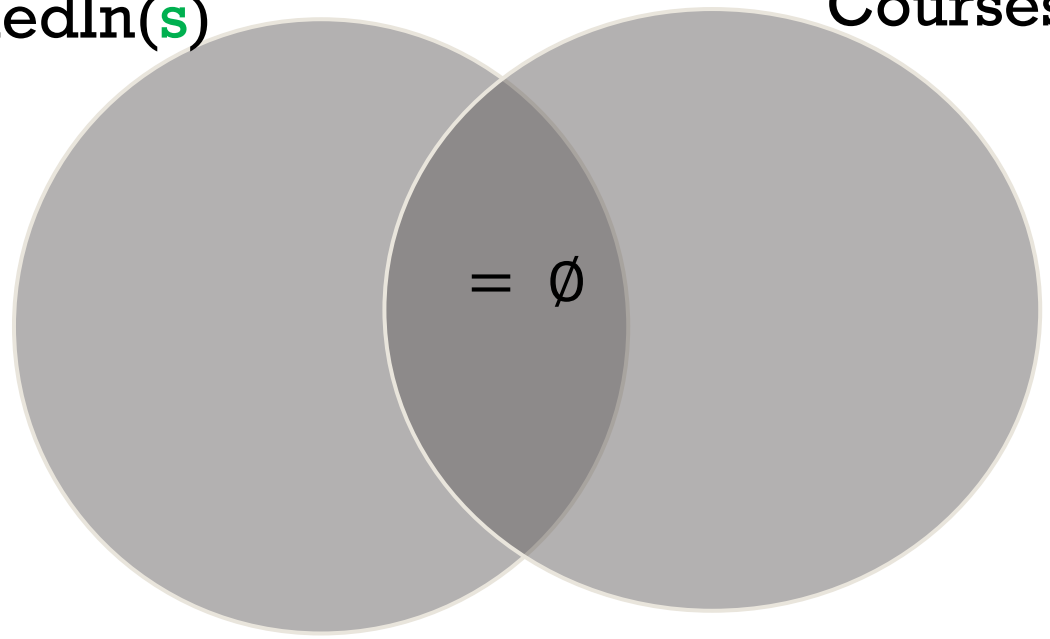
NO FOR PAIRS

$$A \cap B = \emptyset \qquad \neg \exists x (x \in A \cap B)$$

Find each (s,d) pair such that student s takes no courses offered by department d

CoursesEnrolledIn(s)

CoursesOfferedBy(d)



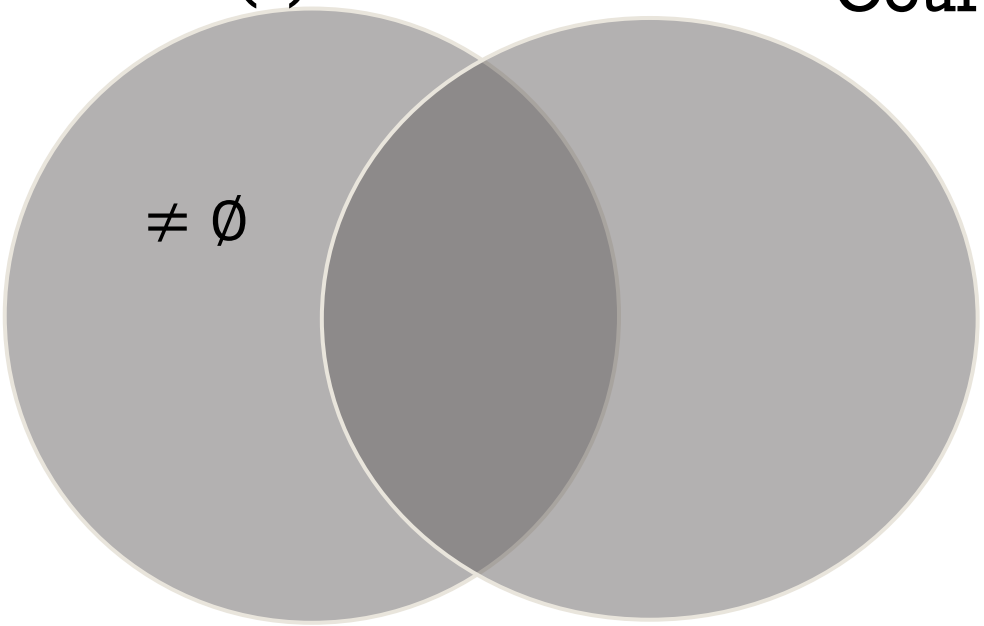
NOT ONLY FOR PAIRS

$$A - B \neq \emptyset \qquad \exists x\,(x \in A - B)$$

Find each (s,d) pair such that student s takes not only courses offered by department d

CoursesEnrolledIn(s)

CoursesOfferedBy(d)



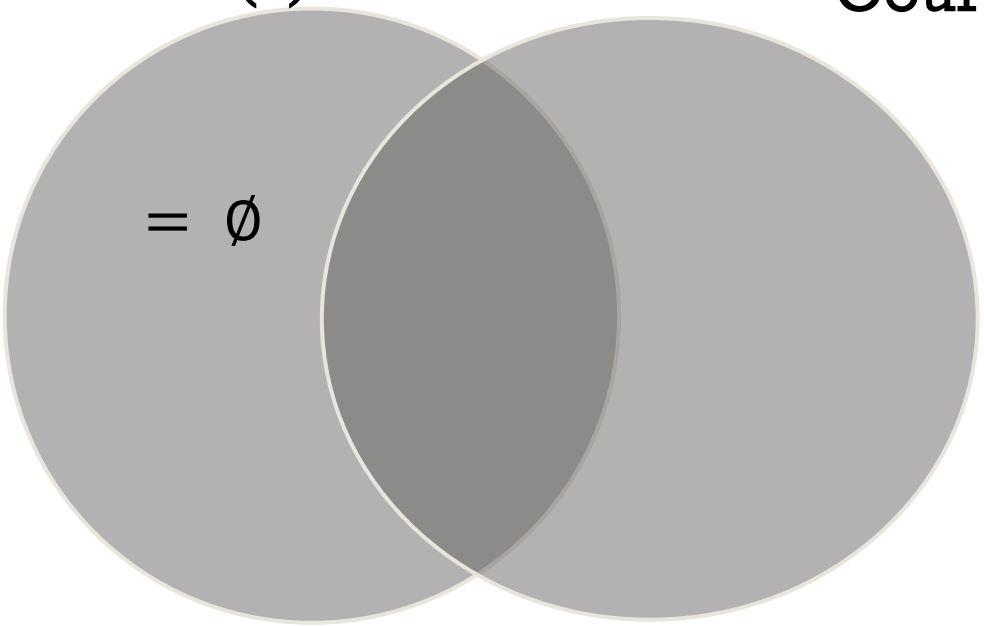
ONLY FOR PAIRS (SUBSET JOIN)

$$A - B = \emptyset \qquad \neg \exists x \, (x \in A - B)$$

Find each (s,d) pair such that student s takes only courses offered by department d

CoursesEnrolledIn(s)

CoursesOfferedBy(d)



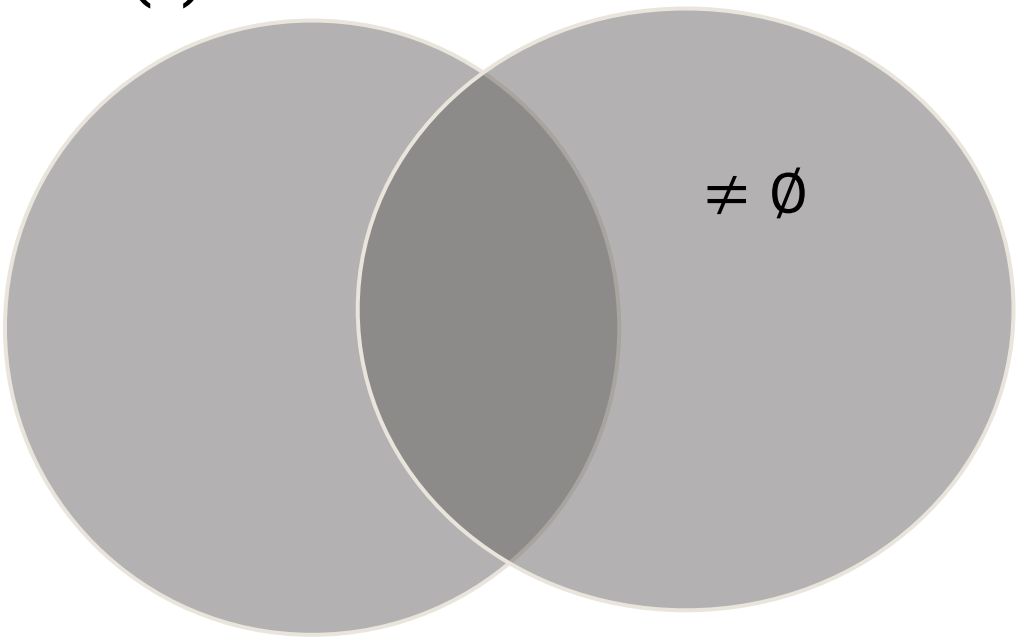
NOT ALL FOR PAIRS

$B - A \neq \emptyset \qquad \exists x\,(x \in B - A)$

Find each (s,d) pair such that student s takes not all courses offered by department d

CoursesEnrolledIn(s)

CoursesOfferedBy(d)

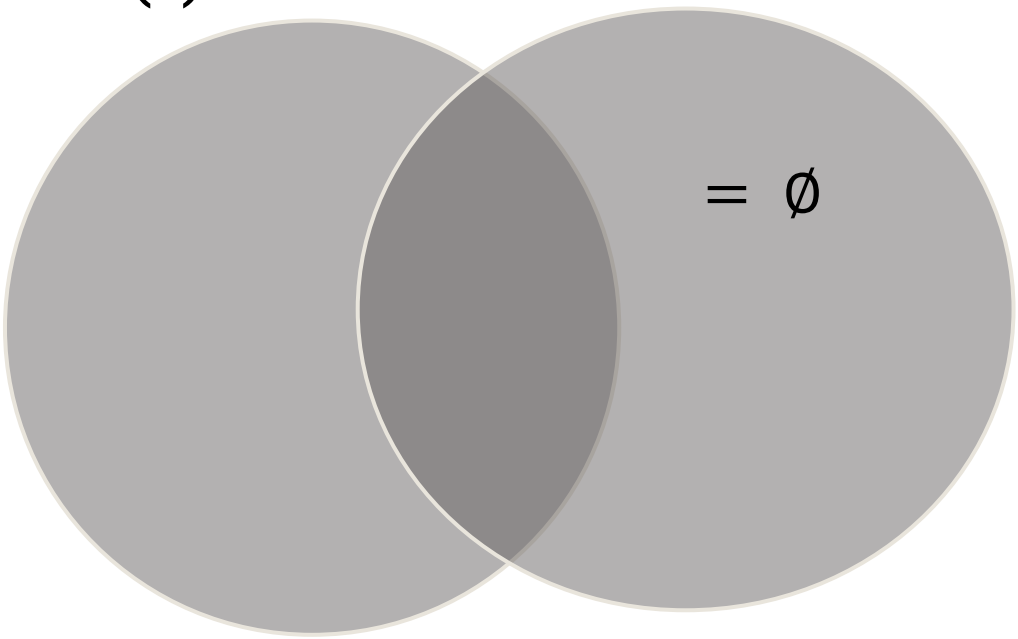


ALL FOR PAIRS

$$B - A = \emptyset \qquad \neg \exists x (x \in B - A)$$

Find each (s,d) pair such that student s takes all courses offered by department d

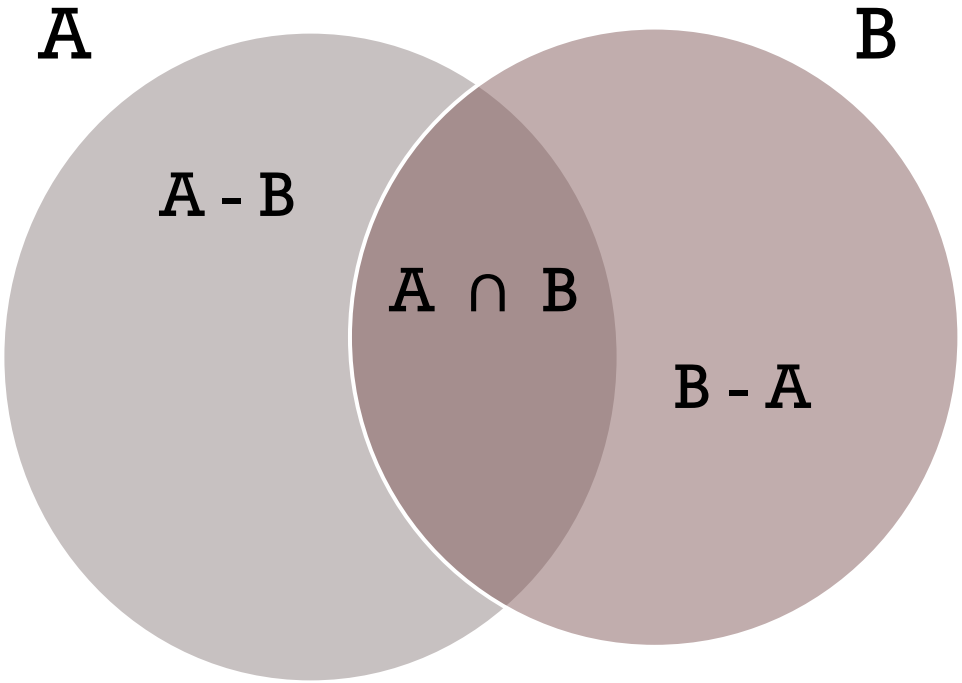
CoursesEnrolledIn(s) CoursesOfferedBy(d)



VENN DIAGRAM AND SQL TEMPLATE

- There is a Venn diagram with conditions to express a query with a quantifier
- These conditions can be expressed as counting conditions
- There is a corresponding SQL statement to express this Venn diagram with condition using the COUNT function

VENN DIAGRAM OF 2 SETS

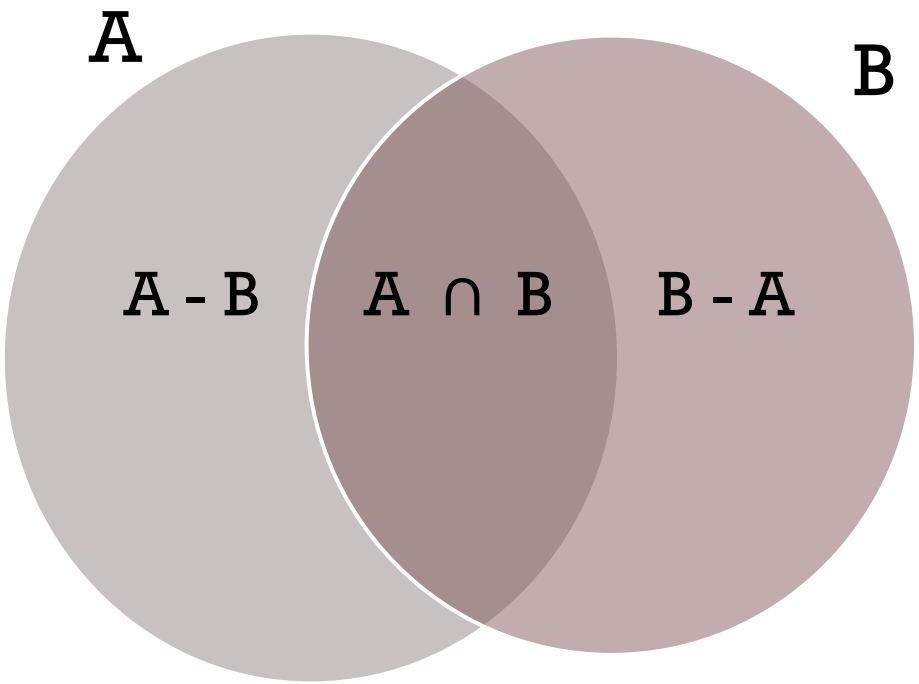


A-B Left Ear

B-A Right Ear

$A \cap B$ Lens

VENN DIAGRAM OF 2 SETS WITH CONDITIONS



Condition
$A \cap B \neq \emptyset \Leftrightarrow \lvert A \cap B \rvert \geq 1$
$A \cap B = \emptyset \Leftrightarrow \lvert A \cap B \rvert = 0$
$A - B \neq \emptyset \Leftrightarrow \lvert A - B \rvert \geq 1$
$A - B = \emptyset \Leftrightarrow \lvert A - B \rvert = 0$
$B - A \neq \emptyset \Leftrightarrow \lvert B - A \rvert \geq 1$
$B - A = \emptyset \Leftrightarrow \lvert B - A \rvert = 0$
$\lvert A - B \rvert = 0$ and $\lvert B - A \rvert = 0$
$\lvert A \cap B \rvert \geq 2$
…

VENN DIAGRAM FOR OUR QUERIES

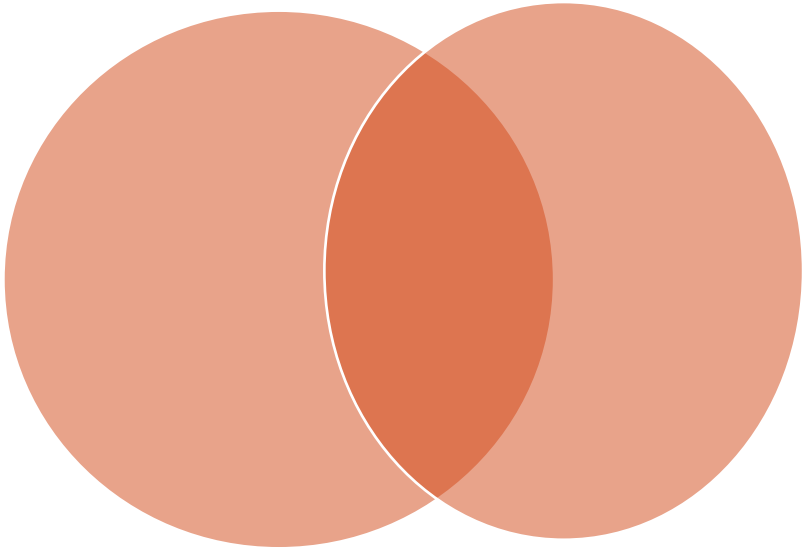
- For a student with key sid, CoursesEnrolledIn(sid) denotes the set of courses taken by this student
- CS_courses denotes the set of courses offered by the 'CS' department
- So in our previous Venn diagram

$$A = \text{CoursesEnrolledIn(sid)}$$
$$B = \text{CS\_Courses}$$

- Note that for different values of sid, CoursesEnrolledIn(sid) denote different sets

VENN DIAGRAM FOR OUR QUERIES

CoursesEnrolledIn(sid)



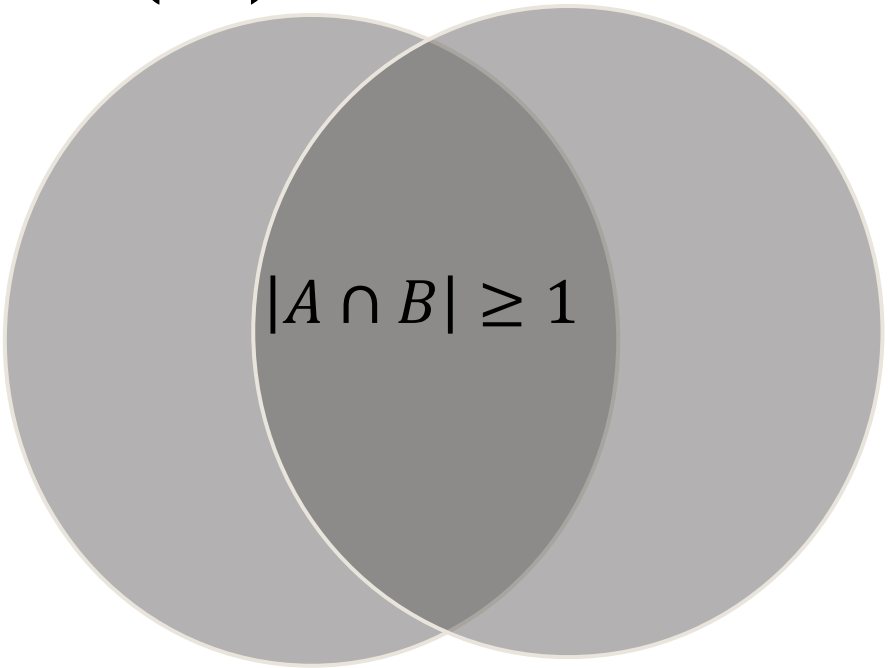
CS_Courses

SOME

Find sid of each student who takes some CS courses

CoursesEnrolledIn(sid)

CS_courses



SOME

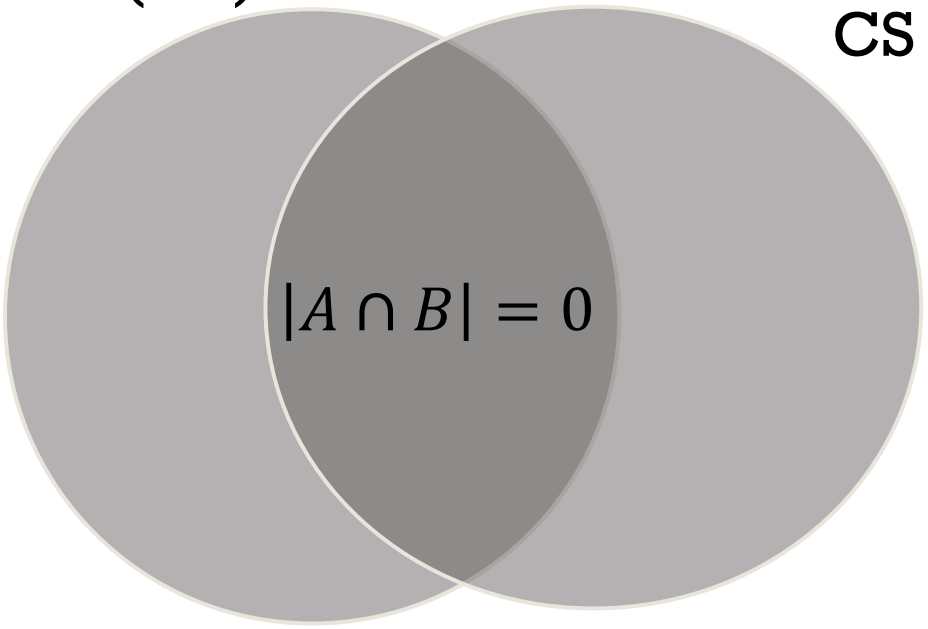
```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CoursesEnrolledIn(sid)
                   INTERSECT
                   SELECT cno
                   FROM    CS_Courses)  q) ≥ 1;
```

NO

Find sid of each student who takes no CS courses

CoursesEnrolledIn(sid)

CS courses



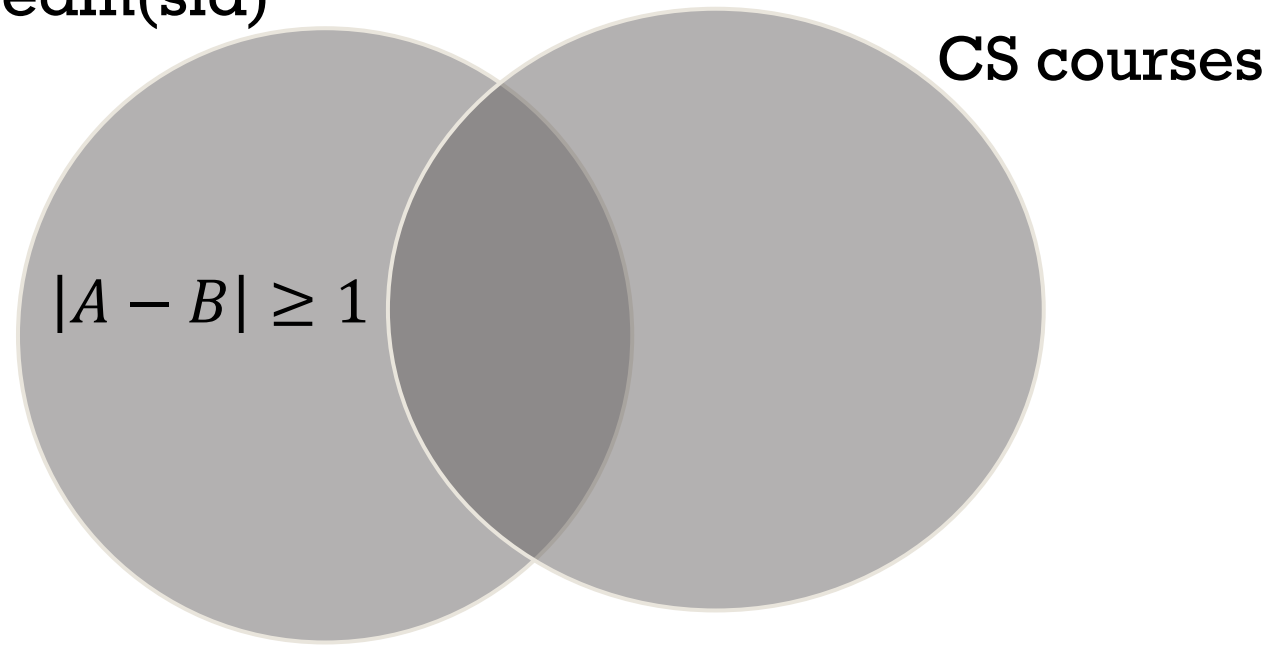
NO

```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CoursesEnrolledIn(sid)
                   INTERSECT
                   SELECT cno
                   FROM    CS_Courses)  q) = 0;
```

NOT ONLY

Find sid of each student who takes not only CS courses

CoursesEnrolledIn(sid)

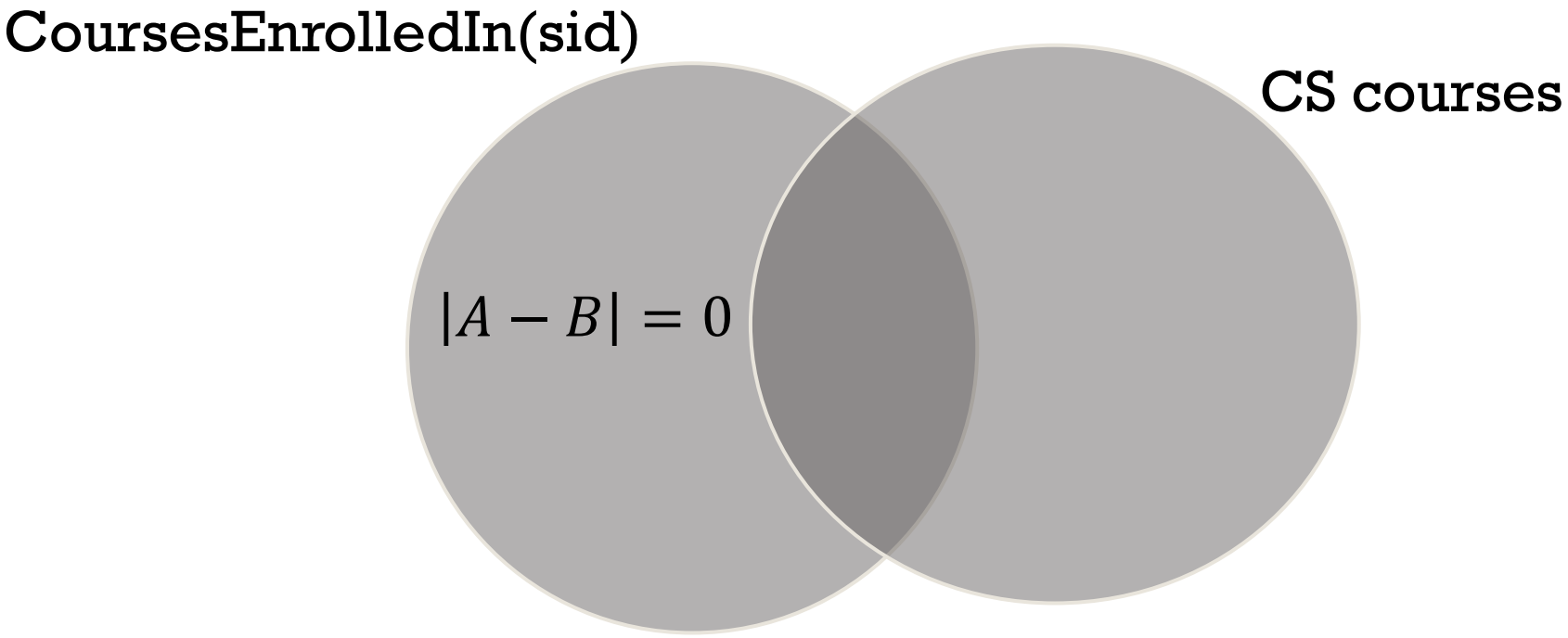


NOT ONLY

```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CoursesEnrolledIn(sid)
                   EXCEPT
                   SELECT cno
                   FROM    CS_Courses)  q) ≥ 1;
```

ONLY

Find sid of each student who takes only CS courses



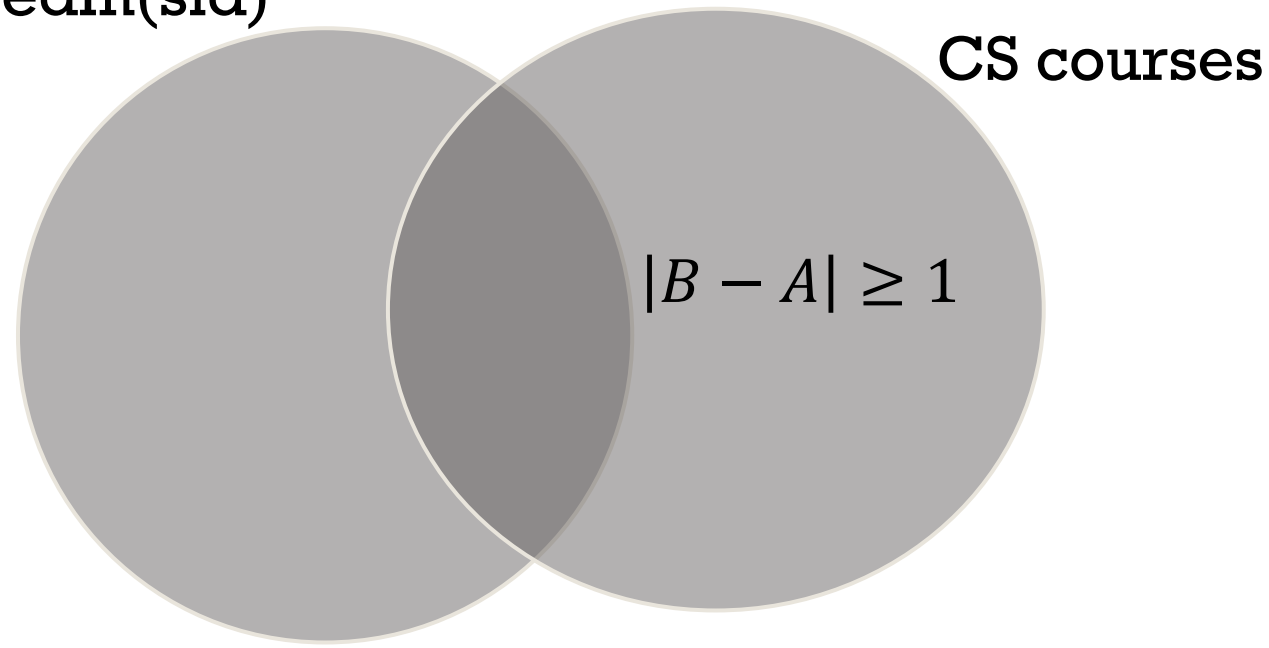
ONLY

```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CoursesEnrolledIn(sid)
                   EXCEPT
                   SELECT cno
                   FROM    CS_Courses)  q) = 0;
```

NOT ALL

Find sid of each student who takes not all CS courses

CoursesEnrolledIn(sid)



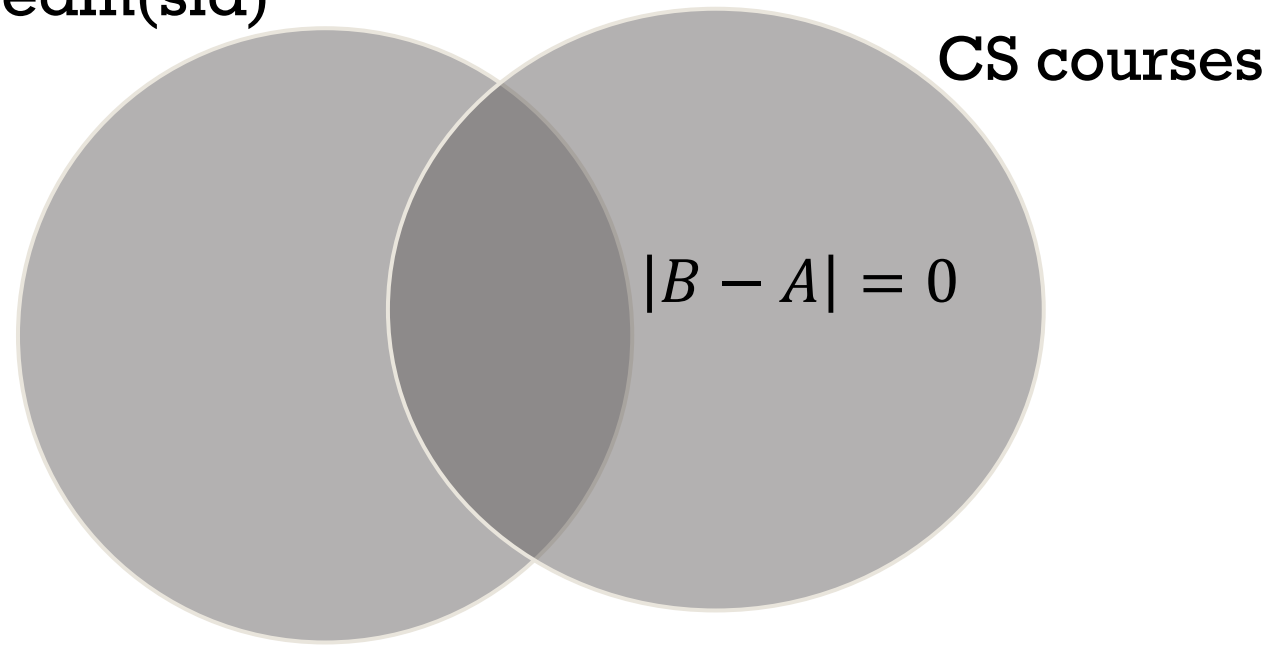
NOT ALL

```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CS_courses
                   EXCEPT
                   SELECT cno
                   FROM   CoursesEnrolledIn(sid)) q) ≥ 1;
```

ALL

Find sid of each student wo takes all CS courses

CoursesEnrolledIn(sid)



ALL

```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CS_courses
                   EXCEPT
                   SELECT cno
                   FROM   CoursesEnrolledIn(sid)) q) = 0;
```

ALL AND ONLY

We have multiple quantifiers: ALL and ONLY

These must both be specified in the WHERE clause using the ALL and ONLY quantifier templates:

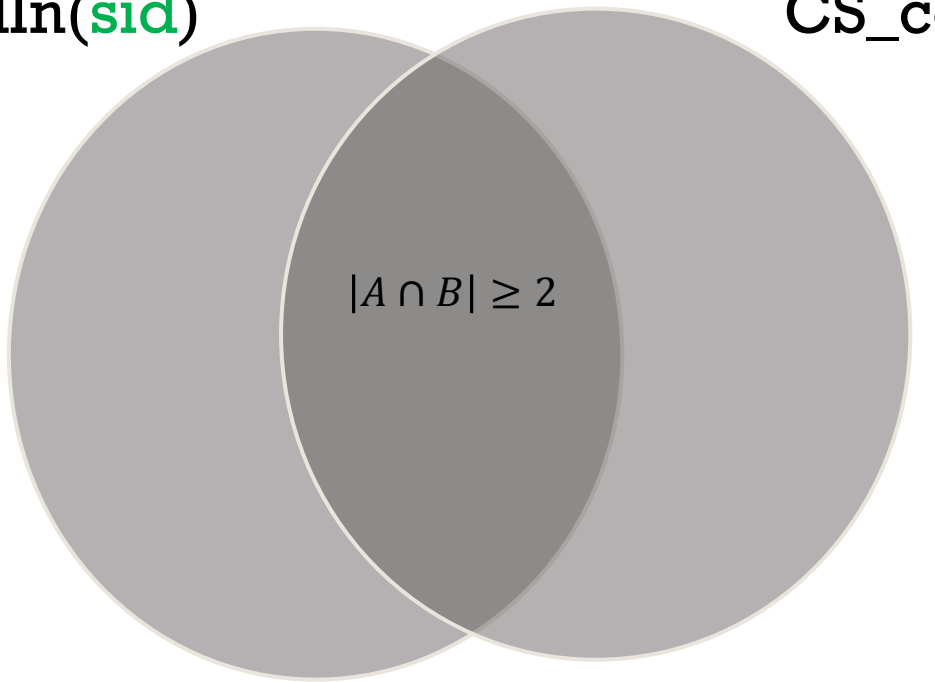
```
SELECT S.Sid
FROM   Student S
WHERE ALL template     AND
          ONLY template
```

AT LEAST TWO

Find sid of each student who takes at least two CS courses

CoursesEnrolledIn(sid)

CS_courses



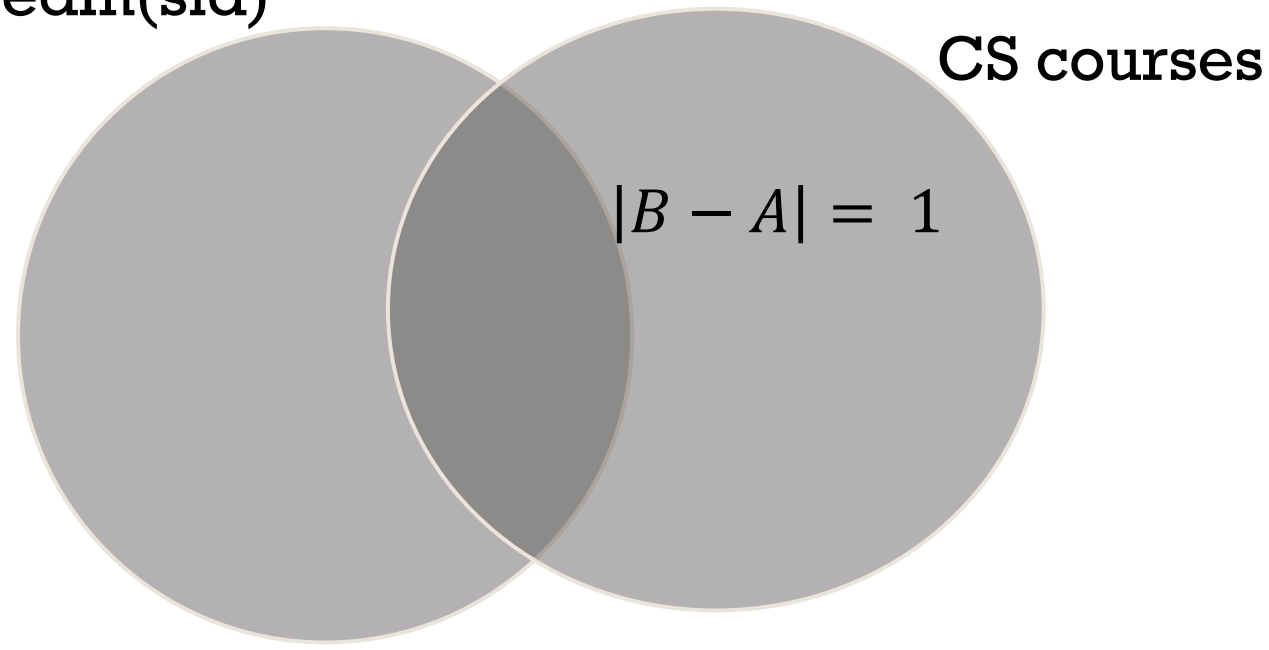
AT LEAST 2

```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CoursesEnrolledIn(sid)
                   INTERSECT
                   SELECT cno
                   FROM    CS_Courses)  q) ≥ 2;
```

ALL BUT ONE

Find sid of each student who takes all but one CS cours

CoursesEnrolledIn(sid)



ALL BUT ONE

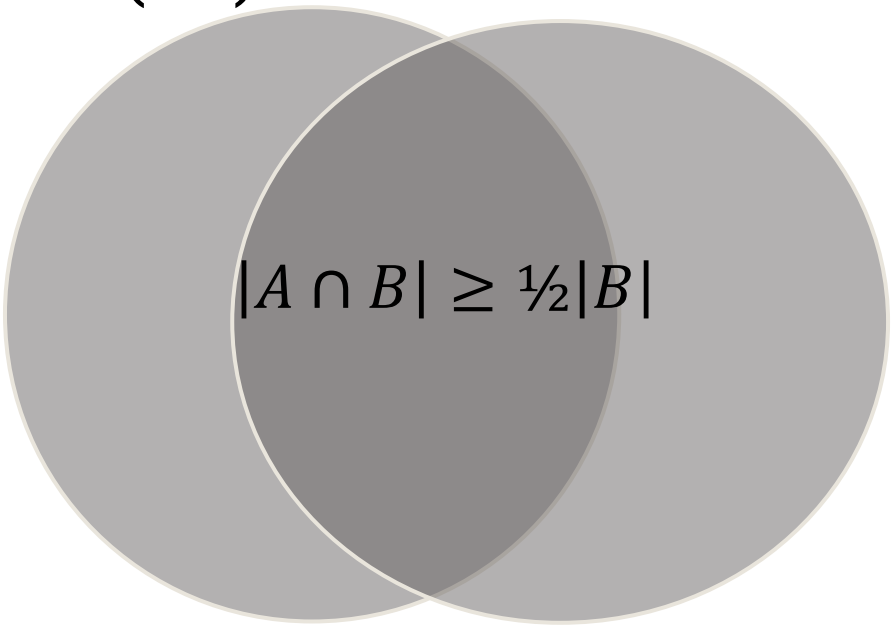
```
SELECT  sid
FROM    Student S
WHERE (SELECT COUNT(1)
          FROM (SELECT  cno
                   FROM    CS_courses
                   EXCEPT
                   SELECT cno
                   FROM   CoursesEnrolledIn(sid)) q) = 1;
```

AT LEAST HALF OF

Find the sid of each student who takes at least half of the CS courses

CoursesEnrolledIn(sid)

CS courses



AT LEAST HALF OF

```
SELECT  sid
FROM    Student S
WHERE  2* (SELECT COUNT(1)
           FROM (SELECT  cno
                 FROM    CoursesEnrolledIn(sid)
                 INTERSECT
                 SELECT cno
                 FROM    CS_Courses)  q) ≥ (SELECT
                                              COUNT (1)
                                            FROM CS_courses)
```

$|A \cap B| \geq ½|B|$

$2\,|A \cap B| \geq |B|$