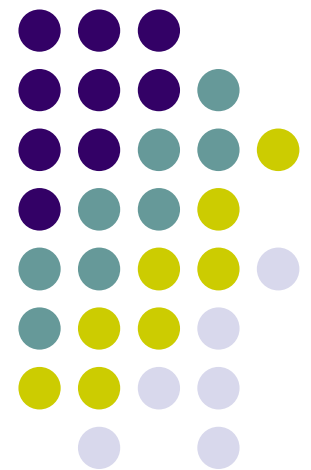


Nested statements

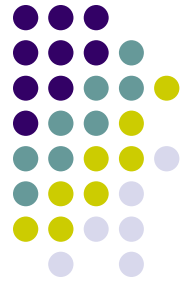
**CIS 331:
Introduction to
Database Systems**





Topics:

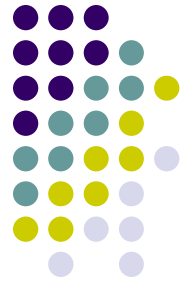
- **Nested statements, subqueries**
- **Indentation**
- **More joins**
- **All**



Nested statements

- To get the names of students who attended CIS616, we have to come up with a two-step strategy:

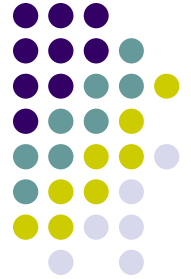
```
SELECT first_name, middle_name, last_name
      FROM students
      WHERE ssn IN
      (
        SELECT student
          FROM students_classes
         WHERE class = 'CIS616'
      )
;
```



Nested statements vs. joins

- Alternatively, the same thing could have been done using a join (although using subqueries is the preferable way to solve this problem):

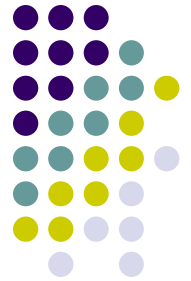
```
SELECT S.first_name, S.middle_name, S.last_name
      FROM students S, students_classes SC
      WHERE S.ssn = SC.student
      AND SC.class = 'CIS616'
;
```



Indentation

- As the queries become more and more complicated, as with all programming, using proper **indentation** saves your time when it comes to debugging:

```
SELECT ...  
    → FROM ...  
        → WHERE ...  
            → AND ...  
;
```



Nested statements

- Get class names for classes with more than 3 registered students:

```
SELECT name
  FROM classes
  WHERE class_code IN
    (
      SELECT class
        FROM students_classes
        GROUP BY class
        HAVING COUNT(student) > 3
    )
;
```



Nested statements vs. joins

- Same thing, using a join:

```
SELECT C.name
      FROM classes C, students_classes SC
      WHERE C.class_code = SC.class
      GROUP BY C.name
      HAVING COUNT(SC.student) > 3
;
```

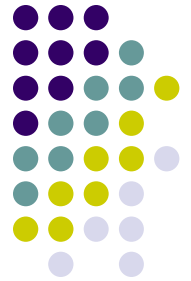


Nested statements

- But if we modify this only slightly, it will not work:

```
SELECT C.name
      FROM classes C, students_classes SC
      WHERE C.class_code = SC.class
      GROUP BY C.class_code
      HAVING COUNT(SC.student) > 3
;
```

- Why?

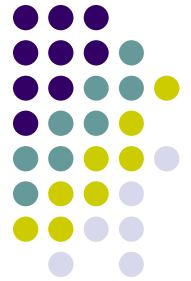


Nested statements

- Get the names of all Vasilis' students.

```
SELECT S.first_name, S.last_name
      FROM students S, students_classes SC,
           professors_classes PC, professors P
      WHERE S.ssn = SC.student
            AND SC.class = PC.class
            AND PC.professor = P.ssn
            AND P.first_name = 'Vasilis'
;

```

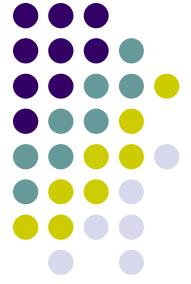


Nested statements

- Get the names of all students that do not attend Vasilis's lectures.
- Your reflex reaction might be to just change = to !=. Do you see why is this wrong?

```
SELECT S.first_name, S.last_name
       FROM students S, students_classes SC,
           professors_classes PC, professors P
       WHERE S.ssn = SC.student
             AND SC.class = PC.class
             AND PC.professor = P.ssn
             AND P.first_name != 'Vasilis'
;

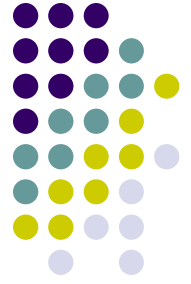
```



Nested statements

- The correct solution is:

```
SELECT first_name, last_name
  FROM students
  WHERE ssn NOT IN
    (
      SELECT SC.student
        FROM students_classes SC,
             professors_classes PC, professors P
        WHERE SC.class = PC.class
              AND PC.professor = P.ssn
              AND P.first_name = 'Vasilis'
    )
;
```



Nested statements

- Or, for those more inclined to play with logic, students that do not attend Vasilis' lectures are all students excluding the ones who do attend his lectures:

```
SELECT first_name, last_name
  FROM students
  MINUS
  (
    SELECT S.first_name, S.last_name
      FROM students S, students_classes SC,
           professors_classes PC, professors P
     WHERE S.ssn = SC.student
           AND SC.class = PC.class
           AND PC.professor = P.ssn
           AND P.first_name = 'Vasilis'
  )
;
```



Nested statements, All

- Find the student with the highest GPA:

```
SELECT first_name, middle_name, last_name
  FROM students
   WHERE ssn IN
     (
       SELECT student
        FROM students_classes
       GROUP BY student
      HAVING AVG(grade) >=
      ALL (
        SELECT AVG(grade)
         FROM students_classes
        GROUP BY student
       )
     )
;
```