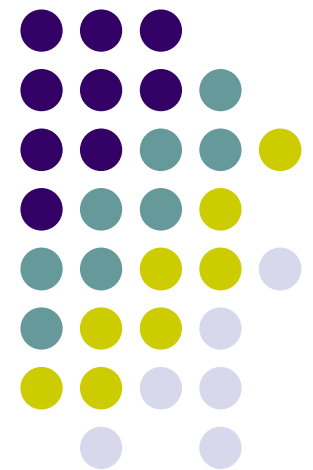


Joins and outer joins

**CIS 331:
Introduction to
Database Systems**





Topics:

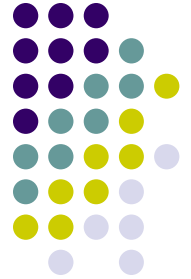
- **JOIN**
- **LEFT OUTER JOIN**
- **RIGHT OUTER JOIN**
- **FULL OUTER JOIN**



Oracle 8i vs. Oracle 9i

- **Oracle9i** has introduced the ANSI standard join syntax (JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, FULL OUTER JOIN). New syntax is SQL92 compliant, and more intuitive. This presentation will outline the old Oracle syntax along with the new one.
- Note that Temple has Oracle 8i **server** (even though your **client** may be Oracle 9i), and that some of the examples will not work.

Join



- Let us see which student is registered for which class:

```
SELECT S.ssn, SC.class
      FROM students S, students_classes SC
      WHERE S.ssn = SC.student
      ORDER BY class
;
```

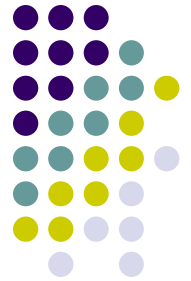


Join

- This is exactly the same as:

```
SELECT S.ssn, SC.class
      FROM students S JOIN students_classes SC
      ON S.ssn = SC.student
      ORDER BY class
;
```

- even though we do not use the word **JOIN** explicitly.



Left outer join

- But for some reason we would also like to include the students who are not registered for any course. This is where an outer join comes handy:

```
SELECT S.ssn, SC.class
      FROM students S LEFT OUTER JOIN students_classes SC
      ON S.ssn = SC.student
      ORDER BY class
;
```



Left outer join

- Or, in Oracle's traditional syntax:

```
SELECT S.ssn, SC.class
      FROM students S, students_classes SC
      WHERE S.ssn = SC.student (+)
      ORDER BY class
;
```

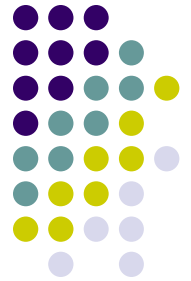
- You are bound to meet old school Oracle programmers who swear by the (+) syntax. So, to be on the safe side, learn both syntaxes and avoid getting into a fight with a senior colleague over a (+).



Right outer join

- What if in addition we would like to see all classes, disregarding if someone has registered for them or not?

```
SELECT S.ssn, SC.class
      FROM students S RIGHT OUTER JOIN
      students_classes SC
      ON S.ssn = SC.student
      ORDER BY class
;
```



Right outer join

- Or, in Oracle's syntax:

```
SELECT S.ssn, SC.class
      FROM students S, students_classes SC
      WHERE S.ssn (+) = SC.student
      ORDER BY class
;
```

- This however does not yield anything new. Why?



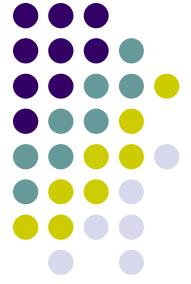
Outer join

- But if we do this:

```
SELECT SC.student, C.class
      FROM students_classes SC RIGHT OUTER JOIN classes C
      ON C.class_code = SC.class
      ORDER BY class_code
;
```

- Or, in Oracle's syntax:

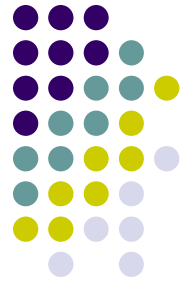
```
SELECT SC.student, C.class_code
      FROM students_classes SC, classes C
      WHERE SC.class (+) = C.class_code
      ORDER BY class_code
;
```



Outer join

- And let us see all students and classes they are taking, plus students who are not taking any classes and classes for which nobody registered.

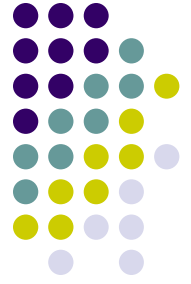
```
SELECT S.ssn, SC.class
      FROM students S, students_classes SC
      WHERE S.ssn = SC.student (+)
UNION
SELECT SC.student, C.class_code
      FROM students_classes SC, classes C
      WHERE SC.class (+) = C.class_code
;
```



Outer join

- A word of caution, though. Let's say you would like to get all students who are taking CIS525 plus all other students, even though they are not taking CIS525:

```
SELECT S.ssn, SC.class
      FROM students S, students_classes SC
      WHERE S.ssn = SC.student (+)
      AND SC.class = 'CIS525'
      ORDER BY SC.class
;
```



Outer join

- We did not get the outer join because AND overrides the outer join condition. This is the correct syntax:

```
SELECT S.ssn, SC.class
      FROM students S, students_classes SC
      WHERE S.ssn = SC.student (+)
      AND SC.class (+) = 'CIS525'
      ORDER BY SC.class
;
```