Principles of Database Systems

Intermediate SQL（1）

主讲教师：

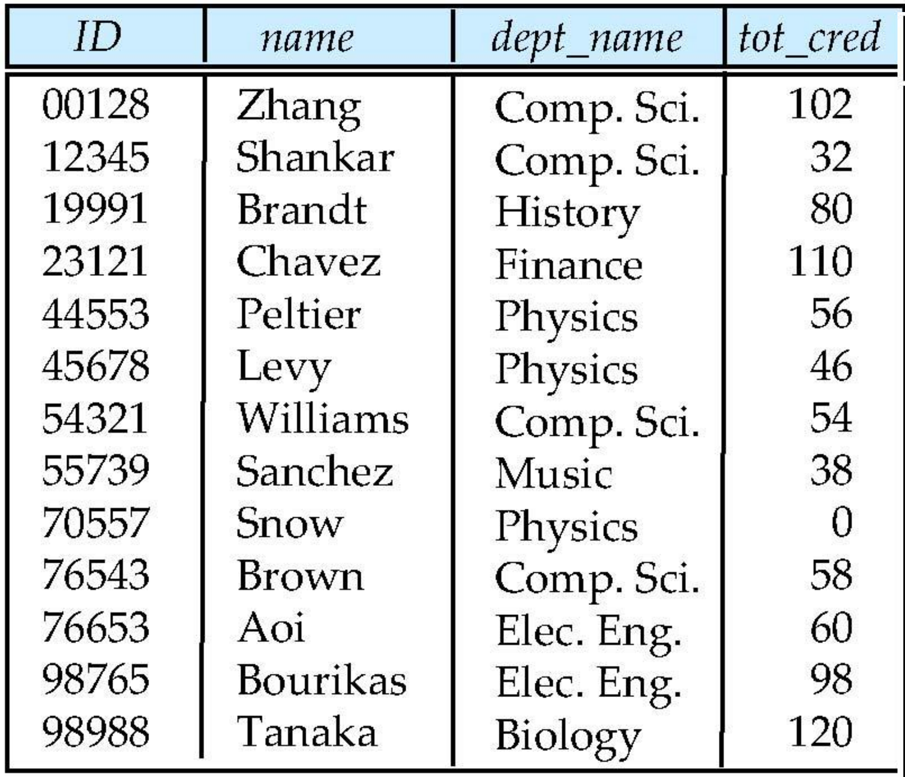
薛昕惟

办公室：信息楼320B

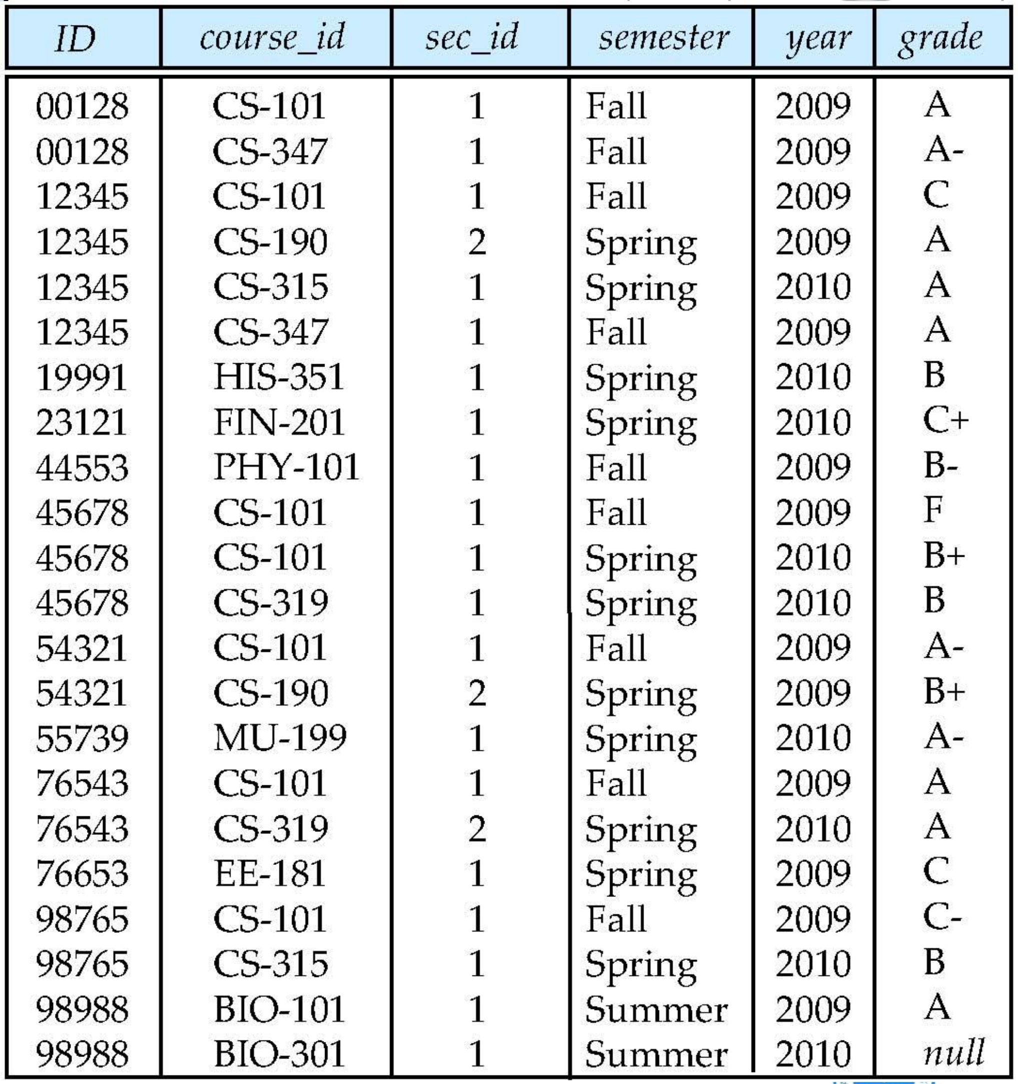
xuexinwei@dlut.edu.cn

Join Expressions

student(ID, name, dept_name, tot_cred)
takes(ID, course_id, sec_id, semester, year, grade)



ID	*name*	*dept_name*	*tot_cred*
00128	Zhang	Comp. Sci.	102
12345	Shankar	Comp. Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp. Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp. Sci.	58
76653	Aoi	Elec. Eng.	60
98765	Bourikas	Elec. Eng.	98
98988	Tanaka	Biology	120



ID	*course_id*	*sec_id*	*semester*	*year*	*grade*
00128	CS-101	1	Fall	2009	A
00128	CS-347	1	Fall	2009	A-
12345	CS-101	1	Fall	2009	C
12345	CS-190	2	Spring	2009	A
12345	CS-315	1	Spring	2010	A
12345	CS-347	1	Fall	2009	A
19991	HIS-351	1	Spring	2010	B
23121	FIN-201	1	Spring	2010	C+
44553	PHY-101	1	Fall	2009	B-
45678	CS-101	1	Fall	2009	F
45678	CS-101	1	Spring	2010	B+
45678	CS-319	1	Spring	2010	B
54321	CS-101	1	Fall	2009	A-
54321	CS-190	2	Spring	2009	B+
55739	MU-199	1	Spring	2010	A-
76543	CS-101	1	Fall	2009	A
76543	CS-319	2	Spring	2010	A
76653	EE-181	1	Spring	2009	C
98765	CS-101	1	Fall	2009	C-
98765	CS-315	1	Spring	2010	B
98988	BIO-101	1	Summer	2009	A
98988	BIO-301	1	Summer	2010	*null*

Review Join and Natural Join

- **select** *
  from student, takes
  where student.ID = takes.ID;

- **select** *
  from student **natural join** takes;

- **select** *
  from student **join** takes **using** (ID);

Join Conditions

- SQL supports another form of join, in which an arbitrary join condition can be specified.
- **select** *
  from student **join** takes **on** student.ID=takes.ID;
- The difference between **join…on** and **natural join** is that the result of join…on has the ID attribute listed twice.

Join Conditions

- **select ***
  from student **join** takes **on** student.ID=takes.ID;

- **select ***
  from student, takes
  where student.ID = takes.ID;

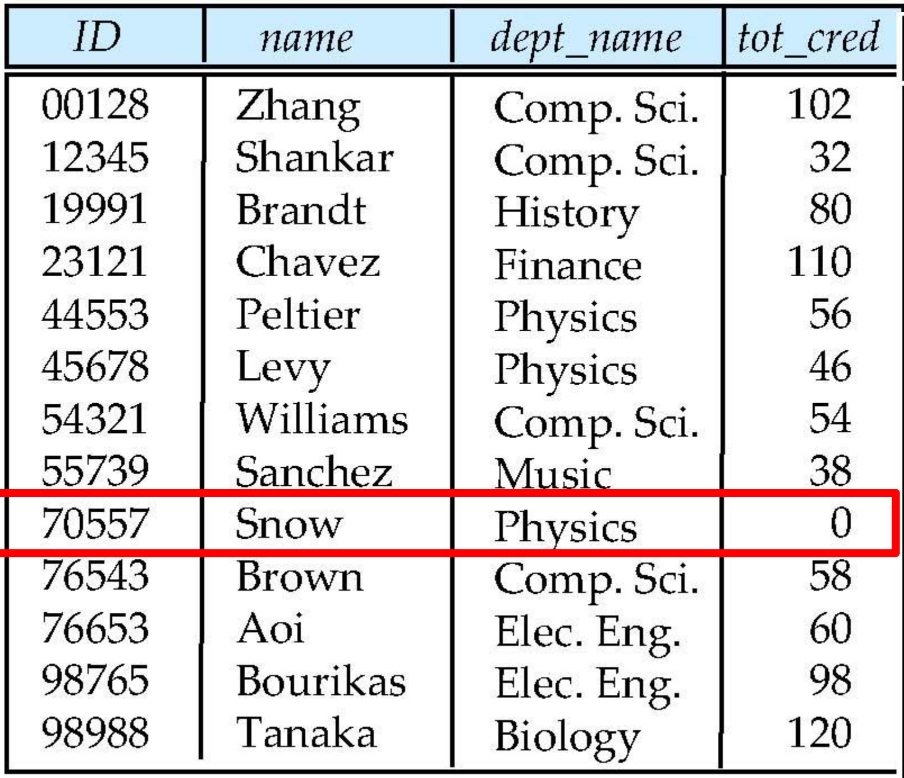
- Good reasons for introducing the **on** condition**:**
  - an SQL query is often more readable if the join condition is specified in the **on clause** and the rest of the conditions appear in the **where clause**
  - In **outer join**, on conditions do behave in a manner different from where conditions

Try…

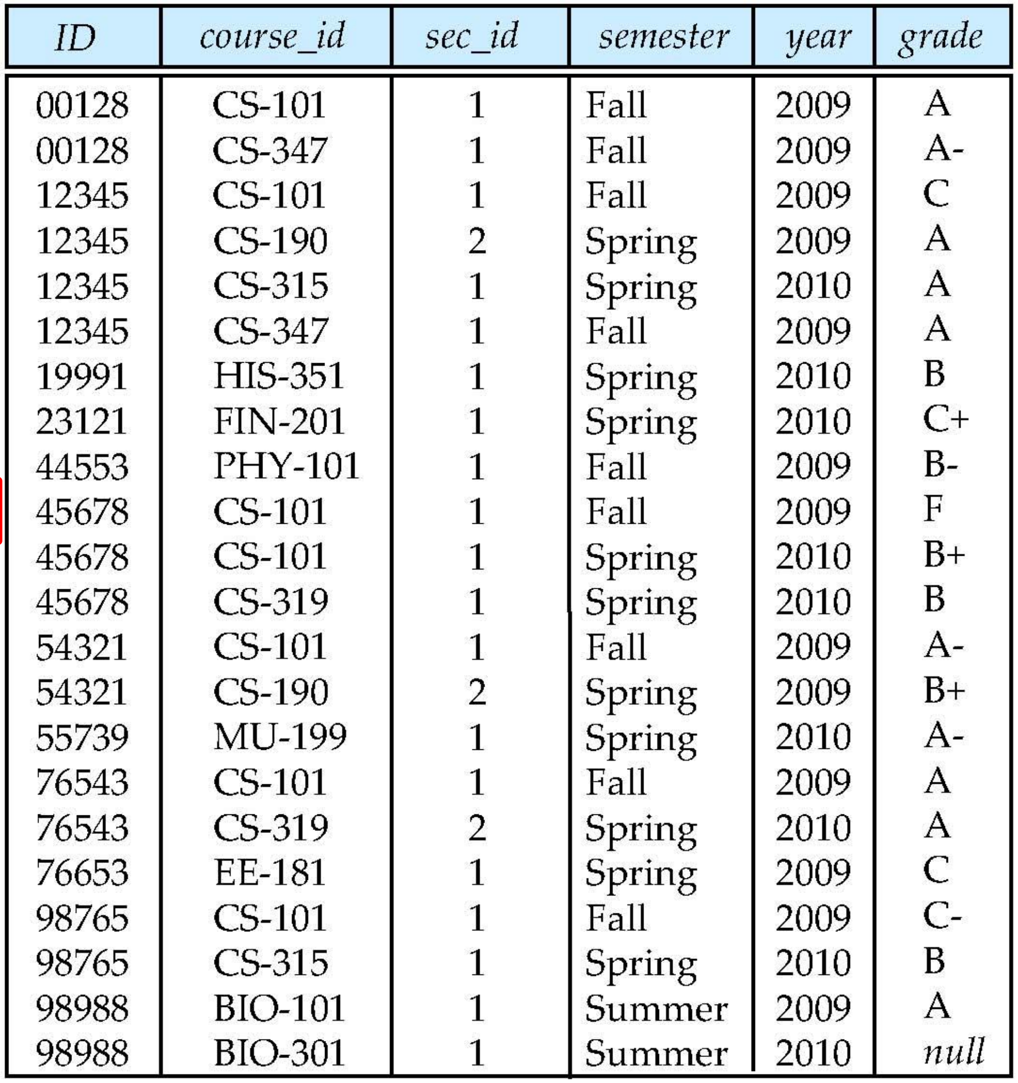
- **For all students**, find their ID, name, dept name, and tot_cred, along with the courses that they have taken.

- Incorrect version
- **select** *
  from student, takes
  where student.ID = takes.ID;

Outer Joins



ID	*name*	*dept_name*	*tot_cred*
00128	Zhang	Comp. Sci.	102
12345	Shankar	Comp. Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp. Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp. Sci.	58
76653	Aoi	Elec. Eng.	60
98765	Bourikas	Elec. Eng.	98
98988	Tanaka	Biology	120



ID	*course_id*	*sec_id*	*semester*	*year*	*grade*
00128	CS-101	1	Fall	2009	A
00128	CS-347	1	Fall	2009	A-
12345	CS-101	1	Fall	2009	C
12345	CS-190	2	Spring	2009	A
12345	CS-315	1	Spring	2010	A
12345	CS-347	1	Fall	2009	A
19991	HIS-351	1	Spring	2010	B
23121	FIN-201	1	Spring	2010	C+
44553	PHY-101	1	Fall	2009	B-
45678	CS-101	1	Fall	2009	F
45678	CS-101	1	Spring	2010	B+
45678	CS-319	1	Spring	2010	B
54321	CS-101	1	Fall	2009	A-
54321	CS-190	2	Spring	2009	B+
55739	MU-199	1	Spring	2010	A-
76543	CS-101	1	Fall	2009	A
76543	CS-319	2	Spring	2010	A
76653	EE-181	1	Spring	2009	C
98765	CS-101	1	Fall	2009	C-
98765	CS-315	1	Spring	2010	B
98988	BIO-101	1	Summer	2009	A
98988	BIO-301	1	Summer	2010	*null*

Observe that student Snow, with ID 70557, has not taken any courses

Outer Joins

- An extension of the join operation that **avoids loss of information**. (避免信息丢失)
- Computes the join and then adds tuples form one relation that does not match tuples in the other relation to the result of the join. (首先进行连接，之后加入一个关系中与另一关系任何元组都不匹配的元组)
- Uses null values.

Left Outer Join

select *
from student **natural left outer join** takes;

select *
from student **left outer join** takes **on** student.ID=takes.ID

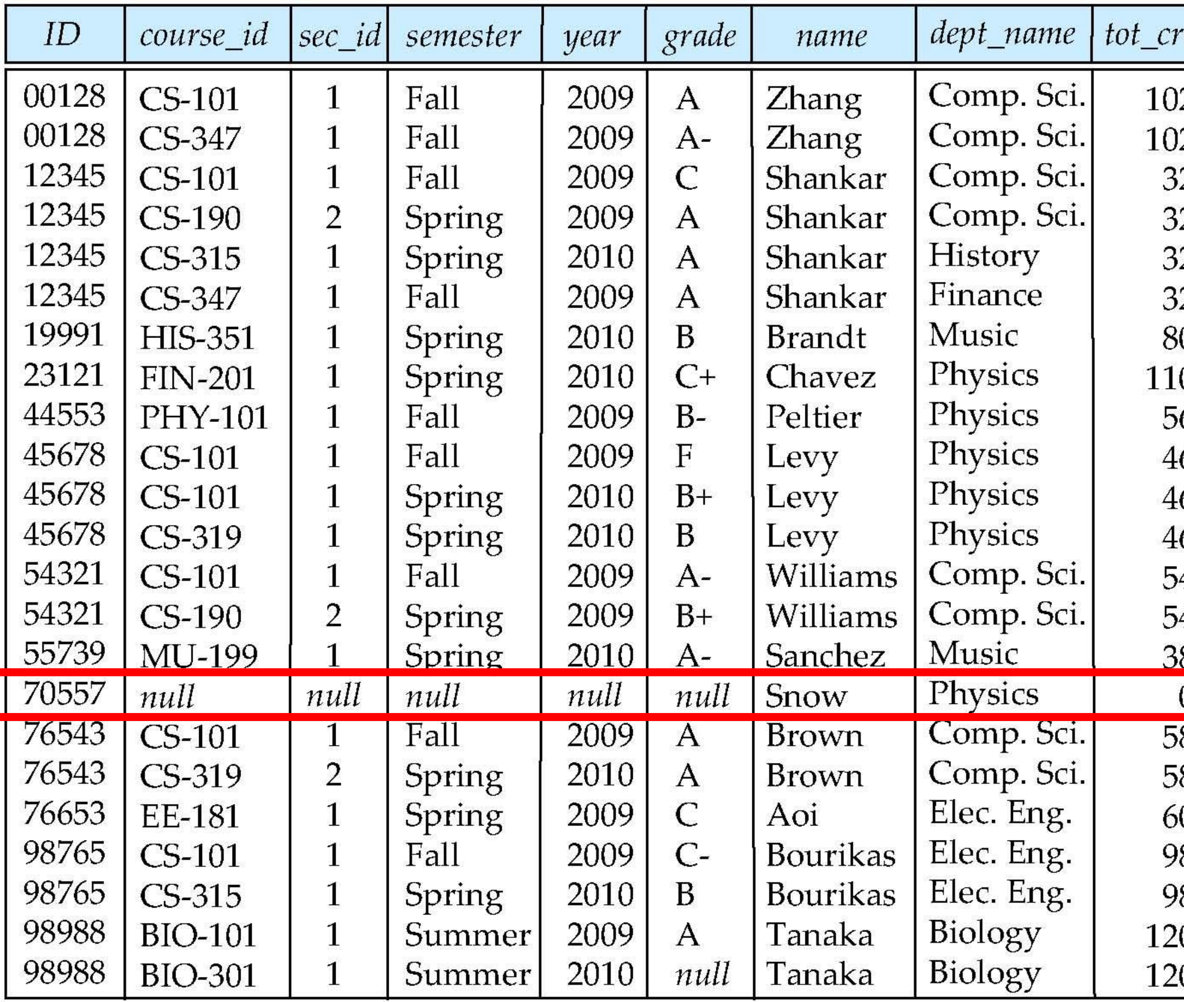
- The **left outer join** preserves tuples only in the relation named before (to the left of) the **left outer join** operation.

ID	*name*	*dept_name*	*tot_cred*	*course_id*	*sec_id*	*semester*	*year*	*grade*
00128	Zhang	Comp. Sci.	102	CS-101	1	Fall	2009	A
00128	Zhang	Comp. Sci.	102	CS-347	1	Fall	2009	A-
12345	Shankar	Comp. Sci.	32	CS-101	1	Fall	2009	C
12345	Shankar	Comp. Sci.	32	CS-190	2	Spring	2009	A
12345	Shankar	History	32	CS-315	1	Spring	2010	A
12345	Shankar	Finance	32	CS-347	1	Fall	2009	A
19991	Brandt	Music	80	HIS-351	1	Spring	2010	B
23121	Chavez	Physics	110	FIN-201	1	Spring	2010	C+
44553	Peltier	Physics	56	PHY-101	1	Fall	2009	B-
45678	Levy	Physics	46	CS-101	1	Fall	2009	F
45678	Levy	Physics	46	CS-101	1	Spring	2010	B+
45678	Levy	Physics	46	CS-319	1	Spring	2010	B
54321	Williams	Comp. Sci.	54	CS-101	1	Fall	2009	A-
54321	Williams	Comp. Sci.	54	CS-190	2	Spring	2009	B+
55739	Sanchez	Music	38	MU-199	1	Spring	2010	A-
70557	Snow	Physics	0	*null*	*null*	*null*	*null*	*null*
76543	Brown	Comp. Sci.	58	CS-101	1	Fall	2009	A
76543	Brown	Comp. Sci.	58	CS-319	2	Spring	2010	A
76653	Aoi	Elec. Eng.	60	EE-181	1	Spring	2009	C
98765	Bourikas	Elec. Eng.	98	CS-101	1	Fall	2009	C-
98765	Bourikas	Elec. Eng.	98	CS-315	1	Spring	2010	B
98988	Tanaka	Biology	120	BIO-101	1	Summer	2009	A
98988	Tanaka	Biology	120	BIO-301	1	Summer	2010	*null*

Right Outer Join

select *
from takes **right outer join** student **on** student.ID=takes.ID

- The **right outer join** preserves tuples only in the relation named after (to the right of) the **right outer join operation.**



ID	*course_id*	*sec_id*	*semester*	*year*	*grade*	*name*	*dept_name*	*tot_cr*
00128	CS-101	1	Fall	2009	A	Zhang	Comp. Sci.	102
00128	CS-347	1	Fall	2009	A-	Zhang	Comp. Sci.	102
12345	CS-101	1	Fall	2009	C	Shankar	Comp. Sci.	32
12345	CS-190	2	Spring	2009	A	Shankar	Comp. Sci.	32
12345	CS-315	1	Spring	2010	A	Shankar	History	32
12345	CS-347	1	Fall	2009	A	Shankar	Finance	32
19991	HIS-351	1	Spring	2010	B	Brandt	Music	80
23121	FIN-201	1	Spring	2010	C+	Chavez	Physics	110
44553	PHY-101	1	Fall	2009	B-	Peltier	Physics	56
45678	CS-101	1	Fall	2009	F	Levy	Physics	46
45678	CS-101	1	Spring	2010	B+	Levy	Physics	46
45678	CS-319	1	Spring	2010	B	Levy	Physics	46
54321	CS-101	1	Fall	2009	A-	Williams	Comp. Sci.	54
54321	CS-190	2	Spring	2009	B+	Williams	Comp. Sci.	54
55739	MU-199	1	Spring	2010	A-	Sanchez	Music	38
70557	*null*	*null*	*null*	*null*	*null*	Snow	Physics	0
76543	CS-101	1	Fall	2009	A	Brown	Comp. Sci.	58
76543	CS-319	2	Spring	2010	A	Brown	Comp. Sci.	58
76653	EE-181	1	Spring	2009	C	Aoi	Elec. Eng.	60
98765	CS-101	1	Fall	2009	C-	Bourikas	Elec. Eng.	98
98765	CS-315	1	Spring	2010	B	Bourikas	Elec. Eng.	98
98988	BIO-101	1	Summer	2009	A	Tanaka	Biology	120
98988	BIO-301	1	Summer	2010	*null*	Tanaka	Biology	120

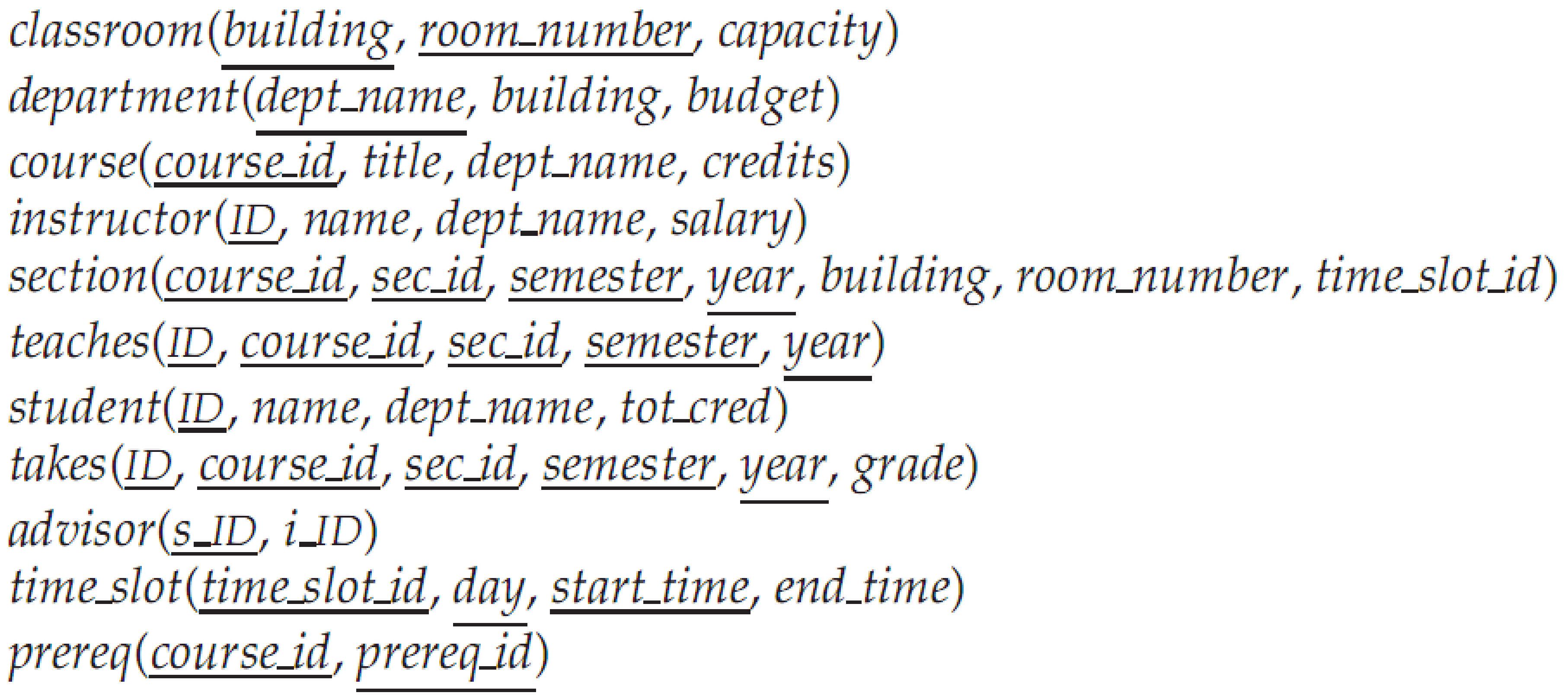
Full Outer Join

- The **full outer join** preserves tuples in both relations.
- Display a list of all students in the Comp. Sci. department, along with the course sections, if any, that they have taken in Spring 2009; all course sections from Spring 2009 must be displayed, even if no student from the Comp. Sci. department has taken the course section.

```
select *
from (select *
        from student
        where dept name= ’Comp. Sci’)
        natural full outer join
       (select *
        from takes
        where semester = ’Spring’ and year = 2009);
```

Try…

- Find all students who have not taken a course



classroom(building, room_number, capacity)
department(dept_name, building, budget)
course(course_id, title, dept_name, credits)
instructor(ID, name, dept_name, salary)
section(course_id, sec_id, semester, year, building, room_number, time_slot_id)
teaches(ID, course_id, sec_id, semester, year)
student(ID, name, dept_name, tot_cred)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(s_ID, i_ID)
time_slot(time_slot_id, day, start_time, end_time)
prereq(course_id, prereq_id)

Try…

- Find all students who have not taken a course
  - **select** ID
    from student **left outer join** takes **on** student.ID=takes.ID
    where course_id **is** null;

Comparison

- **select ***

  from student **left outer join** takes **on** student.ID= takes.ID**;**

- **select ***

  from student **left outer join** takes **on** true

  where student.ID= takes.ID**;**

Joined Relations

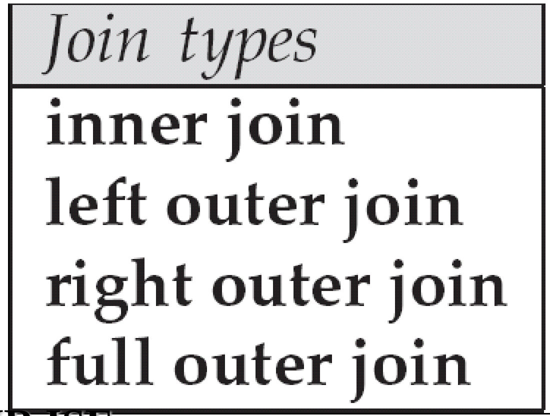
- The default **join** type, when the join clause is used without the outer prefix is the **inner join.**

- **select** *
  from student ***join*** takes ***on*** student.ID=takes.ID

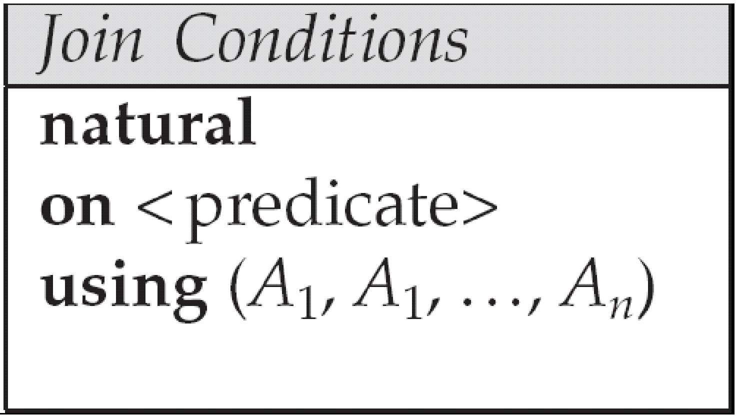
- **select** *
  from student ***inner join*** takes ***on*** student.ID=takes.ID**;**

Joined Relations

- **Join operations** take two relations and return as a result another relation.
- These additional operations are typically used as subquery expressions in the **from** clause
- **Join condition** (连接条件)– defines which tuples in the two relations match, and what attributes are present in the result of the join.
- **Join type**(连接类型) – defines how tuples in each relation that do not match any tuple in the other relation (based on the join condition) are treated.



Join types
inner join
left outer join
right outer join
full outer join



Join Conditions
natural
on <predicate>
using $(A_1, A_1, \ldots, A_n)$

Try…

- Find the information of all courses, along with their prerequisite course ID.

classroom(building, room_number, capacity)
department(dept_name, building, budget)
course(course_id, title, dept_name, credits)
instructor(ID, name, dept_name, salary)
section(course_id, sec_id, semester, year, building, room_number, time_slot_id)
teaches(ID, course_id, sec_id, semester, year)
student(ID, name, dept_name, tot_cred)
takes(ID, course_id, sec_id, semester, year, grade)
advisor(s_ID, i_ID)
time_slot(time_slot_id, day, start_time, end_time)
prereq(course_id, prereq_id)

Try…

- Find the information of all courses, along with their prerequisite course ID.

  – **select** *
    from course **left outer join** prereq **on** course.course_id=prereq.course_id