

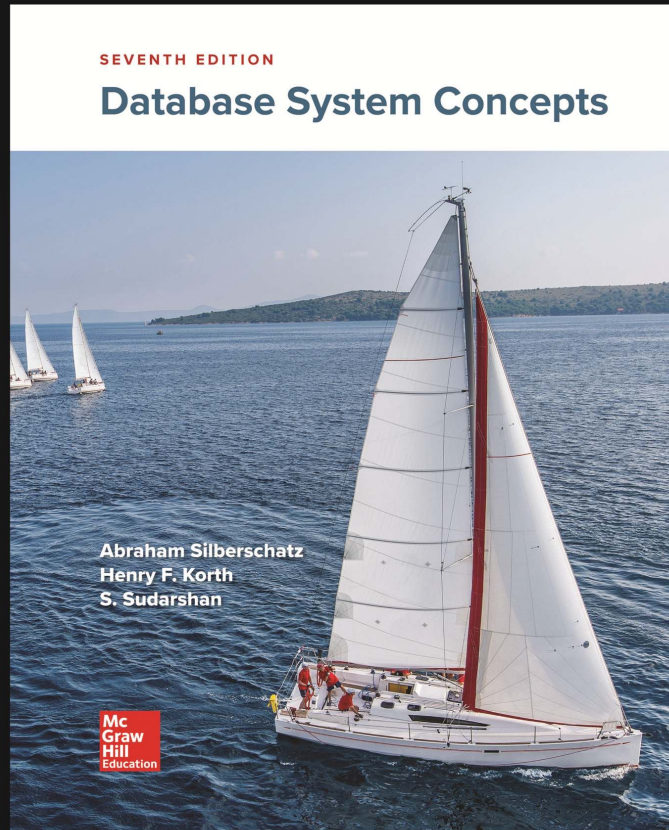


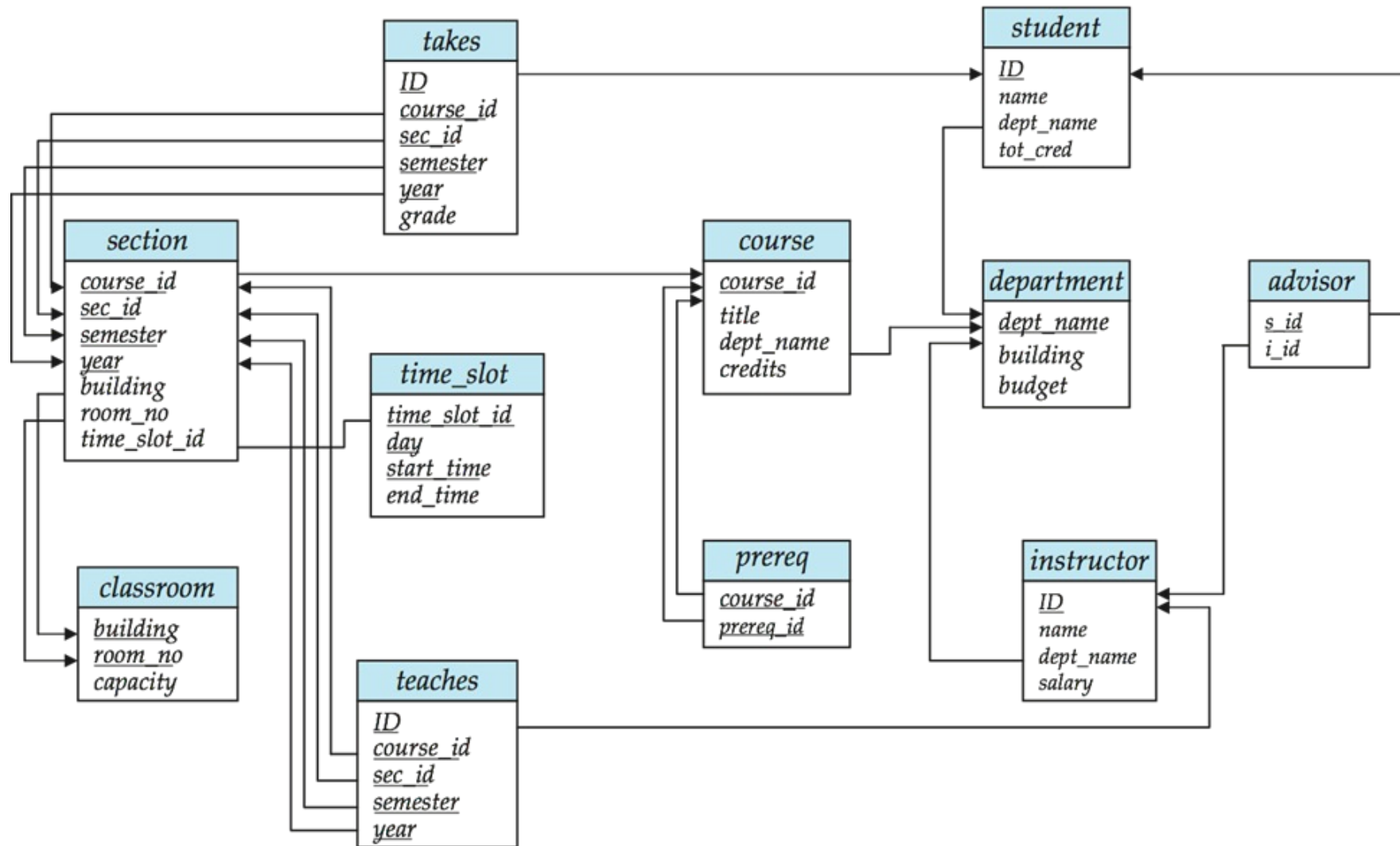
IF2240 – Basis Data SQL (Part 2)

Sumber

Silberschatz, Korth, Sudarshan: "Database System Concepts", 7th Edition

- Chapter 3 : Introduction to SQL





Aggregate Functions

These functions operate on the multiset of values of a column of a relation, and return a value

avg: average value

min: minimum value

max: maximum value

sum: sum of values

count: number of values

Aggregate Functions Examples

Find the average salary of instructors in the Computer Science department

- `select avg (salary)`
`from instructor`
`where dept_name= 'Comp. Sci.';`

Find the total number of instructors who teach a course in the Spring 2018 semester

- `select count (distinct ID)`
`from teaches`
`where semester = 'Spring' and year = 2018;`

Find the number of tuples in the *course* relation

- `select count (*)`
`from course;`

Aggregate Functions – Group By

Group By: Group tuple that have the same values into summary rows

- Often used with aggregate functions

Find the average salary of instructors in each department

- `select dept_name,
 avg (salary) as avg_salary
from instructor
group by dept_name;`

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
76766	Crick	Biology	72000
45565	Katz	Comp. Sci.	75000
10101	Srinivasan	Comp. Sci.	65000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000
12121	Wu	Finance	90000
76543	Singh	Finance	80000
32343	El Said	History	60000
58583	Califieri	History	62000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
22222	Einstein	Physics	95000

<i>dept_name</i>	<i>avg_salary</i>
Biology	72000
Comp. Sci.	77333
Elec. Eng.	80000
Finance	85000
History	61000
Music	40000
Physics	91000

Aggregation (Cont.)

Attributes in **select** clause outside of aggregate functions must appear in **group by** list

- */* erroneous query */*
select *dept_name, ID, avg (salary)*
from *instructor*
group by *dept_name;*

Aggregate Functions – Having Clause

HAVING clause was added to SQL because the WHERE keyword could not be used with **aggregate functions**

Find the names and average salaries of all departments whose average salary is greater than 42000

```
select dept_name, avg (salary) as avg_salary  
from instructor  
group by dept_name  
having avg (salary) > 42000;
```

Note: predicates in the **having** clause are applied after the formation of groups whereas predicates in the **where** clause are applied before forming groups

How About Some More Queries?

6. Find the highest salary earned by instructors from Comp. Sci. Department.
7. Find the enrollment of each section that was offered in Autumn 2009.
8. Find all departments with more than 1000 student enrollments, together with the number of students for each of them. The list should be ordered by dept_name
9. Find all instructors earning the highest salary (there may be more than one with the same salary)

Joined Relations

Join operations take two relations and return as a result another relation.

A join operation is a Cartesian product which requires that tuples in the two relations match (under some condition). It also specifies the attributes that are present in the result of the join

The join operations are typically used as subquery expressions in the **from** clause

Three types of joins:

- Natural join
- Inner join
- Outer join

Natural Join in SQL

Natural join matches tuples with the same values for all common attributes, and retains only one copy of each common column.

List the names of instructors along with the course ID of the courses that they taught

- `select name, course_id`
`from students, takes`
`where student.ID = takes.ID;`

Same query in SQL with “natural join” construct

- `select name, course_id`
`from student natural join takes;`

Natural Join in SQL (Cont.)

The **from** clause can have multiple relations combined using natural join:

```
select   $A_1, A_2, \dots, A_n$   
from     $r_1$  natural join  $r_2$  natural join .. natural join  $r_n$   
where    $P$  ;
```

Student Relation

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>tot_cred</i>
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

Takes Relation

<i>ID</i>	<i>course_id</i>	<i>sec_id</i>	<i>semester</i>	<i>year</i>	<i>grade</i>
00128	CS-101	1	Fall	2017	A
00128	CS-347	1	Fall	2017	A-
12345	CS-101	1	Fall	2017	C
12345	CS-190	2	Spring	2017	A
12345	CS-315	1	Spring	2018	A
12345	CS-347	1	Fall	2017	A
19991	HIS-351	1	Spring	2018	B
23121	FIN-201	1	Spring	2018	C+
44553	PHY-101	1	Fall	2017	B-
45678	CS-101	1	Fall	2017	F
45678	CS-101	1	Spring	2018	B+
45678	CS-319	1	Spring	2018	B
54321	CS-101	1	Fall	2017	A-
54321	CS-190	2	Spring	2017	B+
55739	MU-199	1	Spring	2018	A-
76543	CS-101	1	Fall	2017	A
76543	CS-319	2	Spring	2018	A
76653	EE-181	1	Spring	2017	C
98765	CS-101	1	Fall	2017	C-
98765	CS-315	1	Spring	2018	B
98988	BIO-101	1	Summer	2017	A
98988	BIO-301	1	Summer	2018	<i>null</i>

*student natural
join takes*

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>tot_cred</i>	<i>course_id</i>	<i>sec_id</i>	<i>semester</i>	<i>year</i>	<i>grade</i>
00128	Zhang	Comp. Sci.	102	CS-101	1	Fall	2017	A
00128	Zhang	Comp. Sci.	102	CS-347	1	Fall	2017	A-
12345	Shankar	Comp. Sci.	32	CS-101	1	Fall	2017	C
12345	Shankar	Comp. Sci.	32	CS-190	2	Spring	2017	A
12345	Shankar	Comp. Sci.	32	CS-315	1	Spring	2018	A
12345	Shankar	Comp. Sci.	32	CS-347	1	Fall	2017	A
19991	Brandt	History	80	HIS-351	1	Spring	2018	B
23121	Chavez	Finance	110	FIN-201	1	Spring	2018	C+
44553	Peltier	Physics	56	PHY-101	1	Fall	2017	B-
45678	Levy	Physics	46	CS-101	1	Fall	2017	F
45678	Levy	Physics	46	CS-101	1	Spring	2018	B+
45678	Levy	Physics	46	CS-319	1	Spring	2018	B
54321	Williams	Comp. Sci.	54	CS-101	1	Fall	2017	A-
54321	Williams	Comp. Sci.	54	CS-190	2	Spring	2017	B+
55739	Sanchez	Music	38	MU-199	1	Spring	2018	A-
76543	Brown	Comp. Sci.	58	CS-101	1	Fall	2017	A
76543	Brown	Comp. Sci.	58	CS-319	2	Spring	2018	A
76653	Aoi	Elec. Eng.	60	EE-181	1	Spring	2017	C
98765	Bourikas	Elec. Eng.	98	CS-101	1	Fall	2017	C-
98765	Bourikas	Elec. Eng.	98	CS-315	1	Spring	2018	B
98988	Tanaka	Biology	120	BIO-101	1	Summer	2017	A
98988	Tanaka	Biology	120	BIO-301	1	Summer	2018	null

Dangerous in Natural Join

Beware of unrelated attributes with same name which get equated incorrectly

Example -- List the names of students instructors along with the titles of courses that they have taken

- Correct version

```
select name, title
from student natural join takes, course
where takes.course_id = course.course_id;
```

- Incorrect version

```
select name, title
from student natural join takes natural join course;
```

- This query omits all (student name, course title) pairs where the student takes a course in a department other than the student's own department.
- The correct version (above), correctly outputs such pairs.

Outer Join

An extension of the join operation that avoids loss of information.

Computes the join and then adds tuples from one relation that does not match tuples in the other relation to the result of the join.

Uses *null* values.

Three forms of outer join:

- left outer join
- right outer join
- full outer join

Outer Join Examples

Relation *course*

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>
BIO-301	Genetics	Biology	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3

Relation *prereq*

<i>course_id</i>	<i>prereq_id</i>
BIO-301	BIO-101
CS-190	CS-101
CS-347	CS-101

Observe that

course information is missing CS-347

prereq information is missing CS-315

Left Outer Join

course natural left outer join *prereq*

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>	<i>prereq_id</i>
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-315	Robotics	Comp. Sci.	3	<i>null</i>

In relational algebra: *course* ⋈ *prereq*

Right Outer Join

course natural right outer join *prereq*

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>	<i>prereq_id</i>
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-347	<i>null</i>	<i>null</i>	<i>null</i>	CS-101

In relational algebra: *course* ⋈ *prereq*

Full Outer Join

course natural full outer join *prereq*

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>	<i>prereq_id</i>
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-315	Robotics	Comp. Sci.	3	<i>null</i>
CS-347	<i>null</i>	<i>null</i>	<i>null</i>	CS-101

In relational algebra: *course* ⋈ *prereq*

Joined Types and Conditions

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 – defines which tuples in the two relations match.

- **natural**
- **on** <predicate>
- **using** (A_1, A_2, \dots, A_n)

Join type – defines how tuples in each relation that do not match any tuple in the other relation (based on the join condition) are treated.

- **inner join**
- **left outer join**
- **right outer join**
- **full outer join**

Joined Relations – Examples

course natural right outer join *prereq*

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>	<i>prereq_id</i>
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-347	<i>null</i>	<i>null</i>	<i>null</i>	CS-101

course full outer join *prereq* using (*course_id*)

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>	<i>prereq_id</i>
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-315	Robotics	Comp. Sci.	3	<i>null</i>
CS-347	<i>null</i>	<i>null</i>	<i>null</i>	CS-101

Joined Relations – Examples

course inner join *prereq* on
course.course_id = *prereq.course_id*

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>	<i>prereq_id</i>	<i>course_id</i>
BIO-301	Genetics	Biology	4	BIO-101	BIO-301
CS-190	Game Design	Comp. Sci.	4	CS-101	CS-190

What is the difference between the above, and a natural join?

course left outer join *prereq* on
course.course_id = *prereq.course_id*

<i>course_id</i>	<i>title</i>	<i>dept_name</i>	<i>credits</i>	<i>prereq_id</i>	<i>course_id</i>
BIO-301	Genetics	Biology	4	BIO-101	BIO-301
CS-190	Game Design	Comp. Sci.	4	CS-101	CS-190
CS-315	Robotics	Comp. Sci.	3	<i>null</i>	<i>null</i>