课程名称：数据库系统

基本SQL

单 位：重庆大学计算机学院

计算机内文件的操作

- 数据存储在文件中，程序语言如何实现数据访问？

主要目标

- SQL语言包含的类型和功能
- 基本的SQL查询语言。

- 通过本节学习，可以用SQL语言实现数据库设计，进行基本查询

思考问题

- 关系模型应该提供的操作有哪些？
- 关系代数

一 SQL数据定义语言DDL

1.1 定义数据结构

SQL如何实际建立一个关系模式结构？

一个 create table 语句的能力有多强(包含哪些部分)?

定义属性特征＋说明语义约束。

```
create table department
    (dept_name        varchar(20),
     building         varchar (15),
     budget           numeric (12,2),
     primary key (dept_name));

create table course
    (course_id        varchar (7),
     title            varchar (50),
     dept_name        varchar (20),
     credits          numeric (2,0),
     primary key (course_id),
     foreign key (dept_name) references department);
```

```
create table instructor
    (ID               varchar (5),
     name             varchar (20) not null,
     dept_name        varchar (20),
     salary           numeric (8,2),
     primary key (ID),
     foreign key (dept_name) references department);
create table section
    (course_id        varchar (8),
     sec_id           varchar (8),
     semester         varchar (6),
     year             numeric (4,0),
     building         varchar (15),
     room_number      varchar (7),
     time_slot_id     varchar (4),
     primary key (course_id, sec_id, semester, year),
     foreign key (course_id) references course);

create table teaches
    (ID               varchar (5),
     course_id        varchar (8),
     sec_id           varchar (8),
     semester         varchar (6),
     year             numeric (4,0),
     primary key (ID, course_id, sec_id, semester, year),
     foreign key (course_id, sec_id, semester, year)
references section,
     foreign key (ID) references instructor);
```

属性说明　值类型约束　非空约束　主码(主键)约束　外码(外键)约束

案例1　大学数据库的部分 SQL 数据定义

1.2 定义主键约束和外键约束

解释这些关系模式的主码约束和外码约束的具体含义?

主键含一个属性

外键(参照约束)仅包含一个属性

```
create table instructor
        (ID         varchar (5),
        name        varchar (20) not null,
        dept_name   varchar (20),
        salary      numeric (8,2),
        primary key (ID),
        foreign key (dept_name) references department);
create table section
        (course_id      varchar (8),
        sec_id          varchar (8),
        semester        varchar (6),
        year            numeric (4,0),
        building        varchar (15),
        room_number     varchar (7),
        time_slot_id    varchar (4),
        primary key (course_id, sec_id, semester, year),
        foreign key (course_id) references course);

create table teaches
        (ID         varchar (5),
        course_id   varchar (8),
        sec_id      varchar (8),
        semester    varchar (6),
        year        numeric (4,0),
        primary key (ID, course_id, sec_id, semester, year),
        foreign key (course_id, sec_id, semester, year)
references section,
        foreign key (ID) references instructor);
```

主键包含多个属性

```
create table department
   (dept_name      varchar(20),
   building        varchar (15),
   budget          numeric (12,2),
   primary key (dept_name));

create table course
   (course_id      varchar (7),
   title           varchar (50),
   dept_name       varchar (20),
   credits         numeric (2,0),
   primary key (course_id),
   foreign key (dept_name) references department);
```

外键包含三个属性且它包含两个外键

案例1 大学数据库的部分 SQL 数据定义

1.3 创建数据库的实例(略讲，上机学习)

- CREATE DATABASE 数据库名
- ON
- (NAME = 逻辑文件名,
- FILENAME= ‘mdf数据文件路径’,
- SIZE = 10 MB, /*数据文件初始大小*/
- MAXSIZE = 20 MB, /*数据文件最大值*/
- FILEGROWTH =2 MB), /*数据文件增长值*/

- LOG ON /*创建日志文件，可省略*/
- (NAME = 日志文件名,
- FILENAME= ‘日志文件名’,
- SIZE = 10 MB,
- MAXSIZE = 20MB,
- FILEGROWTH =10%)
- GO

命令格式

- CREATE DATABASE Bank
- ON
- (NAME = Bank_data1,
- FILENAME= ‘D:\Bank_data1.mdf’,
- SIZE = 20 MB,
- MAXSIZE =100MB,
- FILEGROWTH =2 MB)

- LOG ON
- (NAME = Bank_log1,
- FILENAME= ‘D:\Bank_log1.ndf’,
- SIZE = 4 MB,
- MAXSIZE = 25MB,
- FILEGROWTH =1MB)
- GO

案例2 一个实例

(讲解) create database到底做了什么？

1. 产生了一个数据库（空仓库，仅包括系统数据字典）
2. 初始库小，数据增长需要时才增大库空间
3. 同时，还产生了一个日志存放的空仓库（备份恢复用）
4. 还涉及到物理设计工作：库放在何位置、库大小、库增量
   而且日志仓库位置可用与数据仓库位置不同(保证安全)!

二 SQL数据查询语言QL

2.1 基本关系运算的SQL实现

$\Pi_{name}(\sigma_{depat\_name='Comp.Sci' \text{ and } salaty>70000}(instuctor))$

投影

选择

如何理解这些查询语句，SQL为何为描述性语言？

- **(P.35-36 SQL语句)**

```
Select name
From instuctor
Where depat_name = 'Comp.Sci' and salary > 70000;

Select name, instructor.dept_name, building
From instructor, department
Where instructor.dept_name = department.dept_name;
```

'自然连接'
(等值连接)

$\Pi_{name,\ instructor.dept\_name,\ building}(\sigma_{instructor.dept\_name = department.dept\_name}(instructor \times department))$

```
(Select course_id
From section
Where semster = 'Fall' and 'year' = 2009)
```

并

```
Union
(Select course_id
From section
Where semster = 'Spring' and 'year' = 2010);
```

p.44：差 **Except**，交 **Intersact**

[案例3.a] 关系模式上的数据查询例子

SQL如何表示基本关系运算?

$(\Pi_{course\_id}(\sigma_{semster='Fall' \text{ and } 'year'=2009}(section))) \cup (\Pi_{course\_id}(\sigma_{semster='Spring' \text{ and } 'year'=2010}(section)))$

更名

- **(补充案例)**

```
Select name as '教师姓名'      / as instuctor_name
From instuctor;
```

as可用于属性和表(参P.40)

```
Select name, instructor.dept_name, building
From instructor, department;
```

[案例3.b] 更简单的数据查询例子

迪卡儿积

$\Pi_{name,\ instructor.dept\_name,\ building}(instructor \times department)$

2.2 Where子句的重要作用

Where子语句在关系代数操作上的作用?

1)关系记录的筛选
2)两关系间的连接

选择

• **(P.35-36 SQL语句)**

```
Select name
From instuctor
Where depat_name = 'Comp.Sci' and salaty > 70000;

Select name, instructor.dept_name, building
From instructor, department
Where instructor.dept_name = department.dept_name;

(Select course_id
From section
Where semster = 'Fall' and 'year' = 2009)
Union
(Select course_id
From section
Where semster = 'Spring' and 'year' = 2010);
```

'自然连接'
等值连接

注:两关系连接时可以使用大于、小于等比较符号

等效p.38

注:自然连接Natural join与迪卡儿积X两点最大不同(比较p. 36图36p. 38图38): 并
1）仅包含符号连接条件的元组
2）连接属性仅出现一次

差 p.44 Except

[案例3.a] 关系模式上的数据查询例子

SQL如何实现多个关系上的数据查询?

以及后面介绍的嵌套子查询

迪卡儿积

• **(补充案例)**

```
Select name
From instuctor;

Select name, instructor.dept_name, building
From instructor, department;
```

[案例3.b] 更简单的数据查询例子

三 SQL的数据查询能力

3.1 聚集函数

SQL聚集函数的使用方法？

(教材P.46)
平均值avg
最小值min
值大值max
总和sum
计数count

$g_{avg\ (salary)}(\sigma_{dept\_name='Comp.\ Sci.'}(instuctor))$

```
select avg (salary)
from instructor                          仅计算一个系的平均工资
where dept_name= 'Comp. Sci.';

select count (distinct ID)               计数前先去除重复元组
from teaches
where semester ='Spring' and year=2010;

select count (*)                         *代表选择所有属性
from course;

select dept_name, avg (salary) as avg_salary
from instructor                          第1个为平均工资显示部门名
group by dept_name;                      第2个用于指定计算范围(分组)
select dept_name, avg (salary)
from instructor
group by dept_name
having avg (salary) > 42000;
```

限定输出哪些平均工资(结果筛选)

案例4

红色标注之处的作用，意义你清楚吗？

*SQL能够满足应用对数据查询的需要吗？

SQL查询能力很强：

1）实现了基本代数运算
2）灵活的表间连接方式
3）实现了代数运算复合(下面的嵌套子查询)
4）灵活的where条件
5）聚集函数等常用函数
6）嵌入式和动态SQL

3.2 嵌套子查询

什么是SQL嵌套子查询？

案例5.a

(P.49&P.24)找出在2009年秋季，但不在2010年春季同时开课的所有课程

```
select distinct course_id
from section
where semester = ’Fall’ and year= 2009 and
   course_id not in (select course_id
                     from section
                     where semester = ’Spring’ and year= 2010);
```

(集合成员资格)

嵌套子查询、允许嵌在何处及作用？

案例5.b

(p.50)查出这些老师的姓名，他的工资要比Biology系某教师工资高

```
select name
from instructor
where salary > some (select salary
                     from instructor
                     where dept_name ='Biology');
```

(集合的比较)

嵌套子句可以多种方式用在where中！并显著增强了SQL的查询能力！

案例5.c

(P.50&P.24)找出在2009年秋季和2010年春季同时开课的所有课程

```
select course_id
from section as S
where semester= ‘Fall' and year=2009 and
exist (select *
       from section as T
       where semester='Sring' amd year=2010 and
       S.course_id=T.course_id);
```

(空关系测试)

3.2 嵌套子查询(续)

用在having子句中-输出结果筛选

嵌套子查询还允许嵌在何处?

案例5.d

(p.50)找出平均工资最高的系

```
select name
from instructor
group by depat_name
having avg(salary)>= all(select avg(salary)          (集合的比较)
                         from instuctor
                         group by depat_name);
```

外部查询：$\Pi_{dept\_name, , avg\_salary}(\sigma_{avg\_salary>42000}(A))$

嵌套子查询:$A= \rho_{B(dept\_name, avg\_slary)}({}_{dept\_name}g_{dept\_name,avg\ (salary)}(instuctor))$

案例5.e

(p.52)找出‘系平均工资超过42000美元的那些系'的教师平均工资

```
select dept_name, avg_salary
from (select dept_name, avg(salary) as avg_slary
      from instructor
      group by depat_name)  as B                  (属性的别名)
where avg_salary >42000;
```

用在from子句中-生成中间关系

一个查询语句的作用相当于编写一段程序

综可上述，可用看出：SQL查询能力的确强!

案例5.f

(p.54)列出所有系以及它们拥有的教师数

```
select dept_name,                                  (表的别名)
       (select count(*)
        from instructor
        where department.dept_name=instructor.dept_name)
        as num_instructors
from department;
```

嵌套子句可用在select子句中-生成标量值

四 SQL数据操纵语言DML

删除数据
(可利用嵌套子句)

数据修改操作包含哪些方面?

案例**6.a**

(P.54)

```
delete from instructor
where dept_name= ’ Finance’ ;

delete from instructor
where dept_name in (select dept_name
                    from department
                    where building = ’ Watson’ );
```

插入数据
(三种常用方式)

案例**6.b**

P.57~58

```
insert into course (course_id, title, dept_name, credits)
values (’ CS-437’ , ’ Database Systems’ , ’ Comp. Sci.’ , 4);

insert into student
values (’ 3003’ , ’ Green’ , ’ Finance’ , null);

insert into instuctor
        select ID, name, dept_name, 18000
   from   student
          where dept_name =  ‘Music’  and tot_cred > 144;
```

这些示例起到什么作用?

更新数据
(可用case结构)

案例**6.c**

P.58~58

```
update instructor
set salary = salary * 1.03
where salary > 100000;

update instructor
set salary = case
      when salary <= 100000 then salary * 1.05
      else salary * 1.03
      end;
```

五 SQL支持的表间连接方式

5.1 自然/等值连接的不同方式

（方式1）

```
select name, course_id
from instructor, teaches
where instructor.ID= teaches.ID;
```

案例7

有何不同？

查询结果表虽然相同，但**Where**允许按指定属性(可不同名)连接，且在连接表中连接属性都会出现(**ID**两次) 。

（方式2）

```
select name, course_id
from instructor natural join teaches;
```

有何不同？

虽然查询结果表相同，但**where**的连接表中连接属性**ID**出现**2**次，而在**join**中出现**1**次。

有何不同？

虽然都是按相同属性连接，但**using**允许按指定属性、而**natural**按两表同名属性

（方式3）

```
select name, title
from (instructor natural join teaches)
join course using(course_id);
```

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

作用及查询结果表都相同，在连接表中连接属性都会出现(**ID**重复出现两次)。

有何不同？

（方式4）

```
select *
from student join takes on student.id=takes.id;
```

on连接示例

＊SQL支持哪些类型的表间连接方式？

ON连接示例 p. 64

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

案例8。a *takes* 关系

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.	
98765	Bourikas	Elec. Eng.	
98988	Tanaka	Biology	120

案例8。b *student* 关系

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	Comp. Sci.	32	CS-315	1	Spring	2010	A
12345	Shankar	Comp. Sci.	32	CS-347	1	Fall	2009	A
19991	Brandt	History	80	HIS-351	1	Spring	2010	B
23121	Chavez	Finance	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-
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

案例8。c *student* **join** *takes* **on** *student. ID* = *takes. ID* 的结果，其中省略了 *ID* 的第二次出现

5.2 外连接的不同方式

给出这两个表左外连接、右外连接和全外连接的结果？

course

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

prereq

course_id	prereq_id
BIO-301	BIO-101
CS-190	CS-101
CS-347	CS-101

course **natural left outer join** *prereq*

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

*外连接以可与**on**和**using**一起使用吗，其作用？

course **natural right outer join** *prereq*

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

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

Join Conditions
natural
on <predicate>
using (A_1, A_1, …, A_n)

可以，作用与前面join情形类似(p.67)
(这4种连接类型和3种条件可任意组合)

练习

- Department（dname, dphone, daddress）
- Employee（eid, ename, egender, eage, esalary, dname）

1. 查询年龄在20到25之间的员工姓名
2. 查询每个部门的男员工人数
3. 按照从小到大的顺序显示每个部门的工资总额
4. 查找每个部门得到最高工资的员工信息

本节小结

- DDL
- DQL
- DML

课后作业安排

- 作业
  - 第3章：3.16 a)c)， 3.17 a)c)， 3.21 a)c)；
- 预习
  - 第9讲-高级SQL（课前预习资料）