

# 数据库的修改-删除

- ❑ 除了数据库信息的抽取外，SQL还定义了增加、删除和更新数据库信息的操作
- ❑ 删除请求的表达与查询非常类似。我们只能删除整个元组，而不能只删除某些属性上的值。SQL用如下语句表示删除：

```
delete from r  
where P;
```

其中P代表一个谓词，*r*代表一个关系

- 例1，从instructor关系中删除与Finance系教师相关的所有元组

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

## 数据库的修改-删除

- 例2，从instructor关系中删除所有这样的教师元组，他们在位于Watson大楼的系工作

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

# 数据库的修改-删除

- 例3，删除工资低于大学平均工资的教师记录

```
delete from instructor
where salary < (select avg (salary)
                from instructor);
```

- 问题：当我们从instructor关系中删除元组时，平均工资就会改变
- SQL中的解决方案：
  - 首先，计算出平均工资，并找出要删除的所有元组
  - 然后，删除上述找到的所有元组（不重新计算平均工资，也不重新测试元组是否符合删除条件）
  - 在同一SQL语句内，除非外层查询的元组变量引入内层查询，否则内层查询只进行一次

# 数据库的修改-插入

- ❑ SQL允许使用insert语句，向关系中插入元组，形式如下：

```
insert into r [(c1, c2, ...)]  
  values (e1, e2, ...);  
insert into r [(c1, c2, ...)]  
  select e1, e2, ... from ...;
```

- 例1，假设我们要插入的信息是Computer Science系开设的名为“Database Systems”的课程CS-437，它有4个学分

```
insert into course  
  values ( 'CS-437' , 'Database Systems' , 'Comp. Sci.' , 4);
```

- SQL允许在insert语句中指定属性，所以上述语句还可写为：

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

# 数据库的修改-插入

- 若上例中， Database Systems” 课程的学分未知，插入语句还可写为：

```
insert into course
values ( 'CS-437' , 'Database Systems' , 'Comp. Sci.' , null );
```

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

- 假设我们想让Music系每个修满144学分的学生成为Music系的教师，其工资为18000美元

```
insert into instructor
select ID, name, dept_name, 18000
from student
where dept_name = 'Music' and tot_cred > 144;
```


# 数据库的修改-更新

- SQL允许使用update语句，在不改变整个元组的情况下改变其部分属性的值，形式如下：

```
update r
set <c1=e1 , [c2=e2, ... ]>
[where <condition>] ;
```

- 例1，假设给工资超过100 000美元的教师涨3%的工资，其余教师涨5%  
我们可以写两条update语句

```
update instructor
set salary = salary * 1.03
where salary > 100000;
update instructor
set salary = salary * 1.05
where salary <= 100000;
```



**注意：**这两条update语句的顺序十分重要。如果调换顺序，可能导致工资略少于100 000美元的教师将增长8%的工资

## 数据库的修改-更新

- 针对上例查询，我们也可以使用SQL提供的case结构，避免更新次序引发的问题，形式如下：

```
case
when pred1 then result1
when pred2 then result2
. . .
when predn then resultn
else result0
end
```

因此上例查询可重新为：

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

## ❑ SQL查询语句的通用形式:

select <[distinct]  $c1, c2, \dots$ >

from < $r1, \dots$ >

[where <*condition*>]

[group by < $c1, c2, \dots$ > [having <*cond2*>] ]

[order by < $c1$ [desc] , [ $c2$ [desc|asc],  $\dots$ ]]>

## ❑ SQL查询语句执行顺序:

from → where → group (aggregate) → having → select → order by

## ❑ SQL支持关系上的基本集合运算，包括并、交、和差运算



# 总结

- ❑ SQL支持聚集，可以把关系进行分组，还支持在分组上的集合运算
- ❑ SQL支持在外层查询的where和from子句中嵌套子查询
- ❑ SQL提供了用于更新、插入、删除信息的结构

谢谢！