



数据库系统概论

An Introduction to Database System

第三章 关系数据库标准语言SQL





3.3 数据查询

- 学习目标

- 掌握SQL数据查询语句语法格式，并能够熟练使用SQL语句实现案例数据库应用中相关数据查询问题



3.3 数据查询

- 数据查询就是根据用户的需求，从数据库中提取所需要的数据，并将查询结果显示出来
- SQL中使用**SELECT**语句实现数据查询



3.3 数据查询

- 一个完整的数据查询语句格式 指定查询对象(基本表或视图)

SELECT [ALL|DISTINCT] <目标列表达式>
[, <目标列表达式>] ...

FROM <表名或视图名>[, <表名或视图名>] ...

[**WHERE** <条件表达式>] 指定查询条件

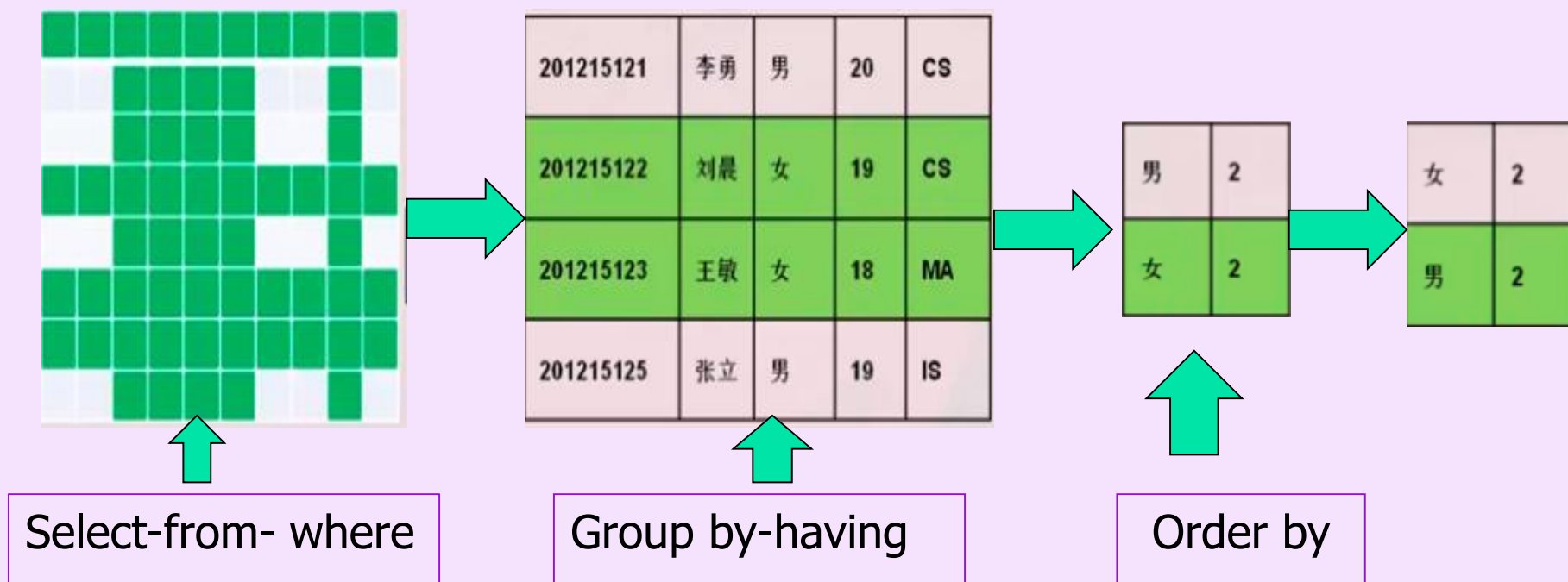
[**GROUP BY** <列名1> [**HAVING** <条件表达式>]]

[**ORDER BY** <列名2> [ASC|DESC]];

对查询结果表按指定的一个或多个列值进行
组为 升序或降序排序

3.3 数据查询

◆ 查询语句的功能分组



3.3 数据查询

◆ 学生选课数据库

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市

	cno	cname	cpno	credit
1	001	数据库原理	002	4
2	002	离散数学		3
3	003	管理信息系统.	001	2
4	004	操作系统		4
5	005	数据结构	004	4
6	006	Java程序设计	005	4
7	007	DB_Design		2

sno	cno	grade
2016010101	001	85
2016010101	003	80
2016010101	002	90
2016010102	001	85
2016010102	002	80
2016010102	003	82
2016010102	005	89
2016020101	001	72
2016020101	002	59
2016020101	004	80
2016020102	007	86
2016030101	002	55
2016030101	001	80
2016010101	004	87



1. 单表查询

◆ 查询仅涉及一个表，是一种最简单的查询操作

一、选择表中的若干列

二、选择表中的若干元组

三、对查询结果排序(**ORDER BY**)

四、使用聚集函数

五、对查询结果分组(**GROUP BY**)

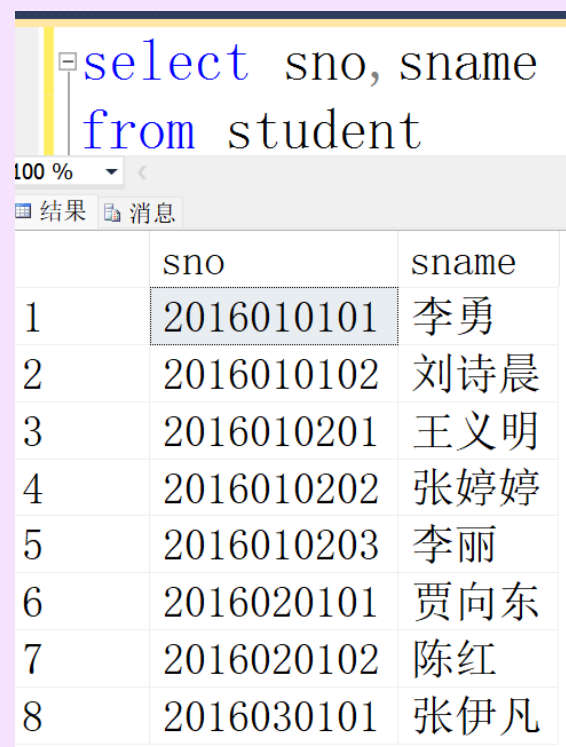
一、选择表中的若干列

■ 查询指定列

[例1] 查询全体学生的学号与姓名

```
select sno, sname  
from student
```

sno	sname	ssex	birthdate	clno
2016010101	李勇	男	2000-01-05	20160101
2016010102	刘诗晨	女	2001-08-15	20160101
2016010201	王义明	男	2000-04-17	20160102
2016010202	张婷婷	女	2000-10-12	20160102
2016010203	李丽	女	1999-12-01	20160102
2016020101	贾向东	男	2001-08-18	20160201
2016020102	陈红	女	2001-11-15	20160201
2016030101	张伊凡	女	2001-01-25	20160301



The screenshot shows a database query interface. At the top, the SQL query `select sno, sname from student` is entered. Below the query, there are tabs for '结果' (Results) and '消息' (Messages). The '结果' tab is selected, displaying a table with 8 rows of data. The table has two columns: 'sno' and 'sname'. The data is as follows:

	sno	sname
1	2016010101	李勇
2	2016010102	刘诗晨
3	2016010201	王义明
4	2016010202	张婷婷
5	2016010203	李丽
6	2016020101	贾向东
7	2016020102	陈红
8	2016030101	张伊凡



一、选择表中的若干列

■ 查询所有列

[例2] 查询所有课程的详细信息

```
select cno, cname, cpno, credit  
from course
```

```
select * from course
```

	cno	cname	cpno	credit
1	001	数据库原理	002	4
2	002	离散数学		3
3	003	管理信息系统.	001	2
4	004	操作系统		4
5	005	数据结构	004	4
6	006	Java程序设计	005	4
7	007	DB_Design		2



一、选择表中的若干列

- 查询经过计算的值
- **SELECT**子句的<目标列表达式>不仅可以是表中的属性列名，也可以是算术表达式、字符串常量、函数等

一、选择表中的若干列

■ 查询经过计算的值

[例3] 查询全体学生的姓名及其年龄。

➤ 获取系统当前日期函数: **Getdate()**

```
select sname, year(getdate()) - year(birthdate)
from student
```



sname	(无列名)
李勇	23
刘诗晨	22
王义明	23
张婷婷	23
李丽	24
贾向东	22
陈红	22
张伊凡	22

```
select sname, year(getdate()) - year(birthdate)
from student
```

```
select sname, datediff(year, birthdate, getdate())
from student
```

一、选择表中的若干列

- 使用**列别名**改变查询结果的列标题:
- 语法格式: **<目标列表表达式> [As] <列别名>**
<列别名>= <目标列表表达式>

```
select sname, age=datediff(year, birthdate, getdate())  
from student
```

	sname	age
1	李勇	23
2	刘诗晨	22
3	王义明	23
4	张婷婷	23
5	李丽	24
6	贾向东	22
7	陈红	22
8	张伊凡	22

二、选择表中的若干元组

- (1) 消除取值重复的行
- 使用 **DISTINCT** 关键词，去掉表中重复

[例4] 查询选修了课程的学生学号

```
Select distinct sno from sc
```

sno
2016010101
2016010102
2016020101
2016020102
2016030101

sno	cno	gmark
2016010101	001	85
2016010101	003	80
2016010101	005	90
2016010102	001	85
2016010102	002	80
2016010102	003	82
2016010102	005	89
2016020101	001	72
2016020101	006	59
2016020101	004	80
2016020102	007	86
2016030101	006	55
2016010101	002	80
2016010101	004	87
2016010101	006	76



二、选择表中的若干元组

- (2) 查询满足条件的元组
- 查询满足条件的元组可以通过**where子句**来实现

查 询 条 件	谓 词
比较大小	=, >, <, >=, <=, !=, <>, !>, !<; NOT+上述比较运算符
确定范围	BETWEEN AND, NOT BETWEEN AND
确定集合	IN, NOT IN
字符匹配	LIKE, NOT LIKE
空 值	IS NULL, IS NOT NULL
多重条件（逻辑运算）	AND, OR, NOT

(1) 比较大小

◆ =, >, <, >=, <=, !=, <>, !=, !=; NOT+上述比较运算符

[例5] 查询出生日期在2000年以后的所有学生的姓名及其出生日期。

```
select sname,birthdate
from student
where birthdate>='2000-01-01'
```

where year(birthdate) >= 2000

sno	sname	ssex	birthdate	clno
2016010101	李勇	男	2000-01-05	20160101
2016010102	刘诗晨	女	2001-08-15	20160101
2016010201	王义明	男	2000-04-17	20160102
2016010202	张婷婷	女	2000-10-12	20160102
2016010203	李丽	女	1999-12-01	20160102
2016020101	贾向东	男	2001-08-18	20160201
2016020102	陈红	女	2001-11-15	20160201
2016030101	张伊凡	女	2001-01-25	20160301



(2) 确定范围

◆ **BETWEEN AND, NOT BETWEEN AND**

<列名> **BETWEEN** 范围下限（低值） **AND** 范围上限（高值）
<列名> **NOT BETWEEN** （低值） **AND** （高值）



(2) 确定范围

◆ BETWEEN AND, NOT BETWEEN AND

[例6] 查询年龄在20~23岁（包括20岁和23岁）之间的学生的姓名、年龄和所在班级

```
select sname, year(getdate())-year(birthdate) as age, clno
from student
where year(getdate())-year(birthdate) between 20 and 23
```

sno	sname	ssex	birthdate	clno
2016010101	李勇	男	2000-01-05	20160101
2016010102	刘诗晨	女	2001-08-15	20160101
2016010201	王义明	男	2000-04-17	20160102
2016010202	张婷婷	女	2000-10-12	20160102
2016010203	李丽	女	1999-12-01	20160102
2016020101	贾向东	男	2001-08-18	20160201
2016020102	陈红	女	2001-11-15	20160201
2016030101	张伊凡	女	2001-01-25	20160301



(3) 多重条件查询

◆ AND, OR, NOT

[例6] 查询年龄在20~23岁（包括20岁和23岁）之间的学生的姓名、年龄和所在班级

```
select sname, year(getdate())-year(birthdate) as age, clno
from student
where year(getdate())-year(birthdate)>=20 and
      year(getdate())-year(birthdate)<=23
```

逻辑运算符**AND**的优先级高于**OR**，可以用括号改变优先级。



(4) 确定集合

◆ **IN , NOT IN**

<列名> **IN**(<值表>),
<列名> **NOT IN**(<值表>)



(4) 确定集合

◆ IN , NOT IN

[例7] 查询20160101、20160102和20160201班的所有女学生的姓名和性别。

```
select sname, ssex  
from student  
where clno in('20160101', '20160102', '20160201') and ssex='女'
```

sno	sname	ssex	birthdate	clno
2016010101	李勇	男	2000-01-05	20160101
2016010102	刘诗晨	女	2001-08-15	20160101
2016010201	王义明	男	2000-04-17	20160102
2016010202	张婷婷	女	2000-10-12	20160102
2016010203	李丽	女	1999-12-01	20160102
2016020101	贾向东	男	2001-08-18	20160201
2016020102	陈红	女	2001-11-15	20160201
2016030101	张伊凡	女	2001-01-25	20160301



(5) 字符匹配

◆ **LIKE , NOT LIKE**

- 查找指定的属性列值与<匹配串>相匹配的元组
- <匹配串>可以是一个完整地字符串
- <匹配串>也可以含有通配符%和_
 - %（百分号）代表任意长度（长度可以为0）的字符串
 - _（下横线）代表任意单个字符，一个汉字占两个字符

<列名> [NOT] LIKE '<匹配串>' [ESCAPE '<换码字符>']

使用换码符将通配符
转义为普通字符



(5) 字符匹配

◆ LIKE, NOT LIKE

[例8] 查询以"DB_"开头，且倒数第3个字符为 i 的课程的具体情况。

```
select *  
from course  
where cname like 'DB\__%i__' escape '\'
```

cno	cname	cpno	credit
001	数据库原理	002	4
002	离散数学		3
003	管理信息系统.	001	2
004	操作系统		4
005	数据结构	004	4
006	Java程序设计	005	4
007	DB_Design		2



(6) 涉及空值的查询

- **NULL**

- 不知道、不存在、无意义

- **空值的产生有其实际需求**

- 学生在选课后，产生选课表，但是还没有成绩，成绩为空值，它和0不一样（不是0分）
 - 某个属性不应该有值
 - 由于某种原因不便于填写
 - 外连接会产生空值
 - 空值的关系运算也会产生空值



(6) 涉及空值的查询

- 空值的约束条件
- 属性定义（或者域定义）中
 - 有NOT NULL约束条件的不能取空值
 - 加了UNIQUE限制的属性只能有一个空值
 - 主属性不能取空值



(6) 涉及空值的查询

- 空值的算术运算、比较运算和逻辑运算
 - 空值与另一个值（包括另一个空值）的算术运算的结果为空值
 - 空值与另一个值（包括另一个空值）的比较运算的结果为UNKNOWN
 - 有UNKNOWN后，传统的逻辑运算中的二值（TRUE,FALSE）逻辑就扩展成了三值逻辑



逻辑运算符真值表

x y	xANDy	xORy	NOTx
T T	T	T	F
T U	U	T	F
T F	F	T	F
U T	U	T	U
U U	U	U	U
U F	F	U	U
F T	F	T	T
F U	F	U	T
F F	F	F	T



(6) 涉及空值的查询

[例9] 查询选课表中所有缺考记录中学生的学号和课程号。

空值判断谓词: **IS NULL**

非空值判断谓词: **IS NOT NULL**

"IS" 不能用 **"="** 代替

```
select sno, cno
from sc
where grade is null
```

sno	cno	grade
2016010101	001	88
2016010101	003	80
2016010101	002	90
2016010102	001	88
2016010102	002	80
2016010102	003	82
2016010102	005	89



三、ORDER BY子句

- 语句格式

SELECT [ALL|DISTINCT] <目标列表达式>[, <目标列表达式>] ...

FROM <表名或视图名>[, <表名或视图名>] ...

[**WHERE** <条件表达式>]

[**ORDER BY** <列名 1>[ASC|DESC], <列名 2>[ASC|DESC]];



三、ORDER BY子句

- ORDER BY子句的作用
 - 对查询结果表按指定的一个或多个属性列值进行升序或降序排序
 - 升序：ASC；降序：DESC；缺省值为升序
- 对于空值，排序时显示的次序由具体系统实现来决定
 - ASC：排序列为空值的元组最后显示
 - DESC：排序列为空值的元组最先显示

三、ORDER BY子句

[例10] 查询选修了002号课程的学生学号及其成绩，查询结果按成绩降序排列。

```
select top 10 sno, grade
```

```
select sno, grade  
from sc  
where cno='002'  
order by grade desc
```

可以通过在**select**后面加上**TOP n**选项指定返回结果集中的前**n**行

sno	cno	grade
2016010101	001	88
2016010101	003	80
2016010101	002	90
2016010102	001	88
2016010102	002	80
2016010102	003	82
2016010102	005	89
2016020101	001	75
2016020101	002	59
2016020101	004	80

三、ORDER BY子句

[例11] 查询全体学生情况，查询结果按所在班级的班级号升序排列，同一班级中的学生按出生日期降序排列。

```
select *  
from student  
order by clno asc, birthdate desc
```

```
select *  
from student  
order by clno asc, birthdate desc
```

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市



四、聚集函数

- 计数

COUNT ([DISTINCT|ALL] *)

COUNT ([DISTINCT|ALL] <列名>)

- 计算总和

SUM ([DISTINCT|ALL] <列名>此列必须为数值型)

- 计算平均值

AVG ([DISTINCT|ALL] <列名>此列必须为数值型)

- 最大最小值

MAX ([DISTINCT|ALL] <列名>)

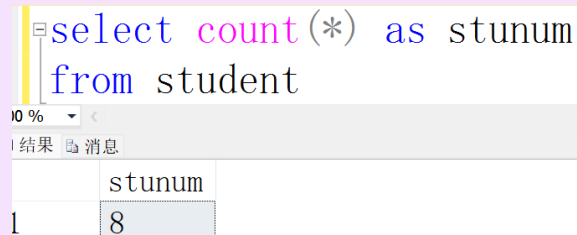
MIN ([DISTINCT|ALL] <列名>)

- ◆ **Distinct**的作用是计算时取消指定列中的重复值。
- ◆ 聚集函数只能用于**SELECT**子句和**GROUP BY**中的**HAVING** 子句中
- ◆ 聚集函数不能用在**WHERE**子句中

四、聚集函数

[例12] 统计学生的总人数。

```
select count(*) as stunum
from student
```



select count(*) as stunum from student	
10 %	<
结果	消息
	stunum
1	8

当聚集函数遇到空值时，除COUNT(*)外，都跳过空值而只处理非空值。

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市



五、GROUP BY子句

- GROUP BY子句的作用
 - 将查询结果按指定的一列或多列值分组，把属性列值相等的作为一组
- GROUP BY子句分组的目的
 - 细化聚集函数的作用对象
 - 未对查询结果分组，聚集函数将作用于整个查询结果
 - 对查询结果分组后，聚集函数将分别作用于每个分组，即每一个分组都有一个函数值。



五、GROUP BY子句

- 语句格式

SELECT [ALL|DISTINCT] <目标列表表达式>
[, <目标列表表达式>] ...

FROM <表名或视图名>[,<表名或视图名>] ...

[**WHERE** <条件表达式>]

[**GROUP BY** <列名1>[,<列名2>,...][**HAVING** <组选择条件表达式>]

[**ORDER BY** <列名1>[,<列名2>,...][**ASC|DESC**]]

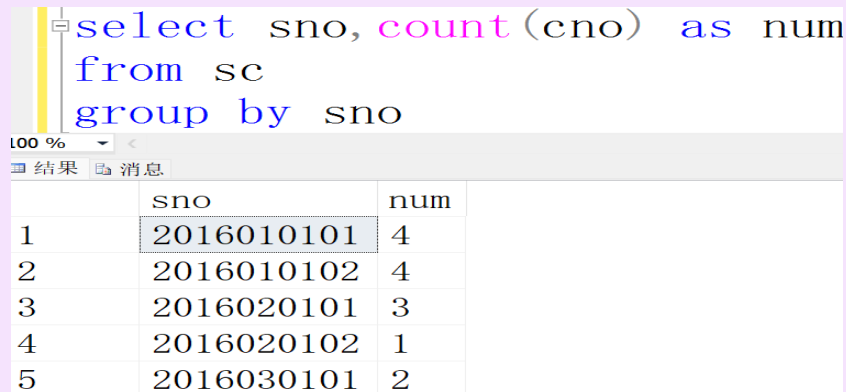
对查询结果依次按列名1、列名2....的值进行分组

HAVING短语对分组进行筛选，选出满足指定条件的组

五、GROUP BY子句

[例13] 查询选修了3门以上课程的学生学号及选课门数。

```
select sno, count(cno) as num
from sc
group by sno having count(cno) >= 3
```



	sno	num
1	2016010101	4
2	2016010102	4
3	2016020101	3
4	2016020102	1
5	2016030101	2

	sno	cno	grade
1	2016010101	001	88
2	2016010101	003	80
3	2016010101	002	90
4	2016010102	001	88
5	2016010102	002	80
6	2016010102	003	82
7	2016010102	005	89
8	2016020101	001	75
9	2016020101	002	59
10	2016020101	004	80
11	2016020102	007	86
12	2016030101	002	55
13	2016030101	001	83
14	2016010101	004	87



五、GROUP BY子句

- ◆ **HAVING**短语与**WHERE**子句作用对象不同
 - **WHERE**子句作用于基表或视图，从中选择满足条件的元组，不可与聚集函数连用
 - **HAVING**短语作用于组，从分组后的组中选择满足条件的组，可与聚集函数连用



五、GROUP BY子句

- ◆ **SELECT**子句所要显示输出的值，必须在分组中是唯一的
 - 除了聚集函数内部的属性，没有出现在**GROUP BY**子句中的属性，不能出现在**SELECT**子句中
 - 任何出现在**HAVING**子句但没有被聚集的属性，必须出现在**GROUP BY**子句中



五、GROUP BY子句

- ◆ 查询选修了3门以上课程且成绩都在80分以上的学生学号及选课门数，并将查询结果按选课门数升序排序

Select sno,count(cno) as cnonum

第五步：select子句汇总数据

From sc

第一步：从from子句取出全部数据

Where gmark>=80

第二步：根据where条件过滤数据

Group by sno

第三步：过滤后根据group by分组

Having count(cno)>=3

第四步：根据having条件筛选出满足条件的组

Order by cnonum

第六步：根据order by子句进行排序

SQL
语句的
执行
顺序



2 连接查询

- ◆ 关系代数中“连接”是用一个特殊符号来表达的
- ◆ 在SQL中“连接”是用“**连接条件**”来表达
- ◆ 同时涉及多个表的查询称为**连接查询**
- ◆ 用来连接两个表的条件称为**连接条件或连接谓词**

一般格式:

[<表名1>.]<列名1> <比较运算符> [<表名2>.]<列名2>
[<表名1>.]<列名1> **BETWEEN** [<表名2>.]<列名2> **AND**
[<表名2>.]<列名3>

各连接字段类型必须是可比的，但名字不必相同。

一、等值与非等值连接查询

◆ 等值连接：连接运算符为=

[例14]查询每个学生及其选修课程的情况

```
select student.*, sc.*  
from student, sc  
where student.sno=sc.sno
```

sno	cno	grade
2016010101	001	88

```
select student.*, sc.*
from student, sc
where student.sno=sc.sno
```

100 %

结果消息

	sno	sname	ssex	birthdate	clno	adress	sno	cno	grade
1	2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市	2016010101	001	88
2	2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市	2016010101	003	80
3	2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市	2016010101	002	90
4	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	2016010102	001	88
5	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	2016010102	002	80
6	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	2016010102	003	82
7	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	2016010102	005	89
8	2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市	2016020101	001	75
9	2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市	2016020101	002	59
10	2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市	2016020101	004	80
11	2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市	2016020102	007	86
12	2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市	2016030101	002	55

一、等值与非等值连接查询

- 自然连接：把目标列中重复的属性列去掉

[例15]对[例14]用自然连接完成

```
select student.*, cno, grade
from student, sc
where student.sno=sc.sno
```

```
[select student.*, cno, grade
from student, sc
where student.sno=sc.sno
```

如果属性名在参加连接的各表中是唯一的，则可以省略表名前缀。

2	2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市	002	88
3	2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市	002	90
4	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	001	88
5	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	002	80
6	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	003	82
7	2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	005	89
8	2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市	001	75
9	2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市	002	59
10	2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市	004	80



连接操作的执行过程

(1) 嵌套循环法

- 首先在表**1**中找到第一个元组，然后从头开始扫描表**2**，逐一查找满足连接件的元组，找到后就将表**1**中的第一个元组与该元组拼接起来，形成结果表中一个元组。
- 表**2**全部查找完后，再找表**1**中第二个元组，然后再从头开始扫描表**2**，逐一查找满足连接条件的元组，找到后就将表**1**中的第二个元组与该元组拼接起来，形成结果表中一个元组。
- 重复上述操作，直到表**1**中的全部元组都处理完毕



连接操作的执行过程

(2) 排序合并法

- 常用于=连接
- 首先按连接属性对表1和表2排序
- 对表1的第一个元组，从头开始扫描表2，顺序查找满足连接条件的元组，找到后就将表1中的第一个元组与该元组拼接起来，形成结果表中一个元组。当遇到表2中第一条大于表1连接字段值的元组时，对表2的查询不再继续。



连接操作的执行过程

(3) 索引连接 (**INDEX-JOIN**)

- 对表2按连接字段建立索引
- 对表1中的每个元组，依次根据其连接字段值查询表2的索引，从中找到满足条件的元组，找到后就将表1中的第一个元组与该元组拼接起来，形成结果表中的一个元组。
- 该方法可以视作嵌套循环法的一个变种

二、自身连接

◆ 自身连接：一个表与其自己进行连接

[例16] 查询每一门课的间接先修课（即先修课的先修课）

```
select c1.cno, c2.cjno
from course as c1, course c2
where c1.cjno=c2.cno
```

◆ 需要给表起别名以示区别

◆ 由于所有属性名都是同名属性，因此必须使用别名前缀

002	离散数学	008	3
003	管理信息系统	001	2
004	操作系统		4
005	数据结构	004	4
006	Java程序设计	005	4
007	DB_Design		2
008	高等数学	NULL	3

002	离散数学	008	3
003	管理信息系统	001	2
004	操作系统		4
005	数据结构	004	4
006	Java程序设计	005	4
007	DB_Design		2
008	高等数学	NULL	3

三、复合条件连接

- ◆ 一条SQL语句可以同时完成选择和连接查询，这时where子句是由连接谓词和选择谓词组成的复合条件。

[例17]查询选修002号课程且成绩在90分以上的所有学生的学号和姓名

```
select student.sno, sname
from student, sc
where student.sno=sc.sno and cno='002' and grade>90
```

sno	sname	ssex	birthdate	clno	adress	sno	cno	grade
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市	2016010101	001	88
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市	2016010101	003	80
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市	2016010101	002	90
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市	2016010102	001	88
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市	2016010102	002	80
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市	2016010102	003	82
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市	2016010102	005	89
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市	2016020101	001	75
						2016020101	002	59
						2016020101	004	80



四、外连接

- ◆ 外链接与普通连接的区别？
 - ◆ 普通连接操作只输出满足条件的元组
 - ◆ 外连接操作以指定表为连接主体，将主体表中不满足连接条件的元组一并输出
- ◆ 普通连接语法格式：
Select From <表名1> (inner) join <表名2> on <连接条件>



四、外连接

◆ 左外连接

- ◆ 列出左边关系中所有的悬浮元组
- ◆ 语法格式：

Select From <表名1> **left outer join** <表名2> **on** <连接条件>

◆ 右外连接

- ◆ 列出右边关系中所有的悬浮元组
- ◆ 语法格式：

Select From <表名1> **right outer join** <表名2> **on** <连接条件>



四、外连接

- ◆ 全外连接

- ◆ 列出左边关系和右边关系中所有的悬浮元组
- ◆ 语法格式:

Select From <表名1> **full outer join** <表名2> **on** <连接条件>

五、多表连接

◆ 多表连接：两个以上的表进行连接

[例18]查询每个学生的学号、姓名、选修的课程名及成绩

```
select student.sno, sname, cname, grade
from student, sc, course
where student.sno=sc.sno and
      sc.cno=course.cno
```

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市

cno	cname	sno	cno	grade
001	数据库原理	2016010101	001	88
002	离散数学	2016010101	003	80
003	管理信息系	2016010101	002	90
004	操作系统	2016010102	001	88
005	数据结构	2016010102	002	80
006	Java程序设计	2016010102	003	82
007	DB_Design	2016010102	005	89
008	高等数学	2016020101	001	75
		2016020101	002	59
		2016020101	004	80



3 嵌套查询

- ◆ 一个SELECT-FROM-WHERE语句称为一个**查询块**。
- ◆ 将一个查询块嵌套在另一个查询块的WHERE子句或HAVING短语的条件中的查询称为**嵌套查询**。

```
Select sname  
From student  
Where sno in
```

```
  (select sno  
   From sc  
   Where cno='002')
```



上层的查询块称为
外层查询/父查询



下层的查询块称为
内层查询/子查询



3 嵌套查询

- ◆ SQL语言允许多层嵌套查询
- ◆ 子查询的限制
 - ◆ 子查询语句中不能使用**ORDER BY** 子句
- ◆ 嵌套查询的分类
 - ◆ 根据子查询中处理的数据是否与父查询的当前元组有关，可以把嵌套查询分为**相关子查询**和**不相关子查询**



一、带有IN谓词的子查询

- 不相关子查询（独立子查询）
 - 子查询的查询条件不依赖于父查询
 - 由里向外逐层处理。即每个子查询在上一级查询处理之前求解，子查询的结果用于建立其父查询的查找条件



一、带有IN谓词的子查询

[例19]查询与“李勇”在同一个班级学习的学生

- 第一步：查询“李勇”所在的班级号
- 第二步：查询所有在该班级学习的学生
- 将第一步查询结果嵌入到第二步查询的条件中

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市



一、带有IN谓词的子查询

[例19]查询与“李勇”在同一个班级学习的学生

```
select *  
from student  
where clno=  
(select clno  
  from student  
  where sname= ‘李勇’)
```




一、带有IN谓词的子查询

- ◆ 在嵌套查询中，子查询的结果往往是一个**集合**
[练习]查询选修了“数据库原理”课程的学生学号和姓名（试写出两种方法）

连接查询可以用嵌套查询实现，
但嵌套查询不一定都能用连接查询替代。



二、带有比较运算符的子查询

- 相关子查询
 - 子查询的查询条件依赖于父查询
 - 整个嵌套语句称为相关嵌套查询
 - 首先取外层查询中表的第一个元组，根据它与内层查询相关的属性值处理内层查询，若**WHERE**子句返回值为真，则取此元组放入结果表
 - 然后再取外层表的下一个元组
 - 重复这一过程，直到外层表全部检查完为止



二、带有比较运算符的子查询

[例20]找出每个学生超过他选修课程平均成绩的学号、课程号及成绩

- 内层查询是求一个学生所有选修课程的平均成绩
- 至于是哪个学生，要看外层参数sno的值，该值与父查询相关

sno	cno	grade
2016010101	001	88
2016010101	003	80
2016010101	002	90
2016010102	001	88
2016010102	002	80
2016010102	003	82
2016010102	005	89
2016020101	001	75
2016020101	002	59
2016020101	004	80



二、带有比较运算符的子查询

[例20]找出每个学生超过他选修课程平均成绩的学号、课程号及成绩

- 一种可能的执行过程是：
 - 从外层查询中取出SC的第一个元组x，将元组x的 sno(2016010101)传送给内层查询，计算平均成绩

```
Select avg(gmark)
From sc y
Where y.sno='2016010101'
```



二、带有比较运算符的子查询

[例20]找出每个学生超过他选修课程平均成绩的学号、课程号及成绩

■ 一种可能的执行过程是：

- 执行内层查询，得到平均成绩，然后用该值代替内层查询，得到外层查询

```
Select sno,cno  
From sc x  
Where gmark>=88
```

- 执行外层查询，得到结果
- 然后外层查询取出下一个元组重复上述步骤，直到外层的SC元组全部处理完毕。



二、带有比较运算符的子查询

[例20]找出每个学生超过他选修课程平均成绩的学号、课程号及成绩

```
select sno, cno, grade
from sc x
where grade >
(select avg(grade)
 from sc y
 where y.sno=x.sno)
```



三、带有ANY(SOME)或ALL谓词的子查询

- ◆ 当内层查询返回多值时，要用**ANY**或**ALL**谓词修饰符，同时必须使用**比较运算符**
 - ◆ **>=ANY** 大于等于子查询结果中的某个值
 - ◆ **>=ALL** 大于等于子查询结果中的所有值（大于最大值）
 - ◆ **<=ANY** 小于等于子查询结果中的某个值
 - ◆ **<=ALL** 小于等于子查询结果中的所有值（小于最小值）
 - ◆ **!=(或<>)ANY** 不等于子查询结果中的某个值
 - ◆ **!=(或<>)ALL** 不等于子查询结果中的任何一个值

三、带有ANY(SOME)或ALL谓词的子查询

[例21]查询20160102班中比20160101班任意一个学生年龄小的学生姓名和年龄

执行过程：

- ◆ 第一步：处理子查询，查询20160101班所有学生的年龄
- ◆ 第二步：处理父查询，找20160102班学生年龄比第一步查询结果中任意一个值小的学生姓名和年龄

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市



三、带有ANY(SOME)或ALL谓词的子查询

[例21]查询20160102班中比20160101班任意一个学生年龄小的学生姓名和年龄

```
select sname, year(getdate())-year(birthdate) as age
from student
where clno='20160102' and
    year(getdate())-year(birthdate)<any
(select year(getdate())-year(birthdate)
from student
where clno='20160101')
```



三、带有ANY(SOME)或ALL谓词的子查询

[例22]查询20160102班中比20160101班所有学生年龄都小的学生姓名和年龄

```
select sname, year(getdate())-year(birthdate) as age
from student
where clno='20160102' and
    year(getdate())-year(birthdate)<all
(select year(getdate())-year(birthdate)
from student
where clno='20160101')
```



三、带有ANY(SOME)或ALL谓词的子查询

[例22]查询20160102班中比20160101班所有学生年龄都小的学生姓名和年龄

```
select  sname, year(getdate())-year(birthdate) as age
from student
where clno='20160102' and
      year(getdate())-year(birthdate) <
      (select min(year(getdate())-year(birthdate))
       from student
       where clno='20160101')
```



四、带有EXISTS谓词的子查询

- ◆ EXISTS谓词用于判断一个子查询的结果是否为空

[NOT] EXISTS(子查询)

- ◆ 其语义为：
 - ◆ 带有EXISTS谓词的子查询不返回任何数据，只产生逻辑真值“true”或逻辑假值“false”
 - ◆ 若子查询的查询结果非空，则EXISTS为真，否则为假。
 - ◆ NOT是对EXISTS谓词运算结果进行非运算

四、带有EXIST谓词的子查询

[例23]查询所有选修了001号课程的学生姓名

```
select sname
from student
where sno in
(select sno
from sc
where cno='001')
```

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市

sno	cno	grade
2016010101	001	88
2016010101	003	80
2016010101	002	90
2016010102	001	88
2016010102	002	80
2016010102	003	82
2016010102	005	89
2016020101	001	75
2016020101	002	59
2016020101	004	80

四、带有EXIST谓词的子查询

[例23]查询所有选修了001号课程的学生姓名

```
select sname
from student
where exists
(select *
from sc
where cno='001' and
sc.sno=student.sno)
```

sno	sname	ssex	birthdate	clno	adress
2016010101	李勇	男	2000-01-05	20160101	内蒙古呼和浩特市
2016010102	刘诗晨	女	2001-08-15	20160101	内蒙古呼和浩特市
2016010201	王义明	男	2000-04-17	20160102	内蒙古鄂尔多斯市
2016010202	张婷婷	女	2000-10-12	20160102	内蒙古鄂尔多斯市
2016010203	李丽	女	1999-12-01	20160102	内蒙古鄂尔多斯市
2016020101	贾向东	男	2001-08-18	20160201	内蒙古包头市
2016020102	陈红	女	2001-11-15	20160201	内蒙古包头市
2016030101	张伊凡	女	2001-01-25	20160301	内蒙古呼和浩特市

sno	cno	grade
2016010101	001	88
2016010101	003	80
2016010101	002	90
2016010102	001	88
2016010102	002	80
2016010102	003	82
2016010102	005	89
2016020101	001	75
2016020101	002	59
2016020101	004	80

exists中子查询的目标列表达式通常为*, 因为带**exists**的子查询只与查询的结果是否有值有关, 给出列名无实际意义。



四、带有EXIST谓词的子查询

[例24]查询没有选修001号课程的学生姓名

```
select sname
from student
where sno not in
(select sno
from sc
where cno='001')
```

```
select sname
from student
where not exists
(select *
from sc
where cno='001' and
sc.sno=student.sno)
```



四、带有EXIST谓词的子查询

[例25] 查询选修了全部课程的学生姓名

$$\Pi_{\text{name}}(\text{student} \bowtie (\Pi_{\text{sno}, \text{cno}}(\text{sc}) \div \Pi_{\text{cno}}(\text{course})))$$

```
select  sname
from    student
where   not exists
(select *
from    course
where   not exists
(select * from sc
where   sc.cno=course.cno and sc.sno=student.sno))
```

表示不存在
某一课程记录

表示没有该学生
选修该课程的记录



四、带有EXIST谓词的子查询

[例26]查询所选课程包含学生“2016010101”所选课程的学生的姓名

```
select sname
from student
where not exists
(select *
from sc x
where sno='2016010101' and not exists
(select *
from sc y
where y.cno=x.cno and y.sno=student.sno))
```



四、带有**EXISTS**谓词的子查询

- ◆ 使用谓词**EXISTS**可以实现连接查询，以及其他方式的嵌套查询等无法实现的**select**查询
- ◆ 所有带**IN**谓词、比较运算符、**ANY**和**ALL**谓词的子查询都能用带**EXISTS**谓词的子查询等价替换
- ◆ 某些带**EXISTS**或**NOT EXISTS**谓词的子查询不能被其他形式的子查询等价替换



4 集合查询

- ◆ 标准SQL直接支持的集合操作种类
并操作(UNION)
- ◆ 一般商用数据库支持的集合操作种类
并操作(UNION)
交操作(INTERSECT)
差操作(EXCEPT)
- ◆ 参加集和操作的各查询结果的列数必须相同；对应项的数据类型也必须相同



4 集合查询

- ◆ 并操作 (UNION) 运算的语法格式:

```
Select 语句1  
UNION[ALL]  
Select 语句1
```

- ◆ 查询结果会自动去除重复的元组
- ◆ ALL用来保留取值重复的元组



4 集合查询

- ◆ 交操作 (INTERSECT) 运算的语法格式:

```
Select 语句1  
INTERSECT  
Select 语句1
```

- ◆ 获取两个查询结果中相同的元组集合



4 集合查询

- ◆ 差操作 (EXCEPT) 运算的语法格式:

```
Select 语句1  
EXCEPT  
Select 语句1
```

- ◆ 查询属于SELECT语句1但不属于SELECT语句2查询结果的元组集合



4 集合查询

- ◆ 查询选修了课程001或者选修了课程002的学生的学号

```
Select cno,gmark  
From sc  
Where cno='001'  
union  
Select cno,gmark  
From sc  
Where cno='002'
```



4 集合查询

- ◆ 查询选修了课程001又选修了课程002的学生

```
Select sno  
From sc  
Where cno='001'  
Intersect  
Select sno  
From sc  
Where cno='002'
```



有没有其他实现语句

- ◆ 查询选修了课程001但没有选修课程002的学生



5 基于派生表的查询

- 子查询不仅可以出现在WHERE子句中，还可以出现在FROM子句中，这时子查询生成的临时派生表(derived table)成为主查询的查询对象。

- 例如：

```
Select Sno,Cno
```

```
From sc,(select sno,avg(grade) from sc group by sno) as  
avg_sc(avg_sno,avg_grade)
```

```
Where sc.sno=avg_sc.avg_sno  
and sc.grade>=avg_sc.avg_grade
```



5 基于派生表的查询

- 如果子查询中没有聚集函数，派生表可以不指定属性列，子查询**SELECT**子句后面的列名为其默认属性。
- 例如：

Select sname

From student,(select sno from sc where cno='1') as sc1

Where student.sno=sc1.sno

注：通过**FROM**子句生成派生表时，**AS**关键字可以省略，但必须为派生关系指定一个别名。



6 SELECT语句的一般格式

- **Select**语句的一般格式

SELECT [ALL|DISTINCT] <目标列表表达式>
[, <目标列表表达式>] ...

FROM <表名或视图名>[AS]<别名>
[,<表名或视图名>] ...

[**WHERE** <条件表达式>]

[**GROUP BY** <列名1>[**HAVING** <条件表达式>]]

[**ORDER BY** <列名2> [ASC|DESC]];



6 SELECT语句的一般格式

■ 目标列表达式格式

- (1) *
- (2) <表名>.*
- (3) **COUNT([DISTINCT|ALL]*)**
- (4) [<表名>.<属性列名表达式>[,<表名>.<属性列名表达式>]...

其中<属性列名表达式>可以由属性列、作用于属性列的聚集函数和常量的任意算术运算（+，-，*，/）组成的运算公式



6 SELECT语句的一般格式

- 聚集函数的一般格式

COUNT
SUM
AVG
MAX
MIN

(**[DISTINCT|ALL]** <列名>)



6 SELECT语句的一般格式

- **Where**子句的条件表达式的可选格式

(1)

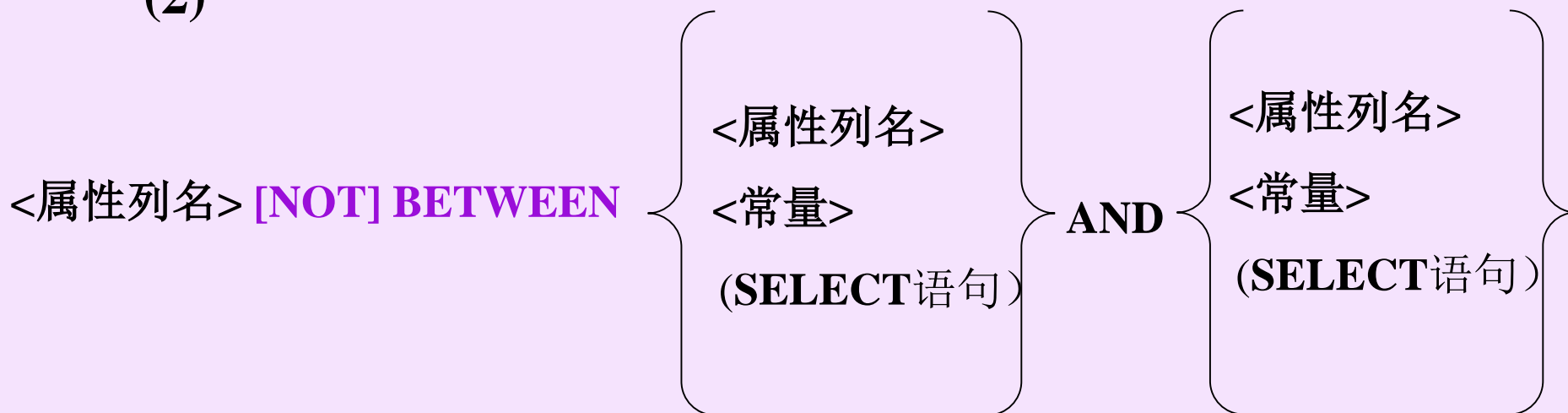
<属性列名> **θ** {
 <属性列名>
 <常量>
 [**ANY|ALL**](**SELECT**语句)



6 SELECT语句的一般格式

- **Where**子句的条件表达式的可选格式

(2)





6 SELECT语句的一般格式

- **Where**子句的条件表达式的可选格式

(3)

<属性列名> **[NOT] IN** $\left\{ \begin{array}{l} (<\text{值1}>[, <\text{值2}>] \dots) \\ (\text{SELECT语句}) \end{array} \right\}$

(4) <属性列名> **[NOT] LIKE** <匹配串>

(5) <属性列名> **IS [NOT] NULL**

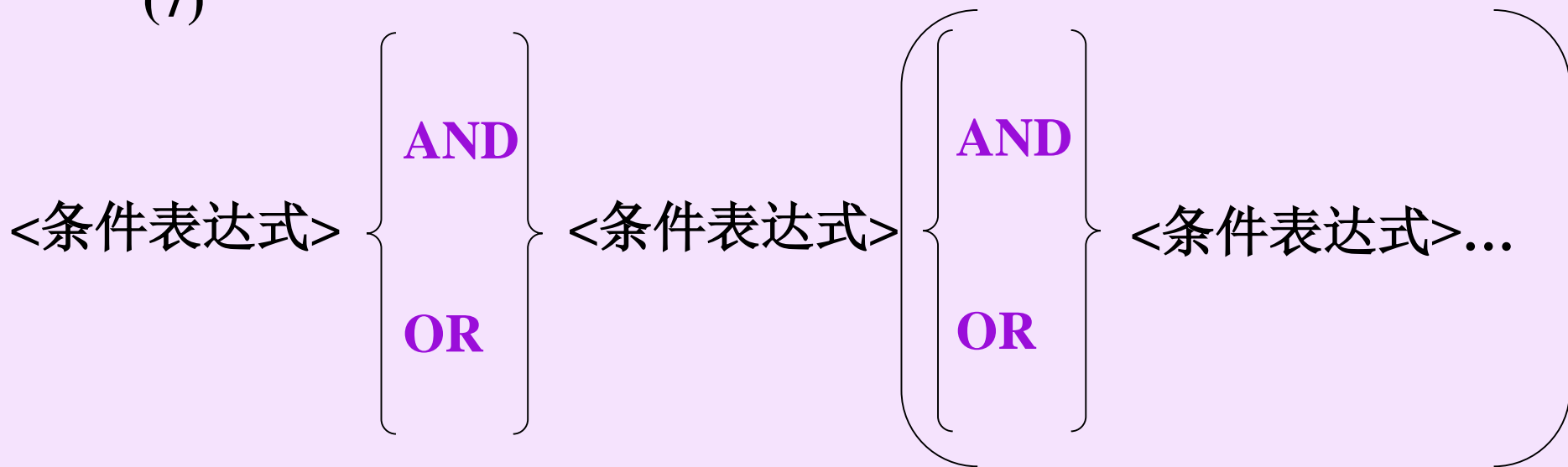
(6) **[NOT] EXISTS** (SELECT语句)



6 SELECT语句的一般格式

- **Where**子句的条件表达式的可选格式

(7)





下课了。。。

追
求



休息一会儿。。。