关系数据库标准语言
Structure Query Language (SQL)

陈杰
chenjie@szu.edu.cn

深圳大学・致理楼
2023.10

提纲

- 数据更新 3.5
- 单表查询 3.4.1

提纲

- 数据更新 3.5
- 单表查询 3.4.1

插入元组

- 语句格式

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

INSERT

INTO <表名> [(<属性列1>[,<属性列2 >…)]

VALUES (<常量1> [,<常量2>]…);

[例3.69]将一个新学生元组（学号：201215128;姓名：陈冬;性别：男;所在系：IS;年龄：18岁）插入到Student表中。

```
INSERT
    INTO  Student (Sno,Sname,Ssex,Sdept,Sage)
    VALUES ('201215128','陈冬','男','IS',18);
```

插入元组

学号 Sno	课程号 Cno	成绩 Grade

[例3.71] 插入一条选课记录（ '200215128','1 '）。

```
INSERT
INTO SC(Sno,Cno)
VALUES ('201215128 ',' 1 ');
```

关系数据库管理系统将在新插入记录的Grade列上自动地赋空值。

```
INSERT
   INTO SC
   VALUES (' 201215128 ',' 1 ',NULL);
```

插入子查询结果

[例3.72] 对每一个系，求学生的平均年龄，并把结果存入数据库

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

第一步：建表

```
CREATE  TABLE  Dept_age
        ( Sdept     CHAR(15)                /*系名*/
          Avg_age SMALLINT);                /*学生平均年龄*/
```

第二步：插入数据

```
INSERT
INTO  Dept_age(Sdept,Avg_age)
            SELECT  Sdept,   AVG(Sage)
            FROM     Student
            GROUP BY Sdept;
```

修改数据

- 语句格式

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

UPDATE <表名>

SET <列名>=<表达式>[,<列名>=<表达式>]…

[WHERE <条件>];

[例3.73] 将学生201215121的年龄改为22岁

```
UPDATE  Student
        SET Sage=22
        WHERE  Sno=' 201215121 ';
```

修改数据

- Increase all students with GPA over 3.5 by 6%, all other students receive 5%.

```
update student
    set gpa= gpa  * 1.06
    where gpa> 3.5

update student
    set gpa = gpa   * 1.05
    where gpa   <= 3.5
```

删除数据

- 语句格式

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

DELETE

FROM <表名>s

[WHERE <条件>];

[例3.76] 删除学号为201215128的学生记录。

```
DELETE
        FROM Student
        WHERE Sno= '201215128 ';
```

带子查询的删除语句

[例3.78] 删除计算机科学系所有学生的选课记录。

学号 Sno	课程号 Cno	成绩 Grade

SC

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

Student

```
DELETE
    FROM  SC
    WHERE  Sno  IN
      (SELETE  Sno
          FROM   Student
          WHERE  Sdept= 'CS') ;
```

带子查询的删除语句

- Delete the record of all students with GPA below the average at the school.

- Problem: as we delete tuples from student, the average GPA changes

- Solution used in SQL:
  - First, compute avg GPA and find all tuples to delete
  - Next, delete all tuples found above (without recomputing avg or retesting the tuples)

```
delete from student
        where gpa< (select avg (gpa)
                    from student)
```

提纲

- 数据更新 3.5
- 单表查询 3.4.1

数据查询

- 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>]…|(SELECT 语句)

[AS]<别名>

[WHERE <条件表达式>]

[GROUP BY <列名1> [HAVING <条件表达式>]]

[ORDER BY <列名2> [ASC|DESC]];

数据查询

- SELECT子句：指定要显示的属性列
- FROM子句：指定查询对象（基本表或视图）
- WHERE子句：指定查询条件
- GROUP BY子句：对查询结果按指定列的值分组，该属性列值相等的元组为一个组。通常会在每组中作用聚集函数。
- HAVING短语：只有满足指定条件的组才予以输出
- ORDER BY子句：对查询结果表按指定列值的升序或降序排序

数据查询

（1）单表查询

（2）连接查询

（3）嵌套查询

（4）集合查询

（5）基于派生表的查询

单表查询

- 查询仅涉及一个表
  - 1.选择表中的若干列
  - 2.选择表中的若干元组
  - 3.ORDER BY子句
  - 4.聚集函数
  - 5.GROUP BY子句

选择表中的若干列

❑ 查询指定列

[例3.16] 查询全体学生的学号与姓名。

```
SELECT Sno,Sname
FROM Student;
```

❑ 查询全部列

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

[例3.18] 查询全体学生的详细记录

```
SELECT  Sno,Sname,Ssex,Sage,Sdept
FROM Student;
```

或

```
SELECT  *
FROM Student;
```

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>不仅可以为表中的属性列，也可以是表达式

[例3.19] 查全体学生的姓名及其出生年份。

```
SELECT Sname,2014-Sage          /*假设当时为2014年*/
FROM Student;
```

Sname	2014-Sage
李勇	1994
刘晨	1995
王敏	1996
张立	1995

Student

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept
201215121	李勇	男	20	CS
201215122	刘晨	女	19	CS
201215123	王敏	女	18	MA
201215125	张立	男	19	IS

选择表中的若干列

[例3.20] 查询全体学生的姓名、出生年份和所在的院系，要求用小写字母表示系名。

SELECT Sname,'Year of Birth: ',2014-Sage, LOWER(Sdept)

FROM Student;

输出结果：

Sname	'Year of Birth:'	2014-Sage	LOWER(Sdept)
李勇	Year of Birth:	1994	cs
刘晨	Year of Birth:	1995	cs
王敏	Year of Birth:	1996	ma
张立	Year of Birth:	1995	is

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>不仅可以为表中的属性列，也可以是表达式

字符串相关运算： **LOWER**(string)，**UPPER**(string)，**SUBSTR(**string,startposition,length)，**INSTR(**string1,string2)，**str_to_date (string[,format])**，LPAD(str,len,padstr)，**RPAD**(str,len,padstr)，**LTRIM(**string)，**RTRIM(**string)，**TRIM** ，**LENGTH(ename)** （见实验指导书**1-2**，**23-27**页 ）

算术运算： **ABS(numeric)**，**MOD(num1, num2)**，**ROUND(numeric,d)**，**TRUNCATE(numeric,d)** **CEIL(numeric)**，**FLOOR(numeric)**，**SQRT(numeric)**，**TO_CHAR(numeric,format)** ，**DATE_FORMAT(date,format)**，**SIGN**， （见实验指导书**1-2**，**28-29**页 ）

请注意，如果任何变量包含空值，则任何涉及算术的SQL语句都将忽略它

选择表中的若干列

- 使用列别名改变查询结果的列标题:

```
SELECT Sname NAME,'Year of Birth:'  BIRTH,
   2014-Sage  BIRTHDAY,LOWER(Sdept)  DEPARTMENT
FROM Student;
```

输出结果:

NAME	BIRTH	BIRTHDAY	DEPARTMENT
李勇	Year of Birth:	1994	cs
刘晨	Year of Birth:	1995	cs
王敏	Year of Birth:	1996	ma
张立	Year of Birth:	1995	is

单表查询

- 查询仅涉及一个表
  - 1.选择表中的若干列
  - 2.选择表中的若干元组
  - 3.ORDER BY子句
  - 4.聚集函数
  - 5.GROUP BY子句

选择表中的若干元组

□ 消除取值重复的行

如果没有指定DISTINCT关键词，则缺省为ALL

[例3.21] 查询选修了课程的学生学号。

SELECT Sno FROM SC;

等价于：

SELECT ALL Sno FROM SC;

执行上面的SELECT语句后，结果为：

Sno
201215121
201215121
201215121
201215122
201215122

SC

学号 Sno	课程号 Cno	成绩 Grade
201215121	1	92
201215121	2	85
201215121	3	88
201215122	2	90
201215122	3	80

选择表中的若干元组

- 指定DISTINCT关键词，去掉表中重复的行

```
SELECT DISTINCT Sno
 FROM SC;
```

执行结果：

Sno
201215121
201215122

选择表中的若干元组

表3.6 常用的查询条件

SELECT * FROM table WHERE F;　F 是查询条件

查 询 条 件	谓　词
比　较	=, >, <, >=, <=, !=, <>, !>, !<; 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

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

[例3.22] 查询计算机科学系全体学生的名单。

```
SELECT Sname
FROM    Student
WHERE  Sdept=‘CS’;
```

[例3.23]查询所有年龄在20岁以下的学生姓名及其年龄。

```
SELECT Sname,Sage
FROM    Student
WHERE  Sage < 20;
```

[例3.24]查询考试成绩有不及格的学生的学号。

```
SELECT DISTINCT Sn
FROM  SC
WHERE Grade<60;
```

② 确定范围

- 谓词： BETWEEN … AND …

NOT BETWEEN … AND …

[例3.25] 查询年龄在20~23岁（包括20岁和23岁）之间的学生的姓名、系别和年龄

```
SELECT Sname, Sdept, Sage
FROM   Student
WHERE  Sage BETWEEN 20 AND 23;
```

[例3.26] 查询年龄不在20~23岁之间的学生姓名、系别和年龄

```
SELECT Sname, Sdept, Sage
FROM   Student
WHERE Sage NOT BETWEEN 20 AND 23;
```

③ 确定集合

□ 谓词：IN <值表>，NOT IN <值表>

[例3.27]查询计算机科学系（CS）、数学系（MA）和信息系（IS）学生的姓名和性别。

```
SELECT Sname, Ssex
FROM  Student
WHERE Sdept IN ('CS','MA’,'IS' );
```

[例3.28]查询既不是计算机科学系、数学系，也不是信息系的学生的姓名和性别。

```
SELECT Sname, Ssex
FROM Student
    WHERE Sdept NOT IN ('IS','MA’,'CS' );
```

④字符匹配

□ 匹配串为含通配符的字符串

[例3.30] 查询所有姓刘学生的姓名、学号和性别。

```
SELECT Sname, Sno, Ssex
FROM Student
WHERE  Sname LIKE '刘%';
```

[例3.31] 查询姓"欧阳"且全名为三个汉字的学生的姓名。

```
SELECT Sname
FROM   Student
WHERE  Sname LIKE '欧阳__';
```

④字符匹配

[例3.32] 查询名字中第2个字为"阳"字的学生的姓名和学号。

```
SELECT Sname，Sno
FROM    Student
WHERE  Sname LIKE '__阳%';
```

[例3.33] 查询所有不姓刘的学生姓名、学号和性别。

```
SELECT Sname, Sno, Ssex
FROM    Student
WHERE  Sname NOT LIKE '刘%';
```

④字符匹配

[例] To match strings with that begin with ‘zhang’ consisting of exactly eight characters

```
SELECT SNAME

FROM STUDENT

WHERE SNAME LIKE 'zhang _ _ _’;
```

[例] To match strings that there are must be exactly four characters in the string, the END of which must be an ‘g’

```
SELECT SNAME

FROM STUDENT

WHERE SNAME LIKE ' _ _ _g’;
```

④字符匹配

- 使用换码字符将通配符转义为普通字符

[例3.34] 查询DB_Design课程的课程号和学分。

```
SELECT Cno，Ccredit
FROM    Course
WHERE  Cname LIKE 'DB\_Design' ESCAPE '\ ' ;
```

[例3.35] 查询以"DB_"开头，且倒数第3个字符为 i的课程的详细情况。

```
SELECT  *
FROM    Course
WHERE  Cname LIKE  'DB\_%i_ _' ESCAPE '\ ' ;
```

ESCAPE ' \ ' 表示“ \ ” 为换码字符

⑤ 涉及空值的查询

□ 谓词： IS NULL 或 IS NOT NULL

[例3.36] 某些学生选修课程后没有参加考试，所以有选课记录，但没 有考试成绩。查询缺少成绩的学生的学号和相应的课程号。

```
SELECT Sno，Cno
FROM    SC
WHERE  Grade IS NULL
```

[例3.37] 查所有有成绩的学生学号和课程号。

```
SELECT Sno，Cno
FROM    SC
WHERE  Grade IS NOT NULL;
```

⑥多重条件查询

- 逻辑运算符：AND和 OR来连接多个查询条件
  - AND的优先级高于OR
  - 可以用括号改变优先级

[例3.38] 查询计算机系年龄在20岁以下的学生姓名。

```
SELECT Sname
 FROM   Student
 WHERE Sdept= 'CS' AND Sage<20;
```

⑥多重条件查询

改写[例3.27]

[例3.27] 查询计算机科学系（CS）、数学系（MA）和信息系（IS）学生的姓名和性别。

```
SELECT Sname, Ssex
FROM     Student
WHERE  Sdept IN ('CS ','MA ','IS')
```

可改写为：

```
SELECT Sname, Ssex
FROM     Student
WHERE  Sdept= ' CS' OR Sdept= ' MA' OR Sdept= 'IS ';
```

单表查询

- 查询仅涉及一个表
  - 1.选择表中的若干列
  - 2.选择表中的若干元组
  - 3.ORDER BY子句
  - 4.聚集函数
  - 5.GROUP BY子句

ORDER BY子句

- ORDER BY子句
  - 可以按一个或多个属性列排序
  - 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

ORDER BY子句

学号 Sno	课程号 Cno	成绩 Grade

[例3.39]查询选修了3号课程的学生的学号及其成绩，查询结果按分数降序排列

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept

```
SELECT Sno, Grade
    FROM   SC
    WHERE  Cno= ' 3 '
    ORDER BY Grade DESC;
```

[例3.40]查询全体学生情况，查询结果按所在系的系号升序排列，同一系中的学生按年龄降序排列。

```
SELECT  *
    FROM  Student
    ORDER BY Sdept, Sage DESC;
```

单表查询

- 查询仅涉及一个表
  - 1.选择表中的若干列
  - 2.选择表中的若干元组
  - 3.ORDER BY子句
  - 4.聚集函数
  - 5.GROUP BY子句

聚集函数

- 聚集函数（AGGREGATE OR GROUPING FUNCTIONS）：
  - 统计元组个数

    COUNT(*)

  - 统计一列中值的个数

    COUNT([DISTINCT|ALL] <列名>)

  - 计算一列值的总和（此列必须为数值型）

    SUM([DISTINCT|ALL] <列名>)

聚集函数

- 聚集函数（AGGREGATE OR GROUPING FUNCTIONS）：
  - 计算一列值的平均值（此列必须为数值型）

    AVG([DISTINCT|ALL] <列名>)
  - 求一列中的最大值和最小值

    MAX([DISTINCT|ALL] <列名>)

    MIN([DISTINCT|ALL] <列名>)

聚集函数

[例3.41] 查询学生总人数。

```
SELECT COUNT(*)
FROM  Student;
```

[例3.42] 查询选修了课程的学生人数。

```
SELECT COUNT(DISTINCT Sno)
  FROM SC;
```

[例3.43] 计算1号课程的学生平均成绩。

```
SELECT AVG(Grade)
        FROM    SC
        WHERE Cno= ' 1 ';
```

聚集函数

[例3.44] 查询选修1号课程的学生最高分数。

```
SELECT MAX(Grade)
  FROM SC
  WHERE Cno='1';
```

[例3.45] 查询学生201215012选修课程的总学分数。

```
SELECT SUM(Ccredit)
        FROM  SC,Course
        WHERE Sno='201215012' AND SC.Cno=Course.Cno;
```

总结：SELECT 聚合函数　FROM　表格　 WHERE 条件; 不能用于WHERE。

GROUP 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>]…|(SELECT 语句)

[AS]<别名>

[WHERE <条件表达式>]

[GROUP BY <列名1> [HAVING <条件表达式>]]

[ORDER BY <列名2> [ASC|DESC]];

GROUP BY子句

- GROUP BY子句分组：
  - 细化聚集函数的作用对象
  - 如果未对查询结果分组，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组
  - 按指定的一列或多列值分组，值相等的为一组

SC

学号 Sno	课程号 Cno	成绩 Grade
201215121	1	92
201215121	2	85
201215121	3	88
201215122	2	90
201215122	3	80

GROUP BY子句

[例3.46] 求各个课程号及相应的选课人数。

```
SELECT Cno，COUNT(Sno)
FROM    SC
GROUP BY Cno;
```

查询结果可能为：

Cno	COUNT(Sno)
1	22
2	34
3	44
4	33
5	48

SC

学号 Sno	课程号 Cno	成绩 Grade
201215121	1	92
201215121	2	85
201215121	3	88
201215122	2	90
201215122	3	80

GROUP BY子句

- [例3.47] 查询选修了3门以上课程的学生学号。

```
SELECT Sno
FROM  SC
GROUP BY Sno
HAVING  COUNT(*) >3;
```

SC

学号 Sno	课程号 Cno	成绩 Grade
201215121	1	92
201215121	2	85
201215121	3	88
201215122	2	90
201215122	3	80

GROUP BY子句

[例3.48]查询平均成绩大于等于90分的学生学号和平均成绩

下面的语句是不对的：

```
SELECT Sno, AVG(Grade)
   FROM  SC
   WHERE AVG(Grade)>=90
   GROUP BY Sno;
```

GROUP BY子句

[例3.48]查询平均成绩大于等于90分的学生学号和平均成绩

因为WHERE子句中是不能用聚集函数作为条件表达式

正确的查询语句应该是：

```
SELECT  Sno, AVG(Grade)
   FROM  SC
   GROUP BY Sno
   HAVING AVG(Grade)>=90;
```

GROUP BY子句

- HAVING短语与WHERE子句的区别：
  - 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
  - HAVING短语作用于组，从中选择满足条件的组。

小结

- ✓数据更新：插入、修改、删除
- ✓数据查询：一般格式
- ✓单表查询：选择若干列、选择若干组、ORDER BY子句、聚集函数、GROUP BY子句

聚集函数使用规则

1) An aggregate function must be specified for an explicitly named column or expression.

2) Each aggregate function returns only one value for the set of selected rows.

3) If you apply an aggregate function to one column in a SELECT statement, you must apply column functions to any other columns specified in the same SELECT statement, unless you also use the GROUP BY clause.

4) Use GROUP BY to apply an aggregate function to a group of named columns. Any other column named in the SELECT statement must be operated on by an aggregate function.

聚集函数使用规则

5) When using the AVG, MAX, MIN and SUM functions on nullable columns, all occurrences of NULL are eliminated before applying the function.

6) You can use the DISTINCT keyword with all aggregate functions to eliminate duplicates before applying the given function. DISTINCT has no effect, however, on the MAX and MIN functions.

7) You can use the ALL keyword to indicate that duplicates should not be eliminated. ALL is the default.

8) An aggregate function can be specified in a WHERE clause only if that clause is part of a sub query of a HAVING clause. Additionally, every column name specified in the expression of the aggregate function must be a correlated reference to the same group.