



# 福昕PDF编辑器

· 永久 · 轻巧 · 自由

点击升级会员

点击批量购买



**永久使用**

无限制使用次数



**极速轻巧**

超低资源占用，告别卡顿慢



**自由编辑**

享受Word一样的编辑自由



扫一扫，关注公众号

如果查询的数据涉及两个或多个表，可以使用联接操作，称为联接查询。





联接条件



决定了两个表中哪些行是匹配的，  
以及联接结果中出现哪些列



联接类型



按照对悬浮行的不同处理方式来的











## 查询每个考生及其报考的试卷

```
FROM examinee, eeexam  
WHERE examinee.eeid = eeexam.eeid;
```

或者:

```
FROM examinee CROSS JOIN eeexam  
WHERE examinee.eeid = eeexam.eeid;
```





## 查询每个考生及其报考的试卷

```
SELECT *  
FROM examinee, eeexam  
WHERE examinee.eeid = eeexam.eeid;  
或者:  
SELECT *  
FROM examinee CROSS JOIN eeexam  
WHERE examinee.eeid = eeexam.eeid;
```



## 自然联接

即自然内联接，是在笛卡尔积的基础上选取所有同名列上取值相等的行，结果表中同名列只出现一次。





# 查询每个考生及其报考的试卷

```
SELECT *  
FROM examinee NATURAL JOIN eeexam;
```

SELECT \*



即属性内联接，是在笛卡尔积的基础上选取指定同名属性上取值相等的行，结果表中这些指定同名属性只出现一次。







# 属性联接与自然联接的区别

自然联接



所有同名属性全部取值相等



属性联接



指定其中若干同名属性取值相等

即条件内联接，是在笛卡尔积运算的基础上选取满足给定条件的行。





## 查询每个考生及其报考的试卷

```
FROM examinee JOIN eeexam ON examinee.eeid = eeexam.eeid;
```

```
examinee JOIN eeexam ON examinee.eeid = eeexam.eeid
```



左表当中的一些行在右表中没有相匹配的行;  
右表当中的一些行在左表中没有相匹配的行。





# 内联接和外联接的区别

内联接



内联接抛弃所有悬浮行



外联接



左外联接, 右外联接, 全外联接



**内联接：** 计算内联接的结果

**外联接：** 把左侧表中的悬浮行加入结果表，  
这些行中来自右侧表的属性赋为空值null







## pgSQL的联接的计算

- ※ 内联接用INNER而外联接用OUTER
- ※ 默认为INNER
- ※ LEFT、RIGHT、FULL均隐含外联接





# 查询表EREXAM和表EXAMPAPER自然左外联接

SELECT \*

FROM erexam NATURAL LEFT OUTER JOIN exampaper;

或者:

SELECT \*

FROM erexam NATURAL LEFT JOIN exampaper;

FROM erexam NATURAL LEFT JOIN exampaper



一个SELECT-FROM-WHERE语句称为一个查询块。





# 嵌套查询





## 表式嵌套

查询块可以出现在另外一个查询中表名可以出现的任何地方。







## 查询平均成绩良好( $\geq 80$ )的考生人数

```
WITH avgach(eeid,avgachieve) AS
    ( SELECT eeid,AVG(achieve)
      FROM eeexam
      GROUP BY eeid
    )
SELECT COUNT(*)
FROM avgach
WHERE avgachieve $\geq$ 80;
```

WITH子句只在包含它自己的查询语句中有效，WITH子句中的AS不能省略。



—— 查询平均成绩 $\geq 80$ 的考生人数 ——

```
SELECT COUNT(*)  
FROM ( SELECT eeid,avg(achieve)  
        FROM eeexam  
        GROUP BY eeid  
      ) AS avgach (eeid,avgachieve)  
WHERE avgachieve >= 80;
```



## 集合式嵌套

查询块也可以出现在集合能够出现的任何的地方。





查询218811011013号考生报考的试卷号和试卷名

中国大学MOOC

```
SELECT eid,ename  
FROM exampaper  
WHERE eid IN  
    (SELECT eid  
     FROM eeexam  
     WHERE eeid='218811011013');
```

# 考生答卷表 eeexam

<u>eeid</u>	<u>eid</u>	achieve
218811011013	0205000002	92
218811011013	0210000001	85
218811011013	0201020001	88
218811011117	0210000001	90
218811011117	0201020001	80



<u>eeid</u>	eeename	eesex	eeage	eedepa
218811011013	刘诗诗	男	20	历史学院
218811011014	刘诗诗	男	21	历史学院
218811011219	王琳懿	女	18	文学院
218811011220	王琳懿	女	19	文学院
218811011221	刘慧杰	女	19	文学院
218811011117	刘慧杰	女	19	教育学部
218811011025	张立帆	男	20	心理学院
218811011027	张立帆	男	19	心理学院
218811011028	刘慧杰	男	20	心理学院



# 查询所有报考了0205000002号试卷的考生详细信息

中国大学MOOC

```
SELECT *  
FROM examinee  
WHERE EXISTS  
    (SELECT *  
     FROM eeexam  
     WHERE eeid=examinee.eeid AND eid= '0205000002');
```

FROM examinee

若内层查询结果非空，则EXISTS产生“true”



若内层查询结果为空，则EXISTS产生“false”





# 查询有考生名叫刘诗诗的学院的考生平均年龄

```
SELECT eedepa, AVG(eeage)
FROM examinee
GROUP BY eedepa
HAVING eedepa IN
    (SELECT eedepa
     FROM examinee
     WHERE eename='刘诗诗');
```

如果能确定查询块只返回单行单列的单个值，查询块可以出现在单个属性名、单个表达式、单个常量，即单值表达式能够出现的任何地方。







## 查询各个院系名及其考官人数

```
SELECT dname,  
       (SELECT COUNT(*)  
        FROM examiner  
        WHERE examiner.erdepa=department.dname)  
FROM department;
```





查询与218811011028号考生同院系的考生的报考号、姓名

```
SELECT eeid, eename
FROM examinee
WHERE eeid =
    (SELECT eeid
     FROM examinee
     WHERE eeid='218811011028');
```



## 查询每个院系的平均分

上午9:37

40% 中国大学MOOC

```
SELECT (
    SELECT eedepa
    FROM examinee
    WHERE eeexam.eeid=examinee.eeid) ,
    avg(achieve)
FROM eeexam
GROUP BY (
    SELECT eedepa
    FROM examinee
    WHERE eeexam.eeid=examinee.eeid);
```



查询每个院每个考生得分，列出院名、考生名和分数，  
按院名升序排列，同院按分数升序排列

```
SELECT (
    SELECT eedepa
    FROM examinee
    WHERE eeexam.eeid=examinee.eeid) ,
    ( SELECT eename
    FROM examinee
    WHERE eeexam.eeid=examinee.eeid) ,
    achieve
FROM eeexam
ORDER BY (
    SELECT eedepa
    FROM examinee
    WHERE eeexam.eeid=examinee.eeid ),
    achieve ASC;
```





对EXAMINEE表按照EEID升序，从考生人数的四分之一处开始，列出四分之一的考生信息

```
SELECT *  
FROM examinee  
ORDER BY eeid  
LIMIT (  
    (SELECT count(*)  
    FROM examinee)/4)  
OFFSET (  
    (SELECT count(*)  
    FROM examinee)/4);
```

