数据库基础实验（2-4）实验报告

实验二 查询 (2 课时)

1.无条件查询

1. 查找 authors 表的全部信息。

```
SELECT * FROM authors
```

2. 查找其他样例表中全部信息:sales,titles,employee,publishers。

```
SELECT * FROM sales
```

```
SELECT * FROM titles
```

```
SELECT * FROM employee
```

```
SELECT * FROM publishers
```

2.简单条件查询

1. 查找 titles 表中全部书号及书名。

```
SELECT title,title FROM titles
```

2. 查找 titles 表中价格在 $10 ~ 15 元之间的书的书名。

```
SELECT title FROM titles
where price BETWEEN 10 AND 15
```

3. 查找 titles 表中书名以 T 开头的书号, 书名。

```
SELECT title,title_id from titles
where title LIKE 'T%'
```

4. 对 authors 样例表构造一个简单查询条件，进行查询。

   例如：查找authors表中城市在Oakland的作者的全名（inname+fname)

```
SELECT au_lname,au_fname from authors
where city = 'Oakland'
```

3.多条件查询

1. 查找书名起始字符为 T，价格小于 $16 元的书名及价格。

```
SELECT title,price from titles
where price<16 And title LIKE 'T%'
```

2. 查找书名起始字符不为 T 的, 价格大于 $16 元的书号, 书名及价格。

```
SELECT title,price from titles
where price>16 And title not LIKE  'T%'
```

3. 列出已出售书的书号和数量。

```
SELECT title_id,ytd_sales from titles
where ytd_sales is not null
```

4. 对 publishers 样例表构造一个多条件查询，进行查询。

   例如：查找出版号大于等于9000，国家为美国的出版商名字

```
SELECT pub_name from publishers
where pub_id>=9000 and country ='USA'
```

4．使用函数进行查找

使用 titles 表进行查询

1. 列出有多少种类型的书。

```
SELECT COUNT(DISTINCT type) as type_count from titles
```

2. 列出书的定价有多少种。

```
SELECT COUNT(DISTINCT price) as price_count from titles
```

3. 查出书价最高的书价。

```
SELECT MAX( price) as MAX_price from titles
```

4. 列出当年销量的总和。

```
SELECT SUM( ytd_sales) as sum_sales from titles
```

5. 计算这些书籍的最高书价、最低书价及平均书价。

   最高书价：

```
SELECT MAX( price) as MAX_price from titles
```

   最低书价：

```
SELECT MIN( price) as min_price from titles
```

   平均书价：

```
SELECT AVG( price) as avg_price from titles
```

6. 按出版社号分组，列出各个出版社当年销量(ytd_sales)的总合。

```
SELECT pub_id,SUM(ytd_sales) as everypub_sum_sales  from titles
GROUP BY pub_id
```

7. 计算不同的书名共有多少种；

```
SELECT COUNT(DISTINCT title) as book_count from titles
```

8. 对样例表 sales 构造一个统计查询，给出查询结果。

   例如：列出stor_id有多少种。

```
SELECT COUNT(DISTINCT stor_id) as stor_id_count from sales
```

5．得到排序的查询结果

1. 查找作者的姓、名、电话号码，并按作者姓、名排列。

```
SELECT au_id,au_lname,au_fname  from authors
ORDER BY au_lname,au_fname
```

2. 查找书名和书的价格，按书价由大到小的次序排列。

```
SELECT title,price from titles
where price is not NULL
ORDER BY price DESC
```

3. 列出烹调书（类别名含有 cook）的种类和该类的平均价格。

```
SELECT type,AVG(price) from titles
where type LIKE '%cook%'
GROUP BY type
```

4. 对其他样例表构造查询条件、排序要求，给出查询结果。

   例如：列出出版社的国家为美国的出版社名字，并按照名字进行排列(升序)

```
SELECT pub_name from publishers
where country='USA'
ORDER BY pub_name
```

6.用嵌套或连接进行查询

1. 使用样例表 titles, publishers 进行查询: 查找出版社的名称以及 所出的书名。

```
SELECT title,pub_name from titles,publishers
where titles.pub_id=publishers.pub_id
```

2. 使用样例表 authors, titleauthor, titles 进行查询: 查找作者的 姓、名和所写的书名。

```
SELECT au_lname,au_fname,title from public.titleauthor
JOIN authors  ON public.titleauthor.au_id=public.authors.au_id
JOIN public.titles ON public.titleauthor.title_id=public.titles.title_id
```

3. 从 titles,sales 中找出定单量(qty)最大的那一行定单的书的书名、 价格、定单量。

```
SELECT title,price,qty from public.titles
JOIN public.sales ON public.titles.title_id=public.sales.title_id
ORDER BY qty DESC
LIMIT 1;
```

4. 构造其他条件，在 sales 和 stores 样例表中进行连接或嵌套查询。

   例如：从sales,stores中找出payterms='ON invoice',且定单量(qty)最大那一行的所有信息。

```
SELECT * from public.sales
JOIN public.stores ON sales.stor_id=public.stores.stor_id
where sales.payterms='ON invoice'
ORDER BY qty DESC
LIMIT 1;
```

实验三 建表(2 课时)

1. 按照下面的的结构与内容建两张表。 表名分别以 T、S 开头，后面是建表人的学号(以下简记为 T、S)。 要求：先用 create table 命令建立表 T** 的初始结构（初始结构只包 括下面 T** 中前五个属性），然后再用 alter table ...add... 添加一 个属性: 出版时间，并插入相应的内容。

   表TPB23071377

```
CREATE TABLE TPB23071377(
书名 VARCHAR(100),
作者 VARCHAR(100),
书号 VARCHAR(10),
价格 FLOAT,
出版社 VARCHAR(100)
);
```

```
ALTER TABLE TPB23071377 ADD 出版时间 int;`
```

```
INSERT INTO tpb23071377 (书名, 作者, 书号, 价格, 出版社, 出版时间) VALUES
('计算机原理', '张一平', 'S3092', 20.80, '中国科技大学', 1986),
('C 语言程序设计', '李华', 'H1298', 15.30, '电子工业', 1993),
('数据库原理', '王家树', 'D1007', 22.70, '高等教育', 1987),
('计算机网络', '高明', 'S5690', 18.90, '高等教育', 1993),
('Artificial intelligence', 'P.Winston', 'D2008', 20.50, '电子工业', 1989),
('Expert systems', 'R.Ullman', 'H3067', 17.00, '清华大学', 1994),
('软件工程', '鲁廷璋', 'S2005', 35.00, '中国科技大学', 1995),
('Fortran 程序设计', '顾学峰', 'S5006', 18.00, '高等教育', 1995);
```

表SPB23071377

```
CREATE TABLE SPB23071377(
书号 VARCHAR(6),
页数 int,
库存量 int,
仓库号 INT
);

INSERT INTO spb23071377 (书号, 页数, 库存量, 仓库号) VALUES
('S3092', 304, 300, 1),
('D1007', 280, 200, 3),
('S5006', 315, 240, 2),
('S5690', 300, 300, 2),
('H1298', 210, 470, 5),
('D2008', 358, 342, 2),
('S2005', 298, 200, 2),
```

```
('H3067', 307, 510, 1);
```

2.用子查询方式建新表。 使用命令： create table <新表> as select * from <旧表> 新表名以 ST 开头, 后面为建表人学号 (简记为 ST**)。 新表内须包括“书名”和“价格”两个属性。

```
CREATE TABLE STPB23071377 as SELECT 书名,价格 from student.tpb23071377
```

3. 按“书号”建索引，索引名为 IT** (**表示建表人的学号，下同)。**

```
CREATE INDEX ITpb23071377 ON spb23071377 (书号);
```

4. 用子查询方式建视图，视图名为 VT** , 并在视图上查找所需信息。

建立价格小于20的书的书名,作者,价格,出版社视图。使用子查询的语法为：`create veiw ... as select ...`，会将子查询的结果建立试图。

```
create view VTPB23071377 (书名,作者,价格,出版社) as select 书名,作者,价格,出版社
from TPB23071377 where 价格<20;
```

5. 删除以 ST** 命名的表。

```
DROP TABLE student.stpb23071377;
```

6. 删除以 VT** 命名的视图。

```
DROP VIEW vtpb23071377;
```

7. 删除以 IT** 命名的索引。

```
DROP INDEX ITpb23071377;
```

8. 对表 T** 和 S** 进行其他操作。

创建索引可以加快数据查询速度。创建视图可以使查询更加便捷，并提高安全性。视图虽然没有表的存储结构，但对于查询语句来说，视图可以几乎当作一个表来处理。

对SPB23071377表建立一个书号索引，然后对页数大于300的书建立书号、页数、仓库号视图，并无条件查询这个视图。

```
create index ISPB23071377 on SPB23071377(书号);
create view VSPB23071377(书号,页数,仓库号) as select 书号,页数,仓库号 from SPB23071377
where 页数>300;
select * from VSPB23071377;
```

实验四 插入、删除、更新与存储过程或函数(4 课时)

1. 在 T** 表中插入一元组: Digital Image Processing, S7028, 36.00

```
INSERT INTO student.tpb23071377 (书名,书号,价格)VALUES
('Digital Image Processing', 'S7028', 36.00);
```

2. 删除书名为“Fortran 程序设计”的那个元组。

```
DELETE FROM student.tpb23071377
WHERE 书名 ='Fortran 程序设计' ;
```

3. 删除书号以 H 开头的元组。

```
DELETE FROM student.spb23071377
WHERE 书号 Like'H%'
```

4. 把书价调整到原来价格的 90 % 。

```
UPDATE student.tpb23071377
SET 价格 = 价格 * 0.9;
```

5. 把书号以 D 开头的那些书的书价减掉 2.00 元。

```
UPDATE student.tpb23071377
SET 价格=价格-2.00
where 书名 like 'D%'
```

6. 将“计算机原理”的书号改为 S1135。

```
UPDATE student.tpb23071377
SET 书号 = 'S1135'
WHERE 书名 = '计算机原理';
```

7. 对所建的表，进行其他插入、删除、更新操作各一个。 提示：每次修改表后，可用 select 查看一下修改后表中的内容，检查 是否满足要求。

   插入：

```
INSERT INTO student.spb23071377 (书号,页数,库存量,仓库号)values
('D9852',300,985,211);

select * from student.spb23071377;
```

删除:

```
DELETE FROM student.spb23071377
WHERE 库存量>=300;

select * from student.spb23071377;
```

8. 已知下面百分制成绩和 GPA 绩点、等级成绩之间的关系，建立名为 PB** (即以 PB 开头，后面是建过程人的学号)的函数/过程，用于实现对 给定的百分制成绩，输出其 GPA 绩点和等级成绩。输入参数为百分制成绩， 返回参数为绩点、等级成绩。如输出成绩不在 0~100 直接，存储过程返回 -1，否则返回 0。

```
-- ========= 第一部分: 创建存储过程 (必须单独执行) =========
CREATE PROCEDURE PB23071377
  @input_grade NUMERIC(3,0) ,
  @out_jd NUMERIC(3,1) output ,
  @out_djcj varchar(5) output
AS
BEGIN
    --成绩不在0-100范围
    if @input_grade>100 or @input_grade<0
    return -1
    --成绩在0-100范围，从表中查询
    SELECT @out_jd=jd,@out_djcj=djcj from public.cjdzb
    where @input_grade>=startfz and @input_grade <=endfz
    return 0
 END
-- ========= 第二部分: 执行存储过程 (在创建成功后执行) =========
  DECLARE @out_jd1 NUMERIC(3,1) ,
          @out_djcj1 VARCHAR(5);
   --执行存储过程
  EXECUTE PB23071377 100, @out_jd1 output, @out_djcj1 output;
  --显示结果
  select @out_jd1, @out_jdcj1;
```