sql

> github https://github.com/jbgod/--jambolt apache+php+postgresql连接，kali linux 连接postgresql，包含一些没有过滤的简单pg sql注入

SQL语法

1.语句的分类

sql语句分为DQL，DML，DDL

DQL：数据查询语句，SELECT

DML：Data Manipulation Language，数据操纵语言，插入，更新，删除数据，INSERT UPDATE DELETE

DDL：Data Definition Language，数据定义语言，用于创建，删除，以及修改表，索引等数据库对象语言。

2.语法结构

多条sql命令由分号;分隔。每条sql语句由一些列记号组成，记号包括关键字，标识符，双引号包围的标识符，常量，单引号包围的文本常量和特殊的字符组成。sql语言可以有注释

DDL语句

DDL语句是创建，修改和删除表的语句。

1.建表语句

```
CREATE TABLE table_name (
column_name1 data_type,
column_name2 data_type,
.....
);
```

建表时，可以在列定义后添加'primary key'将列设置为主键

2.删除表语句

```
DROP TABLE table_name;
```

DML语句

1.插入语句

```
INSERT INTO table_name VALUES(data1, data2, data3, .......);
```

2.更新语句

```
UPDATE table_name SET column = data +(WHERE 子句)
```

3.删除语句

```
DELETE FROM column_name WHERE column_name = data ;
```

查询语句

1.单表查询语句

select column_name1,column_name2,column_name3 from table_name;

select 后可以跟列名，也可以跟上无关的表达式。

2.过滤条件查询

where子句来指定某一条记录

3.排序

order by子句，在where子句之后，order by column_name 按照某列的值来排序

4.分组查询

GROUP BY，需要使用聚合函数 count sum等

5.表join

多表关联查询

其他sql语句

1.INSERT INTO ... SELECT语句

把数据从一张表插到另一张表

2.UNION语句

联合查询

3.TRUNCATE TABLE语句

清空表的内容。

什么是数据库？

数据库（Database）是按照数据结构来组织、存储和管理数据的仓库。

每个数据库都有一个或多个不同的 API 用于创建，访问，管理，搜索和复制所保存的数据。

我们也可以将数据存储在文件中，但是在文件中读写数据速度相对较慢。

所以，现在我们使用关系型数据库管理系统（RDBMS）来存储和管理的大数据量。所谓的关系型数据库，是建立在关系模型基础上的数据库，借助于集合代数等数学概念和方法来处理数据库中的数据。

ORDBMS（对象关系数据库系统）是面向对象技术与传统的关系数据库相结合的产物，查询处理是ORDBMS 的重要组成部分，它的性能优劣将直接影响到DBMS 的性能。

ORDBMS在原来关系数据库的基础上，增加了一些新的特性。

RDBMS 是关系数据库管理系统,是建立实体之间的联系,最后得到的是关系表。

OODBMS 面向对象数据库管理系统,将所有实体都看着对象,并将这些对象类进行封装,对象之间的通信通过消息 OODBMS 对象关系数据库在实质上还是关系数据库 。

ORDBMS 术语

在我们开始学习 PostgreSQL 数据库前，让我们先了解下 ORDBMS 的一些术语：

- **数据库:** 数据库是一些关联表的集合。
- **数据表:** 表是数据的矩阵。在一个数据库中的表看起来像一个简单的电子表格。
- **列:** 一列(数据元素) 包含了相同的数据, 例如邮政编码的数据。
- **行：**一行（=元组，或记录）是一组相关的数据，例如一条用户订阅的数据。
- **冗余**：存储两倍数据，冗余降低了性能，但提高了数据的安全性。
- **主键**：主键是唯一的。一个数据表中只能包含一个主键。你可以使用主键来查询数据。
- **外键：**外键用于关联两个表。
- **复合键**：复合键（组合键）将多个列作为一个索引键，一般用于复合索引。
- **索引：**使用索引可快速访问数据库表中的特定信息。索引是对数据库表中一列或多列的值进行排序的一种结构。类似于书籍的目录。
- **参照完整性:** 参照的完整性要求关系中不允许引用不存在的实体。与实体完整性是关系模型必须满足的完整性约束条件，目的是保证数据的一致性。

PostgreSQL 特征

- **函数**：通过函数，可以在数据库服务器端执行指令程序。
- **索引**：用户可以自定义索引方法，或使用内置的 B 树，哈希表与 GiST 索引。
- **触发器**：触发器是由SQL语句查询所触发的事件。如：一个INSERT语句可能触发一个检查数据完整性的触发器。触发器通常由INSERT或UPDATE语句触发。 多版本并发控制：PostgreSQL使用多版本并发控制（MVCC，Multiversion concurrency control）系统进行并发控制，该系统向每个用户提供了一个数据库的"快照"，用户在事务内所作的每个修改，对于其他的用户都不可见，直到该事务成功提交。
- **规则**：规则（RULE）允许一个查询能被重写，通常用来实现对视图（VIEW）的操作，如插入（INSERT）、更新（UPDATE）、删除（DELETE）。
- **数据类型**：包括文本、任意精度的数值数组、JSON 数据、枚举类型、XML 数据

  等。
- **全文检索**：通过 Tsearch2 或 OpenFTS，8.3版本中内嵌 Tsearch2。
- **NoSQL**：JSON，JSONB，XML，HStore 原生支持，至 NoSQL 数据库的外部数据包装器。
- **数据仓库**：能平滑迁移至同属 PostgreSQL 生态的 GreenPlum，DeepGreen，HAWK 等，使用 FDW 进行 ETL。

centos下PostgreSQL搭建

centos7 添加prm

```
yum install https://download.postgresql.org/pub/repos/yum/10/redhat/rhel-7-
x86_64/pgdg-centos10-10-2.noarch.rpm
```

安装postgresql的客户端，服务端

```
[root@softbank220010146128 jambolt]# yum install postgresql10

[root@softbank220010146128 jambolt]# yum install postgresql10-server
```

启动数据并初始化：

```
service postgresql-10 initdb
chkconfig postgresql-10 on
service postgresql-10 start
```

Ubuntu 安装 PostgreSQL

Ubuntu 可以使用 apt-get 安装 PostgreSQL：

```
sudo apt-get update
sudo apt-get install postgresql postgresql-client
```

安装完毕后，系统会创建一个数据库超级用户 postgres，密码为空。

切换用户postgres,打开psql

PostgreSQL数据类型

网络地址类型

PostgreSQL 提供用于存储 IPv4 、IPv6 、MAC 地址的数据类型。

用这些数据类型存储网络地址比用纯文本类型好， 因为这些类型提供输入错误检查和特殊的操作和功能。

名字	存储空间	描述
cidr	7 或 19 字节	IPv4 或 IPv6 网络
inet	7 或 19 字节	IPv4 或 IPv6 主机和网络
macaddr	6 字节	MAC 地址

在对 inet 或 cidr 数据类型进行排序的时候， IPv4 地址总是排在 IPv6 地址前面，包括那些封装或者是映射在 IPv6 地址里的 IPv4 地址， 比如 ::10.2.3.4 或 ::ffff:10.4.3.2。

XML 类型

xml 数据类型可以用于存储XML数据。 将 XML 数据存到 text 类型中的优势在于它能够为结构良好性来检查输入值， 并且还支持函数对其进行类型安全性检查。 要使用这个数据类型，编译时必须使用 **configure --with-libxml**。

xml 可以存储由XML标准定义的格式良好的"文档"， 以及由 XML 标准中的 **XMLDecl? content** 定义的"内容"片段， 大致上，这意味着内容片段可以有多个顶级元素或字符节点。 xmlvalue IS DOCUMENT 表达式可以用来判断一个特定的 xml 值是一个完整的文件还是内容片段。

创建XML值

使用函数 xmlparse: 来从字符数据产生 xml 类型的值：

```
XMLPARSE (DOCUMENT '<?xml version="1.0"?><book><title>Manual</title><chapter>...
</chapter></book>')
XMLPARSE (CONTENT 'abc<foo>bar</foo><bar>foo</bar>')
```

数组类型

PostgreSQL 允许将字段定义成变长的多维数组。

数组类型可以是任何基本类型或用户定义类型，枚举类型或复合类型。

声明数组

创建表的时候，我们可以声明数组，方式如下：

```
CREATE TABLE sal_emp (
    name            text,
    pay_by_quarter  integer[],
    schedule        text[][]
);
```

pay_by_quarter 为一位整型数组、schedule 为二维文本类型数组。

我们也可以使用 "ARRAY" 关键字，如下所示：

```
CREATE TABLE sal_emp (
   name text,
   pay_by_quarter integer ARRAY[4],
   schedule text[][]
);
```

插入值

插入值使用花括号 {}，元素在 {} 使用逗号隔开：

```
INSERT INTO sal_emp
    VALUES ('Bill',
    '{10000, 10000, 10000, 10000}',
    '{{"meeting", "lunch"}, {"training", "presentation"}}');

INSERT INTO sal_emp
    VALUES ('Carol',
    '{20000, 25000, 25000, 25000}',
    '{{"breakfast", "consulting"}, {"meeting", "lunch"}}');
```

访问数组

现在我们可以在这个表上运行一些查询。

首先，我们演示如何访问数组的一个元素。 这个查询检索在第二季度薪水变化的雇员名：

```
SELECT name FROM sal_emp WHERE pay_by_quarter[1] <> pay_by_quarter[2];

 name
-------
 Carol
(1 row)
```

数组的下标数字是写在方括弧内的。

修改数组

我们可以对数组的值进行修改：

```
UPDATE sal_emp SET pay_by_quarter = '{25000,25000,27000,27000}'
    WHERE name = 'Carol';
```

或者使用 ARRAY 构造器语法：

```
UPDATE sal_emp SET pay_by_quarter = ARRAY[25000,25000,27000,27000]
    WHERE name = 'Carol';
```

数组中检索

要搜索一个数组中的数值，你必须检查该数组的每一个值。

比如：

```
SELECT * FROM sal_emp WHERE pay_by_quarter[1] = 10000 OR
                            pay_by_quarter[2] = 10000 OR
                            pay_by_quarter[3] = 10000 OR
                            pay_by_quarter[4] = 10000;
```

另外，你可以用下面的语句找出数组中所有元素值都等于 10000 的行：

```
SELECT * FROM sal_emp WHERE 10000 = ALL (pay_by_quarter);
```

或者，可以使用 generate_subscripts 函数。例如：

```
SELECT * FROM
   (SELECT pay_by_quarter,
           generate_subscripts(pay_by_quarter, 1) AS s
      FROM sal_emp) AS foo
 WHERE pay_by_quarter[s] = 10000;
```

psql命令行操作

1.\l

查看所有数据库

```
postgres=# \l
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |   Access
privileges
-----------+----------+----------+-------------+-------------+------------------
-----
 postgres  | postgres | UTF8     | zh_CN.UTF-8 | zh_CN.UTF-8 |
 template0 | postgres | UTF8     | zh_CN.UTF-8 | zh_CN.UTF-8 | =c/postgres
   +
           |          |          |             |             |
postgres=CTc/postgres
 template1 | postgres | UTF8     | zh_CN.UTF-8 | zh_CN.UTF-8 | =c/postgres
   +
           |          |          |             |             |
postgres=CTc/postgres
(3 rows)
```

2.\c

\c database_name 连接数据库

3.\d

1 \d 查看所有表

2 \d + table_name 查看指定表的数据结构

3 \d + table_name _pkey 显示治党的索引信息

4.\d+命令

\dt 只显示匹配表

\di 只显示索引

\ds 只显示序列

\dv 只显示视图

\df 显示函数

数据类型

表 5-1　PostgreSQL 支持的数据类型分类

分类名称	说　明	与其他数据库的对比
布尔类型	PostgreSQL 支持 SQL 标准的 boolean 数据类型	与 MySQL 的 BOOL、BOOLEAN 类型相同，使用一字节存储空间
数值类型	整数类型有 2 字节的 smallint、4 字节的 int、8 字节的 bigint，十进制精确类型有 numeric，浮点类型有 real 和 double precision。还有 8 字节的货币（money）类型	无 MySQL 的 unsigned 整数类型，也无 MySQL 1 字节长的 tinyint 整数类型和 3 字节长的 mediumint 整数类型
字符类型	有 varchar(n)、char(n)、text 三种类型	PostgreSQL 中的 varchar(n) 最大可以存储 1GB，而 MySQL 中的 varchar(n) 最大只能是 64KB。PostgreSQL 中的 text 类型相当于 MysQL 中的 LONGTEXT 类型
二进制数据类型	只有一种 bytea	对应 MySQL 的 BLOB 和 LONGBLOB 类型
位串类型	位串就是一串 1 和 0 的字符串，有 bit(n)、bit varying(n) 两种	其他数据库没有此类型
日期和时间类型	有 date、time、timestamp，而 time 和 timestamp 又分是否包括时区的两种类型	在 PostgreSQL 中，可以精确到秒以下，如毫秒。而 MySQL 的时间类型最多只能精确到秒，其日期时间的范围也与 MySQL 差异较大
枚举类型	枚举类型是一种包含一系列有序静态值集合的数据类型，等于某些编程语言中的 enum 类型。	PostgreSQL 使用枚举类型前需要先使用 CREATE TYPE 创建这个类型；MySQL 也有枚举类型（ENUM）
几何类型	包括了点（point）、直线（line）、线段（lseg）、路径（path）、多边形（polygon）、圆（cycle）等类型	PostgreSQL 特有的类型，其他数据库一般没有此类型，可以认为是一种数据库内置的自定义类型
网络地址类型	有 cidr、inet、macaddr 三种类型	PostgreSQL 特有类型，其他数据库一般没有此类型，可以认为是一种数据库内置的自定义类型
数组类型	可以存储一个数组	PostgreSQL 特有类型，其他数据库一般没有此类型
复合类型	可以把已有的简单类型组合成用户自定义的类型，就如 C 语言中的结构体一样	对应其他数据库的自定义类型
xml 类型	可以存储 XML 数据的类型	N/A
json 类型	可以存储 json 类型的数据	N/A
range 类型	范围类型，可以存储范围数据	其他数据库无此类型
对象标识符类型	PostgreSQL 内部标识对象的类型，如 oid 类型、regproc 类型、regclass 类型等	N/A
伪类型	伪类型不能作为字段的数据类型，但是它可以用于声明一个函数的参数或者结果类型。有 any、anyarray、anyelement 、cstring、internal、language_handler、record 、trigger、void 、opaque	N/A
其他类型	一些不好分类的类型都放到这里，如 UUID 类型、pg_lsn 类型	N/A

类型的输入与转换

简单的数据类型

```
postgres=# select 1, 3.1415, '3.14444';
 ?column? | ?column? | ?column?
----------+----------+----------
        1 |   3.1415 | 3.14444
(1 row)
```

用类型名转换

```
postgres=# select int '1'+int '3';
 ?column?
----------
        4
(1 row)
postgres=# select bit '10101111';
   bit
----------
 10101111
(1 row)
postgres=# select date '10101111';
    date
------------
 1010-11-11
(1 row)
postgres=# select cidr '1.1.1.1';
    cidr
------------
 1.1.1.1/32
(1 row)
postgres=# select 'xff'::bit(16);
       bit
------------------
 1111111100000000
(1 row)
```

1.布尔型数据

boolean 有 true false 不带引号的 TRUE FALSE

布尔型的操作符

AND 与 OR 或 NOT 非

2.数值类型

数值型

类型名称	存储空间	描　述	范　围
smallint	2 字节	小范围整数。Oracle 中没有此类型，使用 number 代替	-215 ～ 215-1
int 或 integer	4 字节	常用的整数。Oracle 中 integer 等效于 number(38)，与此类型的意义不同	-231 ～ 231-1
bigint	8 字节	大范围的整数。Oracle 中没有此类型，使用 number 代替	-263 ～ 263-1
numeric 或 decimal	变长	用户声明的精度，精确。注意 Oracle 中叫 NUMBER，与 PostgreSQL 中的名称不一样	无限制
real	4 字节	变精度，不精确	6 位十进制数字精度
double precision	8 字节	变精度，不精确	15 位十进制数字精度
serial	4 字节	自增整数	1 ～ 2^{31}-1
bigserial	8 字节	大范围的自增整数	1 ～ 2^{63}-1

整形int smallint bigint

序列类型 sequence

字符型

类型名称	描　述
character varying(n) varchar(n)	变长，最大 1GB。存储空间为：4+ 实际的字符串长度。 与 MySQL 中的 varchar(n) 或 text(n)，以及 Oracle 中的 varchar2(n) 类似，但是在 MySQL 中 varchar 最多 64KB 长，在 Oracle 中 varchar2 最多 4000 字节，而 PostgreSQL 可以达到 1GB
character(n), char(n)	定长，不足补空白，最大 1GB。存储空间为：4+n
text	变长，无长度限制。与 MySQL 中的 LONGTEXT 类似

枚举类型

有序的静态值集合的数据类型

XML类型

xml函数处理xml数据