**UUID类型**

UUID（Universally Unique Identifier：通用唯一标识符）用于储存全局唯一标识符UUID。对分布式系统来说，UUID比序列更能保证唯一性。

UUID由32个16进制数字组成，标准格式为一组8位字符+一组4位字符+一组4位字符+一组4位字符+12组字符，每组字符由连字符连接，示例如下：

a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11

UUID还支持以大写字母、花括号包围标准格式、省略部分或所有连字符、在任意一组四位数字之后加一个连字符的方式输入，例如：

A0EEBC99-9C0B-4EF8-BB6D-6BB9BD380A11

{a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11}

a0eebc999c0b4ef8bb6d6bb9bd380a11

a0ee-bc99-9c0b-4ef8-bb6d-6bb9-bd38-0a11

{a0eebc99-9c0b4ef8-bb6d6bb9-bd380a11}

**UUID函数**

gen\_random\_uuid()

此函数返回一个版本4(随机)的UUID。这是最常用的UUID类型，适用于大多数应用程序。

**uuid-ossp插件使用**

uuid-ossp模块提供了额外的功能，用于实现生成UUIDs的其他标准算法。

安装UUID-OSSP

create extension "uuid-ossp";

插件函数

|  |  |
| --- | --- |
| 函数 | 描述 |
| uuid\_generate\_v1() | 此函数会生成v1版本的UUID。算法使用了计算机的MAC地址和时间戳。  **说明** 该UUID会泄露计算机标识和生成时间，不适合对安全性要求较高的应用。 |
| uuid\_generate\_v1mc() | 此函数会生成一个v1版本的UUID。和uuid\_generate\_v1()的区别在于uuid\_generate\_v1mc()使用的是一个随机多播MAC地址，uuid\_generate\_v1()  使用的是计算机的真实的MAC地址。 |
| uuid\_generate\_v3(namespace uuid, name text) | 此函数会生成一个v3版本的UUID。这个函数会使用指定输入名称name  在指定的命名空间namespace中生成。  指定的命名空间是调用下表中的函数uuid\_ns\_\*()返回的常量。  参数name是一个指定命名空间namespace中的标识符。  例如：  SELECT uuid\_generate\_v3(uuid\_ns\_url(), 'example.com');    **name**会使用MD5算法进行哈希，从产生的UUID中不能反向获得明文。利用这个方法生成的UUID不需要随机算法且不依赖任何运行相关的环境因素，生成过程是可重复的。 |
| uuid\_generate\_v4() | 此函数会生成一个v4版本的UUID。算法完全依靠随机数。 |
| uuid\_generate\_v5(namespace uuid, name text) | 此函数会生成一个v5版本的UUID。工作过程类似于v3版本的 UUID，但是v5版本使用的是SHA-1的哈希算法，因为SHA-1算法被认为比MD5算法更安全，所以应该尽量使用v5版本而不是v3版本。 |

返回UUID常量的函数

|  |  |
| --- | --- |
| 函数 | 描述 |
| uuid\_nil() | 代表nil的UUID常量，此处不应该看作一个真正的UUID。 |
| uuid\_ns\_dns() | 代表DNS命令空间的UUID常量。 |
| uuid\_ns\_url() | 代表URL命名空间的UUID常量。 |
| uuid\_ns\_oid() | 代表ISO对象标识符（OID）命名空间的UUID常量。    **说明** 此处OID是ASN.1的OID，和PostgreSQL用的OID没有关系。 |
| uuid\_ns\_x500() | 代表X.500识别名字（DN）命名空间的UUID常量。 |

**示例**

生成一个V1版本的UUID。

SELECT uuid\_generate\_v1();

生成一个V3版本的UUID。

SELECT uuid\_generate\_v3(uuid\_ns\_url(), 'aaa');

生成一个V4版本的UUID。

SELECT uuid\_generate\_v4();

生成一个V5版本的UUID

SELECT uuid\_generate\_v5(uuid\_ns\_url(), 'bbb');

PostgreSQL中提供了UUID类型的比较操作符和函数， 示例如下：

select uuid ‘a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11’ >= uuid ‘a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11’;

其他函数和操作符： http://www.postgres.cn/docs/14/functions-comparison.html#FUNCTIONS-COMPARISON-OP-TABLE

**pg\_lsn类型**

pg\_lsn数据类型可以被用来存储 LSN（日志序列号）数据，LSN 是一个指向WAL中的位置的指针。一些系统表中某些字段的类型就是pg\_lsn类型。

\d pg\_stat\_replication

\d pg\_replication\_slots

pg\_stat\_replication的“sent\_location”“write\_location”等字段的类型就是pg\_lsn类型， 而pg\_replication\_slots视图中的“restart\_lsn”字段也是pg\_lsn类型。

在数据库内部， LSN是一个64bit的大整数， 其输出类似如下格式：

16/B374D848

pg\_lsn类型可以使用基本的比较运算符， 如“=”“>”“<”等， 两个pg\_lsn的值可以相减得到两个WAL日志相差的字节数。

计算两个预写式日志位置间的差别

select pg\_wal\_lsn\_diff('0/8BF6C80','0/8BF6C68');

pg\_wal\_lsn\_diff以字节数计算两个预写日志位置之间的差别。

**对象标识符类型**

对象标识符（OID）被PostgreSQL用来在内部作为多个系统表的主键。 类型oid表示一个对象标识符。oid类型为一个无符号4字节整数。

使用方式

SELECT \* FROM pg\_attribute WHERE attrelid = ‘mytable’::regclass;--类型转换

SELECT \* FROM pg\_attribute WHERE attrelid = (SELECT oid FROM pg\_class WHERE relname = 'mytable');

|  |  |  |  |
| --- | --- | --- | --- |
| 名字 | 引用 | 描述 | 值示例 |
| oid | 任意 | 数字形式的对象标识符 | 564182 |
| regclass | pg\_class | 关系名字 | pg\_type |
| regcollation | pg\_collation | 排序规则名称 | "POSIX" |
| regconfig | pg\_ts\_config | 文本搜索配置 | english |
| regdictionary | pg\_ts\_dict | 文本搜索字典 | simple |
| regnamespace | pg\_namespace | 命名空间名称 | pg\_catalog |
| regoper | pg\_operator | 操作符名字 | + |
| regoperator | pg\_operator | 带参数类型的操作符 | \*(integer,​integer) or -(NONE,​integer) |
| regproc | pg\_proc | 函数名字 | sum |
| regprocedure | pg\_proc | 函数与参数类型 | sum(int4) |
| regrole | pg\_authid | 角色名 | smithee |
| regtype | pg\_type | 数据类型名称 | integer |