# PostgreSQL分表参考文档

使用postgres的模式进行分表管理如下



所有主表均在public模式下

分表分别储存在其他各自的模式下

### 添加模式语句

CREATE SCHEMA "模式名" AUTHORIZATION "拥有者";

拥有者可以省略，语句为

CREATE SCHEMA "模式名";

**模式名**为： 分表模式头+被分表表名

例：

submeter\_hcep\_device\_info\_100

分表模式头：submeter

表名：hcep\_device\_info\_100

### 创建分表语句

create table "模式名"."分表1"(check (分表1约束)) inherits(继承的主表);

create table "模式名"."分表2"(check (分表2约束)) inherits(继承的主表);

create table "模式名"."分表3"(check (分表3约束)) inherits(继承的主表);

create table "模式名"."分表4"(check (分表4约束)) inherits(继承的主表);

。。。。

**分表名**：

分表名为：主表名+分表开始数值+包括跨度

例：

hcep\_device\_info\_101\_2019\_08\_1

表名：hcep\_device\_info\_101

开始时间: 2019\_08

时间跨度为 1个月

**约束语句**：

节点字段 >= '开始数值' and 节点字段 < '结束数值'

节点字段：分表以某个字段为节点分表，该字段就是该分表模式下的节点字段

### 创建或重写触发器使用函数

**语法：**

create or replace function 函数名()

returns trigger

as

$$

**declare** --声明

变量 变量类型 := NEW.节点字段;

**Begin** --函数体开始包裹

if 条件1 then

符合条件1执行的语句(将数据添加到分表1

elsif 条件2 then

符合条件2执行的语句(将数据添加到分表2

elsif 条件3 then

符合条件3执行的语句(将数据添加到分表3

。。。。

else

默认执行的语句，通常是报错语句

end if; --判断结束

return null; --收尾防漏

**end;** --函数体结束包裹

$$

language plpgsql; --该函数所使用的语言(默认即可

**函数名**：表名\_条件\_作用

例: hcep\_device\_info\_101\_insert\_trigger

表名：hcep\_device\_info\_101

添加数据时 ：insert

作用：触发器

**声明**： 声明内定义变量，供if判断条件使用 变量名及类型尽量和节点字段相同

**NEW**: 该常量代表当前添加的数据，获取该数据字段时使用 ‘NEW.字段‘获取字段值

**判断条件**：

例：

NEW.节点字段 >= '开始节点' and NEW.节点字段<'结束节点'

**执行语句**：

例：

insert into "分表模式名"."分表名" values (NEW.\*);

**抛出异常**:

raise exception ‘错误字符’,替换变量;

例：

raise exception 'id: % created\_at out of date\_scope', NEW.id;

% : 变量替换标识符

NEW.id : 替换变量

created\_at out of date\_scope：添加时间超出日期范围

抛出的异常显示为

‘id: 1 created\_at out of date\_scope’

说明:

id 为 1 的数据 created\_at(创建时间)超出分表日期范围

**创建示例：**

create or replace function hcep\_device\_info\_101\_insert\_trigger()

returns trigger

as

$$

declare

created\_at varchar := NEW.created\_at;

begin

if NEW.created\_at >= '2019-07' and

NEW.created\_at < '2019-08' then

insert into "submeter\_hcep\_device\_info\_101".

"hcep\_device\_info\_101\_2019\_07\_1" values (NEW.\*);

elsif NEW.created\_at >= '2019-08' and NEW.created\_at<'2019-09' then

insert into "submeter\_hcep\_device\_info\_101".

"hcep\_device\_info\_101\_2019\_08\_1" values (NEW.\*);

elsif NEW.created\_at >= '2019-09' and

NEW.created\_at<'2019-10' then

insert into "submeter\_hcep\_device\_info\_101".

"hcep\_device\_info\_101\_2019\_09\_1" values (NEW.\*);

。。。。

else

raise exception

'id: % created\_at out of date\_scope', NEW.id;

end if;

return null;

end

$$

language plpgsql;

### 创建触发器，关联主表

**语法：**

create trigger 触发器名称 before insert on

主表表名 for each row execute procedure 函数();

**触发器名**：条件+表名+类型

例：insert\_hcep\_device\_info\_101\_parttion\_trigger

条件：insert(添加

主表名：hcep\_device\_info\_101

类型：parttion\_trigger

**例：**

create trigger insert\_hcep\_device\_info\_101\_parttion\_trigger before insert on hcep\_device\_info\_101 for each row execute procedure hcep\_device\_info\_101\_insert\_trigger();

### 创建分表索引

**语法:**

CREATE INDEX "索引名" ON "模式名"."分表名"

USING 索引方式("索引字段");

索引名：分表名+索引字段+index

例：hcep\_device\_info\_101\_2019\_07\_1\_name\_index

分表名：hcep\_device\_info\_101\_2019\_07\_1

索引字段：name

索引模式：btree

**例：**

CREATE INDEX

"hcep\_device\_info\_101\_2019\_07\_1\_name\_index" ON "submeter\_hcep\_device\_info\_101".

"hcep\_device\_info\_101\_2019\_07\_1"

USING btree ("name");

**\*\*分表均需要建立索引**

### 语句自动生成接口

接口地址：<http://60.30.137.2:11718/api/postgres/get/submeter/sql>

提交方式：get

参数：

**说明 变量名 默认 取值范围**

被分表表名 ：table\_name

模式归属用户 ：user postgres

开始节点 ：start

结束节点 ：end

节点字段 ：column created\_at

字段类型 ：type varchar

间隔类型 ：interval month day,month,year

报错信息 ：error created\_at out of date\_scope

分表索引字段 ：index

索引类型 ：index\_type btree btree,hash,gist,gin,spgist,brin

返回参数:

添加模式 ：submetet

创建分表 ：create

创建函数 ：function

创建关联 ：relation

创建索引 ：index