PostgreSQL数据库中特有的数据类型： Range类型， 此类型可以进行范围快速搜索， 因此在一些场景中非常有用。

Range类型用于表示范围， 如一个整数的范围、 一个时间的范围， 而范围底下的基本类型（如整数、 时间） 被称为Range类型的subtype。

**创建Range类型**

PostgreSQL中内置了一些常用的Range类型， 这些类型不需要创建就可以直接使用，具体如下。

int4range： 4字节整数的范围类型。

int8range： 8字节大整数的范围类型。

numrange： numeric的范围类型。

tsrange： 无时区的时间戳的范围类型。

tstzrange： 带时区的时间戳的范围类型。

daterange： 日期的范围类型。

还可以使用CREATE TYPE来自定义范围数据类型。

**Range类型的输入与输出**

()：表示范围内不包括此元素

[]：表示范围内包括此元素

稀疏类型的Range，其内部存储的格式为[)，连续类型的range则为精确存储。

empty：表示空， 即范围内不包括任何内容。

int4range类型为稀疏类型的Range

select '(0,6)'::int4range;

select '[0,6)'::int4range;

select 'empty'::int4range;

select int4range(1,10);

Numrange为连续类型的Range， 内部存储则是精确存储的

select '[0,6]'::numrange;

select '[0,6)'::numrange;

select '(0,6]'::numrange;

select '(0,6)'::numrange;

极值的使用：

select '[1,)'::int4range;--表示从1到int4可以表示的最大数值

select '[,1)'::int4range;--表示从int4的最小值到1的范围

**Range类型的操作符**

|  |  |  |
| --- | --- | --- |
| 操作符 | 描述 | 例子 |
| @> | 包含 | int4range(2,4) @> int4range(2,3) → t |
| <@ | 被包含 | int4range(2,4) <@ int4range(1,7) → t |
| && | 重叠 | int8range(3,7) && int8range(4,12) → t |
| << | 严格在左 | int8range(1,10) << int8range(100,110) → t |
| >> | 严格在右 | int8range(50,60) >> int8range(20,30) → t |
| &< | 扩展到左 | int8range(1,20) &< int8range(18,20) → t |
| &> | 扩展到右 | int8range(7,20) &> int8range(5,10) → t |
| -|- | 范围相邻 | numrange(1.1,2.2) -|- numrange(2.2,3.3) → t |
| + | 计算范围的并集。 | numrange(5,15) + numrange(10,20) → [5,20) |
| \* | 计算范围的交集。 | int8range(5,15) \* int8range(10,20) → [10,15) |
| - | 计算范围的差异。 | int8range(5,15) - int8range(10,20) → [5,10) |

**Range类型的函数**

|  |  |  |
| --- | --- | --- |
| 函数 | 描述 | 例子 |
| lower | 提取范围的起始值 | lower(numrange(1.1,2.2)) → 1.1 |
| upper | 提取范围的结束值 | upper(numrange(1.1,2.2)) → 2.2 |
| isempty | 范围为空吗? | isempty(numrange(1.1,2.2)) → f |
| lower\_inc | 起始值是否在范围内 | lower\_inc(numrange(1.1,2.2)) → t |
| upper\_inc | 结束值是否在范围内 | upper\_inc(numrange(1.1,2.2)) → f |

其他操作符和函数：http://www.postgres.cn/docs/14/functions-range.html

**Range类型的索引和约束**

在Range类型的列上可以创建GiST和SP-GiST索引， 创建语法如下：

CREATE INDEX index\_name ON table\_name USING gist (range\_column);

在Range的列上也可以建立约束， 使其范围永不重叠， 示例如下：

CREATE TABLE t1 (

idrange int4range,

EXCLUDE USING gist (idrange WITH &&)

);

插入数据插入数据测试：

insert into t1 values(int4range '[1,5)');--正常插入

insert into t1 values(int4range '[4,5)');--报错