PostgreSQL类型系统包含了一些特殊目的的项，它们被统称为伪类型。伪类型不能被用作一个列的数据类型，但是它可以被用来定义一个函数的参数或者结果类型。每一种可用的伪类型都有其可以发挥作用的情况，这些情况的特点是一个函数的行为并不能符合于简单使用或者返回一种特定SQL数据类型的值。

|  |  |
| --- | --- |
| 名字 | 描述 |
| any | 表示一个函数可以接受任意输入数据类型。 |
| anyelement | 表示一个函数可以接受任意数据类型。 |
| anyarray | 表示一个函数可以接受任意数组数据类型。 |
| anynonarray | 表示一个函数可以接受任意非数组数据类型。 |
| anyenum | 表示一个函数可以接受任意枚举数据类型。 |
| anyrange | 表示一个函数可以接受任意范围数据类型。 |
| anycompatible | 指示函数接受任何数据类型，并自动将多个参数提升为通用数据类型 。 |
| anycompatiblearray | 指示函数接受任何数组数据类型，并自动将多个参数提升为通用数据类型 。 |
| anycompatiblenonarray | 指示函数接受任何非数组数据类型，并将多个参数自动提升为通用。 |
| anycompatiblerange | 指示函数接受任何范围数据类型，并将多个参数自动提升为通用数据类型 。 |
| cstring | 表示一个函数接受或者返回一个非空结尾的C字符串。 |
| internal | 表示一个函数接受或返回一个服务器内部数据类型。 |
| language\_handler | 一个被声明为返回language\_handler的过程语言调用处理器。 |
| fdw\_handler | 一个被声明为返回fdw\_handler的外部数据包装器处理器。 |
| table\_am\_handler | 一种表访问方法处理程序，声明要返回 table\_am\_handler. |
| index\_am\_handler | 一个被声明为返回index\_am\_handler索引访问方法处理器。 |
| tsm\_handler | 一个被声明为返回tsm\_handler的表采样方法处理器。 |
| record | 标识一个接收或者返回一个未指定的行类型的函数。 |
| trigger | 一个被声明为返回trigger的触发器函数。 |
| event\_trigger | 一个被声明为返回event\_trigger的事件触发器函数。 |
| pg\_ddl\_command | 标识一种对事件触发器可用的 DDL 命令的表达。 |
| void | 表示一个函数不返回值。 |
| unknown | 标识一种还未被解析的类型，例如一个未修饰的字符文本。 |

用C语言编写的函数（不管是内置的还是动态装载的） 都可以声明为接受或返回上面任意一种伪数据类型。 在把伪类型用作函数参数类型时， PostgreSQL数据库本身对类型的检查就少了很多， 保证类型正确的任务就交给了写函数的开发人员。

用过程语言编写的函数不一定都能使用上面列出的全部伪类型， 具体能使用哪些伪类型需要查看相关的过程语言文档， 或者查看过程语言的实现。 通常情况下， 过程语言不支持使用any类型， 但基本都能支持使用void和record作为结果类型， 能支持多态函数的过程语言还支持使用anyarray、 anyelement、 anyenum和anynonarray类型。

伪类型internal用于声明只能在数据库系统内部调用的函数， 不能直接在SQL查询中调用它们。 如果函数至少有一个internal类型的参数， 那么就不能从SQL中调用它。 为了保留这个限制的类型安全， 我们一定要遵循以下编码规则： 对于没有任何internal参数的函数， 不要把返回类型创建为internal类型。