日期/时间类型除包括不同日期/时间范围和精度的类型外， 还包括了时间间隔类型。

# 日期/时间类型介绍

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名字 | 存储尺寸 | 描述 | 最小值 | 最大值 | 解析度 |
| timestamp [ (*p*) ] [ without time zone ] | 8字节 | 包括日期和时间（无时区） | 4713 BC | 294276 AD | 1微秒 |
| timestamp [ (*p*) ] with time zone | 8字节 | 包括日期和时间，有时区 | 4713 BC | 294276 AD | 1微秒 |
| date | 4字节 | 日期（没有一天中的时间） | 4713 BC | 5874897 AD | 1日 |
| time [ (*p*) ] [ without time zone ] | 8字节 | 一天中的时间（无日期） | 00:00:00 | 24:00:00 | 1微秒 |
| time [ (*p*) ] with time zone | 12字节 | 仅仅是一天中的时间  （没有日期），带有时区 | 00:00:00+1459 | 24:00:00-1459 | 1微秒 |
| interval [ *fields* ] [ (*p*) ] | 16字节 | 时间间隔 | -178000000年 | 178000000年 | 1微秒 |

注意： 时间类型可以精确到秒以下， time、 timestamp、 interval接受一个可选的精度值p以指明秒中小数部分的位数。 p的取值范围是0～6，如果不指定精度，默认为6。

# 日期输入

在SQL中， 任何日期或者时间的文本输入都需要由“日期/时间”类型加单引号括起来的字符串组成， 语法如下：

type [ (p) ] 'value‘

日期和时间的输入几乎可以是任何合理的格式， 包括ISO-8601格式、 SQL-兼容格式、 传统的Postgres格式及其他形式。 对于一些格式， 日期输入中的月和日可能会使人产生疑惑， 因此系统支持自定义这些字段的顺序。 如果DateStyle参数默认为“MDY”， 则表示按“月-日-年”的格式进行解析， 如果参数设置为“DMY”， 则按照“日-月-年”的格式进行解析， 设置为， “YMD”表示按照“年-月-日”的格式进行解析。 示例如下：

show datestyle;--查看日期格式

set datestyle=‘YMD’;--设置日期格式

create table t1(col1 date);

insert into t1 values(date '12-10-10');

select \* from t1;

更多的日期输入示例

|  |  |
| --- | --- |
| 例子 | 描述 |
| 1999-01-08 | ISO 8601; 任何模式下的1月8日 （推荐格式） |
| January 8, 1999 | 在任何datestyle输入模式下都无歧义 |
| 1/8/1999 | MDY模式中的1月8日；DMY模式中的8月1日 |
| 1/18/1999 | MDY模式中的1月18日；在其他模式中被拒绝 |
| 01/02/03 | MDY模式中的2003年1月2日； DMY模式中的2003年2月1日； YMD模式中的2001年2月3日 |
| 1999-Jan-08 | 任何模式下的1月8日 |
| Jan-08-1999 | 任何模式下的1月8日 |
| 08-Jan-1999 | 任何模式下的1月8日 |
| 99-Jan-08 | YMD模式中的1月8日，否则错误 |
| 08-Jan-99 | 1月8日，除了在YMD模式中错误 |
| Jan-08-99 | 1月8日，除了在YMD模式中错误 |
| 19990108 | ISO 8601; 任何模式中的1999年1月8日 |
| 990108 | ISO 8601; 任何模式中的1999年1月8日 |
| 1999.008 | 年和一年中的日子 |
| J2451187 | 儒略日期 |
| January 8, 99 BC | 公元前99年 |

对于中国人来说， 最好使用“-”， 然后以“年-月-日”的格式输入日期。

# 时间输入

输入时间时需要注意时区的输入。 time被认为是time without time zone的类型， 这样即使字符串中有时区也会被怱略， 示例如下：

select time ‘04:05:06’;--不带时区

select time ‘04:05:06 PST’;--不带时区

select time with time zone‘04:05:06 PST’;--带时区

时间字符串可以使用冒号作分隔符， 即输入格式为“hh:mm:ss”， 如“10:23:45”， 也可以不用分隔符， 如“102345”表示10点23分45秒。

更多的时间类型的输入示例

|  |  |
| --- | --- |
| 例子 | 描述 |
| 04:05:06.789 | ISO 8601 |
| 04:05:06 | ISO 8601 |
| 04:05 | ISO 8601 |
| 040506 | ISO 8601 |
| 04:05 AM | 和04:05一样，AM并不影响值 |
| 04:05 PM | 和16:05一样，输入的小时必须为 <= 12 |
| 04:05:06.789-8 | ISO 8601 |
| 04:05:06-08:00 | ISO 8601 |
| 04:05-08:00 | ISO 8601 |
| 040506-08 | ISO 8601 |
| 04:05:06 PST | 缩写指定的时区 |
| 2003-04-12 04:05:06 America/New\_York | 全名指定的时区 |

最好不要用时区缩写来表示时区， 因为这样有可能给阅读者带来困扰， 如CST时间有可能有以下几种含义：

Central Standard Time (USA) UT-6:00， 即美国标准时间。

Central Standard Time (Australia) UT+9:30， 即澳大利亚标准时间。

China Standard Time UT+8:00， 即中国标准时间。

Cuba Standard Time UT-4:00， 即古巴标准时间。

# 特殊值

为方便起见， PostgreSQL中用了一些特殊的字符串输入值表示特定的意义， 具体见表

|  |  |  |
| --- | --- | --- |
| 输入串 | 合法类型 | 描述 |
| epoch | date, timestamp | 1970-01-01 00:00:00+00（Unix系统时间0）  SELECT EXTRACT(EPOCH FROM NOW()); |
| infinity | date, timestamp | 比任何其他时间戳都晚 |
| -infinity | date, timestamp | 比任何其他时间戳都早 |
| now | date, time, timestamp | 当前事务的开始时间 |
| today | date, timestamp | 今日午夜 (00:00) |
| tomorrow | date, timestamp | 明日午夜 (00:00) |
| yesterday | date, timestamp | 昨日午夜 (00:00) |
| allballs | time | 00:00:00.00 UTC |

# 日期/时间操作符

|  |  |  |
| --- | --- | --- |
| 操作符 | 描述 | 例子 |
| date + integer → date | 给日期加上天数 | date '2001-09-28' + 7 → 2001-10-05 |
| date + interval → timestamp | 为日期添加时间间隔 | date '2001-09-28' + interval '1 hour' → 2001-09-28 01:00:00 |
| date + time → timestamp | 给日期添加时间 | date '2001-09-28' + time '03:00' → 2001-09-28 03:00:00 |
| date - date → integer | 减去日期，生成天数 | date '2001-10-01' - date '2001-09-28' → 3 |
| date - integer → date | 从日期中减去天数 | date '2001-10-01' - 7 → 2001-09-24 |
| date - interval → timestamp | 日期减去时间间隔 | date '2001-09-28' - interval '1 hour' → 2001-09-27 23:00:00 |
| time - time → interval | 减去时间 | time '05:00' - time '03:00' → 02:00:00 |
| time - interval → time | 时间减去时间间隔 | time '05:00' - interval '2 hours' → 03:00:00 |
| timestamp - interval → timestamp | 从时间戳中减去时间间隔 | timestamp '2001-09-28 23:00' - interval '23 hours' → 2001-09-28 00:00:00 |
| timestamp - timestamp → interval | 减去时间戳 | timestamp '2001-09-29 03:00' - timestamp '2001-07-27 12:00' → 63 days 15:00:00 |

日期/时间函数

|  |  |  |
| --- | --- | --- |
| 函数 | 描述 | 例子 |
| age ( timestamp, timestamp ) → interval | 第一个时间减去第二个时间的结果 | age(timestamp '2001-04-10', timestamp '1957-06-13') → 43 years 9 mons 27 days |
| age ( timestamp ) → interval | 从 current\_date 减去参数 | age(timestamp '1957-06-13') → 62 years 6 mons 10 days |
| clock\_timestamp ( ) → timestamp with time zone | 实时时钟的当前时间戳（在语句执行期间变化） | clock\_timestamp() → 2019-12-23 14:39:53.662522-05 |
| current\_date → date | 当前日期 | current\_date → 2019-12-23 |
| current\_time → time with time zone | 一天中的当前时间 | current\_time → 14:39:53.662522-05 |
| current\_time ( integer ) → time with time zone | 一天中的当前时间；有限精度 | current\_time(2) → 14:39:53.66-05 |
| current\_timestamp → timestamp with time zone | 当前日期和时间 (当前事务的开始) | current\_timestamp → 2019-12-23 14:39:53.662522-05 |
| current\_timestamp ( integer ) → timestamp with time zone | 当前日期和时间 (当前事务的开始) | current\_timestamp(0) → 2019-12-23 14:39:53-05 |
| localtime → time | 一天中当前时间 | localtime → 14:39:53.662522 |
| localtime ( integer ) → time | 一天中的当前时间，有限精度 | localtime(0) → 14:39:53 |
| localtimestamp → timestamp | 当前日期和时间（当前事务的开始） | localtimestamp → 2019-12-23 14:39:53.662522 |
| localtimestamp ( integer ) → timestamp | 当前日期和时间（当前事务的开始）；有限精度 | localtimestamp(2) → 2019-12-23 14:39:53.66 |
| now ( ) → timestamp with time zone | 当前日期和时间(当前事务的开始) | now() → 2019-12-23 14:39:53.662522-05 |
| statement\_timestamp ( ) → timestamp with time zone | 当前日期和时间(当前语句的开始) | statement\_timestamp() → 2019-12-23 14:39:53.662522-05 |
| timeofday ( ) → text | 当前的日期和时间 （类似 clock\_timestamp, 但是采用 text 字符串） | timeofday() → Mon Dec 23 14:39:53.662522 2019 EST |
| transaction\_timestamp ( ) → timestamp with time zone | 当前日期和时间(当前事务的开始) | transaction\_timestamp() → 2019-12-23 14:39:53.662522-05 |
| to\_timestamp ( double precision ) → timestamp with time zone | 将Unix纪元转换为带时区的时间戳(从1970-01-01 00:00:00+00开的的秒) | to\_timestamp(1284352323) → 2010-09-13 04:32:03+00 |

其他日期/时间操作符和函数：http://www.postgres.cn/docs/14/functions-datetime.html

返回当前事务开始的时间/日期：

CURRENT\_DATE

CURRENT\_TIME

CURRENT\_TIMESTAMP

CURRENT\_TIME(precision)

CURRENT\_TIMESTAMP(precision)

now()

不带时区：

LOCALTIME--不带时区

LOCALTIMESTAMP--不带时区

CURRENT\_TIME、 CURRENT\_TIMESTAMP、 LOCALTIME、 LOCALTIMESTAMP可以选择性地给予一个精度参数， 该精度会导致结果的秒数域被四舍五入到指定的小数位。 如果没有精度参数， 将给予所能得到的全部精度。

返回实时时间的函数：

statement\_timestamp()

clock\_timestamp()

timeofday()

# extract函数

extract函数格式如下：

extract (field FROM source)

extract函数可以从日期/时间数值中抽取子域， 比如年、月、日、时、分、秒等。

|  |  |  |
| --- | --- | --- |
| Field值 | 描述 | 例子 |
| year | 年 | SELECT EXTRACT(year FROM now()); |
| month | 月 | SELECT EXTRACT(month FROM now()); |
| day | 日 | SELECT EXTRACT(day FROM now()); |
| hour | 时 | SELECT EXTRACT(hour FROM now()); |
| minute | 分 | SELECT EXTRACT(minute FROM now()); |
| second | 秒 | SELECT EXTRACT(second FROM now()); |
| week | 周 | SELECT EXTRACT(week FROM now()); |
| doy | 一年中的第几天 | SELECT EXTRACT(doy FROM now()); |