二进制类型可以存放任意数据，PostgreSQL中只有一种二进制类型： bytea类型。

二进制字符串是一个字节序列。 二进制字符串和普通字符串的区别有两个：

第一， 二进制字符串可以存储字节零值， 以及其他“不可打印”的字节（定义在32到126范围之外的字节） 。 普通字符串不允许字节零值， 而且也不允许存储那些不符合选定的字符集编码的非法字节值或者字节序列。

第二， 对二进制字符串的处理实际上就是对字节的处理，而对字符串的处理， 则取决于区域设置。 简单说， 二进制字符串适于存储那些程序员认为是“原始字节”的数据， 比如图片内容， 而字符串则适合存储文本。

**二进制数据类型表示**

bytea类型支持两种用于输入和输出的格式：“十六进制”格式和PostgreSQL的历史的“转义”格式。输出格式则取决于配置参数bytea\_output，其默认值为十六进制。

**bytea的十六进制格式**

整个串以序列\x开头（用以和转义格式区分）。

例子：

SELECT '\x61';

**bytea的转义格式**

转义格式采用将二进制串表示成ASCII码的方法，要转义一个字节，需要把它的数值转换成对应的三位八进制数， 并且加1个前导反斜杠。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 十进制字节值 | 描述 | 转义输入表示 | 例子 | 十六进制表示 |
| 0 | 0字节 | '\000' | '\000'::bytea | \x00 |
| 39 | 单引号 | ''''或'\047' | ''''::bytea | \x27 |
| 92 | 反斜线 | '\\'或'\134' | '\\'::bytea | \x5c |
| 0到31和127到255 | “不可打印的”字节 | '\***xxx'***（八进制值） | '\001'::bytea | \x01 |

**bytea输出转义字节**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 十进制字节值 | 描述 | 转义的输出表示 | 例子 | 输出结果 |
| 92 | 反斜线 | \\ | '\134'::bytea | \\ |
| 0到31和127到255 | “不可打印的”字节 | \***xxx***（八进制值） | '\001'::bytea | \001 |
| 32到126 | “可打印的”字节 | 客户端字符集表示 | '\176'::bytea | ~ |

**二进制数据类型的函数**

|  |  |  |
| --- | --- | --- |
| 函数 | 描述 | 例子 |
| || | 连接两个二进制字符串。 | '\x123456'::bytea || '\x789a00bcde'::bytea → \x123456789a00bcde |
| bit\_length | 返回二进制字符串中的位数 (8 倍于 octet\_length). | bit\_length('\x123456'::bytea) → 24 |
| octet\_length | 返回二进制字符串中的字节数。 | octet\_length('\x123456'::bytea) → 3 |
| position | 返回之串的位置 | position('\x5678'::bytea in '\x1234567890'::bytea) → 3 |
| substring | 抽取子字符串 | substring('\x1234567890'::bytea,3,2) → \x5678 |
| get\_byte | 从二进制字符串中提取 n‘th 字节。 | get\_byte('\x1234567890'::bytea, 4) → 144 |
| set\_byte | 设置二进制字符串中的 n‘th 字节到 newvalue。 | set\_byte('\x1234567890'::bytea, 4, 64) → \x1234567840 |
| length | 返回二进制字符串中的字节数 | length('\x1234567890'::bytea) → 5 |

其他函数参考这里：http://www.postgres.cn/docs/14/functions-binarystring.html