Встроенные функции Oracle

Функции для работы со строками в Oracle.

Для упрощения работы со строками имеется ряд встроенных функций, что значительно облегчает такие операции как преобразование строк к данным других типов, поиск подстроки в строке, определение длины строки и т. д. В данной статье мы рассмотрим самые распространенные функции для работы со строками.

1) Функция определения длины строки LENGTH(строка), возвращает количество символов в строке, включая концевые пробелы.

SELECT LENGTH(‘string ’) FROM DUAL вернет значение 7.

2) Функции преобразования регистров символов UPPER(строка), LOWER(строка), INITCAP(строка). Для преобразования символов к верхнему регистру используется функция UPPER().

SELECT UPPER(‘string’) FROM DUAL вернет STRING.

Если необходимо преобразовать символы строки к нижнему регистру используется функция LOWER().

SELECT LOWER(‘STrinG’) FROM DUAL

вернет string.

Функция INITCAP преобразовывает каждый первый символ слова к верхнему регистру, а все остальные символы к нижнему при условии, что символ-разделитель между словами пробел.

SELECT INITCAP(‘string1 string2’) FROM DUAL

вернет строку String1 String2.

3) Функции для обрезания начальных и концевых пробелов LTRIM(строка), RTRIM(строка), TRIM(строка). Соответственно первая функция обрезает все начальные пробелы строки, вторая – все концевые, а третья все начальные и концевые.

SELECT LTRIM(‘ str1’) FROM DUAL вернет строку str1,  
SELECT RTRIM(‘str2 ’) FROM DUAL вернет строку str2,  
SELECT TRIM(‘ str3 ’) FROM DUAL вернет строку str3.

4) Функция замены части строки другой строкой REPLACE(исходная\_строка, заменяемая\_подстрока, заменяющая\_подстрока). Для большей ясности рассмотрим пример, в некотором текстовом поле таблицы хранится число. Причем символ-разделитель между целой и дробной частью в некоторых полях «.», а нам для дальнейшей обработки данных нужно, чтобы он во всех полях должен быть «,». Для этого воспользуемся функцией REPLACE следующим образом. REPLACE(field1, ’.’, ’,’) и все символы «.» в поле field будут заменены на символ «,».

SELECT REPLACE(‘My\_string’,’\_’,’@’) FROM DUAL вернет строку My@string.

5) Функции преобразования данных к другим типам данных. TO\_CHAR(число) преобразует число в текст. TO\_NUMBER(строка) преобразует текст в число. TO\_DATE(строка, формат\_даты) преобразует строку в дату определенного формата.

SELECT TO\_CHAR(123) FROM DUAL вернет строку 123,  
SELECT TO\_NUMBER(‘12345’) FROM DUAL вернет число 12345,  
SELECT TO\_DATE(’01.01.2010’,’dd.mon.yyyy’) FROM DUAL вернет дату 01.JAN.2010.

6) Функция определения вхождения подстроки в строку INSTR(исходная\_строка, подстрока, номер\_символа). Даная функция позволяет определять номер символа в исходной строке с которого начинается искомая подстрока (если такая есть). Иначе возвращается 0. Например нам нужно определить все должности в таблице Table1, в наименовании которых встречается подстрока «менеджер». Для этого вполне подойдет следующий оператор

SELECT \* FROM TABLE1 WHERE INSTR(POST, ‘менеджер’, 1) > 0.

То есть оператор SELECT выведет только те записи из таблицы TABLE1 где искомая подстрока «менеджер» будет найдена. Причем поиск будет осуществляться с первого символа. Если поиск нужно осуществлять с другой позиции, то номер символа для начала поиска указывается в третьем параметре.

SELECT INSTR(‘Small string’, ‘string’, 1) FROM DUAL вернет значение 7,  
SELECT INSTR(‘Small string’, ‘String’, 1) FROM DUAL вернет значение 0.

7) Функция выделения в исходной строке подстроки SUBSTR(исходная\_строка, номер\_начального\_символа, количество\_символов). Рассмотрим такой пример, в пользовательской таблице хранится адрес в виде наименование населенного пункта, название улицы, номер дома. Причем мы точно знаем, что для наименования населенного пункта отводится строго 20 символов (если наименовании населенного пункта меньше чем 20 символов, то остальная часть заполняется пробелами), для наименования улицы 30 символов, для номера дома 3 символа. Далее нам необходимо перенести все адреса из нашей таблицы в другую и при этом все 3 компонента адреса должны быть в разных полях. Для выделения компонент адреса применим функцию SUBSTR().

SELECT SUBSTR(TABLE\_1.ADDRESS, 1, 20) CITY, SUBSTR(TABLE\_1.ADDRESS, 21, 30) STREET, SUBSTR(TABLE\_1.ADDRESS, 52, 3) TOWN FROM TABLE\_1

Конечно для переноса данных необходимо воспользоваться оператором INSERT, но для понимания работы функции SUBSTR вполне подойдет рассмотренный пример.  
SELECT SUBSTR(‘My\_string’, 4, 3) FROM DUAL вернет строку str.

Рассмотренные выше функции можно использовать во входных параметрах. Так если нам нужно выделить все символы, после какого-то определенного, то в функцию SUBSTR можно передать номер искомого символа из функции INSTR. Например если нужно перенести все символы из поля таблицы, которые расположены после «,» то можно использовать такую конструкцию  
SELECT SUBSTR(My\_string, INSTR(My\_string, ‘,’, 1), LENGTH(My\_string)- INSTR(My\_string, ‘,’, 1)+1) FROM DUAL.  
Для определения начального символа мы вызываем функцию INSTR(), которая вернет номер символа первого вхождения подстрок «,». Далее мы определяем количество символов до конца строки как разницу длины строки и номера первого вхождения подстроки.

8) Для определения кода символа используется функция ASCII(строка), которая возвращает код 1 символа строки. Например

SELECT ASCII(W) FROM DUAL вернет значение 87.

9) Обратная функция преобразования кода символа в символ CHR(число).

SELECT CHR(87) FROM DUAL вернет символ W.

Функции для работы с числами в Oracle.

В СУБД Oracle имеется ряд функций для работы с числами. К ним относятся функции возведение числа в степень POWER(), округление ROUND() и т. д.

1) Функция ABS(число) возвращает абсолютное значение аргумента.  
SELECT ABS(-3) FROM DUAL вернет значение 3.

2) Функция CEIL(число) возвращает наименьшее целое, большее или равное переданному параметру.  
SELECT CEIL(4.5) FROM DUAL вернет значение 5.

3) Функция FLOOR(число) возвращает наибольшее целое, меньшее или равное переданному параметру.   
SELECT FLOOR(3.8) FROM DUAL вернет значение 3.

4) Функция MOD(число\_1, число\_2) возвращает остаток от деления первого параметра на второй.  
SELECT MOD(5, 3) FROM DUAL вернет значение 2. Примечание. Если второй параметр равен 0, то функция возвращает первый параметр.

5) Функция округления ROUND(число\_1, число\_2). Округляет первый переданный параметр до количества разрядов, переданного во втором параметре. Если второй параметр не указан, то он принимается равным 0, то есть округление производится до целого значения. Примеры  
SELECT ROUND(101.34) FROM DUAL вернет значение 101,  
SELECT ROUND(100.1268, 2) FROM DUAL вернет значение 100.13  
SELECT ROUND(1234000.3254, -2) FROM DUAL вернет значение 1234000,  
SELECT ROUND(-100.122, 2) FROM DUAL вернет значение -100.12.

6) Функция усечения значения TRUNC(число\_1, число\_2). Возвращает усеченное значение первого параметра до количества десятичных разрядов, указанного во втором параметре. Примеры  
SELECT TRUNC(150.58) FROM DUAL вернет значение 150  
SELECT TRUNC(235.4587, 2) FROM DUAL вернет значение 235.45  
SELECT TRUNC(101.23, -1) FROM DUAL вернет значение 100

7) В СУБД Oracle имеется ряд тригонометрических функций SIN(число), COS(число), TAN(число) и обратные им ACOS(число), ASIN(число), ATAN(число). Они возвращают значение соответствующей названию тригонометрической функции. Для прямых функции параметром является значение угла в радианах, а для обратных – значение функции. Примеры  
SELECT COS(0.5) FROM DUAL вернет значение 0.877582561890373  
SELECT SIN(0.5) FROM DUAL вернет значение 0.479425538604203  
SELECT TAN(0.5) FROM DUAL вернет значение 0.546302489843791  
SELECT ACOS(0.5) FROM DUAL вернет значение 1.0471975511966  
SELECT ASIN(0.5) FROM DUAL вернет значение 0.523598775598299  
SELECT ATAN(0.5) FROM DUAL вернет значение 0.463647609000806

8) Гиперболические функции. SINH(число),  
COSH(число), TANH(число). SINH() возвращает гиперболический синус переданного параметра, COSH() возвращает гиперболический косинус переданного параметра, TANH() возвращает гиперболический тангенс переданного параметра. Примеры  
SELECT COSH(0.5) FROM DUAL вернет значение 1.12762596520638  
SELECT SINH(0.5) FROM DUAL вернет значение 0.521095305493747 SELECT TANH(0.5) FROM DUAL вернет значение 0.46211715726001

9) Функция возведения в степень POWER(число\_1, число\_2). Примеры  
SELECT POWER(10, 2) FROM DUAL вернет значение 100  
SELECT POWER(100, -2) FROM DUAL вернет значение 0.0001

10) Логарифмические функции. LN(число) возвращает натуральный логарифм переданного параметра, LOG(число\_1, число\_2) возвращает логарифм второго переданного параметра по основанию, переданному первом параметре. Причем первый параметр должен быть больше нуля и не равен 1. Примеры  
SELECT LN(5) FROM DUAL вернет значение 1.6094379124341  
SELECT LOG(10, 3) FROM DUAL вернет значение 0.477121254719662

11) Функция извлечения квадратного корня SQRT(число). Пример  
SELECT SQRT(4) FROM DUAL вернет значение 2.

12) Функция возведение числа е в степень EXP(число). Пример  
SELECT EXP(2) FROM DUAL вернет значение 7.38905609893065.

Функции для работы с датами в Oracle

На практике очень часто необходимо анализировать данные в виде дат, производить некоторые операции над ними, изменять формат. Все эти операции уже реализованы в виде встроенных функций. Рассмотрим самые основные из них.

1) ADD\_MONTHS(дата, количество\_месяцев) возвращает дату, отстоящую от даты, переданной в первом параметре на количество месяцев, указанном во втором параметре. Примеры  
SELECT ADD\_MONTHS(’01-JAN-2010’, 2) FROM DUAL вернет дату ’01.03.2010’  
SELECT ADD\_MONTHS(’01-JAN-2010’, -3) FROM DUAL вернет дату ’01.10.2009’  
SELECT ADD\_MONTHS(’30-JAN-2010’, 1) FROM DUAL вернет дату ’28.02.2010’

2) Для определения текущей даты и времени применяется функция SYSDATE. Область применения данной функции намного шире чем может показаться на первый взгляд. В первую очередь это контроль за вводом данных в БД. Во многих таблицах выделяется отдельное поля для сохранения даты последнего внесения изменений. Также очень удобно контролировать некие входные параметры для отчетов, особенно если они не должны быть больше чем текущая дата. Помимо даты данная функция возвращает еще и время с точностью до секунд. Пример  
SELECT SYSDATE FROM DUAL вернет дату ‘22.05.2010 14:51:20’

3) Если необходимо определить последний день месяца, то для этого вполне подойдет функции LAST\_DAY(дата). Её можно использовать для определения количества дней, оставшихся в месяце.  
SELECT LAST\_DAY(SYSDATE) – SYSDATE FROM DUAL.  
В результате выполнения данного оператора будет выведено количество дней от текущей даты до конца месяца. Пример  
SELECT LAST\_DAY(’15-FEB-2010’) FROM DUAL вернет дату ’28.02.2010’.

4) Функция для определения количества месяцев между датами MONTHS\_BETWEEN(дата\_1, дата\_2). Примеры  
SELECT MONTHS\_BETWEEN(’01-JUL-2009’, ’01-JAN-2010’) FROM DUAL вернет значение -6  
SELECT MONTHS\_BETWEEN(’01-JUL-2009’, ’10-JAN-2010’) FROM DUAL вернет значение -6.29032258064516.   
Примечание. Если дни месяцев совпадают, то функция возвращает целое число, в противном случае результат будет дробным, причем количество дней в месяце будет принято 31.

5) Функция NEXT\_DAY(дата, день\_недели) позволяет определить следующую дату от даты, переданной в первом параметре, которая соответствует дню недели, переданном во втором параметре. Пример  
SELECT NEXT\_DAY(’01-JUL-2009’, ’mon’) FROM DUAL вернет дату ‘06.07.2009’, то есть следующий понедельник после 1 июля 2009 наступил 6 числа.

|  |  |
| --- | --- |
| **Значение параметра** | **День недели** |
| mon | Понедельник |
| tue | Вторник |
| wed | Среда |
| thu | Четверг |
| fri | Пятница |
| sat | Суббота |
| sun | воскресенье |

6) Округление даты ROUND(дата, формат). Второй параметр не обязателен, если его не указывать, то он принимается за ‘DD’, то есть округление будет произведено до ближайшего дня. Примеры   
SELECT ROUND(SYSDATE) FROM DUAL вернет дату ‘23.05.2010’  
SELECT ROUND(SYSDATE, MONTH) FROM DUAL вернет дату ‘01.06.2010’, округляется до ближайшего первого дня месяца.

**Запрос(примеры) на вычисление глубины заданной от даты относительно текущего времени:**

SELECT TO\_DATE('30102013','DDMMYYYY')- TO\_DATE('27102013','DDMMYYYY') FROM DUAL

SELECT TO\_DATE(SYSDATE) - TO\_DATE('27042013','DDMMYYYY') DEPTH FROM DUAL SELECT TO\_DATE(SYSDATE) - (SELECT TRUNC(MIN(DQ\_RUN\_END\_TIME))FROM DQ\_PARAMS)FROM DUAL

**SQL Server:**

Select DATEDIFF(DAY,Getdate()-1,Getdate()) получение разницы в днях

Select Convert(datetime,Convert(varchar,Getdate(),104),104)вывод текущей даты без времени

|  |  |
| --- | --- |
| **Формат** | **Единица округления** |
| СС, SCC | Век |
| SYYYY, YYYY, YEAR | Год |
| Q | Квартал |
| MM, MONTH | Месяц |
| WW | Тот же день недели, что и первый день года |
| W | Тот же день недели, что и первый день месяца |
| DD, J | День |
| Day, DY | Первый день недели |
| HH, HH12, HH24 | Час |
| MI | Минута |

7) Усечение даты. Функция TRUNC(дата, формат). Также как и рассмотренная выше может не иметь второго параметра. В таком случае усечение будет производиться до ближайшего дня. Примеры   
SELECT TRUNC(SYSDATE) FROM DUAL вернет дату ’22.05.2010’  
SELECT TRUNC(SYSDATE, ‘WW’) FROM DUAL вернет дату ’01.05.2010’  
SELECT TRUNC(SYSDATE, ‘Day’) FROM DUAL вернет дату ‘16.05.2010’.

Функции преобразования данных в Oracle

Данный раздел посвящен рассмотрению преобразования данных в различные форматы. На практике довольно распространены ситуации, когда необходимо строковые величины рассматривать как числа и наоборот. Несмотря на небольшое количество функции их возможностей вполне хватает для решения очень сложных прикладных задач.

1) TO\_CHAR(данные, формат). На первый взгляд синтаксис довольно прост, но за счет второго параметра можно очень точно описать в какой формат преобразовать данные. Итак в строку можно преобразовать как дату, так и числовое значение. Рассмотрим вариант преобразования даты к строке. Значения самых распространенных форматов приведены в таблице, более полная информация содержится в технической документации.

|  |  |
| --- | --- |
| **Формат** | **Описание формата** |
| D | День недели |
| DD | День месяца |
| DDD | День года |
| MM | Номер месяца |
| MON | Сокращенное название месяца |
| MONTH | Полное название месяца |
| Q | Квартал |
| YY, YYY, YYYY | Год |
| HH, HH12, HH24 | Час |
| MI | Минут |
| SS | Секунда |

Таблица значений форматов для преобразования числа в строку.

|  |  |
| --- | --- |
| **Формат** | **Описание формата** |
| 99D9 | Указание позиции разделителя десятичной точки. Число девяток соответствует максимальному количеству цифр |
| 999G99 | Указание позиции группового разделителя |
| 99,999 | Возвращает запятую в указанной позиции |
| 99.999 | Возвращает точку в указанной позиции |
| 99V9999 | Возвращает значение умноженное на 10 в степени n, где n число девяток после V. |
| 0999 | Возвращает ведущие нули, а не пробелы |
| 9990 | Возвращает конечные нули, а не пробелы |
| 9.99EEEE | Возвращает число в экспоненциальной форме |
| RM | Возвращает число в римской системе исчисления |

Примеры

SELECT TO\_CHAR(SYSDATE, ‘D-MONTH-YY’) FROM DUAL вернет строку ‘7-MAY -10’  
SELECT TO\_CHAR(SYSDATE, ‘DDD-MM-YYYY’) FROM DUAL вернет строку ‘142-05-2010’  
SELECT TO\_CHAR(SYSDATE, ‘Q-D-MM-YYY’) FROM DUAL вернет строку ‘2-7-05-010’  
SELECT TO\_CHAR(1050, ‘9.99EEEE) FROM DUAL вернет строку ‘ 1.050E+03’  
SELECT TO\_CHAR(1400, ‘9999V999’) FROM DUAL вернет строку ‘1400000’  
SELECT TO\_CHAR(48, ‘RM’) FROM DUAL вернет строку ‘ XLVIII’

2) Функция преобразования строки в дату TO\_DATE(строка, формат). Возможные значения форматов уже рассмотрены выше, поэтому приведу несколько примеров использования данной функции. Примеры  
SELECT TO\_DATE(’01.01.2010’, ‘DD.MM.YYYY’) FROM DUAL вернет дату ‘01.01.2010’  
SELECT TO\_DATE(’01.JAN.2010’, ‘DD.MON.YYYY’) FROM DUAL вернет дату ‘01.01.2009’  
SELECT TO\_DATE(’15-01-10’, ‘DD-MM-YY’) FROM DUAL вернет дату ‘15.01.2010’.

3) Функция преобразования строки в числовое значение TO\_NUMBER(строка, формат). Самые распространенные значения форматов перечислены в таблице, поэтому рассмотрим применение данной функции на примерах. Примеры   
SELECT TO\_NUMBER(‘100’) FROM DUAL вернет число 100  
SELECT TO\_NUMBER(‘0010.01’, ’9999D99’) FROM DUAL вернет число 10.01  
SELECT TO\_NUMBER('500,000','999G999') FROM DUAL вернет число 500000.

Если Вам понравилась статья, проголосуйте за нее

[Голосов](http://www.quizful.net/post/oracle-inline-functions): 5Голосовать loading...

http://www.quizful.net/user/javadev75?avatar[**javadev75**](http://www.quizful.net/user/javadev75) http://www.quizful.net/user/Crabar?avatar[**Crabar**](http://www.quizful.net/user/Crabar) http://www.quizful.net/user/st_dent?avatarst\_dent http://www.quizful.net/user/TIMON?avatarTIMON http://www.quizful.net/user/dinamio?avatardinamio

Комментариев: 3

↑обновить

http://www.quizful.net/user/victor_homyakov?avatar[victor\_homyakov](http://www.quizful.net/user/victor_homyakov)03.06.2010 | 10:51:54[#](http://www.quizful.net/post/oracle-inline-functions#4287801111544587925)

У функции SUBSTR второй аргумент необязателен, насколько я помню. Поэтому пример  
[code]  
SELECT SUBSTR(My\_string, INSTR(My\_string, ‘,’, 1), LENGTH(My\_string)- INSTR(My\_string, ‘,’, 1)+1) FROM DUAL  
[/code]  
можно упростить до  
[code]  
SELECT SUBSTR(My\_string, INSTR(My\_string, ‘,’, 1)) FROM DUAL  
[/code]

ответить

http://www.quizful.net/user/victor_homyakov?avatar[victor\_homyakov](http://www.quizful.net/user/victor_homyakov)03.06.2010 | 10:49:02[#](http://www.quizful.net/post/oracle-inline-functions#1934822417099750720)

Мелкие опечатки и пропущенные пробелы:  
TO\_DATE(строка, формт\_даты)  
SELECT \* FROMTABLE1 WHEREINSTR(POST, ‘менеджер’, 1) > 0.  
для понимания работы функции SUBSRTвполне подойдет  
вернетстроку  
SELECTNEXT\_DAY(’01-JUL-2009’, ’mon’) FROMDUALвернет  
вернетзначение  
по основанию, переданному первом параметре

ответить

http://www.quizful.net/user/art?avatarart03.06.2010 | 15:56:28[#](http://www.quizful.net/post/oracle-inline-functions#451772439094062554)

Спасибо, поисправляли