Oracle PL/SQL

CREATING NEW VALUES WITH CHARACTER FUNCTIONS
# CHARACTER FUNCTIONS

<table>
<thead>
<tr>
<th>Function</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`</td>
<td></td>
<td>`</td>
</tr>
<tr>
<td>ASCII</td>
<td>ASCII(<code>'string'</code>)</td>
<td>Returns the ASCII value of the first character of the string.</td>
</tr>
<tr>
<td>CHR</td>
<td>CHR(integer)</td>
<td>Returns the character equivalent ASCII value of the specified integer.</td>
</tr>
<tr>
<td>INITCAP</td>
<td>INITCAP(<code>'string'</code>)</td>
<td>Changes the first letter of the string to uppercase. The remaining letters are made lowercase.</td>
</tr>
<tr>
<td>INSTR</td>
<td>INSTR(<code>'string'</code>, set [, Start[, occurrence]])</td>
<td>Determines the beginning location of a set of characters in a string that match a specified character set.</td>
</tr>
<tr>
<td>LENGTH</td>
<td>LENGTH(<code>'string'</code>)</td>
<td>Returns the length of the string.</td>
</tr>
<tr>
<td>LOWER</td>
<td>LOWER(<code>'string'</code>)</td>
<td>Converts the entire string to lowercase.</td>
</tr>
<tr>
<td>LPAD</td>
<td>LPAD(<code>'string'</code>, length, [,<code>set</code>])</td>
<td>Changes the target string into a string with a specific length by adding a specified set of characters to the left of the string.</td>
</tr>
<tr>
<td>LTRIM</td>
<td>LTRIM(<code>'string'</code>, [,<code>set</code>])</td>
<td>Trims characters from the left side of a string. The characters are trimmed if they match any character in the specified character set.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>LPAD</td>
<td>Changes the target string into a string with a specific length by adding a specified set of characters to the left of the string.</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>Trims characters from the left side of a string. The characters are trimmed if they match any character in the specified character set.</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>Exchanges a set of characters with a replacement set of characters.</td>
<td></td>
</tr>
<tr>
<td>RPAD</td>
<td>Changes the target string into a string with a specific length by adding a specified set of characters to the right of the string.</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>Trims characters from the right side of a string. The characters are trimmed if they match any character in the specified character set.</td>
<td></td>
</tr>
<tr>
<td>SOUNDEx</td>
<td>Converts a string to a code value. It is used to compare strings that might have small differences in spelling but sound alike (or have the same soundex value).</td>
<td></td>
</tr>
<tr>
<td>SUBSTR</td>
<td>Extracts a piece of a string beginning at the start position in the specified string. The number of characters to extract is determined by the count parameter.</td>
<td></td>
</tr>
<tr>
<td>TO_CHAR</td>
<td>Changes a non-character value into a character value.</td>
<td></td>
</tr>
<tr>
<td>TRANSLATE</td>
<td>Changes a string, character by character based on a positional matching of characters in the if string with characters in the then string.</td>
<td></td>
</tr>
<tr>
<td>TRIM</td>
<td>Removes a specified character from the beginning, end, or both the beginning and end of a string of characters.</td>
<td></td>
</tr>
<tr>
<td>UPPER</td>
<td>Converts the entire string to uppercase.</td>
<td></td>
</tr>
</tbody>
</table>
The Initcap Function

Using the Initcap function to capitalize the first letter of a string

```
SQL> select initcap('WILLIAM CLINTON'), initcap('william clinton')
       2  from dual;
INITCAP('WILLIAM INITCAP('WILLIA
--------------- ---------------
William Clinton William Clinton

SQL>
```

End listing

The DUAL keyword in the From clause takes the place of a data table name. It is a pseudo table in Oracle. This is a real table that has one value. It is used when you want to execute a Select statement but do not want to use a data table name. It can be used at any time in the From clause of the Select statement.

The Instr Function

Using the Instr function to determine the position of “OOS” in a character string

```
SQL> select last_name, instr(last_name, 'OOS')
       2  from employee
       3  where fk_department = 'POL';
LAST_NAME       INSTR(LAST_NAME,'OOS')
--------------- ----------------------
WILSON                   0
DWORCZAK                 0
JOHNSON                  0
JOHNSON                  0
CLINTON                  0
NIXON                    0
KENNEDY                  0
ROOSEVELT                2

8 rows selected.
SQL>
```

End listing
The Length Function

Using the Length function to calculate the number of positions in last_name values

SQL> select last_name, length(last_name)
2  from employee
3  where fk_department = 'POL';

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>LENGTH(LAST_NAME)</th>
</tr>
</thead>
<tbody>
<tr>
<td>WILSON</td>
<td>6</td>
</tr>
<tr>
<td>DWORCZAK</td>
<td>8</td>
</tr>
<tr>
<td>JOHNSON</td>
<td>7</td>
</tr>
<tr>
<td>JOHNSON</td>
<td>7</td>
</tr>
<tr>
<td>CLINTON</td>
<td>7</td>
</tr>
<tr>
<td>NIXON</td>
<td>5</td>
</tr>
<tr>
<td>KENNEDY</td>
<td>7</td>
</tr>
<tr>
<td>ROOSEVELT</td>
<td>9</td>
</tr>
</tbody>
</table>

8 rows selected.

SQL>

End Listing

The Lower function

Using the Lower function to change last_name values to lowercase

SQL> select lower('George'), lower('BUSH'), lower('Jr')
2  from dual;

LOWER( LOWE LO
------- ---- --
george bush jr

SQL>

End listing
The LPAD function

Using the Lpad function to pad the left side of the last_name values

```
SQL> select lpad(last_name, 25, '*')
2  from employee
3  where fk_department = 'POL';

LPAD(LAST_NAME,25,'*')
-------------------------
*******************WILSON
*****************DWORCZAK
******************JOHNSON
******************JOHNSON
******************CLINTON
******************NIXON
******************KENNEDY
******************ROOSEVELT
8 rows selected.
```

SQL>

The LTRIM function

Using the Ltrim function to trim the characters “R”, “O”, and “N” from the left side of the last_name values

```
SQL> select last_name, ltrim(last_name, 'RON')
2  from employee
3  where fk_department = 'POL';

LAST_NAME       LTRIM(LAST_NAME
--------------- ---------------
WILSON          WILSON
DWORCZAK        DWORCZAK
JOHNSON         JOHNSON
JOHNSON         JOHNSON
CLINTON         CLINTON
NIXON           IXON
KENNEDY         KENNEDY
ROOSEVELT       SEVELT
8 rows selected.
```

SQL>

End listing
The REPLACE function

The Replace function is used to exchange a character string with another character string. The function has three parameters. These are:

1. The target character string.
2. The search character string.
3. The replacement character string.

Using the Replace function to exchange characters

```
SQL> select replace(last_name, 'OO', 'AA')
2     from employee
3     where last_name like 'ROOS%';

REPLACE(LAST_NAME,'OO','AA')
----------------------------
RAASEVELT
RAASEVELT
RAASEVELT

SQL>
```
End listing
The RPAD function

Using the Rpad function to place the * symbol to the right of the last_name

SQL> select rpad(last_name, 25, '*')
  2  from employee
  3  where fk_department = 'POL';

RPAD(LAST_NAME,25,'*')
-------------------------
WILSON*******************
DWORCZAK*****************
JOHNSON******************
JOHNSON******************
CLINTON******************
NIXON*********************
KENNEDY*******************
ROOSEVELT****************

8 rows selected.

SQL>

End listing

The RTRIM function

Listing 6.9 – Using the Rtrim function to remove “ONL” characters from the right side of the last_name values

SQL> select last_name, rtrim(last_name, 'ONL')
  2  from employee
  3  where fk_department = 'POL';

LAST_NAME       RTRIM(LAST_NAME
--------------- ---------------
WILSON          WILS
DWORCZAK        DWORCZAK
JOHNSON         JOHNS
JOHNSON         JOHNS
CLINTON         CLINT
NIXON           NIX
KENNEDY         KENNEDY
ROOSEVELT       ROOSEVELT

8 rows selected.

SQL>

End Listing
The SOUNDEX function

Using the Soundex function to retrieve records that have a last_name value sounding like “JOHNSON”

```sql
SQL> select last_name, soundex(last_name), soundex('JOHNSEN')
2   from employee
3   where soundex(last_name) = soundex('JOHNSEN');
```

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>SOUN</th>
<th>SOUN</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOHNSON</td>
<td>J525</td>
<td>J525</td>
</tr>
<tr>
<td>JOHNSON</td>
<td>J525</td>
<td>J525</td>
</tr>
</tbody>
</table>

SQL>
End listing

The following describes how the Soundex algorithm encodes a word.

1. The first character of the string is retained as the first character of the soundex code.

2. The following letters are discarded from the string: a, e, i, o, u, h, w, and y.

3. If consonants having the same code number appear consecutively, the number will be coded once (i.e. “J244” becomes “J24”).

4. The following is a list of consonants and codes:
   b, p, f, v     1
   c, s, k, g, j, q, x, z  2
   d, t           3
   l              4
   m, n           5

5. The resulting code is then modified so that it becomes exactly four characters. If it is less than 4 characters, zeros are added to the end. If it is more than 4 characters, the code is truncated.
The SUBSTR function

The function has three parameters. These are:

? String Name

? Extraction Start Position

? Number of Characters to extract

Using the Substr function to extract characters from the last_name values

SQL> select last_name, substr(last_name, 3,3)  
    2  from employee  
    3  where fk_department = 'POL';
LAST_NAME       SUB
--------------- ---
WILSON          LSO
DWORCZAK        ORC
JOHNSON         HNS
JOHNSON         HNS
CLINTON         INT
NIXON           XON
KENNEDY         NNE
ROOSEVELT       OSE
8 rows selected.

SQL>

End Listing
The **TO_CHAR** function

This function is used to convert a value to a character string. A common use of this function is to convert a date into a different format or date picture. This will be discussed in the next chapter.

---

The **TRANSLATE** function

**Using the Translate function to change the letter “O” to “A”**

```sql
SQL> select last_name, translate(last_name, 'O', 'A')
2  from employee
3  where fk_department = 'POL';
```

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>TRANSLATE(LAST_</th>
</tr>
</thead>
<tbody>
<tr>
<td>WILSON</td>
<td>WILSAN</td>
</tr>
<tr>
<td>DWORCZAK</td>
<td>DWARCZAK</td>
</tr>
<tr>
<td>JOHNSON</td>
<td>JAHNSAN</td>
</tr>
<tr>
<td>JOHNSON</td>
<td>JAHNSAN</td>
</tr>
<tr>
<td>CLINTON</td>
<td>CLINTAN</td>
</tr>
<tr>
<td>NIXON</td>
<td>NIXAN</td>
</tr>
<tr>
<td>KENNEDY</td>
<td>KENNEDY</td>
</tr>
<tr>
<td>ROOSEVELT</td>
<td>RAASEVELT</td>
</tr>
</tbody>
</table>

8 rows selected.

SQL>
Trim

The Trim function is similar to the Rtrim and Ltrim functions. The difference is the function can trim characters from the right end, left end, and both ends of the target character string. The function has three options:

- Leading - Trims characters from the beginning of the target string.
- Trailing - Trims characters from the end of the target string.
- Both - Trims characters from both ends of the target string.

Using the Trim function to remove characters.

```
SQL> select trim(leading 'R' from last_name) "Leading Option",
2         trim(trailing 'N' from last_name) "Trailing Option",
3         trim(both 'R' from last_name) "Both Option"
4  from employee
5  where last_name like 'R%'
6     or last_name like '%N';

Leading Option  Trailing Option Both Option
--------------- --------------- ---------------
WILSON          WILSO           WILSON
OOSEVELT        ROOSEVELT       OOSEVELT
OOSEVELT        ROOSEVELT       OOSEVELT
JOHNSON         JOHNSO          JOHNSON
EAGAN           REAGA           EAGAN
JOHNSON         JOHNSO          JOHNSON
CLINTON         CLINTO          CLINTON
NIXON           NIXO            NIXON
TRUMAN          TRUMA           TRUMAN
OOSEVELT        ROOSEVELT       OOSEVELT

10 rows selected.
SQL>
```
End listing
The UPPER function

Using the Upper function to change character strings to uppercase

SQL> select upper('WillIam'), upper('clinton')  
    2  from dual;

UPPER(' UPPER('  
------- -------  
WILLIAM CLINTON

SQL>

End listing

Case Expressions

template of the statement follows:

Case when condition1 then expression1
    When condition2 then expression2
        [Else] expression2a
    End


Using the Case statement to compute a histogram displaying the number of employees in a Wages class.

```sql
SQL> select sum(case when nvl(wages,0) between 0 and 5000 then 1 else 0 end) as "0 to 5000",
3   sum(case when nvl(wages,0) between 5001 and 10000 then 1 else 0 end) as "5001 to 10000",
3   sum(case when nvl(wages,0) between 10001 and 15000 then 1 else 0 end) as "10001 to 15000",
3   sum(case when nvl(wages,0) between 15001 and 20000 then 1 else 0 end) as "15001 to 20000"
5   from employee;
```

<table>
<thead>
<tr>
<th></th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 to 5000</td>
<td>2</td>
</tr>
<tr>
<td>5001 to 10000</td>
<td>9</td>
</tr>
<tr>
<td>10001 to 15000</td>
<td>10</td>
</tr>
<tr>
<td>15001 to 20000</td>
<td>0</td>
</tr>
</tbody>
</table>

SQL>

End Listing

Using Case statements in the Select and Where clauses to count the number of tool purchases per Tool_cost class.

```sql
SQL> select (case when tool_cost >= 0 and tool_cost <= 10 then '0 to 10'
3   when tool_cost > 10 and tool_cost <= 20 then '10.01 to 20'
3   when tool_cost > 20 and tool_cost <= 30 then '20.01 to 30'
3   when tool_cost > 30 then 'Above 30' end) 
3   "Tool Costs", count(*) as amount 
9   from emp_tools
10  group by (case when tool_cost >= 0 and tool_cost <= 10 then '0 to 10'
11   when tool_cost > 10 and tool_cost <= 20 then '10.01 to 20'
12   when tool_cost > 20 and tool_cost <= 30 then '20.01 to 30'
13   when tool_cost > 30 then 'Above 30' end);
```

<table>
<thead>
<tr>
<th>Tool Costs</th>
<th>AMOUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 to 10</td>
<td>15</td>
</tr>
<tr>
<td>10.01 to 20</td>
<td>7</td>
</tr>
<tr>
<td>20.01 to 30</td>
<td>3</td>
</tr>
<tr>
<td>Above 30</td>
<td>9</td>
</tr>
</tbody>
</table>

SQL>
Using Functions in the Where Clause

It is important to understand one of the affects of using a function as an argument in a Where clause. In a Where clause condition, one of the arguments is a table column. Indexes are often placed on table columns used for selection criteria. This helps Oracle identify the records more efficiently. If the query uses a function on a table column argument, it is essentially changing its format. This means Oracle cannot use the table column’s index.