ORACLE sql

**USE ‘execute immediate’ to apply a query result to a variable:**

-------------------------------------  
Declare     
str   varchar2(100);   
record     number;   
begin   
    str:= 'select count(\*)  from  bookinfo';   
    execute   immediate str into record;   
end ;

-------------------------------------

**Define, set and Bind variable:**

-----------------------------------------

Declare

var01 varchar2(100);

Begin

var01:= ‘test’;

Select VALUE INTO var01 FROM tb\_name where cl\_name = ‘vcin';

execute immediate ‘Select VALUE FROM tb\_name where cl\_name = ‘vcin'’ INTO var01;

execute immediate 'update tb\_name set cl\_name=:1'

using var01;

execute immediate 'update tb\_name set cl\_name= ‘||var01 ||’';

execute immediate 'update tb\_name set cl\_name = ‘’’||’string01 ‘||’’’';

End;

-----------------------------------------

**Define whether the table existed, if yes, drop it:**

-----------------------------------------

Declare

vCount number;

Begin

select count(\*) into vCount from user\_tables where table\_name=’xxx';

if vCount >0 then

execute immediate 'drop table xxx';`

end if;

-----------------------------------------

**Output variable:**

------------------------------------------

dbms\_output.put\_line(var\_name);

see the above command output with following setting:

sql develop: View ->Dbms Output-> click ‘plus’ icon->choose db.

-----------------------------------------

**Query current user tables:**

---------------------------------------------------

select count(\*)from user\_tables where table\_name LIKE '%TMP%';

SELECT \* from user\_tables where table\_name LIKE '%TMP%';

--------------------------------------------------------------------------

**Query all user tables with system user:**

---------------------------------------------------

select \* from dba\_tab\_columns;

----------------------------------------------------

**Grant/revoke other user the right to query table ‘dba\_tab\_columns’:**

------------------------------------------------------------------------------------

grant select on dba\_tab\_columns to otherUser; (login with dba)

revoke select on dba\_tab\_columns to otherUser;

**------------------------------------------------------------------------------------**

**CpuMhz=cpuhz/1000/1000 (cast string to numeric: cast(‘string’ as numeric)**

-----------------------------------------------------------------------------------

Select cast(cast(**cpuhz** as numeric)/1000/1000 as numeric) as **CpuMhz** from XXX;

------------------------------------------------------------------------------------

**Substr(), length(): (600 ->6.0.0)**

----------------------------------------------

VCenterServerVerID :=’600’;

verIDLen := length(VCenterServerVerID);

i :=1;

formatedVERID := substr(VCenterServerVerID,i,1);

Loop

i :=i+1;

EXIT WHEN i>verIDLen;

formatedVERID := formatedVERID||'.'||SUBStr(VCenterServerVerID,i,1);

end Loop;

-----------------------------------------------

**Sql parameters:**

-------------------------------------

&1 the first parameter

&2 the second parameter

--------------------------------------

**Sql exp/imp Commands:**

--------------------------------------

exp system/bdnacn file=/home/oracle/xxxTables.dmp statistics=NONE tables='tb\_name1, tb\_name2'

imp: grant sysdba to username; grant imp\_full\_database to username; Then run ‘imp’ to imp dmp file.

---------------------------------------

**Exp table to csv files: (test.sql)**

---------------------------------------------

set serveroutput on size 100000

set long 2000000000 (for long text content for one column)

set trimspool on set trimout on

set term off

set echo off

set verify off

set termout on

set feedback off

set newpage none

set pages 0

set heading off

set linesize 7500 (in case one line was cut)

set term off;

*spool*  &1/LMS\_VCENTERVMS\_&2\.csv

select '"VMNAME","VMGUESTOS","VMGUESTHOSTNAME","VMIPADDRESS"' from *dual*;

select '"' ||VMNAME|| '","' ||VMGUESTOS|| '","' ||VMGUESTHOSTNAME|| '","' ||VMIPADDRESS|| '"' from LMS\_VCENTERVMS;

*spool off*

*exit*

------------------------------------------------

Eg: sqlplus bdna245/bdna245 @test.sql /tmp/LMS 20160517

*[bdna@VMDC8245 OracleLMS]$ ll /tmp/LMS\**

*-rw-rw-r--. 1 bdna bdna 3755 May 17 15:36 LMS\_VCENTERVMS\_20160517.csv*

-------------------------------------------------

**Cursor**

----------------------------------------------------------------------------------------------------------

***Declare* *cursor*** cursor\_name is select distinct owner from tbname WHERE clnname='xxxx';

Var01 varchar (30);

Var02 varchar (100);

***Begin***

Var02 :=’xxxxx’;

Open cursor\_name;

*Loop*

fetch cursor\_name into Var01;

exit when cursor\_name %notfound;

……………….

.………………

*End Loop*;

***End***

--------------------------------------------------------------------------------------------------------------

1. Create a new empty table: CREATE table tb\_name (cln1 cln1Type, cln2 cln2Type)
2. Create a new tb\_name with copying data from other\_tb:

Create table tb\_name as select \* from other\_tb;

1. Create a new empty tb\_name with other\_tb structure:

Create table tb\_name as (select \* from other\_tb where 1==2);

1. Empty all data of a tb\_name: truncate table tb\_name;
2. Drop a tb\_name: drop table tb\_name;
3. Rename a table name: rename tb\_name to tbn;
4. Rename a column name: alter table tb\_name rename column old\_cln to new\_cln;
5. **Copy data from other\_tb to existed tbn:**

INSERT INTO tbn (a,b,c,d)

SELECT

other\_tb.a AS a,

other\_tb.b AS b,

other\_tb.c AS c,

'0' AS d

FROM other\_table;

1. Add column: alter table tbn add (cln3 cln3Type);
2. Modify column attribute: alter table tbn modify (cln3 newType);
3. Update column: update tbn set cln3=’XXX’;
4. Rownum: select rownum, cln1 from tbn where rownum <=5; (print 1-5 row data)
5. Order by (desc/asc);
6. Count(), min(), max(), avg(), Substr(), length();
7. cast(‘string’ as numeric)
8. exp table:

exp system/bdnacn file=/home/oracle/lmsVcTables.dmp statistics=NONE tables='LMS\_VCENTERVMS,LMS\_VCENTERINV'

1. sss

**About cursor:**

192.168.8.152 datastore1

192.168.8.152 datastore2

192.168.9.100 datastore1

192.168.9.100 datastore2

192.168.8.152 datastore1; datastore2

192.168.9.100 datastore1; datastore2

---------------------------------------

declare

cursor hostSt is select distinct HOST\_NAME from TMP\_HostStore;

stores varchar(1024);

stHostName varchar2(1024);

stCount int;

Begin

open hostSt;

Loop

fetch hostSt into stHostName;

EXIT WHEN hostSt%notfound;

declare cursor STORE is select distinct DS\_NAME from TMP\_HostStore where HOST\_NAME= stHostName;

stores varchar(1024);

stcount NUMBER(20);

stname varchar(300);

begin

stcount:=0;

open STORE;

loop

fetch STORE INTO stname;

exit when STORE%notfound;

stcount:=stcount+1;

if stcount=1 then

stores :=stname;

else

stores := stores ||';' || stname;

end if;

end loop;

close STORE;

dbms\_output.put\_line(stores);

insert into TMP\_HostStores values(stHostName,stores);

end;

end loop;

close hostSt;

end;

/

set colsep' ';　　　 //-域输出分隔符

set newp none //设置查询出来的数据分多少页显示，如果需要连续的数据，中间不要出现空行就把newp设

置为none，这样输出的数据行都是连续的，中间没有空行之类的

set echo off;　　　 //显示start启动的脚本中的每个sql命令，缺省为on

set echo on //设置运行命令是是否显示语句

set feedback on; //设置显示“已选择XX行”

set feedback off;　 //回显本次sql命令处理的记录条数，缺省为on即去掉最后的 "已经选择10000行"

set heading off;　　//输出域标题，缺省为on 设置为off就去掉了select结果的字段名，只显示数据

set pagesize 0;　　 //输出每页行数，缺省为24,为了避免分页，可设定为0。

set linesize 80;　　 //输出一行字符个数，缺省为80

set numwidth 12;　 //输出number类型域长度，缺省为10

set termout off;　 //显示脚本中的命令的执行结果，缺省为on

set trimout on;　　//去除标准输出每行的拖尾空格，缺省为off

set trimspool on;　//去除重定向（spool）输出每行的拖尾空格，缺省为off

set serveroutput on; //设置允许显示输出类似dbms\_output

set timing on; //设置显示“已用时间：XXXX”

set autotrace on-; //设置允许对执行的sql进行分析

set verify off //可以关闭和打开提示确认信息old 1和new 1的显示.

**Example:**

**Delete all user of which name contains words ‘BDNA’**

--------------------------------------------------------------------------------------------------------------------

***Declare cursor*** cursor\_name ***is*** select username from dba\_users where username like '%BDNA%';

Var01 varchar (80);

Begin

Open cursor\_name;

Loop

fetch cursor\_name into Var01;

exit when cursor\_name %notfound;

execute immediate 'drop user '||Var01||' cascade';

End Loop;

End;

--------------------------------------------------------------------------------------------------------------------

[**Oracle**](http://lib.csdn.net/base/oracle)中的支持正则表达式的函数主要有下面四个：  
1，REGEXP\_LIKE ：与LIKE的功能相似  
2，REGEXP\_INSTR ：与INSTR的功能相似  
3，REGEXP\_SUBSTR ：与SUBSTR的功能相似  
4，REGEXP\_REPLACE ：与REPLACE的功能相似  
它们在用法上与Oracle SQL 函数LIKE、INSTR、SUBSTR 和REPLACE 用法相同，  
但是它们使用POSIX 正则表达式代替了老的百分号（%）和通配符（\_）字符。  
POSIX 正则表达式由标准的元字符（metacharacters）所构成：  
'^' 匹配输入字符串的开始位置，在方括号表达式中使用，此时它表示不接受该字符集合。  
'$' 匹配输入字符串的结尾位置。如果设置了 RegExp 对象的 Multiline 属性，则 $ 也匹  
配 '\n' 或 '\r'。  
'.' 匹配除换行符之外的任何单字符。  
'?' 匹配前面的子表达式零次或一次。  
'+' 匹配前面的子表达式一次或多次。  
'\*' 匹配前面的子表达式零次或多次。  
'|' 指明两项之间的一个选择。例子'^([a-z]+|[0-9]+)$'表示所有小写字母或数字组合成的  
字符串。  
'( )' 标记一个子表达式的开始和结束位置。  
'[]' 标记一个中括号表达式。  
'{m,n}' 一个精确地出现次数范围，m=<出现次数<=n，'{m}'表示出现m次，'{m,}'表示至少  
出现m次。  
\num 匹配 num，其中 num 是一个正整数。对所获取的匹配的引用。  
字符簇：   
[[:alpha:]] 任何字母。  
[[:digit:]] 任何数字。  
[[:alnum:]] 任何字母和数字。  
[[:space:]] 任何白字符。  
[[:upper:]] 任何大写字母。  
[[:lower:]] 任何小写字母。  
[[:punct:]] 任何标点符号。  
[[:xdigit:]] 任何16进制的数字，相当于[0-9a-fA-F]。  
各种操作符的运算优先级  
\转义符  
(), (?:), (?=), [] 圆括号和方括号  
\*, +, ?, {n}, {n,}, {n,m} 限定符  
^, $, anymetacharacter 位置和顺序  
|   
\*/

--创建表  
create table fzq( id varchar(4), value varchar(10) );  
--数据插入  
insert into fzq values('1','1234560');  
insert into fzq values('2','1234560');  
insert into fzq values('3','1b3b560');  
insert into fzq values('4','abc');  
insert into fzq values('5','abcde');  
insert into fzq values('6','ADREasx');  
insert into fzq values('7','123 45');  
insert into fzq values('8','adc de');  
insert into fzq values('9','adc,.de');  
insert into fzq values('10','1B');  
insert into fzq values('10','abcbvbnb');  
insert into fzq values('11','11114560');  
insert into fzq values('11','11124560');  
--regexp\_like  
--查询value中以1开头60结束的记录并且长度是7位  
select \* from fzq where value like '1\_\_\_\_60';  
select \* from fzq where regexp\_like(value,'1....60');  
--查询value中以1开头60结束的记录并且长度是7位并且全部是数字的记录。  
--使用like就不是很好实现了。  
select \* from fzq where regexp\_like(value,'1[0-9]{4}60');  
-- 也可以这样实现，使用字符集。  
select \* from fzq where regexp\_like(value,'1[[:digit:]]{4}60');  
-- 查询value中不是纯数字的记录  
select \* from fzq where not regexp\_like(value,'^[[:digit:]]+$');  
-- 查询value中不包含任何数字的记录。  
select \* from fzq where regexp\_like(value,'^[^[:digit:]]+$');  
--查询以12或者1b开头的记录.不区分大小写。  
select \* from fzq where regexp\_like(value,'^1[2b]','i');  
--查询以12或者1b开头的记录.区分大小写。  
select \* from fzq where regexp\_like(value,'^1[2B]');  
-- 查询数据中包含空白的记录。  
select \* from fzq where regexp\_like(value,'[[:space:]]');  
--查询所有包含小写字母或者数字的记录。  
select \* from fzq where regexp\_like(value,'^([a-z]+|[0-9]+)$');  
--查询任何包含标点符号的记录。  
select \* from fzq where regexp\_like(value,'[[:punct:]]');

正则表达式用于字符串处理、表单验证等场合，实用高效。现将一些常用的表达式收集于此，以备不时之需。

用户名：/^[a-z0-9\_-]{3,16}$/

密码：/^[a-z0-9\_-]{6,18}$/

十六进制值：/^#?([a-f0-9]{6}|[a-f0-9]{3})$/

电子邮箱：/^([a-z0-9\_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/

URL：/^(https?:\/\/)?([\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]\*)\*\/?$/

IP 地址：/^(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$/

HTML 标签：/^<([a-z]+)([^<]+)\*(?:>(.\*)<\/\1>|\s+\/>)$/

Unicode编码中的汉字范围：/^[u4e00-u9fa5],{0,}$/

匹配中文字符的正则表达式： [\u4e00-\u9fa5]  
评注：匹配中文还真是个头疼的事，有了这个表达式就好办了

匹配双字节字符(包括汉字在内)：[^\x00-\xff]  
评注：可以用来计算字符串的长度（一个双字节字符长度计2，ASCII字符计1）

匹配空白行的正则表达式：\n\s\*\r  
评注：可以用来删除空白行

匹配HTML标记的正则表达式：<(\S\*?)[^>]\*>.\*?</\1>|<.\*? />  
评注：网上流传的版本太糟糕，上面这个也仅仅能匹配部分，对于复杂的嵌套标记依旧无能为力

匹配首尾空白字符的正则表达式：^\s\*|\s\*$  
评注：可以用来删除行首行尾的空白字符(包括空格、制表符、换页符等等)，非常有用的表达式

匹配Email地址的正则表达式：\w+([-+.]\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*  
评注：表单验证时很实用

匹配网址URL的正则表达式：[a-zA-z]+://[^\s]\*  
评注：网上流传的版本功能很有限，上面这个基本可以满足需求

匹配帐号是否合法(字母开头，允许5-16字节，允许字母数字下划线)：^[a-zA-Z][a-zA-Z0-9\_]{4,15}$  
评注：表单验证时很实用

匹配国内电话号码：\d{3}-\d{8}|\d{4}-\d{7}  
评注：匹配形式如 0511-4405222 或 021-87888822

匹配腾讯QQ号：[1-9][0-9]{4,}  
评注：腾讯QQ号从10000开始

匹配中国大陆邮政编码：[1-9]\d{5}(?!\d)  
评注：中国大陆邮政编码为6位数字

匹配身份证：\d{15}|\d{18}  
评注：中国大陆的身份证为15位或18位

匹配ip地址：\d+\.\d+\.\d+\.\d+  
评注：提取ip地址时有用

匹配特定数字：  
^[1-9]\d\*$　 　 //匹配正整数  
^-[1-9]\d\*$ 　 //匹配负整数  
^-?[1-9]\d\*$　　 //匹配整数  
^[1-9]\d\*|0$　 //匹配非负整数（正整数 + 0）  
^-[1-9]\d\*|0$　　 //匹配非正整数（负整数 + 0）  
^[1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*$　　 //匹配正浮点数  
^-([1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*)$　 //匹配负浮点数  
^-?([1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*|0?\.0+|0)$　 //匹配浮点数  
^[1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*|0?\.0+|0$　　 //匹配非负浮点数（正浮点数 + 0）  
^(-([1-9]\d\*\.\d\*|0\.\d\*[1-9]\d\*))|0?\.0+|0$　　//匹配非正浮点数（负浮点数 + 0）  
评注：处理大量数据时有用，具体应用时注意修正

匹配特定字符串：  
^[A-Za-z]+$　　//匹配由26个英文字母组成的字符串  
^[A-Z]+$　　//匹配由26个英文字母的大写组成的字符串  
^[a-z]+$　　//匹配由26个英文字母的小写组成的字符串  
^[A-Za-z0-9]+$　　//匹配由数字和26个英文字母组成的字符串  
^\w+$　　//匹配由数字、26个英文字母或者下划线组成的字符串

select REGEXP\_REPLACE('1200 MHz', '[[:alpha:]]', '') -120 from dual;

(result: 1180)

select REGEXP\_REPLACE('2.00', '[[:alpha:]]', '') -120 from dual;

(result: -118)

CREATE  TABLE caceytest4 PCTFREE 0  as SELECT rownum num, t.\* from ( SELECT inv\_sw\_md\_cpu\_agg1.software\_id "SOFTWARE\_ID", inv\_sw\_md\_cpu\_agg1.software\_label "SOFTWARE\_LABEL", inv\_sw\_md\_cpu\_agg1.cpu\_id "CPU\_ID", (case  when ( (NOT (lower(inv\_sw\_md\_cpu\_agg1.discovered\_speed) like '%' || lower(' ') || '%' escape '\') or inv\_sw\_md\_cpu\_agg1.discovered\_speed is null) AND (REGEXP\_REPLACE(DISCOVERED\_SPEED, '[[:alpha:]]', '') > inv\_sw\_md\_cpu\_agg1.expr\_speed)) then (REGEXP\_REPLACE(DISCOVERED\_SPEED, '[[:alpha:]]', '') - inv\_sw\_md\_cpu\_agg1.expr\_speed)  when ( (NOT (lower(inv\_sw\_md\_cpu\_agg1.discovered\_speed) like '%' || lower(' ') || '%' escape '\') or inv\_sw\_md\_cpu\_agg1.discovered\_speed is null) AND (REGEXP\_REPLACE(DISCOVERED\_SPEED, '[[:alpha:]]', '') <= inv\_sw\_md\_cpu\_agg1.expr\_speed)) then (inv\_sw\_md\_cpu\_agg1.expr\_speed -REGEXP\_REPLACE(DISCOVERED\_SPEED, '[[:alpha:]]', ''))  when inv\_sw\_md\_cpu\_agg1.discovered\_speed is null then inv\_sw\_md\_cpu\_agg1.expr\_speed  end ) "EXPR\_DIFFSPEED", inv\_sw\_md\_cpu\_agg1.discovered\_hardware "DISCOVERED\_HARDWARE", inv\_sw\_md\_cpu\_agg1.expr\_speed "EXPR\_SPEED" FROM G10\_APP\_TMP\_1527 inv\_sw\_md\_cpu\_agg1) t;