ORACLE 学习资料

1 SQL 语言分类

1.1 DDL（数据定义语言）

以下主要是对表操作：
create（创建）
alter（修改）
drop（删除）
truncate（清空表数据）

1.2 DML（数据操纵语言）

以下主要是对表中数据操作：
insert(插入)
delete(删除)
update(更新)
select(查询)

1.3 TCL（事务控制语言）

Commit(提交)
rollback(回滚)

1.4 DCL（数据控制语言）

Grant(授权)
revoke(撤销)

2 Oracle 数据类型

Char:　　字符型（最大长度 2000，定长、不足时以空格补充）

Varchar2:字符型 最大长度 4000，变长，实际长度由存储的数据长度决定（与存储的数据长度一致）

Number(x,y):既可以存储浮点型，也可以存储整形，x 表示有效位数的最大位数，y 表示小数位最大位数。

Date：存储时间类型。默认格式：dd--mm--yy: 天-月-年。

Clob:存储较大的文本，比如存储非结构化 XML 文档，最大为 4G

Blob:存储二进制对象，如图形、视频、声音等。

Long :存储较长字符串，最大长度为 2G

3 表管理

3.1 创建表

Create table 表名（字段名称 类型 约束 ）
例子：create table emp(id number,name varchar2(50));

3.2 修改表结构

3.2.1 添加列

Alter table 表名 add(字段名称 数据类型)

例子：alter table test_tb add(test_name varchar2(200))

3.2.2 删除表中一列

Alter table 表名 set unused column 列名

例子：alter table test_tb set unused column test_name

一般：不建议删除数据库中的列。

3.2.3 修改列名

Alter table 表名 rename column 旧列名 to 新列名

例子：alter table test_tb rename column test_msg to msg

3.2.4 修改表名

Rename 表名 1 （旧名字）to 表名 2（新名字）

例子：rename test_tb to test_tb1

3.2.5 修改表字段名

Alter table 表名 modify(字段名称 新的字段类型)

例子：alter table test_tb1 modify(msg varchar(4000))

3.2.6 添加表约束

Alter table 表名 add Constraint 约束名 约束内容

例子：

alter table test_tb1 add constraint p_k primary key(test_num)-----主键约束

alter table test_tb1 add constraint c_k check(msg='男' or msg='女')---检查约束

Not null----非空约束

4 数据操作语言：DML

4.1 简单查询

Select *|列名|表达式 from 表名 where 条件 order by 列名
1.* 表示所有列
2.列名可以选择若干个表中列名，各个表中列名用逗号分隔。
3.表达式可以是函数，列名，常数等组成表达式。
4.Where 子句是查询的条件
5.Order by 要求在查询结果中排序，默认是升序。
例子：
select * from emp order by sal desc（降序）
select * from emp order by sal （升序）
select * from emp order by sal asc(升序)

4.2 插入

Insert into 表名 values(所有字段对应值)；
Insert into 表名（字段名 1,字段名称 2....） values(对应字段值)
例子：insert into test_tb1(test_num,msg) values(666,'MSG values')

4.3 更新

Update 表名 set 字段名称 =值 where 更新条件
例子：update test_tb1 set msg='hello oracle' where test_num=123

4.4 删除

Delete 表名 where 条件
例子：delete test_tb1 where msg='hello oracle'

Truncate--将表中数据一次性删除
语法： truncate table 表名
Truncate 和 detele 区别：
1、truncate 是 DDL 命令，删除数据不能回复；delete 是 DML 命令，删除数据可以通过数据库的日志文件进行恢复。
2、如果一个表中记录很多，truncate 相对 delete 速度快。
警告：由于 truncate 命令比较危险，所有在实际开发中，truncate 命令慎用。

5 操作符

算术操作符： + 、 - 、 * 、 / （加、减、乘、除）
关系运算和逻辑运算符：
= 等于
> 大于
<>或 != 不等于
<= 小于或等于
< 小于
>= 大于或等于

逻辑运算符：
And 、 or 、 in 、not in
And:且
Or: 或者
Not in 不在...中
In:在.....中
--加法
例子：SELECT ename,job,(sal+comm) FROM emp;

--减法
例子：SELECT ename,job,(sal-comm) FROM emp;
--双竖线 ||
例子：SELECT (ename||'的工资'||(sal+comm)) FROM emp;
字符串连接操作符： || （双竖线）
例子：select (ename||' 的工资是：'||(sal+comm)) from emp

6 常用函数

6.1 数值函数

abs(m) m 的绝对值

mod(m,n) m 被 n 除后的余数

power(m,n) m 的 n 次方

round(m[,n]) m 四舍五入至小数点后 n 位的值（n 缺省为 0）

trunc(m[,n]) m 截断 n 位小数位的值（n 缺省为 0）

6.2 字符函数

initcap(st) 返回 st 将每个单词的首字母大写，所有其他字母小写

lower(st) 返回 st 将每个单词的字母全部小写

upper(st) 返回 st 将每个单词的字母全部大写

concat(st1,st2) 返回 st 为 st2 接 st1 的末尾（可用操作符"||"）

lpad(st1,n[,st2]) 返回右对齐的 st,st 为在 st1 的左边用 st2 填充直至长度为 n,st2 的缺省为空格

rpad(st1,n[,st2]) 返回左对齐的 st,st 为在 st1 的右边用 st2 填充直至长度为 n,st2 的缺省为空格

ltrim(st[,set]) 返回 st,st 为从左边删除 set 中字符直到第一个不是 set 中的字符。缺省时，指的是空格

rtrim(st[,set]) 返回 st,st 为从右边删除 set 中字符直到第一个不是 set 中的字符。缺省时，指的是空格

replace(st,search_st[,replace_st]) 将每次在 st 中出现的 search_st 用 replace_st 替换，返回一个 st。缺省时，删除 search_st

substr(st,m[,n]) n=返回 st 串的子串，从 m 位置开始，取 n 个字符长。缺省时，一直返回到 st 末端

length(st) 数值，返回 st 中的字符数

instr(st1,st2[,m[,n]]) 数值，返回 st1 从第 m 字符开始，st2 第 n 次出现的位置，m 及 n 的缺省值为 1

例： 1.

select initcap('THOMAS'),initcap('thomas') from test;

initca initca

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

Thomas Thomas

2.

select concat('abc','def') "first" from test;

first

-----

abcdef

3.

```
select 'abc'||' '||'def' "first" from test;

first

-----

abc def
```

4.

```
select lpad(name,10),rpad(name,5,'*') from test;

lpad(name,10) rpad(name,5,'*')

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

         mmx mmx**

    abcdef abcde
```

5.

去掉地址字段末端的点及单词 st 和 rd

```
select rtrim(address,'. st rd') from test
```

6.

```
select name,replace(name,'a','*') from test;

name    replace(name,'a','*')

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

great gre*t
```

7.

```
select  substr('archibald  bearisol',6,9)  a,substr('archibald  bearisol',11)  b

from test;

a                b
```

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

bald bear bearisol

8.

select name,instr(name,' ') a,instr(name,' ',1,2) b from test;

name a b

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

li lei 3 0

l i l 2 4 数位的值（n 缺省为 0）

6.3 转换函数

nvl(m,n) 如果 m 值为 null,返回 n,否则返回 m

to_char(m[,fmt]) m 从一个数值转换为指定格式的字符串 fmt 缺省时，fmt 值的宽度正好能容纳所有的有效数字

to_number(st[,fmt]) st 从字符型数据转换成按指定格式的数值，缺省时数值格式串的大小正好为整个数

附：

to_char()函数的格式：

符号 说明

---

9 每个 9 代表结果中的一位数字

0 代表要显示的先导 0

$　　美元符号打印在数的左边

L　　任意的当地货币符号

.　　打印十进制的小数点

,　　打印代表千分位的逗号

例：

1.

```
select to_number('123.45')+to_number('234.56') form test;

to_number('123.45')+to_number('234.56')

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

                                    358.01
```

2.

```
select to_char(987654321) from test;

to_char(987654321)

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

987654321
```

3.

```
select  to_char(123,'$9,999,999')  a,to_char(54321,'$9,999,999')

b,to_char(9874321,'$9,999,999') c from test;

a       b           c

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

$123    $54,321    $9,874,321
```

4.

select to_char(1234.1234,'999,999.999') a,to_char(0.4567,'999,999.999')

b,to_char(1.1,'999,999.999') from test;

a b c

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

1,234.123 .457 1.100

6.4 分组函数

avg([distinct/all] n) 列 n 的平均值

count([all] *) 返回查询范围内的行数包括重复值和空值

count([distinct/all] n) 非空值的行数

max([distinct/all] n) 该列或表达式的最大值

min([distinct/all] n) 该列或表达式的最小值

stdev([distinct/all] n) 该列或表达式的标准偏差，忽略空值

sum([distinct/all] n) 该列或表达式的总和

variance([distinct/all] n) 该列或表达式的方差，忽略空值

6.5 日期函数

add_months(d,n) 日期 d 加 n 个月

last_day(d) 包含 d 的月份的最后一天的日期

month_between(d,e) 日期 d 与 e 之间的月份数，e 先于 d

new_time(d,a,b) a 时区的日期和时间 d 在 b 时区的日期和时间

next_day(d,day)　比日期 d 晚，由 day 指定的周几的日期

sysdate　当前的系统日期和时间

greatest(d1,d2,...dn) 给出的日期列表中最后的日期

least(d1,k2,...dn) 给出的日期列表中最早的日期

to_char(d [,fmt]) 日期 d 按 fmt 指定的格式转变成字符串

to_date(st [,fmt]) 字符串 st 按 fmt 指定的格式转成日期值，若 fmt 忽略，st 要用缺省格式

round(d [,fmt])　日期 d 按 fmt 指定格式舍入到最近的日期

trunc(d [,fmt])　日期 d 按 fmt 指定格式截断到最近的日期

附：

日期格式：

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

格式代码	说明	举例或可取值的范围
DD	该月某一天	1－3
DY	三个大写字母表示的周几	SUN，...SAT
DAY	完整的周几，大写英文	SUNDAY，...SATURDAY
MM	月份	1－12
MON	三个大写字母表示的月份	JAN，...DEC
MONTH	完整	JANUARY,...DECEMBER
RM	月份的罗马数字	I,...XII

YY 或 YYYY 两位，四位数字年

HH:MI:SS　时：分：秒

HH12 或 HH24　　以 12 小时或 24 小时显示

MI　　　　　　分

SS　　　　　　秒

AM 或 PM　　　　上下午指示符

SP　　　　　　后缀 SP 要求拼写出任何数值字段

TH　　　　　　后缀 TH 表示添加的数字是序数　　4th,1st

FM　　　　　　前缀对月或日或年值，禁止填充

————————————————————————————

例：

1.

下一个周五的日期

select next_day(sysdate,6) from test;

2.

两个月前的今天的日期

select add_months(sysdate,-2) from test;

7 其他常用语句

7.1 集合操作

UNION 由每个查询选择的所有不同的行，并集不包含重复值

UNION ALL 由每个查询选择的所有的行，包括所有重复的行，完全并集包含重复值

注：Union all 效率一般比 union 高

7.2 exists 和 not exists 的使用

以下两条语句等价

使用 in

```
select last_name, title
 from s_emp
 where dept_id in
      (select id
        from s_dept
       where name='Sales');
```

使用 exists

```
select last_name,title
        from s_emp e
        where  exists
```

(select 'x' --把查询结果定为 constant，提高效率

from s_dept s where s.id=e.dept_id and s.name='Sales');

7.3 with 子句

1. 使用 with 子句可以重复使用相同的子查询块,通过 select 调用，一般在子查询用到多次情况下。
2. with 子句的返回结果存到用户的临时表空间中
3. with 子句可以提高查询效率
4. 有多个 with 的时候，用逗号隔开
5. 最后一个 with 子句与下面的查询之间不能有逗号，只通过右括号分割,查询必须用括号括起来

   目的是为了重用查询。

   举例：

   如查询销售部门员工的姓名

```
 --with clause
with a as
    (select id from s_dept where name='Sales' order by id)
 select last_name,title
        from s_emp where dept_id in (select * from a);--使用 select 查
```

   询别名

7.4 decode 函数

类似于程序中的 IF..ELSE 功能。

举例：

如果 empno 值为 7369 变为’smith’，如果是 7499 变为’allen’，否则为’unknow’

```
select     empno,decode(empno,7369,'smith',7499,'allen','unknow')     as
       name
from emp
```

7.5 case 表达式

举例：

上面语句改写为 CASE 写法

```
select empno,(case    empno
                        when 7369 then    'smith'
                        when 7499 then 'allen'
                        else 'unknow' end) as name
from emp
```

7.6 rownum—top-N

查询出前 5 行数据

```
select     *    from    emp where    rownum<=5;
```

7.7 row_number 函数使用

说明：ROW_NUMBER() OVER(partition by col1 order by col2) 表示根据 col1 分组，在分组内部根据 col2 排序，而此函数计算的值就表示每组

内部排序后的顺序编号（组内是连续且唯一的）。

举例：

SELECT * FROM T1;

ID	NAME	DATE1
101	aaa	09-SEP-13
101	bbb	10-SEP-13
101	ccc	11-SEP-13
102	ddd	08-SEP-13
102	eee	11-SEP-13

执行语句：

SELECT ID,NAME,DATE1,ROW_NUMBER() OVER(partition by ID order by DATE1 desc) as RN FROM T1;

得出结果：

ID	NAME	DATE1	RN
101	ccc	11-SEP-13	1
101	bbb	10-SEP-13	2
101	aaa	09-SEP-13	3
102	eee	11-SEP-13	1
102	ddd	08-SEP-13	2