# OracleSQL语句大全讲义

Oracle SQL语句大全

desc(描述） emp　　描述emp这张表

2.desc　　dept　　　 部门表

3.desc salgrade　　　薪水等级

4.select \*from table 查找表中的元素

5.dual　　 是系统中的一张空表 （可以用于计算各种数学表达式）

6.select \*from dual

7.select sysdate from dual 取出系统时间

8.select ename,sal\*12 "annul\_sal"(取的别名） from emp; 查找用户姓名和用户的年薪

9.任何含有空值的数学表达式的值都是空值

select ename,sal\*12+comm from emp;

10.select ename||sal from emp 其中的|| (字符串连接符) 相当于将sal全部转化为字符串

11.表示字符串的方法

select ename ||'ajjf' from emp;

12.如果其中有一个单引号就用2个单引号来代替他

select ename||'sakj' 'lds'from emp;

13.select distinct deptno from emp　　 (去除部门字段中重复的部分,关键字distinct)

14.select distinct deptno,job from emp;(去除这2个字段中重复的组合)

15.select \*from dept where deptno=10;　　 取出条件(取出部门编号为10的记录)

16.select \* from emp where ename='CLIRK'; 取出部门中姓名为clirk的记录(注意取出过程中ename用单引号隔开)

17.select ename,sal from emp where sal&gt;1500; 取出部门中薪水大于1500的人的姓名

18.select ename,sal,deptno from emp where deptno&lt;&gt; 10 取出部门中的部门号不等于10的

19.select ename,sal,deptno from emp where ename&gt;'CBA' 取出部门中员工名字大于CBA的员工(实际比较的是ACIIS码)

20.select ename,sal from emp where sal between 800 and 1500

　 select ename,sal from emp where sal&gt;=800 and sal&lt;=1500;　 (取出800和1500之间的数)

21.select ename,sal,comm from emp where comm is null (选出其中的空值)

　 select enmae,sal,comm from emp where comm is not null(选出其中的非空值)

22.select ename,sal,comm from emp where sal in (800,1500,2000);取出这3者之中的

　 select ename,sal,comm from emp where ename in('simth');

23.select ename,sal,hiredate from emp where hiredata&gt;'3-04月-81';宣传符合条件的日期

24.select ename,sal,from emp where sal&gt;1000 or deptno=10;　　　 找出工资薪水大于1000或者部门号等于10的员工

25.select ename,sal from emp where sal not in(500,1000);　　　　查找薪水不在500到1000的员工姓名和月薪

26.select ename,sal from emp where ename like '%ALL%';　　　查找姓名中含有ALL的客户信息 %表示0个或多个字母

　 select ename,sal from emp where ename like '\_A%';　　　 查找第二个字母是A的客户信息

select ename,sal from emp where ename like '\_%$%%' escape '$'; 自己指定转义字符

　 select ename,sal from emp where ename like '\_%\%%';　　　　　　 查找中间含有%相匹配的客户信息，运用转易字符

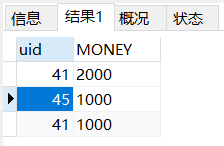
28.select \* from dept order by deptno　　　　　　　　　　　　　　对表中元素按部门号排序

　 select \*from dept order by deptno desc　　　　　　　　　　　　默认为升序asc，可以用desc按降序

29.select ename,sal from emp where sal &lt;&gt;1000 order by sal desc, ename desc　 按照查询条件来查询，并排序（asc升序排列）

{example: select uid, MONEY from account order by MONEY desc, uid desc;

类似 先按money 分组， 再按uid 分组。



}

30.select ename,sal\*12 from emp where ename not like '\_%A%' and sal&gt;800 order by sal desc

31.select lower(ename) from emp 将ename都转化为小写　　lower是函数能将字母转化为小写，upper(ename)

32.select ename from emp where lower(ename) like '\_%a%'; 找出ename 中所有的含有a的字符

33.select substr(ename,2,3) form emp　　　　　　从第2个字符开始截取3个字符

34.select chr(65) from dual;　　　　　将65转化为字符

35.select ascii('A') from dual　　　　 将ACSII码转化为字符串

36.select round(23.565)from dual　　 四舍五入

36.select round(23,4565,2)from dual 四舍五入到第二位(第二个参数默认为0)

37.select to\_char(sal,'$99,999.9999') from emp　按指定格式输出(9代表一位数字)

　 select to\_char(sal,'L99,999.9999') form emp　L代表本地字符(￥) 0代表占位符

38.select hiredate from emp

　 select to\_char(hiredate,'YYYY-MM-DD HH:MI:SS) from emp;　　　　　时间格式的显示

　 select to\_char(sysdate,'YYYY-MM-DD HH:MI:ss) from dual;　　　　　十二小时制显示系统时间

　 select to\_char(sysdate,'YYYY-MM-DD HH24:MI:SS) from dual　　　　 二四小时制显示系统时间

39.select ename,hiredate from emp where hiredate &gt; to\_date('2005-2-3 12:32:23','YYYY-MM-DD HH:MI:SS');　把特定格式字符串转换成日期

40 select sal from emp where sal&gt;to\_number('$1,250.00','$9,999.99'); 取出比它大的一切字符串（把特定格式的数字转化成字符）

41 select ename,sal+nvl(comm,0) from emp;　　 讲comm值为空的用0来替换，单行函数（以一条记录为条件）一条对一条

组函数

42.select Max(sal) from emp;

　 select Min(sal) from emp;

　 select avg(sal) from emp;

　 select sum(sal) from emp;

　 select count(\*) from emp;　　　　　　　　　　　　　查看表中一共有多少条记录

　 select count(\*) from emp where deptno=10;　　　　　查找部门10一共有多少人；

43.select avg(sal),deptno from emp group by deptno;　 按部门号进行分组

　 select deptno,job,max(sal) from emp group by job,deptno; 按工作和部门号进行分组；

44.select ename from emp where sal=(select max(sal) from emp); 子查询，查找部门中薪水最高的员工姓名

45.group by 注意：出现在select列表中的字段，如果没有出现在组函数中必须出现在group by子句中

where是过滤单条语句,having可以过滤分组。

46.select avg(sal),deptno from emp group by deptno having avg(sal)&gt;2000;　　选出部门中平均薪水大于2000的部门，

47.select \* from emp where sal&gt;100 group by deptno having ..........order by........

　 先取数据--过滤数据------分组----对分组限制-------排序

48.select avg(sal) from emp where sal&gt;1200 group by deptno having avg(sal)&gt;1500 order by avg(sal) desc;

　 查找部门中平均薪水大于1200的员工，并按部门号进行排序，查询分组后的平均薪水必须大于1500，查询结果按平均薪水从低到高排列

子查询（把它当成一张表）

49.select ename from emp where sal&gt;(select avg(sal) from emp);

　 查找出员工中薪水位于部门平均薪水之上的所有员工

50.select ename,sal from emp join(select max(sal) max\_sal from emp group by deptno) t on(emp.sal=t,max\_sal and emp.deptno=t.deptno);

　 查找每个部门中薪水最高的

51.select e1.ename,e2.ename from emp e1,emp e2 where e1.mgr=e2.empno;　　　　　 表的自连接(当成两张表来用)

1999 SQL标准

where语句里不写“连接条件”，只写数据过滤条件。

join on(表间连接条件)，where写过滤条件，思路清晰。

52.select dname,ename from emp cross join dept　　　　　　交叉连接，笛卡尔

SQL99中的新语法

53.select ename,dname from emp join dept on(emp.deptno=dept.deptno); //on(连接条件)

54.select ename,dname from emp join dept using(deptno); 不推荐　　　 查找emp和dept表中deptno相同的部分。

55.select ename,dname,grade from emp e join dept d on(e.deptno=d.depno)

　　　　join salgrade s on(e.sal between s.losal and s.hisal)

　　　　where ename not like '\_%A%';　 (三表查找）

56.select e1.ename,e2.ename from emp e1 join emp e2 on(e1.mgr=e2.deptno); 表的自连接

57.select e1.ename,e2.ename from emp e1 left join emp e2 on(e1.mgr=e2.empno) 左外表连接// 可以将左边表多余的数据拿出来，右外表连接：right (outer) join on

　 select ename,dname from emp e right join dept d on(e.deptno=d.deptno)右外连接

　 select ename,dname from emp e full join dept d on(e.deptno=d.deptno)全连接

58.求部门中薪水最高的

　 select ename,sal from emp join (select max(sal) max\_sal, deptno from emp group by deptno) t　on (emp.sal=t.max\_sal and emp.deptno=t.deptno);

---求部门平均薪水的等级

select deptno, avg\_sal, grade from

(select deptno, avg(sal) avg\_sal from emp group by deptno)t

join salgrade s on (t.avg\_sal between s.losal and s.hisal);

----求部门中各成员的平均薪水等级

select deptno,ename,grade from emp

join salgrade s on(emp.sal between s.losal and s.hisal);

59.求部门中薪水等级的平均值

　 select deptno,avg(grade) from(select deptno,ename,grade from emp join salgrade s on(emp.sal between s.losal and s.hisal))t group by deptno;

60.查找雇员中哪些是经理人

　 select ename from emp where empno in(select distinct mgr from emp);

61.select distinct e1.sal from emp e1 join emp e2 on(e1.sal&lt;e2.sal); 自连接(不用组函数求出最高薪水）

　 select distinct sal from emp where sal not in (select ename, e1.sal from emp e1 join emp e2 on(e1.sal&lt;e2.sal));

62.求平均薪水最大的部门的部门编号

select deptno, avg\_sal from

(select avg(sal) avg\_sal, deptno from emp group by deptno)t

where avg\_sal=

(select max(avg\_sal) from

(select avg(sal) avg\_sal, deptno from emp group by deptno));

写法2：

select deptno, avg\_sal from

(select avg(sal) avg\_sal, deptno from emp group by deptno)t

where avg\_sal=

(select max(avg(sal)) from emp group by deptno);　　max(avg(sal))组函数嵌套，最多两层

求平均薪水最大的部门的部门名称

select dname from dept where deptno =

(

　select deptno from

　　(select avg(sal) avg\_sal,deptno from emp group by deptno)

　where avg\_sal=

(select max(avg\_sal) from

　　　(select avg(sal) avg\_sal,deptno from emp group by deptno)

)

);

64,求平均薪水等级最低的部门的部门名称

select dname, t1.deptno, grade, avg\_sal from

　(

　　select deptno, grade, avg\_sal from

　　　(select deptno, avg(sal) avg\_sal from emp group by deptno)t

　　join salgrade s on (t.avg\_sal between s.losal and s.hisal)

　)t1

　join dept on (t1.deptno = dept.deptno)

where t1.grade =

(

　select min(grade) from

　(

　　select deptno, grade, avg\_sal from

　　　(select deptno, avg(sal) avg\_sal from emp group by deptno)t

　　join salgrade s on (t.avg\_sal between s.losal and s.hisal)

　)

)

;

创建一个视图(一张表,虚表)，简化SQL语句

create view v$\_dept\_avg\_sal\_info as

　　select deptno, grade, avg\_sal from

　　　(select deptno, avg(sal) avg\_sal from emp group by deptno)t

join salgrade s on (t.avg\_sal between s.losal and s.hisal);

简化后代码

select dname, t1.deptno, grade, avg\_sal from

　v$\_dept\_avg\_sal\_info　t1

join dept on (t1.deptno = dept.deptno)

where t1.grade =

(

　select min(grade) from

　v$\_dept\_avg\_sal\_info

);

求比普通员工最高薪水还高的经理人

select ename from emp

where empno in (select distinct mgr from emp where mgr is not null)

and

sal &gt;

(

　　select max(sal) from emp where empno not in

　　　(select distinct mgr from emp where mgr is not null)

)

;

DML（数据操作语言）语句：更、删、改、查

创建权限， conn sys/admin as sysdba

　　　　　 grant create table,create view to scott;

首先在C:下面建个文件夹备份文件

一.createNewUser方法

1.--backup scott　---备份scott

　　 exp　　　　 ---执行exp.exe文件///c:\temp&gt;exp ,,.DMP文件。在c:\temp下

2.create user(创建用户）用超级管理员模式进入

　 create user 用户名 identified by 密码 default tablespace users quota 10M on users;

　 grant create session,create table,create view to 用户名 ---授予权限

　 drop user 用户名 ---删除用户

3.import the data(导入备份数据）

　 imp

二.insert

insert into dept values (50,'game','bj') 插入一条记录

insert into dept2 (deptno,dname) values (78,'games'); 插入指定的几条记录

insert into dept2 select \*from dept　　　插入指定的表（表结构要一样）

rollback;　　　　 回退（取消上一步操作）

create table emp2 as select \* from emp;　　备份emp这张表

三. update emp2 set sal=sal\*12 where deptno=10;　　update的用法

四. delete from dept2 where deptno&lt;25 ;　　　　　 删除语句的用法

rownum r 只能和&lt;和&lt;=使用,不能用&gt;或者&gt;=或者=

transaction 事务:一个事务起始于一个DML语句

DDL（数据定义）语言

1.create table t(a varchar2(10));　创建表:

2.drop table t　　　　　　　　 删除表

3.commit　 所有的(上一条事务的修改)提交，所有修改都结束了。对于rollback(rollback会回到原始状态)无效，一个事务开始于第1条DML语句

碰到执行DDL DCL语句事务自动提交(自动执行commit)对于rollback无效，

正常断开连接，自动提交。 非正常提交，自动回滚rollback

DCL：数据控制语句----如grant 授权

建表语句

建学生信息表：

create table stu

(id number(6),

name varchar2(20) constraint stu\_name\_nn not null,

sex number(1),

age number(3),

sdate date,

grade number(2) default 1,

class number(4),

email varchar2(50) unique　　(唯一约束)

);

5个约束条件

非空not null

唯一unique（不能插入重复值），

主键primary key（可以唯一标识整条记录）：　非空，唯一，数字 ，可以使用多个字段的组合作为主键

外键 foreign key(class) references class(id),　references参考 class中的id，被参考的字段必须是主键。 被参考的记录是无法删除的

：两张表，在一张表插入数据的时候，不允许插入另一张表中一个字段没有的数据。默认添加。

check：检查约束

create table stu

(

id number(6) primary key,

name varchar2(20) constraint stu\_name\_nn not null, 约束名constraint

sex number(1),

age number(3),

sdate date,

grade number(2) default 1,

class number(4),

email varchar2(50),

constraint stu\_name\_email\_uui unique(email,name) （表级约束）组合性约束，name和email的组合不能重复

);

主键约束方法二

create table stu

(id number(6),

name varchar2(20) constraint stu\_name\_nn not null,(非空约束)

sex number(1),

age number(3),

sdate date,

grade number(2) default 1,

class number(4) references class(id),(参考class 这张表，参考字段)

email varchar2(50),

constraint stu\_id\_pk primary key(id),

constraint stu\_name\_uui unique(email,name) 组合性约束

);

外键约束

create table class

(

id number(4) primary key,　 (id为被参考字段,被参考的字段必须是主键)

name varchar2(20) not null

)

create table stu

(

id number(6),

name varchar2(20) constraint stu\_name\_nn not null,(非空约束)

sex number(1),

age number(3),

sdate date,

grade number(2) default 1,

class number(4)

email varchar2(50),

constraint stu\_class\_fk foreign key(class) references class(id),　references参考

constraint stu\_id\_pk primary key(id),

constraint stu\_name\_uui unique(email,name) 组合性约束

);

像外键中插入关键字，

1.insert into class values(1000,'c1');

2.insert into stu(id,name,class,email) values(1,'a',1000,'a');

3.alter table stu add(addr varchar(20));添加表的结构

4.alter table stu drop(addr); 删除表结构

5.alter table stu modify(addr varchar2(150));修改精度

6.alter table stu drop constraint stu\_class\_fk; 删除约束条件

7.alter table stu add constraint stu\_class\_fk foreign key(class) references class(id),添加约束条件

查找当前用户下有哪些表和哪些视图及哪些约束:

数据字典表

user\_tables：当前用户下有多少张表。

user\_views：当前用户下有多少个视图。

user\_constraints：当前用户下有多少个约束。

user\_indexes：当前用户下有多少个索引。

数据字典表的表：dictionary

8.select table\_name from user\_names

9.select view\_name from view\_names

10.select constraint\_name,table\_name from user\_constraints;

desc dictionary数据字典表

desc user\_tables当前用户下面有多少张表

select table\_name from user\_tables; 查找当前用户有多少张表

索引： 加主键或者唯一约束的时候oracle会自动为该字段加索引。

创建索引

create index idx\_stu\_email on stu(email);

drop index idx\_stu\_email;

查找索引

select index\_name from user\_indexes;

索引读的速度快了，插入速度变慢。经常需要查询的字段可以建立索引。

优化数据库表首选。

view 视图（子查询，虚表）

视图赠加了维护的量

create view v$\_dept\_avg\_sal\_info as (select ...)

create view v$\_stu as select id, name, age from stu;　----只给客户看非关键的信息

非必要尽量不适用，因为会带来维护繁琐的问题。

尽量不要用视图更新数据。因为视图很可能从多个表拿数据。

序列(sequence)：序列对象　产生唯一的，不间断的序列,一般用于主键

create table arcticle

(id number,

title varchar2(1024),

cont long

);

序列的创建sequence产生独一无二的序列，而且是oracle独有的

create sequence seq;

select seq.nextval from dual; 查找序列号

insert into arcticle values(seq.nextval,'a','b');往表中插入序列

dorp sequence seq;　---删除序列。

数据库设计的3范式（姓范的设定的规则）：实际当中可以打破

不要存在冗余数据：一条数据只存放在一个表里，要用这条数据，通过该数据的主键和其他表联系。

第一范式： 设计任何表都要有主键，列不可分

-----不存在冗余数据，不能使用012345\_张三\_23

第二范式： 如果有2个主键的话，不能存在部分依赖

不是主键的字段，不能依赖于主键的一部分。

第三范式， 不能存在传递依赖

掌握一张语言：1、数据类型；2、语法。

oracle内部的编程语言PL-SQL，用来补充SQL（没有分支，没有循环）

PL：过程语言：带有分支、循环的语言。

PL-SQL

例子1：

SQL&gt; set serveroutput on;

SQL&gt; begin(必要的--程序开始执行)

2 dbms\_output.put\_line('hello world');　 ----输出语句

3 end;(结束)

4 /

例子2：

SQL&gt; declare　　　　------声明变量　v\_xxx

2 v\_name varchar2(20);

3 begin

4 v\_name:='myname';　 ---赋值

5 dbms\_output.put\_line(v\_name);

6 end;

7 /

例子3：

SQL&gt; declare

2 v\_num number:=0;

3 begin

4 v\_num:=2/v\_num;

5 dbms\_output.put\_line(v\_num);

6 end;

7 /

declare

\*

ERROR 位于第 1 行:

ORA-01476: 除数为 0

ORA-06512: 在line 4

例子4：

declare

v\_num number:=0;

begin

v\_num:=2/v\_num;

dbms\_output.put\_line(v\_num);

exception

when others then　　　 ------------当其他的情况出现的时候

dbms\_output.put\_line('error');

end;

/

选数据库：够用为原则。小公司别用oracle，太贵了。

变量声明的规则

1.变量名不能够使用保留字，如from,select等

2.第一字符必须是字母。

3.变量名最多包含30个字符

4.不要与数据库的表或者列同名

5.每一行只能声明一个变量

常用变量类型

1. binary\_interger, 整数，主要用来计数，而不是用来表示字段类型

2. number　　　　数字类型

3. char　　　　　 定长字符串

4. varchar2　　　 变长字符串,4096个字节（2048个字）

5. date　　　　　 日期

6. long　　　　　 长字符串，最长2GB(存大文章)

7. boolean　　　　布尔类型，可以取true、 false 和null（不初始化）的值

例5：

declare

　　v\_temp number(1);

　　v\_count binary\_integer:=0;

　　v\_sal number(7,2):=4000.00;

　　v\_date date:=sysdate;

　　v\_pi constant number(3,2):=3.14;　　----constant 相当于java里的final。常量

　　v\_valid boolean:=false;

　　v\_name varchar2(20) not null:='myname';

begin

　　dbms\_output.put\_line('v\_temp value:'||v\_temp); --- ||字符串连接符

end;

/

用--可以注释一行

---变量声明，使用%type属性

例6：

declare

　　v\_empno number(4);

　　v\_empno2 emp.empno%type;　---声明变了，程序也跟着变

　　v\_empno3 v\_empno2%type;　 ---变量变了，另一个变量也跟着变

begin

　　dbms\_output.put\_line('test');

end;

---table变量类型　，table是一个数组，指定类型

例7

set serveroutput on;

declare

　　 type type\_table\_emp\_empno is table of emp.empno%type index by binary\_integer;　---声明一种新类型type\_table\_emp\_empno。

　　 v\_empnos type\_table\_emp\_empno;

begin

　　 v\_empnos(0):=7369;

　　 v\_empnos(2):=7869;

　　 v\_empnos(-1):=9999;

　　 dbms\_output.put\_line(v\_empnos(-1));

end;

---Record 变量类型， 相当于一个类。

例8

set serveroutput on;

declare

　 type type\_record\_dept is record

　 (

　　　deptno　dept.deptno%type,　 ---相当于成员变量

　　　dname　dept.dname%type,

　　　loc　　 dept.loc%type

　　);

　　v\_temp type\_record\_dept;

begin

　　v\_temp.deptno:=50;

　　v\_temp.loc:='aaaa';

　　v\_temp.loc:='bj';

　　dbms\_output.put\_line(v\_temp.deptno||' '||v\_temp.dname);

end;

例9：

使用%rowtype声明record变量（表结构的变化同时也能代理储存过程的变化）

set serveroutput on;

declare

　　v\_temp dept%rowtype;

begin

　　v\_temp.deptno:=50;

　　v\_temp.loc:='aaaa';

　　v\_temp.loc:='bj';

　　dbms\_output.put\_line(v\_temp.deptno||' '||v\_temp.dname);

end;

---SQL语句的运用

例10；

select语句必须返回一条记录，并且只能返回一条记录。

declare

v\_name emp.ename%type;

v\_sal emp.sal%type;

begin

select ename,sal into v\_name,v\_sal from emp where empno=7369;（将ename和sal的值放在v\_name和v\_sal里面）

declare

v\_emp emp%rowtype;

begin

select \* into v\_empl from emp where empno=7369;

dbms\_ouput.put\_line(v\_emp.ename);

end;

/

例11：

declare

v\_name emp.ename%type;

v\_sal emp.sal%type;

begin

select ename,sal into v\_name,v\_sal from emp where empno=7369;

dbms\_output.put\_line(v\_name||' '||v\_sal);

end;

dbms\_output.put\_line(v\_name||' '||v\_sal);

end;

---插入

例12：

declare

v\_deptno dept.deptno%type:=50;

v\_dname dept.dname%type:='aaaa';

v\_loc　　dept.loc%type:='bj';

begin

insert into dept2 values(v\_deptno,v\_dname,v\_loc);

commit;

end;

例13：

declare

v\_deptno emp2.deptno%type:=50;

v\_count number;

begin

update emp2 set sal=sal/2 where deptno=v\_deptno;

dbms\_output.put\_line(sql%rowcount ||'条记录被影响');(sql为关键字，代表上一条SQL语句，rowcount属性：影响了多少行，主要用于insert,updata,delete）

commit;

end;

/

例14：

declare

v\_deptno emp2.deptno%type:=50;

v\_count number;

begin

--update emp2 set sal=sal/2 where deptno=v\_deptno;

　 select deptno into v\_deptno from emp2 where empno=7369;

dbms\_output.put\_line(sql%rowcount ||'条记录被影响');(sql为关键字，代表上一条语句

commit;

end;

/

例15

declare

v\_deptno emp2.deptno%type:=50;

v\_count number;

begin

--update emp2 set sal=sal/2 where deptno=v\_deptno;

--select deptno into v\_deptno from emp2 where empno=7369;

select count(\*) into v\_count from emp2;　　（select必须和into一起使用）

dbms\_output.put\_line(sql%rowcount ||'条记录被影响');

commit;

end;

/

PL/SQL里面执行DDL语句：必须加execute immediate

begin

execute immediate 'create table T(nnn varchar2(20) default ' 'aaa' ')';

end;

PL/SQL的分支语句：

declare

　 v\_sal emp.sal%type;

begin

　 select sal into v\_sal from emp

　　where empno=7369;

　 if(v\_sal&lt;1200) then

　　 dbms\_output.put\_line('low');

　 elsif(v\_sal&lt;2000) then

　　 dbms\_output.put\_line('middle');

　 else

　　dbms\_output.put\_line('high');

　 end if;

end;

PL/SQL do while的循环

declare

　 i binary\_integer:=1;

begin

　 loop

　　　dbms\_output.put\_line(i);

　　　i:=i+1;

　　　exit when(i&gt;=11);

　 end loop;

end;

while循环

declare

　 j binary\_integer:=1;

begin

　 while j&lt;11 loop

　　　dbms\_output.put\_line(j);

　　　j:=j+1;

　 end loop;

end;

PL/SQL for循环

begin

for k in 1..10 loop

　　dbms\_output.put\_line(k);

end loop;

for k in reverse 1..10 loop　　　---reverse 逆序

　　dbms\_output.put\_line(k);

end loop;

end;

exception 捕获异常

too\_many\_rows：返回过多的数据

declare

　 v\_temp number(4);

begin

　 select empno into v\_temp from emp where deptno=10;　---会出现异常

exception

　 when too\_many\_rows then　　　　　 -catch(too\_many\_rows) {}

　 dbms\_output.put\_line('太多记录了');

　 when others then

　 dbms\_output.put\_line('error');

end;

no\_data\_found：没有数据错误

declare

　 v\_temp number(4);

begin

　 select empno into v\_temp from emp where deptno=2222;

exception

　 when no\_data\_found then

　 dbms\_output.put\_line('没数据');

when others then

　 dbms\_output.put\_line('error');

end;

/

PL\_SQL写的程序可移植性不强。---使用JDBC

一般做项目的时候都会采用“跨数据库”（既能在oracle也能在MSSQL或者其他数据库）的方法。

错误处理

create table errorlog

(

id number primary key,

errcode number,

errmsg varchar2(1024),

errdate date

);

create sequence seq\_errorlog\_id start with 1 increment by 1;　--从1开始，每次递增1

declare

v\_deptno dept.deptno%type:=10;

v\_errcode number;

v\_errmsg varchar2(1024);

begin

delete from dept where deptno=v\_deptno;

exception

when others then

　　rollback;

　　v\_errcode:=SQLCODE;　 ---SQLCODE关键字，出错代码

　　v\_errmsg:=SQLERRM;　 ---SQLERRM，出错信息

　　insert into errorlog values(seq\_errorlog\_id.nextval,v\_errcode,v\_errmsg,sysdate);

　　commit;

end;

游标cursor

cursor的4个属性：

isopen：游标是否打开

nofound：最近的一条fetch语句没有返回记录，则返回true

found：找到记录，返回true

count：找到的总条数

declare

　 cursor c is　　　　　　　--声明游标

　　　　　select \* from emp;

　 v\_emp c%rowtype;

　 begin

　　　open c;　　　　　　--打开游标（打开了游标才会执行select语句）

　　　fetch c into v\_emp;　　　　　 --(fetch取游标的第一条记录插入v\_emp，移动到下一条，再不断的循环--使用exit when)

　　　dbms\_output.put\_line(v\_emp.ename);

　　　close c;　　　　　　--清理内存

　 end;

例子

declare

　 cursor c is

　　　 select \* from emp;

　　　 v\_emp c%rowtype;

　　begin

　　　 open c;

　　loop

　　　 fetch c into v\_emp;

　　　 exit when(c%notfound);

　　　 dbms\_output.put\_line(v\_emp.ename);

　　 end loop;

　　close c;

end;

declare

　 cursor c is

　　　 select \* from emp;

　　　 v\_emp c%rowtype;

　　begin

　　　 open c;

　　　 fetch c into v\_emp;

　　 while(c%found) loop

　　　 dbms\_output.put\_line(v\_emp.ename);

　　　 fetch c into v\_emp;

　　 end loop;

　　close c;

end;

for循环不需要 打开和关闭cursor，v\_emp也不需要声明，也没有语句出现顺序的问题。

declare

　 cursor c is

　　　　 select \* from emp;

　 begin

　　　　 for v\_emp in c loop

　　　　 dbms\_output.put\_line(v\_emp.ename);

　　end loop;

end;

带参数的游标

declare

　 cursor c(v\_deptno emp.deptno%type, v\_job emp.job%type)

　 is

　　select ename,sal from emp where deptno=v\_deptno and job=v\_job;

begin

　 for v\_temp in c(30,'chick') loop

　　　　dbms\_output.put\_line(v\_temp.ename);

　 end loop;

end;

可更新的游标 for update　　　　　 where current of c

存储过程 storager procedure

create or replace procedure p

is

　 cursor c is

　　　　select \* from emp2 for update;

begin

　 for v\_emp in c loop

　　　　if (v\_emp.deptno=10) then

　　　　 update emp2 set sal=sal+10 where current of c;

　　　 elsif(v\_emp.deptno=20) then

　　　　 update emp2 set sal=sal+20 where current of c;

　　　 else

　　　　 update emp2 set sal=sal+50 where current of c;

　　　 end if;

　　end loop;

　　commit;

end;

exec p执行存储过程

begin

p;

end;

带参数的存储过程

create or replace procedure p

(v\_a in number,v\_b number,v\_ret out number, v\_temp in out number)

is

begin

if(v\_a&gt;v\_b) then

　　v\_ret:=v\_a;

　 else

　　v\_ret:=v\_b;

end if;

　　v\_temp:=v\_temp+1;

end;

in:传入参数（默认）：由调用环境给v\_a赋值，可以直接拿来用。

out:传出参数：由自己的程序将v\_a的值赋值到调用环境

in out:

调用存储过程

declare

v\_a number:=3;

v\_b number:=4;

v\_ret number;　　 ---给一个篮子，让调用环境给填值

v\_temp number:=5;

begin

　 p(v\_a,v\_b,v\_ret,v\_temp);

　 dbms\_output.put\_line(v\_ret);

　 dbms\_output.put\_line(v\_temp);

end;

show error返回错误信息

删除存储过程

存储过程中的函数

create or replace function sal\_tax

　　 (v\_sal number)

　　 return number　　　　　 ----返回一个number类型的值

is

begin

if(v\_sal&lt;2000) then

　　return 0.10;

elsif(v\_sal&lt;2750) then

　　return 0.5;

else

　　return 0.20;

end if;

end;

/

函数的调用：和以前的函数调用一样。

select lower(ename), sal\_tax(sal) from emp;

触发器trigger

在执行插入、删除、更新操作时，将操作的信息插入到触发器的表中。

create or replace trigger trig

　after insert or delete or update on emp2 for each row　 ---每执行一行都触发

begin

　if inserting then

　　　insert into emp2\_log values(USER, ‘insert’ sysdate); ---在执行插入时，插入

　elsif updating then

　　　insert into emp2\_log values(USER, ‘insert’ sysdate);

　elsif deleting then

　　　insert into emp2\_log values(USER, ‘insert’ sysdate);

　end if;

end;

/

触发器的副作用：很少用。

create or replace trigger trig

after update on dept　　　　　　---after之后，before之前

for each row

begin

update emp set deptno=:NEW.deptno where deptno=:OLD.deptno; ---可以修改参考值

end;

/

先触发触发器，再检查约束条件

树状结构的存储与展示

drop table article;

create table article

(

id number primary key,

cont varchar2(4000),

pid number,　　　　　---回复的父节点

isleaf number(1),　--0代表非叶子节点，1代表叶子节点(叶子节点：节点下没有子节点)

alevel number(2)

);

insert into article values (1,'蚂蚁大战大象', 0, 0, 0);

insert into article values (2,'大象被打趴下了', 1, 0, 1);

insert into article values (3,'蚂蚁也不好过', 2, 1, 2);

insert into article values (4,'瞎说', 2, 0, 2);

insert into article values (5,'没有瞎说', 4, 1, 3);

insert into article values (6,'怎么可能', 1, 0, 1);

insert into article values (7,'怎么没有可能', 6, 1, 2);

insert into article values (8,'可能性是很大的', 6, 1, 2);

insert into article values (9,'大象进医院了', 2, 0, 2);

insert into article values (10,'护士是蚂蚁', 9, 1, 3);

commit;

蚂蚁大战大象

大象被打趴下了

　　　　蚂蚁也不好过

　　　　瞎说

　　　　　　没有瞎说

　　　　大象进医院了

　　　　　　护士是蚂蚁

怎么可能

　　　　怎么没有可能

　　　　可能性是很大的

递归 recursion

create or replace procedure p (v\_pid article.pid%type, v\_level binary\_integer) is

　cursor c is select \* from article where pid = v\_pid;

　v\_preStr varchar2(1024) := ‘’;

begin

　for i in 0..v\_level loop

　　v\_preStr := v\_preStr || ‘----’;

　end loop;

　for v\_article in c loop

dbms\_output.put\_line(v\_preStr || v\_article.cont);

if(v\_article.isleaf = 0) then　　　 --注意，PL-SQL中if括号内=表示判断

　p (v\_article.id, v\_level + 1);

end if;

　end loop;

end;

/

set serveroutput on;

exec p(0,0);

文本118网：<http://www.txt118.com>

海量资料，等你发现