# Oracle(三)

1. 基本的PL/SQL块

|  |
| --- |
| /\*声明部分，以declare开头\*/  DECLARE  --declare statement  /\*执行部分，以begin开头,以end结束\*/  BEGIN  --execute statement  /\*异常处理部分，以exception开始\*/  EXCEPTION  WHEN exceptioncase THEN  --Exception handle  END; |

1. Oracle中的输入和输出

|  |
| --- |
| dbms\_output.put\_line(‘’);--输出一行内容，换行  dbms\_output.put(‘’);--输出一行内容，不换行  dbms\_output.new\_line();--新增一行  &变量名 –输入内容 |

1. 常见的数据库异常以及自定义异常

|  |
| --- |
| 1、ORA-06530: ACCESS\_INTO\_NULL--企图将值写入未初化对象的属性 2、ORA-06592:CASE\_NOT\_FOUND--case语句格式有误，没有分支语句 3、ORA-06531:COLLECTION\_IS\_NULL--企图将集合填入未初始化的嵌套表中 4、ORA-06511:CURSOR\_ALREADY\_OPEN--企图打开已经打开的指针．指针已经打开，要再次打开必须先关闭  5、ORA-00001:DUP\_VAL\_ON\_INDEX--数据库字段存储重复，主键唯一值冲突 6、ORA-01001:INVALID\_CURSOR--无效指针 7、ORA-01722:INVALID\_NUMBER--无效数字 8、ORA-01403:NO\_DATA\_FOUND--无数据发现 9、ORA-06504:ROWTYPE\_MISMATCH--行类型不匹配 10、ORA-06533:SUBSCRIPT\_BEYOND\_COUNT--子句超出数量 11、ORA-01410:SYS\_INVALID\_ROWID--无效的字段名 12、ORA-01422:TOO\_MANY\_ROWS--返回超过一行 13、ORA-01476:ZERO\_DIVIDE--除0错误 |

1. 分支结构
2. 循环结构
3. Oracle中的游标

|  |
| --- |
| -- 声明游标；CURSOR cursor\_name IS select\_statement |
| --For 循环游标  --（1）定义游标  --（2）定义游标变量  --（3）使用for循环来使用这个游标  declare  --类型定义  cursor c\_job  is  select empno,ename,job,sal  from emp  where job='MANAGER';  --定义一个游标变量v\_cinfo c\_emp%ROWTYPE ，该类型为游标c\_emp中的一行数据类型  c\_row c\_job%rowtype;  begin  for c\_row in c\_job loop  dbms\_output.put\_line(c\_row.empno||'-'||c\_row.ename||'-'||c\_row.job||'-'||c\_row.sal);  end loop;  end; |
| --Fetch游标  --使用的时候必须要明确的打开和关闭  declare  --类型定义  cursor c\_job  is  select empno,ename,job,sal  from emp  where job='MANAGER';  --定义一个游标变量  c\_row c\_job%rowtype;  begin  open c\_job;  loop  --提取一行数据到c\_row  fetch c\_job into c\_row;  --判读是否提取到值，没取到值就退出  --取到值c\_job%notfound 是false  --取不到值c\_job%notfound 是true  exit when c\_job%notfound;  dbms\_output.put\_line(c\_row.empno||'-'||c\_row.ename||'-'||c\_row.job||'-'||c\_row.sal);  end loop;  --关闭游标  close c\_job;  end; |
| 在执行DML（insert，update，delete）语句时，可以用到以下三个隐式游标（游标是维护查询结果的内存中的一个区域，运行DML时打开，完成时关闭，用sql%isopen检查是否打开）：  sql%found （布尔类型，默认值为null）  sql%notfound（布尔类型,默认值为null）  sql%rowcount(数值类型默认值为0)  sql%isopen(布尔类型)  当执行一条DML语句后，DML语句的结果保存在四个游标属性中，这些属性用于控制程序流程或者了解程序的状态。当运行DML语句时，PL/SQL打开一个内建游标并处理结果，游标是维护查询结果的内存中的一个区域，游标在运行DML语句时打开，完成后关闭。隐式游标只使用SQL%FOUND,SQL%NOTFOUND,SQL%ROWCOUNT三个属性.SQL%FOUND,SQL%NOTFOUND是布尔值，SQL%ROWCOUNT是整数值。  SQL%FOUND和SQL%NOTFOUND  在执行任何DML语句前SQL%FOUND和SQL%NOTFOUND的值都是NULL,在执行DML语句后，SQL%FOUND的属性值将是：  . TRUE :INSERT  . TRUE :DELETE和UPDATE，至少有一行被DELETE或UPDATE.  . TRUE :SELECT INTO至少返回一行  当SQL%FOUND为TRUE时,SQL%NOTFOUND为FALSE。  SQL%ROWCOUNT  在执行任何DML语句之前，SQL%ROWCOUNT的值都是NULL,对于SELECT INTO语句，如果执行成功，SQL%ROWCOUNT的值为1,如果没有成功或者没有操作(如update、insert、delete为0条），SQL%ROWCOUNT的值为0.  SQL%ISOPEN  SQL%ISOPEN是一个布尔值，如果游标打开，则为TRUE, 如果游标关闭，则为FALSE.对于隐式游标而言SQL%ISOPEN总是FALSE，这是因为隐式游标在DML语句执行时打开，结束时就立即关闭。  no\_data\_found 与 sql%notfound 以及 sql%rowcount 的区别：  NO\_DATA\_FOUND：该异常可以在两种不同的情况下出现：第一种：当SELECT。。。。INTO语的 WHERE子句 没匹配任何数据行时；第二种：试图引用尚未赋值的PL/SQL index-by表元素时。  SQL%NOTFOUND：是隐匿游标的属性，当没有可检索的数据时，该属性为：TRUE；常作为检索循环退出的条件。若某UPDATE或DELETE语句的WHERE子句不匹配任何数据行，该属性为：TRUE，但不并不出现NO\_DATA\_FOUND异常.  SQL%ROWCOUNT：该数字属性返回了到目前为止，游标所检索数据库行的个数。 |

1. PL/SQL块的注意事项