**表空间**

**创建表空间：**

**CREATE TABLESPACE tbs DATAFILE ‘d:\app\administrator\oradata\tbs\_1.dbf’ SIZE 50M**

**EXTENT MANAGEMENT LOCAL //可省略 另一种是DICTIONARY**

**UNIFORM SIZE 512K //区定制分配**

**SEGMENT SPACE MANAGEMENT MANUAL; //段手动管理**

**修改表空间：**

**ALTER TABLESPACE tbs ADD DATAFILE ‘d:\app\administrator\oradata\tbs\_2.dbf’ SIZE 10M; //添加数据文件**

**ALTER DATABASE DATAFILE ‘d:\app\administrator\oradata\tbs\_1.dbf’**

**AUTOEXTEND ON NEXT 1M MAXSIZE UNLIMITED; //改为自动扩展**

**ALTER DATABASE DATAFILE ‘d:\app\administrator\oradata\tbs\_1.dbf’ RESIZE 8M;**

**//手动修改数据文件大小**

**删除表空间：DROP TABLESPACE tbs INCLUDING CONTENTS AND DATAFILES;**

**查询表空间:V$tablespace dba\_tablespaces**

**改变数据文件的名称和位置：**

1. **alter tablespace tbs offline;**
2. **host copy**

**d:\app\administrator\oradata\tbs\_1.dbf**

**d:\app\administrator\tbs\_1.dbf**

1. **alter tablespace tbs rename datafile**

**‘d:\app\administrator\oradata\tbs\_1.dbf’ to**

**‘d:\app\administrator\tbs\_1.dbf’**

1. **alter tablespace tbs online;**

**查询数据文件：v$datafile dba\_data\_files**

**控制文件：**

**控制文件用于记录和维护数据库的物理结构，包括:数据库名称、数据文件和重做日志文**

**件的名称和位置等。**

**数据库至少要包含一个控制文件, Oracle建议使用多个控制文件避免因单个控制文件损**

**坏而导致的数据库无法启动。**

**控制文件对数据库至关重要，应联机保存多个备份，存储在不同的磁盘上。**

**多路复用控制文件: 4步。**

**（1）alter system set control\_files=**

**'d:\app\administrator\oradata\control01.ctl',**

**'d:\app\administrator\oradata\control02.ctl',**

**'d:\app\administrator\control03.ctl' scope=spfile;**

**（2）shutdown immediate**

**（3）host copy**

**d:\app\administrator\oradata\control01.ctl**

**d:\app\administrator\control03.ctl**

**（4）startup**

**查询控制文件: v$controlfile**

**范围分区：**

**Create table student\_range(**

**Sno number(6) primary key,**

**Sname varchar2(10),**

**Sage int,**

**Score number)**

**Partition by range(score)**

**(**

**Partition p1 values less than(60) tablespace tbs1,**

**Partition p2 values less than(80) tablespace tbs2,**

**Partition p3 values less than(maxvalue) tablespace tbs3**

**);**

**列表分区：**

**Create table student\_list(**

**Sno number(6) primary key,**

**Sname varchar2(10),**

**Sex char(2) constraint s\_ck1 check(sex in(‘M’,’F’)**

**)**

**Partition by list(sex)**

**(**

**Partition stu\_male values(‘M’) tablespace tbs1,**

**Partition stu\_female values(‘F’) tablespace tbs2**

**);**

**用户管理**

**创建用户：create user user1 identified by user1 default tablespace users temporary tablespace temp quota 5M on users password expire account lock;**

**修改用户：alter user user1 identified by user1 account unlock;**

**删除用户：drop user user1 CASCADE;**

**授权管理**

**授系统权限：grant create session,create table to user1;**

**授对象权限：grant update,select on scott.emp to user1;**

**回收权限：revoke create table from user1;**

**修改表：**

**Alter table student ADD(phone varchar2(10));**

**Alter table student ADD(hiredate DATE default sysdate not null);**

**Alter table student ADD constraint p\_ck1 check(sage between 20 and 30);**

**Alter table student ADD constraint p\_ck2 check(sex=’M’ or sex=’F’) modify sex default ‘M’;**

**CREATE TABLE employee(**

**empno NUMBER(5) PRIMARY KEY,**

**ename VARCHAR2(15),**

**deptno NUMBER(3) NOT NULL CONSTRAINT fk\_emp REFERENCES dept(deptno)**

**);**

**模糊查询：**

**SELECT \* FROM employees WHERE last\_name LIKE '%S%'; 名字中含有S的**

**SELECT \* FROM employees WHERE first\_name LIKE '\_a%'; 第二个字母是a的**

**SELECT \* FROM employees WHERE first\_name LIKE '%x\_%' ESCAPE 'x'; 名字中含有\_的**

**空值查询：**

**SELECT \* FROM employees WHERE commission\_pct IS NOT NULL;**

**多列排序**

**SELECT \* FROM employees ORDER BY department\_id,salary DESC; ACS升序，默认**

**统计查询**

**SELECT count(\*),avg(salary),max(salary),min(salary) FROM employees WHERE**

**department\_id=50;**

**分组查询**

**SELECT department\_id,count(\*),avg(salary) FROM employees GROUP BY**

**department\_id HAVING avg(salary)>8000;**

**连接查询**

**内连接：**

**select w.ename,w.empno,w.sal,m.ename,m.empno,m.sal from emp w,emp m where w.mgr=m.empno and w.deptno=10；**

**左外连接**

**SELECT dname,empno,ename FROM dept LEFT JOIN emp ON**

**dept.deptno=emp.deptno AND dept.deptno=10;**

**右外连接**

**SELECT empno,ename,dname FROM dept RIGHT JOIN emp ON**

**dept.deptno=emp.deptno AND dept.deptno=20;**

**INSERT INTO emp\_s SELECT deptno,max(salary), min(salary) FROM emp GROUP BY deptno;**

**INSERT INTO dept VALUES(departments\_seq.nextval, 'Teaching',200,1800);**

**UPDATE emp SET sal=300+ (SELECT avg(sal) FROM emp WHERE deptno=10)**

**WHERE deptno=30;**

**DELETE FROM emp WHERE sal>(SELECT sal FROM emp WHERE empno=7900);**

**创建一个以员工号为参数，返回该员工所在部门平均工资的函数。编写PL/SQL块，输出7788号员工所在部门的平均工资。**

**create or replace function func\_avgsal(**

**p\_empno emp.empno%type)**

**return emp.sal%type**

**as**

**v\_deptno emp.deptno%type;**

**v\_avgsal emp.sal%type;**

**begin**

**select deptno into v\_deptno from emp where empno=p\_empno;**

**select avg(sal) into v\_avgsal from emp where deptno=v\_deptno;**

**return v\_avgsal;**

**end func\_avgsal;**

**创建一个函数，以员工号为参数，返回该员工的工资。并编写PL/SQL块，调用该函数求出所有员工的员工号，员工姓名和工资。**

**create or replace function func\_sal(**

**p\_empno emp.empno%type,**

**p\_ename out emp.ename%type)**

**return emp.sal%type**

**as**

**v\_sal emp.sal%type;**

**begin**

**select sal into v\_sal from emp where empno=p\_empno;**

**select ename into p\_ename from emp where empno=p\_empno;**

**return v\_sal;**

**EXCEPTION**

**WHEN NO\_DATA\_FOUND THEN RAISE\_APPLICATION\_ERROR(-20000,'There is not such an employee!');**

**end func\_sal;**

**declare**

**v\_sal emp.sal%type;**

**p\_ename emp.ename%type;**

**begin**

**for v\_emp in (select distinct empno from emp where empno is not null) loop**

**v\_sal:=func\_sal(v\_emp.empno,p\_ename);**

**dbms\_output.put\_line(v\_emp.empno||' '||p\_ename||' '||v\_sal);**

**end loop;**

**end;**

**创建一个存储过程，以部门号为参数，查询该部门的平均工资，并输出该部门中比平均工资高的员工姓名、工资及所在员工号。编写PL/SQL块，调用此过程求出20号部门平均工资以及20号部门中比平均工资高的员工号、员工名和工资。**

**create or replace procedure proc\_show(**

**p\_deptno emp.deptno%type)**

**as**

**v\_sal emp.sal%type;**

**begin**

**select avg(sal) into v\_sal from emp where deptno=p\_deptno;**

**dbms\_output.put\_line(p\_deptno||' '||'avgsal is:'||v\_sal);**

**for v\_emp in (select \* from emp where deptno=p\_deptno and sal>v\_sal)loop**

**dbms\_output.put\_line(v\_emp.empno||' '||v\_emp.ename||' '||v\_emp.sal);**

**end loop;**

**end proc\_show;**

**创建一个存储过程，以部门号为参数，返回该部门的人数和最高工资和最低工资。编写PL/SQL块求出所有部门的人数、最高工资和最低工资。**

**create or replace procedure proc\_show3(**

**p\_deptno emp.deptno%type,**

**p\_count out number,**

**p\_maxsal out emp.sal%type,**

**p\_minsal out emp.sal%type)**

**as**

**v\_sal emp.sal%type;**

**begin**

**select count(\*),max(sal),min(sal) into p\_count,p\_maxsal,p\_minsal from emp where deptno=p\_deptno;**

**exception**

**when no\_data\_found then**

**dbms\_output.put\_line('the department doesn't exists');**

**end proc\_show3;**

**declare**

**v\_count number;**

**v\_maxsal emp.sal%type;**

**v\_minsal emp.sal%type;**

**begin**

**for v\_emp in (select distinct deptno from emp)loop**

**proc\_show3(v\_emp.deptno,v\_count,v\_maxsal,v\_minsal);**

**dbms\_output.put\_line(v\_emp.deptno||' '||v\_count||' '||v\_maxsal||' '||v\_minsal);**

**end loop;**

**end;**

**create or replace procedure pro\_up(**

**p\_empno emp.empno%type)**

**as**

**v\_deptno emp.deptno%type;**

**v\_inc number(4);**

**begin**

**select deptno into v\_deptno from emp where empno=p\_empno;**

**case v\_deptno**

**when 10 then v\_inc:=140;**

**when 20 then v\_inc:=200;**

**when 30 then v\_inc:=250;**

**else v\_inc:=300;**

**end case;**

**update emp set sal=sal+v\_inc where empno=p\_empno; end pro\_up;**

**create or replace procedure pro\_up(**

**p\_empno emp.empno%type)**

**as**

**v\_deptno emp.deptno%type;**

**v\_inc number(4);**

**begin**

**select deptno into v\_deptno from emp where empno=p\_empno;**

**if v\_deptno=10 then v\_inc:=140;**

**elsif v\_deptno=20 then v\_inc:=200;**

**elsif v\_deptnp=30 then v\_inc:=250;**

**else v\_inc:=300;**

**end if;**

**update emp set sal=sal+v\_inc where empno=p\_empno; end pro\_up;**

**begin**

**for v\_emp in (select distinct empno from emp where empno is not null) loop**

**pro\_up(v\_emp.empno);**

**end loop;**

**end;**