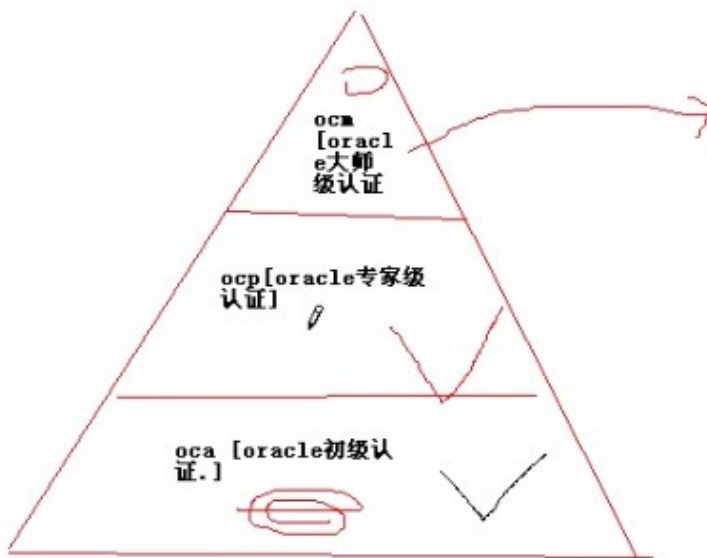


韩顺平—玩转oracle视频教程笔记

一：Oracle认证，与其它数据库比较，安装



小型数据库

access, foxbase

你该用什么数据库:

- 项目的规模:
 - 负载量多大, 用户多大
 - 成本.
 - 安全性.

负载量小, 100人内.
比如留言板, 信息系统.
成本在千元内.
对安全性要求不高.

中型数据库

mysql sql server,
informix

比如在负载 日访问量
5000~15000
成本在万元内.

比如商务网站.

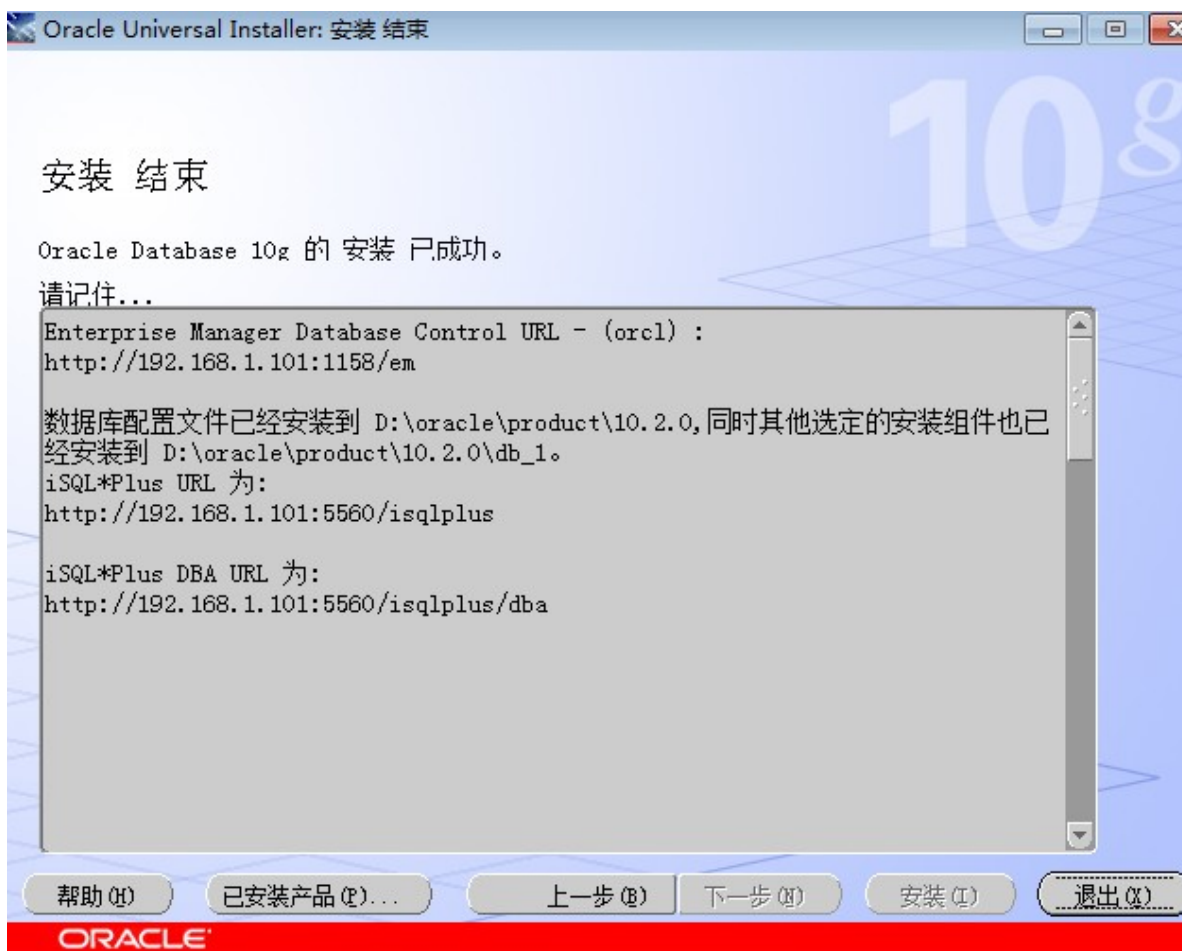
大型数据库

sybase, oracle, db2

负载可以处理 海量数据库.

sybase < oracle < db2

这几个数据库的安全性很高. 相对贵.



连接命令

(1) conn[ect]

用法: conn 用户名/密码@网络服务名[as sysdba/s ysoper]

例如: conn system /manager

当用户特权用户连接时，必须带上（例如sys的登录）as sysdba 或是as sysoper显示当前用户
show user;

(2)disc[onnect]

说明：该命令用来断开与当前数据库的连接

(3)passw[ord]

说明：该命令用于修改用户的密码，如果想修改其他用户的密码，需要用
sys/system登录

(4)show user

说明：显示当前用户名

(5)exit

说明：该命令会断开与数据库的连接，同时会退出sql*plus
文件操作命令

(1) start和@

说明：运行sql脚本

案例：sql>@ d:\a.sql 或者sql>start d:\a.sql

(2) edit

说明：该命令可以编辑指定的sql脚本

案例：sql>edit d:\a.sql

(3) spool

说明：该命令可以将sql*plus屏幕上的内容输出到指定文件中

案例：sql>spool d:\b.sql 并输入sql>spool off

显示和设置环境变量

概述：可以用来控制输出的格式，set show如果希望永久的保存相关的设置，可以去修改
glogin.sql脚本

(1) linesize

说明：设置显示行的宽度，默认是80个字符

Sql>show linesize

Sql>set linesize 90

(2) pagesize

说明：设置每页显示的行数目，默认是14，用法和linesize一样

至于其它环境参数的使用也是大同小异

创建用户

概述：在oracle中要创建一个新的用户使用create user语句，一般具有dba（数据库管理员）的权限才能使用

案例：create user xiaoming identified by m123;

给修改用户密码

概述：如果给自己修改密码可以直接使用

Sql>password 用户名

如果给别人修改密码则需要具有dba的权限，或者拥有alter user的系统权限

Sql>alter user 用户名 identified by 新密码

删除用户

概述：一般以dba的身份去删除某个用户，如果用其它用户去删除用户则需要具有drop user的权限。比如：drop user 用户名 **【cascade】**，在删除用户时注意，如果删除的这个用户，已经创建了表，那么就需要在删除时带一个参数

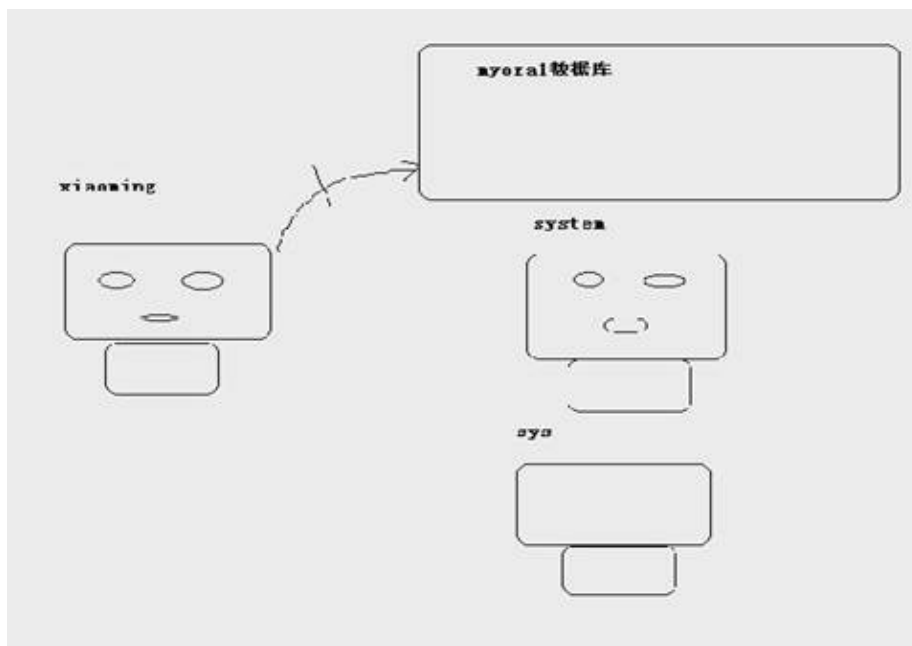
用户解锁：

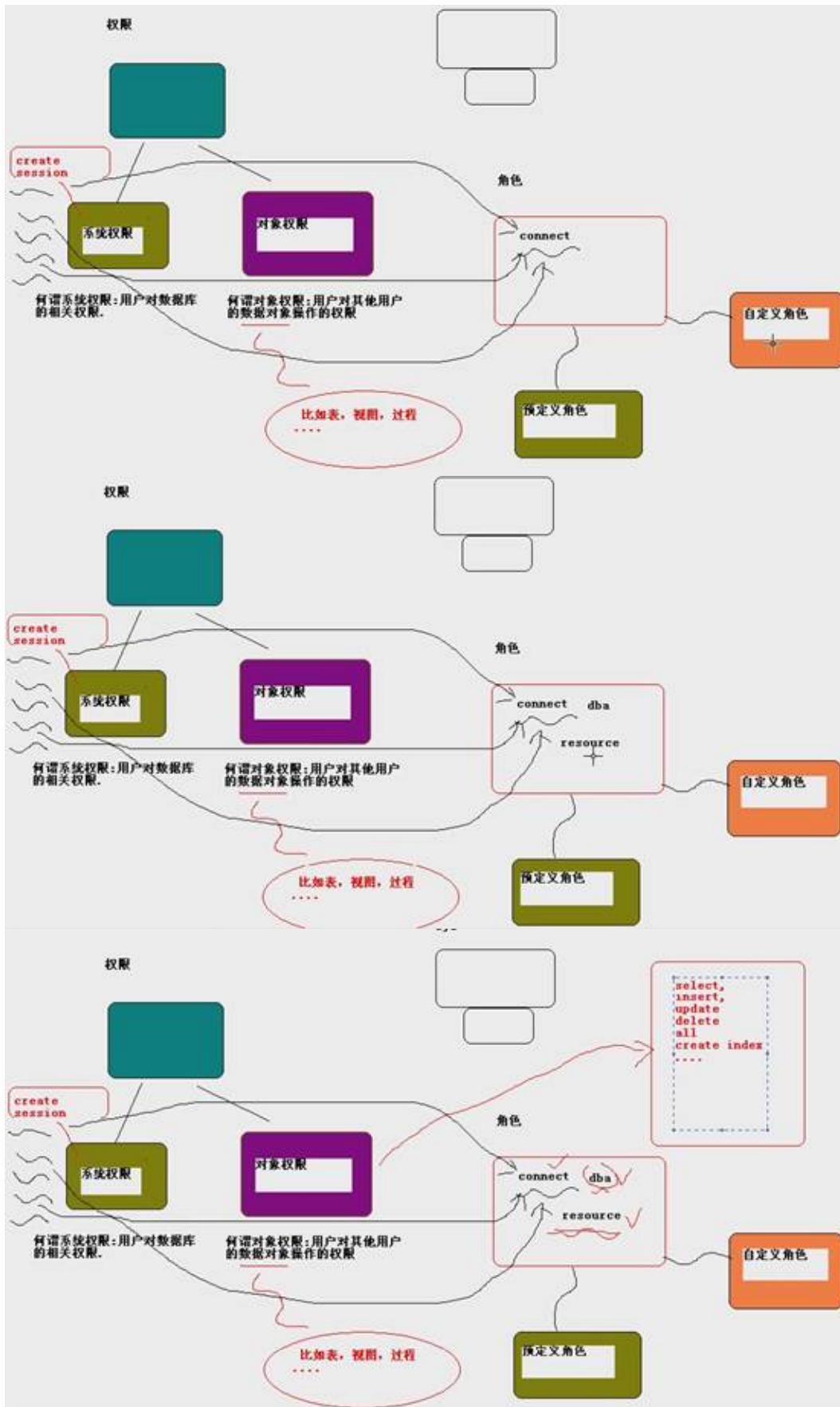
alter user user_name account unlock;

cascade;

用户管理的综合案例

概述：创建的新用户是没有任何权限的，甚至连登录数据库的权限都没有，需要为指定相印的权限。给一个用户赋权限使用命令grant，回收权限使用的命令是revoke。





希望xiaoming这个用户可以去查询emp表

希望xiaoming这个用户可以去查询scott的 emp表

命令是: grant select on emp to xiaoming

希望xiaoming用户可以去修改scott的emp表

命令是: `grant update on emp to xiaoming`

希望xiaoming用户可以修改/删除、查询、添加scott的emp表

命令是: `grant all on emp to xiaoming`

Scott希望收回xiaoming对emp表的查询权限.

命令是: `revoke select on emp from xiaoming`

//对权限的维护.

希望xiaoming这个用户可以去查询scott的 emp表/还希望xiaoming可以把这个权限继续给别人.

如果是对象权限, 就加入with grant option

命令是: `grant select on emp to xiaoming with grant option`

其他的权限类似。

如果是系统权限（和对象权限类似）.

System给xiaoming赋权时:

命令是: `grant connect to xiaoming with admin option`

如果ccott把xiaoming对emp表的查询权限, 那么xiaohong会怎么样?

Xiaohong的权限也被回收了.

使用profile管理用户口令

概述: profile是口令限制, 资源限制的命令集合, 当简历数据库时, oracle会自动建立名称为default的profile.

当建立用户没有指定profile选项, 那么oracle就会将default分配给用户

(1) 账户锁定

概述: 指定该账户（用户）登录时最多可以输入密码的次数, 也是可以指定用户锁定的时间（天）一般用dba的身份去执行该命令

例子: 指定scott这个用户最多只能尝试3次登录, 锁定时间为2天, 让我们看看怎么实现.

创建profile文件

```
sql>create profile lock_account limit  
failed_login_attempts 3 password_lock_time 2;
```

```
sql>alter user scott profile lock_account;
```

(2) 给账户（用户）解锁

```
Sql> alter user scott account unlock; (dba权限才能操作)
```

(3) 终止口令

为了让用户定期修改密码可以使用终止口令的指令来完成, 同样这个命令也需要dba身份来操作.

例子: 给前面创建的用户scott创建一个profile文件, 要求该用户每隔10天要修改自家的登录密码, 宽限期限为2天。如下

```
Sql> creat profile myprofile limit password_life_time 10 password_grace_time 2;
```

```
Sql>alter user scott profile myprofile
```

口令历史

概述：如果希望用户在修改密码时，不能使用以前使用过的密码，可使用命令历史，这样oracle就会将口令修改的信息存放到数据字典中，这样当用户修改密码时，oracle就会对新旧密码进行比较，当发现新旧密码一样时，就提示用户重新输入密码。

1) 建立profile

```
Sql>create profile password_history limit password_life_time 10
```

```
password_grace_time 2 password_reuse_time 10
```

Password_reuse_time //指定口令可重用时间即10天后就可以重用

2) 分配给某个用户.

删除profile

概述：当不需要某个profile文件时，可以删除该文件.

```
Sql> drop profile password_history 【cascade】
```

Oracle的表的管理

表名和列的命名原则

必须以字母开头

长度不能超过30字符

不能使用oracle的保留字

只能使用如下的字符：A-Z, a-z, 0-9, \$, #等

Oracle支持的数据的类型

字符型

Char 定长 最大2000字符.

例子：char(10) ‘小韩’前四个字符放‘小韩’，后添6个空格不全 ‘小韩’

Varchar2(20) 变长 最长4000字符

例子：varchar2(2) ‘小寒’ oracle分配四个字符. 这样可以节省空间

Clob(character large object) 字符型大对象 最大4G

数字型

Number范围-10的38次方到10的38次方

可以表示整数，也可以表示小数.

例：Number(5, 2) 表示一个小数有5位有效数，2位小数位，范围-999.99—999.99

Number(5) 表示一个5位数整数范围为-99999 99999

日期类型

Date 包含年月日和时分秒

Timestamp这个ora9i对数据类型的扩展.

图片

Blob 二进制数据 可以存放图片/声音 4G

处于视频音频的安全性考虑可以直接存放到数据库中。一般的话只是存放在一个文件夹下面，在数据库里面存放路径即可。

建表

—学生表

```
Sql>create table student (-表名
```

```
Xh number(4), -学号
```

```
Xm varchar2(20), -姓名
```

```
Sex char(2), -性别
```

```
Birthday date, 出生日期
```

```
Sal number(7,2)-奖金
```

```
);
```

删除表命令: **drop table** user

再创建一个班级的表!

添加一个字段

```
Sql>alter table student add(classid number(2));
```

修改字段的长度

```
Sql>alter table student modify (xm varchar2(30));
```

修改字段的类型/或是名字(不能有数据)

```
Sql>alter table student modify (xm char(30));
```

删除一个字段

```
Sql>alter table student drop column sal;(一般工作中不用)
```

修改表的名字

```
Sql>drop table student;
```

删除表

```
Sql>drop table student;
```

添加数据

```
insert into student values(2,'小明2','男','2-12月-96',23465.11,2);
```

注意: oracle中默认日期格式' dd-mon-yy' dd 日子(天)mon月份yy2位年' 09-3月-12' ,
该日期的默认格式:

```
Alter session set nls_date_format = ' yyyy-mm-dd' ;
```

修改后, 可以用我们熟悉的格式添加日期类型:

```
insert into student values(2,'小明2','男','1996-12-2',23465.11,2);
```

插入部分字段

Insert into student (xh,xm,sex) values('a003' ,john' , ' 女'); 注: 不能为空的不能为空

插入空值

Insert into student(xh,xm,sex,birthday) values('a004' , ' martin' , ' 男' ,null);

注意怎么查询为部分为空的表: 格式如下

Select * from student where birthday is null;

改一个字段

Update student set sex=' 女' where xh=' A001' ;

修改多个字段

Update student set sex=' 男' ,birthday=' 1980-04-01' where xh=' A002' ;

修改含有null的

删除数据

Delete from student;

删除所有的记录, 表结构还在, 写日志, 可以恢复的, 速度慢

Drop table student; 删除表的结构和数据

Delete from student where xh=' A001' ;删除一条记录

Truncate tablestudent; 删除表中的所有记录, 表结构还在, 不写日志, 无法找回删除的记录, 速度快。

设置保存点: **savepoint** aa;

回滚保存点: **rollback to** aa;

表的查询

查看表的结构

emp

上级的编号

奖金

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	1980-12-17	800.00		20
7499	ALLEN	SALESMAN	7698	1981-2-20	1600.00	300.00	30
7521	WARD	SALESMAN	7698	1981-2-22	1250.00	500.00	30
7566	JONES	MANAGER	7839	1981-4-2	2975.00		20
7654	MARTIN	SALESMAN	7698	1981-9-28	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1981-5-1	2850.00		30
7782	CLARK	MANAGER	7839	1981-6-9	2450.00		10
7788	SCOTT	ANALYST	7566	1987-4-19	3000.00		20
7839	KING	PRESIDENT		1981-11-17	5000.00		10
7844	TURNER	SALESMAN	7698	1981-9-8	1500.00	0.00	30
7876	ADAMS	CLERK	7788	1987-5-23	1100.00		20
7900	JAMES	CLERK	7698	1981-12-3	950.00		30
7902	FORD	ANALYST	7566	1981-12-3	3000.00		20
7934	MILLER	CLERK	7782	1982-1-23	1300.00		10

部门表

```
SQL> select * from dept;
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Sql>desc dept;

查询所有列

Select * from dept;

查询制定列

Select ename, sal, job, deptno, from emp;

如何取消重复行

Select distinct deptno, job from emp;

? 查询SMITH的薪水, 工作, 所在部门

设置执行时间开启命令: set timing

on;

Select SAL, JOB, DEPT from emp where ename=' SMITH' ;

循环创建记录命令:

Insert into users (userid, username, userpass) select *from users;

使用算数表达式

? 显示每个雇员的年工资

Select sal*13+nvl(comm,0)*13 “年工资”, ename from em;

使用列的别名

Select ename “姓名”, sal*12 as “年收入” from emp;

如何处理null值

使用nvl函数来处理

如何连接字符串(||)

```
Select ename || 'is a' || job from emp;
```

使用where子句

? 如何显示工资高于3000的员工

```
Select ename,sal from emp where sal>3000;
```

? 如何查找1982.1.1后入职的员工

```
Select ename,hiredat from emp where hiredat>' 1982.1.1' ;
```

? 如何显示工资在2000到2500的员工的情况

```
Select ename,sal from emp where sal>2000 and sal<2500;
```

如何使用like操作符

%: 表示任意0到多个字符, _: 表示任意字符

? 如何显示首字符为S的员工姓名和工资

```
Select ename,sal from emp where ename like 'S%';
```

? 如何显示第三个字符为大写O的所有员工的姓名和工资

```
Select ename,sal from emp where ename like '__O%';
```

在where条件中使用in

? 如何显示empno为123, 345, 678...的雇员的情况

```
Select ename,sal from emp where empno in(123,345,678);
```

使用is null 的操作符

? 如何显示没有上级雇员的情况

```
Select ename,sal from emp where mgr is null;
```

使用逻辑操作符号

? 查询工资高于500或是岗位为manager的雇员, 同时还要满足他们的姓名首写字母为大写的J

```
select ename,job,sal from emp where (sal>500 or job='manager') and ename like 'j%';
```

使用order by子句

? 如何按照工资的从低到高的顺序显示雇员的信息

```
select * from emp order by sal;
```

? 按照部门号升序而雇员的工资降序排列(默认为升序, 降序的时候用desc).

```
select * from emp order by dept , sal desc;
```

使用列的别名排序

```
select ename, (sal+nvl(comm,0))*12 "年薪" from emp order by "年薪" desc;
```

```
Select ename,sal*12 "年薪" from emp order by "年薪" asc;
```

别名需要使用"号圈中

Oracle 表的复杂查询

说明

在实际应用中经常需要执行复杂的数据统计，经常需要显示多张表的数据，现在我们给大家介绍较为复杂的select语句

数据分组-max, min, avg, sum, count

? 如何显示所有员工中最高工资和最低工资

```
select max(sal),min(sal) from emp;
```

? 显示所有员工的平均工资和工资总和

```
select avg(sal),sum(sal) from emp;
```

? 计算共有多少员工

```
select count(ename) from emp;
```

扩展要求:

? 请显示工资最高的员工的姓名，工作岗位

```
select ename,sal,job from emp where sal=(select max(sal) from emp);
```

? 请显示工资高于平均工资的员工信息

```
select * from emp where sal> (select avg(sal) from emp);
```

group by 和having子句

group by 用于对查询的结果分组统计，

having子句用于限制分组显示结果，

? 如何显示每个部门的平均工资和最高工资

```
select avg(sal),max(sal), dept from emp group by dept;
```

? 显示每个部门的每个岗位的平均工资和最低工资

```
select avg(sal),max(sal), dept,job from emp group by dept,job order by dept;
```

? 显示平均工资低于2000的部门号和它的平均工资

```
SQL> select avg(sal), dept from emp group by dept having avg(sal)>2000 ;
```

```
AVG(SAL) DEPT
```

```
-----
```

```
2175 20
```

```
2916.66666 10
```

扩展要求:

```
SQL> select avg(sal), dept from emp group by dept having avg(sal)>2000  
order by avg(sal) desc;
```

```
AVG(SAL) DEPT
```

```
-----
```

```
2916.66666 10
```

```
2175 20
```

对数据分组的总结

1. 分组函数只能出现在选择列表、having、order by子句中
2. 如果在select 语句中同时含有group by, having, order by 那么他们的顺序是group by, having, order by
3. 在选择列中如果有列、表达式、和分组函数, 那么这些列和表达式必须有一个出现在group by子句中, 否则就会出错

如select dept, avg(sal),max(sal) from emp group by dept having avg(sal)
<2000;这里dept就一定要出现在group by中.

多表的查询

说明

多表查询是指基于两个和两个以上的表或是视图的查询。在实际应用中, 查询单个表中可能不能满足你的需求, (如显示sales部门位置和员工的姓名), 这种情况下需要使用到(dept表和emp表)

? 显示雇员名, 雇员工资及所在部门的名字

【笛卡尔集】

规定: 多表查询的条件是至少不能少于表的个数减一。

```
SQL> select a1.*, a2.* from emp a1,dept a2 where a1.dept=a2.deprno;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT	DEPRNO
DNAME	LOC							
7369	SMITH	CLERK	7902	17/12/1989	800.00		20	20
	RESERCH	DALLAS						
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30	30
	SALES	CHICAGO						
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30	30
	SALES	CHICAGO						
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20	20
	RESERCH	DALLAS						
7654	MARTIN	SALEMAN	7698	28/09/1981	1250.00	1400.00	30	30
	SALES	CHICAGO						
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30	30
	SALES	CHICAGO						
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10	10
	ACCOUNTING	NEW YORK						

7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		20	20
RESERCH		DALLAS						
7839	KING	PRESIDENT		17/11/1981	5000.00		10	10
ACCOUNTING		NEW YORK						
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30	30
SALES		CHICAGO						
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20	20
RESERCH		DALLAS						
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30	30
SALES		CHICAGO						
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20	20
RESERCH		DALLAS						
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10	10
ACCOUNTING		NEW YORK						

? 如何显示部门号为10的部门名、员工名和工资

```
SQL> select  a1.ename,a1.sal, a2.dname from emp a1,dept a2 where
a1.dept=a2.deprno and a2.deprno=10;
```

ENAME	SAL	DNAME
CLARK	2450.00	ACCOUNTING
KING	5000.00	ACCOUNTING
MILLER	1300.00	ACCOUNTING

```
SQL> select  a1.ename,a1.sal, a2.dname from emp a1,dept a2 where
a1.dept=a2.deprno and a1.dept=10;
```

ENAME	SAL	DNAME
CLARK	2450.00	ACCOUNTING
KING	5000.00	ACCOUNTING
MILLER	1300.00	ACCOUNTING

? 显示各个员工的姓名，工资及其工资的级别

```
SQL> select  a1.ename,a1.sal,a2.grade from emp a1,salgrade a2 where a1.sal
between a2.losal and a2.hisal;
```

ENAME	SAL	GRADE
SMITH	800.00	1
JAMES	950.00	1
ADAMS	1100.00	1
WARD	1250.00	2
MARTIN	1250.00	2
MILLER	1300.00	2
TURNER	1500.00	3
ALLEN	1600.00	3
CLARK	2450.00	4
BLAKE	2850.00	4
JONES	2975.00	4
SCOTT	3000.00	4
FORD	3000.00	4
KING	5000.00	5

扩展要求：

? 显示雇员名，雇员工资及所在部门的名称，并按部门排序。

```
SQL> select  a1.ename,a1.sal, a2.dname from emp a1,dept a2 where
a1.dept=a2.deprno order by a2.dname;
```

ENAME	SAL	DNAME
CLARK	2450.00	ACCOUNTING
KING	5000.00	ACCOUNTING
MILLER	1300.00	ACCOUNTING
JONES	2975.00	RESERCH
FORD	3000.00	RESERCH
ADAMS	1100.00	RESERCH
SMITH	800.00	RESERCH
SCOTT	3000.00	RESERCH
WARD	1250.00	SALES
TURNER	1500.00	SALES
ALLEN	1600.00	SALES
JAMES	950.00	SALES
BLAKE	2850.00	SALES
MARTIN	1250.00	SALES

多表查询

自连接

自连接是指在同一张表的连接查询.

?显示某个员工的上级领导的姓名

比如显示' FORD' 的上级.

```
SQL> select work.ename,boss.ename from emp work, emp boss where
work.mgr=boss.empno and work.ename='FORD';
```

```
ENAME      ENAME
-----
      FORD      JONES
```

子查询

什么是子查询

子查询是指嵌入在其它sql语句中的select语句, 也叫嵌套查询

单行子查询

单行子查询是指只返回一行数据的子查询语句

请思考: 如何显示与SMITH 同一部门的所有员工? (数据库在扫描的时候是从左到右, 执行sql 是从左到右, 有括号先括号)

```
SQL> select * from emp where dept=(select dept from emp where
ename='SMITH');
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		20
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20

多行子查询

多行子查询指返回多行数据的子查询

请思考: 如何查询和部门10的工作相同的雇员的名字、岗位、工资、部门号。

```
SQL> select * from emp where job in
2  (select distinct job from emp where dept=10);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20

7839	KING	PRESIDENT		17/11/1981	5000.00		10
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7369	SMITH	CLERK	7902	17/12/1989	800.00		20

在多行查询中使用all操作符

请思考：如何显示工资比部门30的所有员工的工资高的员工的信息。

方法一：

```
SQL> select * from emp where sal> all (select sal from emp where dept=30);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
-----	-----	-----	-----	-----	-----	-----	----
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		20
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7839	KING	PRESIDENT		17/11/1981	5000.00		10

方法二：

```
SQL> select * from emp where sal> (select max(sal) from emp where dept=30);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
-----	-----	-----	-----	-----	-----	-----	----
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		20
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20

在多行子查询中使用any操作符

请思考：如何显示工资比部门30的任意员工的工资高的员工的信息。

方法一：

```
SQL> select * from emp where sal> any (select sal from emp where dept=30);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
-----	-----	-----	-----	-----	-----	-----	----
7839	KING	PRESIDENT		17/11/1981	5000.00		10

7902 FORD	ANALYST	7566 3/12/1981			
3000.00		20			
7788 SCOTT	ANALYST	7566 19/04/1987			
3000.00		20			
7566 JONES	MANAGER	7839 2/04/1981	2975.00		
20					
7698 BLAKE	MANAGER	7839 1/05/1981			
2850.00		30			
7782 CLARK	MANAGER	7839 9/06/1981			
2450.00		10			
7499 ALLEN	SALESMAN	7698 20/02/1981	1600.00	300.00	30
7844 TURNER	SALESMAN	7698 8/09/1981	1500.00	0.00	30
7934 MILLER	CLERK	7782 23/01/1982			
1300.00		10			
7521 WARD	SALESMAN	7698 22/02/1981	1250.00	500.00	30
7654 MARTIN	SALEMAN	7698 28/09/1981	1250.00	1400.00	30
7876 ADAMS	CLERK	7688 23/05/1987	1100.00	0.00	
20					

方法二：

SQL> select * from emp where sal> (select min(sal) from emp where dept=30);

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981			
2975.00				20			
7654	MARTIN	SALEMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981			
2850.00				30			
7782	CLARK	MANAGER	7839	9/06/1981			
2450.00				10			
7788	SCOTT	ANALYST	7566	19/04/1987			
3000.00				20			

7839 KING	PRESIDENT		17/11/1981			
5000.00		10				
7844 TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
7876 ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	
20						
7902 FORD	ANALYST	7566	3/12/1981			
3000.00		20				
7934 MILLER	CLERK	7782	23/01/1982	1300.00		
	10					

多列子查询

单行子查询是指查询只返回单列、单行数据、多行子查询是指返回单列多行数据，都是针对单列而言的，而多列子查询则是指查询返回多个列数据的子查询语句

请思考：如何查询与SMITH的部门和岗位完全相同的所有雇员

```
SQL> select * from emp where (dept,job) = (select dept,job from emp where
ename='SMITH');
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20

在from子句中使用子查询

请思考：如何显示高于自己部门平均工资的员工的信息

在这里需要说明的当在from子句中使用子查询时，该子查询被作为一个视图来对待，因此叫做内嵌视图，当在from子句中使用子查询时，必须给子查询指定别名。

//1. 查询出各个部门的平均工资和部门号

```
select deptno ,avg(sal) mysal from emp group by deptno;
```

//2. 把上面的查询看做是一张 子表

```
select a2.ename,a2.sal,a2.deptno,a1.mysal from emp a2, (select deptno ,avg
(sal) mysal from emp group by deptno) as a1 where a2.deptno=a1.deptno and
a2.sal>a1.mysal|
```

```
SQL> select * from emp a1,(select dept, avg(sal) mysal from emp group by
dept) a2 where a1.dept=a2.dept and a1.sal>a2.mysal;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
DEPT	MYSAL						

7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30	30
1566.66666								
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20	
20	2175							
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30	30
1566.66666								
7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		20	
20	2175							
7839	KING	PRESIDENT		17/11/1981	5000.00		10	10
2916.66666								
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20	
20	2175							

分页查询

按雇员的id号升序取出

1. 根据ROWID来分

```
select * from t_xiaoxi where rowid in(select rid from (select rownum rn,rid  
from(select rowid rid,cid from  
t_xiaoxi order by cid desc) where rownum<10000) where rn>9980) order by cid  
desc;
```

执行时间0.03秒

2. 按分析函数来分

```
select * from (select t.*,row_number() over(order by cid desc) rk from  
t_xiaoxi t) where rk<10000 and rk>9980;
```

执行时间1.01秒

3. 按ROWNUM来分

```
select * from(select t.*,rownum rn from(select * from t_xiaoxi order by cid  
desc) t where rownum<10000) where
```

rn>9980; 执行时间0.1秒

其中t_xiaoxi为表名称，cid为表的关键字段，取按CID降序排序后的第9981-9999条记录，
t_xiaoxi表有70000多条记录

个人感觉1的效率最好，3次之，2最差

//测试通过的分页查询okokook

```
select * from (select a1.*,rownum rn from (select ename,job from emp) a1  
where rownum<=10) where rn>=5 ;
```

```
SQL> select a1.*,rownum rn from (select * from emp) a1;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
-------	-------	-----	-----	----------	-----	------	------

RN

7369	SMITH	CLERK	7902	17/12/1989	800.00		
20	1						
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	
30	2						
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	
30	3						
7566	JONES	MANAGER	7839	2/04/1981	2975.00		
20	4						
7654	MARTIN	SALESMAN	7698	28/09/1981	1250.00	1400.00	
30	5						
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		
30	6						
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		
10	7						
7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		
20	8						
7839	KING	PRESIDENT		17/11/1981	5000.00		
10	9						
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
10							

```

7876 ADAMS    CLERK          7688 23/05/1987    1100.00      0.00      20
11
7900 JAMES    CLERK          7688 3/12/1981      950.00      0.00      30
12
7902 FORD     ANALYST        7566 3/12/1981      3000.00              20
13
7934 MILLER   CLERK          7782 23/01/1982      1300.00              10
14

```

//1. 查询出各个部门的平均工资和部门号

```
select deptno ,avg(sal) mysal from emp group by deptno;
```

//2. 把上面的查询看做是一张 子表

```
select a2.ename,a2.sal,a2.deptno,a1.mysal from emp a2, (select deptno ,avg
(sal) mysal from emp group by deptno) as a1 where a2.deptno=a1.deptno and
a2.sal>a1.mysal|
```

```
SQL> select * from (select a1.*,rownum rn from (select ename sal,dept from
emp order by dept) a1 where rownum<=10 order by rownum) where rn>3;
```

```

SAL      DEPT      RN
-----
SMITH      20          4
FORD       20          5
ADAMS      20          6
SCOTT      20          7
JONES      20          8
ALLEN      30          9
TURNER     30         10

```

用查询的结果创建新表:

这个命令是一种快捷的建表的方式.

```
SQL> create table myemp(
2 id,ename,sal) as select empno,ename,sal from emp;
```

Table created

```
SQL> desc myemp
```

```

Name      Type          Nullable Default Comments
-----
ID         NUMBER(4)      Y

```

```
ENAME VARCHAR2(7) Y
```

```
SAL    NUMBER(7,2) Y
```

```
SQL> select * from myemp;
```

ID	ENAME	SAL
7369	SMITH	800.00
7499	ALLEN	1600.00
7521	WARD	1250.00
7566	JONES	2975.00
7654	MARTIN	1250.00
7698	BLAKE	2850.00
7782	CLARK	2450.00
7788	SCOTT	3000.00
7839	KING	5000.00
7844	TURNER	1500.00
7876	ADAMS	1100.00
7900	JAMES	950.00
7902	FORD	3000.00
7934	MILLER	1300.00

```
14 rows selected
```

合并查询

有时候在应用中, 为了合并多个select语句的结果, 可以使用集合操作符号union, union all, intersect, minus

1) union

该操作符用于取得两个结果集的并集。当使用该操作符时, 会自动去掉结果集中重复行。

```
SQL> SELECT ENAME, SAL, JOB FROM EMP WHERE SAL > 2500 UNION
```

```
2 SELECT ENAME, SAL, JOB FROM EMP WHERE JOB = 'MANAGER';
```

ENAME	SAL	JOB
BLAKE	2850.00	MANAGER
CLARK	2450.00	MANAGER
FORD	3000.00	ANALYST
JONES	2975.00	MANAGER
KING	5000.00	PRESIDENT
SCOTT	3000.00	ANALYST

2) union all

该操作赋予union相似，但是它不会取消重复行，而且不会排序。

```
SQL> SELECT ENAME,SAL,JOB FROM EMP WHERE SAL>2500 UNION all SELECT  
ENAME,SAL,JOB FROM EMP WHERE JOB = 'MANAGER';
```

ENAME	SAL	JOB
JONES	2975.00	MANAGER
BLAKE	2850.00	MANAGER
SCOTT	3000.00	ANALYST
KING	5000.00	PRESIDENT
FORD	3000.00	ANALYST
JONES	2975.00	MANAGER
BLAKE	2850.00	MANAGER
CLARK	2450.00	MANAGER

8 rows selected

该操作符用于取出得两个结果集的并集。当使用该操作符时，会自动去掉结果集中重复行。

3) intersect

使用该操作符用于取得两个结果集的交集。

```
SQL> SELECT ENAME,SAL,JOB FROM EMP WHERE SAL>2500 intersect SELECT  
ENAME,SAL,JOB FROM EMP WHERE JOB = 'MANAGER';
```

ENAME	SAL	JOB
BLAKE	2850.00	MANAGER
JONES	2975.00	MANAGER

Minus

使用该操作符用于取得两个结果集的差集，它只会显示存在第一个集合中，而不存在第二个集合中的数据。

```
SQL> SELECT ENAME,SAL,JOB FROM EMP WHERE SAL>2500 minus SELECT  
ENAME,SAL,JOB FROM EMP WHERE JOB = 'MANAGER';
```

ENAME	SAL	JOB
FORD	3000.00	ANALYST
KING	5000.00	PRESIDENT
SCOTT	3000.00	ANALYST

创建数据库有两种方法:

- 1) 通过oracle提供的向导工具
- 2) 我们可以用手工步骤直接创建.

Java 操作oracle

- 1) java程序如何操作oracle
- 2) 如何在oracle中操作数据
- 3) Oracle事务处理
- 4) Sql函数的使用

Java连接oracle

介绍

前面我们一直在plsql中操作oracle, 那么如何在java程序中操作数据库呢?

下面我们举例说明, 写一个SHOWEMP. JAVA, 分页显示emp表的信息.

Java代码



复制代码

```
1. package com.sp;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.ResultSet;
6. import java.sql.Statement;
7.
8. //演示 如何使用 jdbc_odbc桥连接方式
9. public class TestOracle {
10.
11.     public static void main(String[] args) {
12.         try {
13.
14.             // 1.加载驱动
15.             Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
16.
17.             // 2.得到连接
18.             Connection ct = DriverManager.getConnection(
```

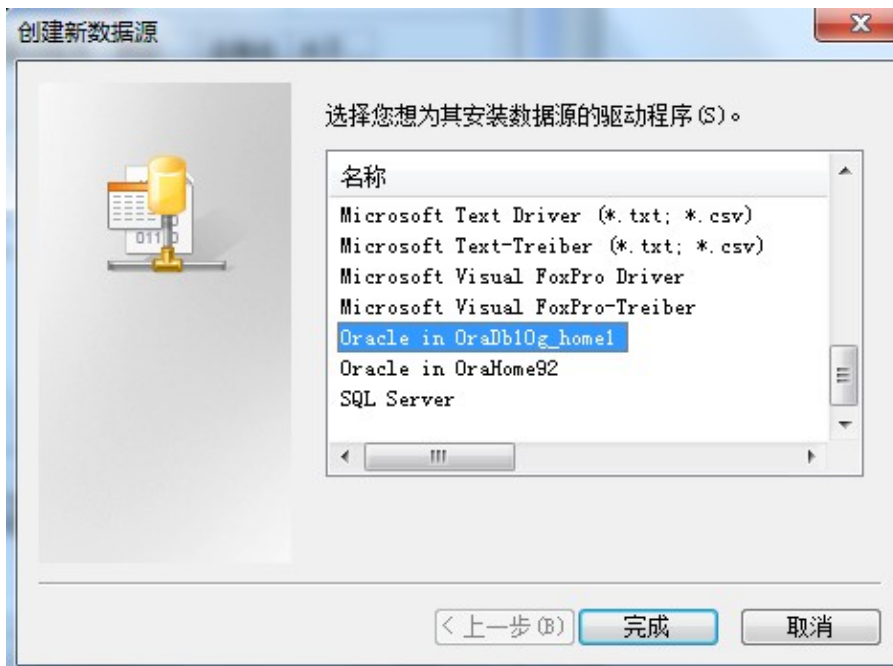
```

19.                                     "jdbc:odbc:testConnectOracle", "scott
",
20.
21. "tiger");
22.
23.                                     // 从下面开始, 和SQL Server一模一样
24.                                     Statement sm = ct.createStatement();
25.
                                     ResultSet rs = sm.executeQuery("select * from em
p");
26.                                     while (rs.next()) {
27.                                         //用户名
28.                                         System.out.println("用户
名: "+rs.getString(2));
29.                                         //默认是从1开始编号的
30.                                     }
31.                                     } catch (Exception e) {
32.                                         e.printStackTrace();
33.                                     }
34.                                     }
35. }

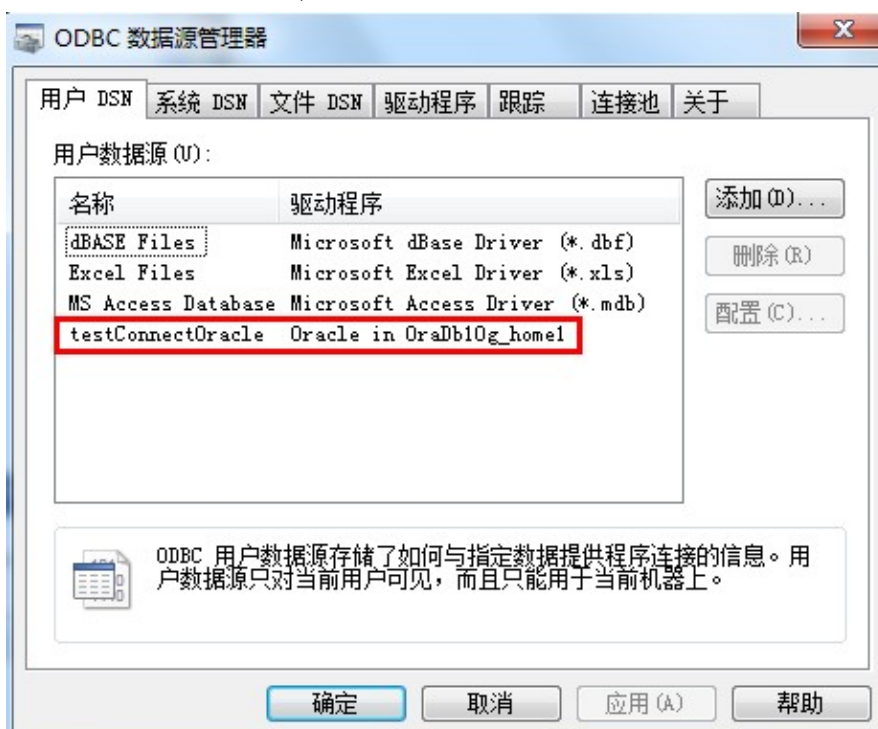
```

在得到连接那里, 要去配置数据源, 点击控制面板-->系统和安全-->管理工具-->数据源(ODBC), 打开后点添加, 如图:

可以看到, 有个Oracle in OraDb10g_home1的驱动, 它是Oracle安装完后自动加上去的。选中后, 点完成, 再填如下信息, 如图:



这样配好后基本就可以了，但为了安全起见，建议大家测试一下，点击 Test Connection按钮，测试通过后点ok, 然后数据源就生成了，如图：



然后把数据源名称写进jdbc.odbc:里。

这里要注意：jdbcodbc能不能远程连接呢？不能远程连接，也就是你这样写的话就意味着java程序和oracle数据库应该是在同一台机器上，因为这里没有指定IP地址，肯定默认就是本地。

如果要远程连，就用jdbc, jdbc是可以远程连的。

运行TestOracle.java, 控制台输出.....

可惜我没运行成功，说

```
java.sql.SQLException: No suitable driver found for jdbc.odbc:testConnectOracle
at java.sql.DriverManager.getConnection(Unknown Source)
at java.sql.DriverManager.getConnection(Unknown Source)
at com.sp.TestOracle.main(TestOracle.java:18)
```

不知道为什么。。。

接下来讲解用JDBC的方式连接Oracle

Java代码



复制代码

```
1. package com.sp;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.ResultSet;
6. import java.sql.Statement;
7.
8. //使用 jdbc连接oracle
9. public class TestOracle2 {
10.
11.     public static void main(String[] args) {
12.         try {
13.
14.             // 1. 加载驱动
15.             Class.forName("oracle.jdbc.driver.OracleDriver");
16.
17.             // 2. 得到连接
18.             Connection ct = DriverManager.getConnection
19.
20. ("jdbc:oracle:thin:@127.0.0.1:1521:orcl", "scott", "tiger");
21.
22.             // 从下面开始, 和SQL Server一模一样
23.             Statement sm = ct.createStatement();
24.
25.             ResultSet rs = sm.executeQuery("select * from emp");
```

```

25.                while (rs.next()) {
26.                    //用户名
27.                    System.out.println("用户
名:  "+rs.getString(2));
28.                    //默认是从1开始编号的
29.                }
30.            } catch (Exception e) {
31.                e.printStackTrace();
32.            }
33.        }
34.    }

```

记得要把驱动包引入，classes12.jar

运行，。。。。 再次可惜，我还是没运行成功，错误是：

java.sql.SQLException: Io 异常: The Network Adapter could not establish the

connection

at oracle.jdbc.dbaccess.DBError.throwSQLException(DBError.java:134)

at oracle.jdbc.dbaccess.DBError.throwSQLException(DBError.java:179)

at oracle.jdbc.dbaccess.DBError.throwSQLException(DBError.java:334)

at oracle.jdbc.driver.OracleConnection.<init>(OracleConnection.java:418)

at oracle.jdbc.driver.OracleDriver.getConnectionInstance

(OracleDriver.java:521)

at oracle.jdbc.driver.OracleDriver.connect(OracleDriver.java:325)

at java.sql.DriverManager.getConnection(Unknown Source)

at java.sql.DriverManager.getConnection(Unknown Source)

at com.sp.TestOracle2.main(TestOracle2.java:18)

我也不知道为什么。。。 幽怨了。。

接下来建个web project，来测试Oracle的分页，挺麻烦，不记录了。。

在oracle中操作数据 – 使用特定格式插入日期值

使用 to_date函数

请大家思考： 如何插入列带有日期的表，并按照年-月-日的格式插入？

```
insert into emp values (9998, 'xiaohong', 'MANAGER', 7782, to_date('1988-12- 12',
'yyyy-mm-dd'), 78.9, 55.33, 10);
```

注意:

```
insert into emp values (9998,'xiaohong','MANAGER', 7782,'12-12月-1988', 78.9,
55.33,10);
```

这句语句是可以成功运行的

使用子查询插入数据

介绍

当使用values子句时，一次只能插入一行数据，当使用子查询插入数据时，一条insert语句可以插入大量的数据。当处理行迁移或者装载外部表的数据到数据库时，可以使用子查询来插入数据。

把emp表中10号部门的数据导入到新表中

```
create table kkk(myId number(4), myName varchar2(50), myDept number(5));
insert into kkk (myId, myName, myDept) select empno, ename, deptno from emp where
deptno = 10;
```

介绍

使用update语句更新数据时，既可以使用表达式或者数值直接修改数据，也可以使用子查询修改数据。

问题：希望员工SCOTT的岗位、工资、补助与SMITH员工一样。

```
update emp set(job, sal, comm)=(select job, sal, comm from emp where ename='SMITH')
where ename='SCOTT';
```

使用to_date函数

思考：如何插入列带有日期的表，并按照年-月-日的格式插入？

```
SQL> INSERT INTO EMP VALUES
(8990,'DSJJ','CLERK',7780,TO_DATE('1986/2/22','YYYY/MM/DD'),888.00,NULL,10);
```

1 row inserted

```
SQL> select * from emp;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7369	SMITH	CLERK	7902	17/12/1989	800.00		20

7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7654	MARTIN	SALEMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		20
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10
8990	DSJJ	CLERK	7780	22/02/1986	888.00		10

使用子查询插入数据

介绍当使用values子句时, 一次能插入一行数据, 当使用子查询插入数据时, 一条insert语句可以大量的数据, 当处理行迁移或者装载外部表的数据到数据库时, 可以使用子查询来插入数据.

```
SQL> INSERT INTO USERS (ED,ENAME,SAL) SELECT DEPT,ENAME,SAL FROM EMP WHERE
DEPT=10;
```

```
4 rows inserted
```

```
SQL> SELECT * FROM EMP;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
-----	-----	-----	-----	-----	-----	-----	-----
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7654	MARTIN	SALEMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7788	SCOTT	ANALYST	7566	19/04/1987	3000.00		20
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30

7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10
8990	DSJJ	CLERK	7780	22/02/1986	888.00		10

15 rows selected

```
SQL> SELECT * FROM USERS;
```

ED	ENAME	SAL
----	-----	-----
10	CLARK	2450.00
10	KING	5000.00
10	MILLER	1300.00
10	DSJJ	888.00

使用子查询更新数据库
介绍

使用update语句更新数据时, 既可以使用表达式或者数值直接修改数据, 也可以使用子查询修改数据.

?希望员工scott的岗位, 工资, 补助与smith员工一样

```
SQL> update emp set (empno,mgr,sal)=(select empno,mgr,sal from emp where
ename='SMITH') WHERE ENAME='SCOTT';
```

1 row updated

ORACLE事务处理

什么是事务

事务用于保证数据的一致性, 它由一组相关的dml语句组成, 该组的dml语句要么全部成功, 要么全部失败.

如:网上转账就是典型的要用事务来处理, 用来保证数据的一致性.

事务和锁

当执行事务操作时 (dml 语句), oracle会被作用的表上加锁, 防止其它用户该表的表的结构. 这里对我们用户来讲是非常重要的.

提交事务

当执使用commit语句可以提交事务. 当执行了commit语句子句, 会确定事务的变化、结束事务、删除保存点、释放锁, 当使用commit语句结束事务子句, 其它会话将可以查看到事务变化

后的新数据

保存点就是为回退做的。保存点的个数没有限制

退出事务

在介绍回退事务前，我们首先介绍一下保存点(savepoint)的概念和作用，保存点是事务中的一点，用于取消部分事务，当结束事务时，会自动的删除该事务所定义的保存点. 当执行rollback时，通过指定保存点可以回退到指定的点这里我们作图说明(这个回退事务必须在commit之前，exit也是自动commit)。

事务的几个重要操作

1)设置保存点

Savepoint a

2)取消部分事务

Rollback to a

3)取消全部事务

Rollback

注意：这个回退事务，必须是没有commit前使用的；如果事务提交了，那么无论你刚才做了多少个保存点，都统统没有。

如果没有手动执行commit,而是exit了，那么会自动提交

Java程序中如何使用事务

在java操作数据库时，为了保证数据的一致性，比如转账操作(1)从一个账户减掉10¥, (2)在另一个账户上加入10¥，我们看看如何使用事务？

Java代码



复制代码

```
1. package com.sp;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.ResultSet;
6. import java.sql.Statement;
7.
8. public class TestTrans {
9.
10.     public static void main(String[] args) {
11.         try {
12.
13.             // 1.加载驱动
```

```

14.          Class.forName("oracle.jdbc.driver.OracleDriver");

15.

16.          // 2. 得到连接
17.          Connection ct = DriverManager.getConnection(

18.

                  "jdbc:oracle:thin:@127.0.0.1:1521:orcl

", "scott", "tiger");
19.
20.          Statement sm = ct.createStatement();
21.
22.          // 从scott的sal中减去100
23.          sm.executeUpdate("update emp set sal=sal-
100 where ename='SCOTT'");
24.
25.          int i = 7 / 0;
26.
27.          // 给smith的sal加上100
28.

          sm.executeUpdate("update emp set sal=sal+100 where
ename='SMITH'");
29.
30.          // 关闭打开的资源
31.          sm.close();
32.          ct.close();
33.      } catch (Exception e) {
34.          e.printStackTrace();
35.      }
36.
37.    }
38.
39. }

```

运行，会出现异常，查看数据库，SCOTT的sal减了100，但是SMITH的sal却不变，很可怕。。。我们怎样才能保证，这两个操作要么同时成功，要么同时失败呢？

Java代码



复制代码

```
1. package com.sp;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.SQLException;
6. import java.sql.Statement;
7.
8. public class TestTrans {
9.
10.     public static void main(String[] args) {
11.         Connection ct = null;
12.         try {
13.             // 1.加载驱动
14.             Class.forName("oracle.jdbc.driver.OracleDriver");
15.
16.             // 2.得到连接
17.             ct = DriverManager.getConnection(
18.
19.                 "jdbc:oracle:thin:@127.0.0.1:1521:orcl
20. ", "scott", "tiger");
21.
22.             // 加入事务处理
23.             ct.setAutoCommit(false); // 设置不能默认提交
24.
25.             Statement sm = ct.createStatement();
26.
27.             // 从scott的sal中减去100
28.             sm.executeUpdate("update emp set sal=sal-
29. 100 where ename=' SCOTT' ");
30.
31.             // 给smith的sal加上100
32.
33.             sm.executeUpdate("update emp set sal=sal+100 where
```

```

    ename=' SMITH' ");
32.
33.          // 提交事务
34.          ct.commit();
35.
36.          // 关闭打开的资源
37.          sm.close();
38.          ct.close();
39.      } catch (Exception e) {
40.          // 如果发生异常，就回滚
41.          try {
42.              ct.rollback();
43.          } catch (SQLException e1) {
44.              e1.printStackTrace();
45.          }
46.          e.printStackTrace();
47.      }
48.
49.  }
50.
51. }

```

再运行一下，会出现异常，查看数据库，数据没变化。。

只读事务

只读事务是指只允许执行查询的操作，而不允许执行的任何其它dml操作的事务，使用只读事务可以确保用户只能取得某时间点的数据。假定机票代售点每天18点开始统计今天的销售情况，这时可以使用只读事务，在设置只读事务后，尽管其它回话可能会提交新的事务，但是只读事务将不会取得最新数据的变化，从而可以保证取得特定时间点的数据信息。

设置只读事务

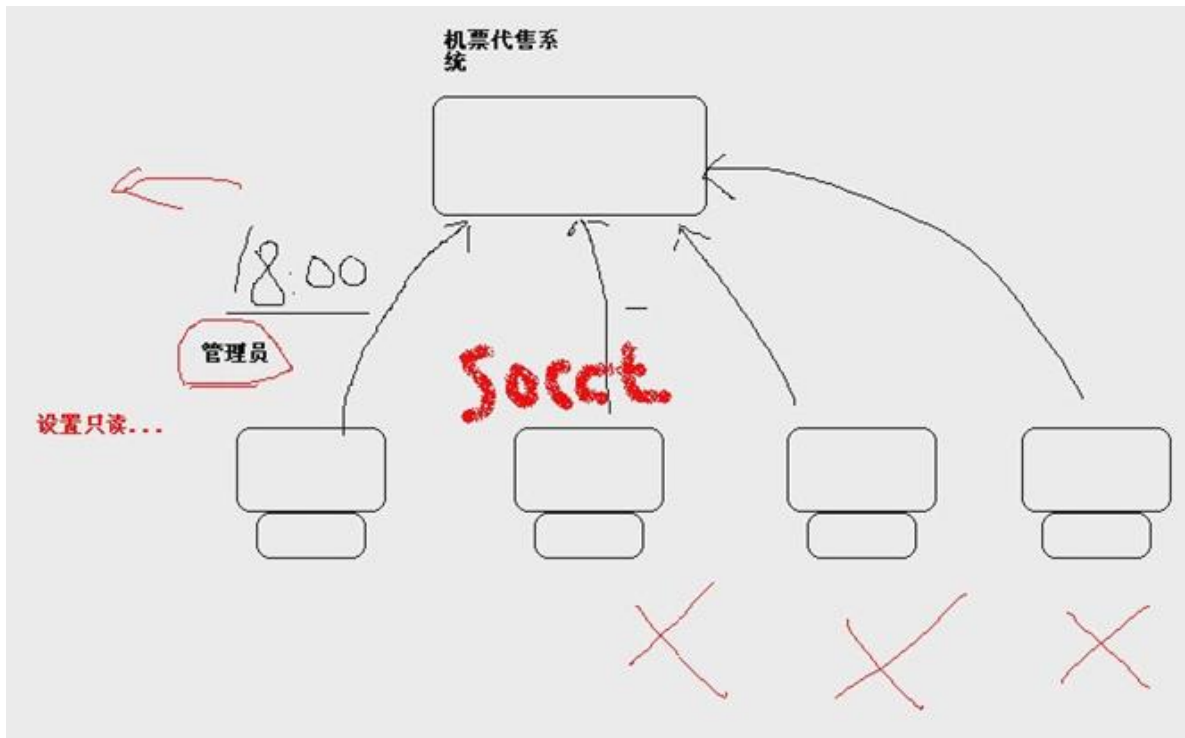
Set transaction read only

```

SQL> set transaction read only;

Transaction set.

```



Sql函数的使用-字符函数

介绍

字符函数是oracle中最常用的函数

Lower(char):将字符串转换为小写的格式

Upper(char):将字符串转化为大写的格式

Length(char):返回字符串的长度

Substr(char,m,n):取字符串的子串

?将所有员工的名字按小写的方式显示

```
SQL> select lower(ename),sal from emp;
```

LOWER(ENAME)	SAL
smith	800.00
allen	1600.00
ward	1250.00
jones	2975.00
martin	1250.00
blake	2850.00
clark	2450.00
scott	800.00
king	5000.00
turner	1500.00
adams	1100.00
james	950.00

ford	3000.00
miller	1300.00

?将所有员工的姓名按大写的方式显示

```
SQL> select upper(ename),sal from emp;
```

UPPER(ENAME)	SAL
-----	-----
SMITH	800.00
ALLEN	1600.00
WARD	1250.00
JONES	2975.00
MARTIN	1250.00
BLAKE	2850.00
CLARK	2450.00
SCOTT	800.00
KING	5000.00
TURNER	1500.00
ADAMS	1100.00
JAMES	950.00
FORD	3000.00
MILLER	1300.00

?显示正好为5个字符的员工姓名

```
SQL> select * from emp where length(ename)=5;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
-----	-----	-----	-----	-----	-----	-----	-----
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7369	SCOTT	ANALYST	7902	19/04/1987	800.00		20
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30

?显示所有员工姓名的前三个字符(注意下面括号里面的1表示从第几个开始,3 表示三个字符,不是第三个字符)

```
SQL> select substr(ename,1,3) from emp;
```

```
SUBSTR(ENAME,1,3)
```

```
-----
```

```
SMI
```

```
ALL
```

```
WAR
```

```
JON
```

```
MAR
```

```
BLA
```

```
CLA
```

```
SCO
```

```
KIN
```

```
TUR
```

```
ADA
```

```
JAM
```

```
FOR
```

```
MIL
```

```
14 rows selected
```

?以首写字母大写的方式显示所有员工的姓名

分析,清晰思路

1. 完成首字母大写

```
select upper(substr(ename,1,1)) from emp;
```

2. 完成后面字母小写

```
select lower(substr(ename,2,length(ename)-1)) from emp;
```

3. 合并

```
select upper(substr(ename,1,1))||lower(substr(ename,2,length(ename)-1)) from emp;
```

```
SQL> select upper(substr(ename,1,1)) from emp;
```

```
UPPER(SUBSTR(ENAME,1,1))
```

```
-----
```

```
S
```

```
A
```

```
W
```

J
M
B
C
S
K
T
A
J
F
M

14 rows selected

```
SQL> select lower(substr(ename,2,length(ename))) from emp;
```

```
LOWER(SUBSTR(ENAME,2,LENGTH(EN
```

```
-----
```

mith
llen
ard
ones
artin
lake
lark
cott
ing
urner
dams
ames
ord
iller

14 rows selected

```
SQL> select upper(substr(ename,1,1)) ||  
lower(substr(ename,2,length(ename))) from emp;
```

```
UPPER(SUBSTR(ENAME,1,1)) || LOWE
```

```
-----
```

Smith

Allen
Ward
Jones
Martin
Blake
Clark
Scott
King
Turner
Adams
James
Ford
Miller

14 rows selected

?以首写字母小写的方式显示所有员工的姓名.

```
SQL> select lower(substr(ename,1,1)) ||  
upper(substr(ename,2,length(ename))) from emp;
```

```
LOWER(SUBSTR(ENAME,1,1)) || UPPE
```

```
-----
```

sMITH
aLLEN
wARD
jONES
mARTIN
bLAKE
cLARK
sCOTT
kING
tURNER
aDAMS
jAMES
fORD
mILLER

14 rows selected

Replace函数

介绍

字符函数是oracle中最常用的函数,

Replace(char1, search_string, replace_string)

(这里的char1是需要替换的字段, search_string需要替换的字符串, replace是替换成那个字符串)

Insert(char1, char2, [, n[, m]])取子串在字符串的位置

?显示所有员工的姓名, 用' 我是A' 替换所有' A' .

```
SQL> select replace(ename, 'A', '我是A') from emp;
```

```
REPLACE(ENAME, 'A', '我是A')
```

```
-----  
SMITH
```

```
我是ALLEN
```

```
W我是ARD
```

```
JONES
```

```
M我是ARTIN
```

```
BL我是AKE
```

```
CL我是ARK
```

```
SCOTT
```

```
KING
```

```
TURNER
```

```
我是AD我是AMS
```

```
J我是AMES
```

```
FORD
```

```
MILLER
```

```
14 rows selected
```

数学函数

介绍

数学函数的输入参数和返回值的数据类型都是数字类型的, 数学函数包括

cos, cosh, exp, ln, log, sin, sinh, sqrt, tan, tanh, acos, asin, atan, round,

Round(n, [m])

Trunc(n, [m])

Mod(m, n)

Floor(n)

Ceil(n)

对数字的处理, 在财务系统或银行系统中用的最多, 不同的处理方法, 对财务报表有不同的结果.

介绍

Round(char1, [m]) 该函数用于执行四舍五入, 如果省掉m, 则四舍五入到整数; 如果m是正数, 则四舍五入到小数点的m位后. 如果是负数, 则四舍五入到小数点的m位前

```
SQL> SELECT * FROM EMP;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7654	MARTIN	SALESMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7369	SCOTT	ANALYST	7902	19/04/1987	800.00		20
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10
7935	JLJK	CLERK	7688	17/10/1981	1000.45	600.68	20

15 rows selected

```
SQL> SELECT ROUND(SAL), SAL FROM EMP WHERE ENAME='JLJK';
```

ROUND(SAL)	SAL
1000	1000.45

```
SQL> SELECT ROUND(SAL), SAL, ROUND(COMM), COMM FROM EMP WHERE ENAME='JLJK';
```

ROUND(SAL)	SAL	ROUND(COMM)	COMM
1000	1000.45	601	600.68

```
SQL> SELECT ROUND(SAL,1),SAL,ROUND(COMM,1),COMM FROM EMP WHERE
ENAME='JLJK';
```

ROUND(SAL,1)	SAL	ROUND(COMM,1)	COMM
1000.5	1000.45	600.7	600.68

```
SQL> SELECT ROUND(SAL,-1),SAL,ROUND(COMM,-1),COMM FROM EMP WHERE
ENAME='JLJK';
```

ROUND(SAL,-1)	SAL	ROUND(COMM,-1)	COMM
1050	1045.45	660	655.68

Trunc(char1,[m])该函数用于截取数字. 如果省掉m, 就截去掉小数部分, 如果m是正数就截取到小数点的m位后, 如果m是负数, 则截取到小数点的前m位

```
SQL> SELECT trunc(SAL),SAL,trunc(COMM),COMM FROM EMP WHERE ENAME='JLJK';
```

TRUNC(SAL)	SAL	TRUNC(COMM)	COMM
1000	1000.45	600	600.68

```
SQL> SELECT trunc(SAL,1),SAL,trunc(COMM,1),COMM FROM EMP WHERE
ENAME='JLJK';
```

TRUNC(SAL,1)	SAL	TRUNC(COMM,1)	COMM
1000.4	1000.45	600.6	600.68

Mod(m,n) 求模

在做oracle测试, 可以使用dual表,

```
SQL> select mod(10,2) from dual;
```

MOD(10,2)
0

```
SQL> select mod(10,3) from dual;
```

MOD(10,3)
1

```
SQL> select mod(11,3) from dual;
```

```
MOD(11,3)
-----
2
```

Floor(CHAR1) 返回小于或是等于n的最大整数

```
SQL> SELECT floor(SAL), SAL, FLOOR(COMM), COMM FROM EMP WHERE ENAME='JLJC';
```

```
FLOOR(SAL)      SAL FLOOR(COMM)      COMM
-----
1045      1045.45      655      655.68
```

Ceil(CHAR1) 返回大于或是等于n的最小整数

```
SQL> SELECT CEIL(SAL), SAL, CEIL(COMM), COMM FROM EMP WHERE ENAME='JLJC';
```

```
CEIL(SAL)      SAL CEIL(COMM)      COMM
-----
1046      1045.45      656      655.68
```

案例数据: 235.56, 45.94

?显示在一个月为30天的情况所有员工的日薪金, 忽略余数.

方法一

```
SQL> SELECT ENAME, TRUNC(SAL/30), SAL/30, TRUNC(COMM/30), COMM/30 FROM EMP;
```

```
ENAME      TRUNC(SAL/30)      SAL/30 TRUNC(COMM/30)      COMM/30
-----
SMITH      26 26.66666666
ALLEN      53 53.33333333      10      10
WARD      41 41.66666666      16 16.66666666
JONES      99 99.16666666
MARTIN     41 41.66666666      46 46.66666666
BLAKE      95      95
CLARK      81 81.66666666
SCOTT      26 26.66666666
KING      166 166.66666666
TURNER     50      50      0      0
ADAMS      36 36.66666666      0      0
JAMES      31 31.66666666      0      0
FORD      100      100
MILLER     43 43.33333333
```

JLJK	33	33.3483333	20	20.0226666
JLJC	34	34.8483333	21	21.856

16 rows selected

方法二

```
SQL> SELECT ENAME, FLOOR(SAL/30), SAL/30, FLOOR(COMM/30), COMM/30 FROM EMP;
```

ENAME	FLOOR(SAL/30)	SAL/30	FLOOR(COMM/30)	COMM/30
SMITH	26	26.6666666		
ALLEN	53	53.3333333	10	10
WARD	41	41.6666666	16	16.6666666
JONES	99	99.1666666		
MARTIN	41	41.6666666	46	46.6666666
BLAKE	95	95		
CLARK	81	81.6666666		
SCOTT	26	26.6666666		
KING	166	166.666666		
TURNER	50	50	0	0
ADAMS	36	36.6666666	0	0
JAMES	31	31.6666666	0	0
FORD	100	100		
MILLER	43	43.3333333		
JLJK	33	33.3483333	20	20.0226666
JLJC	34	34.8483333	21	21.856

16 rows selected

其它的数学函数，

Abs(n) 返回数字n的绝对值（）此处的n可以为列变量。

```
Select abs(-13) from dual;
```

Acos(n):返回数字的反余旋值

Asin(n):返回数字的反正旋值

Atan(n):返回数字的反正切

Cos(n):返回余旋值

Exp(n):返回e的n次方幂

```
SQL> select exp(3) from dual;
```

EXP(3)

20.0855369

Log(m, n) 返回对数值

```
SQL> select log(100,10) from dual;
```

LOG(100,10)

0.5

```
SQL> select log(10,100) from dual;
```

LOG(10,100)

2

Power(m, n): 返回m的n次幂

```
SQL> select power(10,3) from dual;
```

POWER(10,3)

1000

```
SQL> select power(10,2) from dual;
```

POWER(10,2)

100

日期函数

介绍

日期函数用于处理date类型的数据.

默认情况下日期格式是dd-mon-yy即12-7月-78

(1) sysdate: 该函数返回系统时间

```
SQL> select sysdate from dual;
```

SYSDATE

11/08/2012

(2) add_months(日期字段, n)

```
SQL> select * from emp where sysdate>add_months(hiredate,320);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7654	MARTIN	SALESMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10
7935	JLJK	CLERK	7688	17/10/1981	1000.45	600.68	20
7935	JLJC	CLERK	7688	17/10/1981	1045.45	655.68	20

13 rows selected

```
SQL> select * from emp ;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7654	MARTIN	SALESMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7369	SCOTT	ANALYST	7902	19/04/1987	800.00		20
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10
7935	JLJK	CLERK	7688	17/10/1981	1000.45	600.68	20

7935	JLJC	CLERK	7688	17/10/1981	1045.45	655.68	20
------	------	-------	------	------------	---------	--------	----

16 rows selected

(3) last_day(d):返回指定日期所在月份的最后一天

SQL> select hiredate, last_day(hiredate) from emp ;

HIREDATE	LAST_DAY(HIREDATE)
-----	-----
17/12/1989	31/12/1989
20/02/1981	28/02/1981
22/02/1981	28/02/1981
2/04/1981	30/04/1981
28/09/1981	30/09/1981
1/05/1981	31/05/1981
9/06/1981	30/06/1981
19/04/1987	30/04/1987
17/11/1981	30/11/1981
8/09/1981	30/09/1981
23/05/1987	31/05/1987
3/12/1981	31/12/1981
3/12/1981	31/12/1981
23/01/1982	31/01/1982
17/10/1981	31/10/1981
17/10/1981	31/10/1981

16 rows selected

?查找已经入职8个月多的员工

?显示满10年服务年限的员工的姓名和受雇日期

SQL> select * from emp where sysdate>=add_months(hiredate,10);

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
-----	-----	-----	-----	-----	-----	-----	-----
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20

7654	MARTIN	SALEMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30
7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7369	SCOTT	ANALYST	7902	19/04/1987	800.00		20
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
7876	ADAMS	CLERK	7688	23/05/1987	1100.00	0.00	20
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7934	MILLER	CLERK	7782	23/01/1982	1300.00		10
7935	JLJK	CLERK	7688	17/10/1981	1000.45	600.68	20
7935	JLJC	CLERK	7688	17/10/1981	1045.45	655.68	20

16 rows selected

?对于每个员工，显示其加入公司的天数.

```
SQL> select trunc(sysdate-hiredate),ename from emp;
```

TRUNC (SYSDATE-HIREDATE)	ENAME
-----	-----
8273	SMITH
11495	ALLEN
11493	WARD
11454	JONES
11275	MARTIN
11425	BLAKE
11386	CLARK
9246	SCOTT
11225	KING
11295	TURNER
9212	ADAMS
11209	JAMES
11209	FORD
11158	MILLER
11256	JLJK
11256	JLJC

16 rows selected

```
SQL> select sysdate-hiredate,ename from emp;
```

```

SYSDATE-HIREDATE ENAME
-----
8273.93012731481 SMITH
11495.9301273148 ALLEN
11493.9301273148 WARD
11454.9301273148 JONES
11275.9301273148 MARTIN
11425.9301273148 BLAKE
11386.9301273148 CLARK
9246.93012731481 SCOTT
11225.9301273148 KING
11295.9301273148 TURNER
9212.93012731481 ADAMS
11209.9301273148 JAMES
11209.9301273148 FORD
11158.9301273148 MILLER
11256.9301273148 JLJK
11256.9301273148 JLJC

```

16 rows selected

?找出各月倒数第三天受雇的所有员工.

```
SQL> select * from emp where last_day(hiredate)-2=hiredate;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7654	MARTIN	SALEMAN	7698	28/09/1981	1250.00	1400.00	30

转换函数

介绍

转换函数用于将数据类型从一种转为另一种，在某些情况下，oracle server允许值的数据类型和实际的情况不一样，这是oracle server会隐含的转化数据类型

比如：

```
Create table t1(id int);
```

```
Insert into t1 values( '1' ); ——>这样oracle会自动的将'1' ò1
```

```
Create table t2(id varchar2(20));
```

```
Insert into t2 values(1); ò这样oracle就会自动的将1ò'1' ;
```

我们要说的是尽管oracle可以进行隐含的数据类型转换，但是它并不能所有的情况，为了提高程序的可靠性，我们应该使用转换函数进行转换

To_char


你可以使用select ename,hiredate,sal from emp where dept=10;显示信息，可是，在某些情况下，这个并不能满足你的需求。


?日期是否可以显示 时/分/秒

?薪水是否可以显示指定的货币符号

?日期是否可以显示 时/分/秒

?薪水是否可以显示指定的货币符号

 yy: 两位数字的年份 2004-->04
yyyy: 四位数字的年份 2004年
mm : 两位数字的月份 8月-->08
dd: 2位数字的天 30号-->30
hh24: 8点 --> 20
hh12: 8点 --> 08
mi、ss --> 显示分钟\秒



9: 显示数字,并忽略前面0
0: 显示数字,如位数不足,则用0补齐
.: 在指定位置显示小数点
,: 在指定位置显示逗号
\$: 在数字前加美元
L: 在数字前加本地货币符号
C: 在数字前加国际货币符号
G: 在指定位置显示组分隔符.
D: 在指定位置显示小数点符号(.)

```
SQL> select to_char(hiredate,'yyyy-mm-dd hh24:mi:ss') from emp;
```

```
TO_CHAR(HIREDATE,'YYYY-MM-DDHH
```

```
-----  
1989-12-17 00:00:00  
1981-02-20 00:00:00  
1981-02-22 00:00:00  
1981-04-02 00:00:00  
1981-09-28 00:00:00  
1981-05-01 00:00:00  
1981-06-09 00:00:00  
1987-04-19 00:00:00  
1981-11-17 00:00:00  
1981-09-08 00:00:00  
1987-05-23 00:00:00  
1981-12-03 00:00:00  
1981-12-03 00:00:00  
1982-01-23 00:00:00  
1981-10-17 00:00:00  
1981-10-17 00:00:00  
2012-08-11 22:50:40
```

17 rows selected

```
SQL> select to_char(hiredate,'yyyy-mm-dd hh24:mi:ss'),
to_char(sal,'l9999.99') from emp;
```

TO_CHAR(HIREDATE,'YYYY-MM-DDHH TO_CHAR(SAL,'L9999.99') (本地的货币符号)

TO_CHAR(HIREDATE,'YYYY-MM-DDHH	TO_CHAR(SAL,'L9999.99')
1989-12-17 00:00:00	\$800.00
1981-02-20 00:00:00	\$1600.00
1981-02-22 00:00:00	\$1250.00
1981-04-02 00:00:00	\$2975.00
1981-09-28 00:00:00	\$1250.00
1981-05-01 00:00:00	\$2850.00
1981-06-09 00:00:00	\$2450.00
1987-04-19 00:00:00	\$800.00
1981-11-17 00:00:00	\$5000.00
1981-09-08 00:00:00	\$1500.00
1987-05-23 00:00:00	\$1100.00
1981-12-03 00:00:00	\$950.00
1981-12-03 00:00:00	\$3000.00
1982-01-23 00:00:00	\$1300.00
1981-10-17 00:00:00	\$1000.45
1981-10-17 00:00:00	\$1045.45
2012-08-11 22:50:40	\$8786.03

17 rows selected

To_char

?显示1981年入职的所有员工

```
SQL> select * from emp where to_char(hiredate,'yyyy')=1981;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7499	ALLEN	SALESMAN	7698	20/02/1981	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/1981	1250.00	500.00	30
7566	JONES	MANAGER	7839	2/04/1981	2975.00		20
7654	MARTIN	SALEMAN	7698	28/09/1981	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	1/05/1981	2850.00		30

7782	CLARK	MANAGER	7839	9/06/1981	2450.00		10
7839	KING	PRESIDENT		17/11/1981	5000.00		10
7844	TURNER	SALESMAN	7698	8/09/1981	1500.00	0.00	30
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20
7935	JLJK	CLERK	7688	17/10/1981	1000.45	600.68	20
7935	JLJC	CLERK	7688	17/10/1981	1045.45	655.68	20

12 rows selected

?显示所有12月份入职的员工

```
SQL> select * from emp where to_char(hiredate,'mm')=12;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7369	SMITH	CLERK	7902	17/12/1989	800.00		20
7900	JAMES	CLERK	7688	3/12/1981	950.00	0.00	30
7902	FORD	ANALYST	7566	3/12/1981	3000.00		20

To_date

函数to_date用于将字符串转换成date类型的数据. (前面已经做过)

Sys_context

- 1) terminal:当前会话客户所对应的终端的标识符
- 2) language:语言
- 3) db_name:当前数据库名称
- 4) nls_date_format:当前会话客户所对应的日期格式
- 5) session_user:当前会话客户所对应的数据库用户名
- 6) current_schema:当前会话客户所对应的默认方案名
- 7) host:返回函数库所在主机的名称

通过该函数，可以查询一些重要的信息，比如你怎么使用那个数据库？

```
Select sys_context( 'userenv' , ' db_name' ) from dual;
```

```
SQL> select sys_context('userenv','language') from dual;
```

```
SYS_CONTEXT('USERENV','LANGUAG
```

```
AMERICAN_AMERICA.WE8MSWIN1252
```

```
SQL> select sys_context('userenv','db_name') from dual;
```

```
SYS_CONTEXT('USERENV','DB_NAME
```

```
orcl
```

数据库管理员

数据库(表)的逻辑备份和恢复

数据字典和动态性能视图

管理表空间和数据文件

介绍

每个oracle数据库应该至少有一名数据库管理员(dba)，对于一个小的数据库，一个dba就够了，但是对于一个大的数据库可能需要多个dba分担不同的管理职责. 那么一个数据库管理员主要工作是什么呢？

职责：

- 1) 安装和升级oracle数据库
- 2) 建库，表空间，表，视图，索引...
- 3) 制定并实施备份与恢复计划
- 4) 数据库权限管理，**调优，故障排除**
- 5) 对于高级dba，要求能**参与项目开发，会编写sql语句、存储过程、触发器、规则、约束、包**

(dba也是编程高手)

管理数据库的用户主要是sys和system(sys可以比喻董事长, system比喻为总经理)

在前面我们已经提到这两个用户，区别主要是：

- (1) 最重要的区别，存储的数据的重要性不同

Sys:所有oracle的数据字典的基本表和视图都存放在sys用户中，这些基表和视图对于oracle的运行是至关重要的，由数据库自己维护，任何用户都不能手动更改。Sys用户拥有dba，sysdba，sysoper角色或权限，是oracle权限最高的用户。

System:用于存放次一级的内部数据，如oracle的一些特性或工具的管理信息。System用户拥有dba，sysdba角色或系统权限。

- (2) 其次的区别，权限的不同。

Sys用户必须以as sysdba或as sysoper形式登录。不能以normal方式登录数据库

System如果正在登录，它其实就是一个普通的dba用户，但是如果以as sysdba登录，其结果实际上它是作为sys用户登录的，从登录信息里面我们可以看出来。

■ sysdba和sysoper权限区别图

sysdba和sysoper具体的权限可以看一下表：

系统权限	sysdba	sysoper
区别	Startup(启动数据库)	startup
	Shutdown(关闭数据库)	shutdown
	alter database open/mount/backup	alter database open/mount/backup
	改变字符集	none
	create database(创建数据库)	None不能创建数据库
	drop database(删除数据库)	none
	create spfile	create spfile
	alter database archivelog(归档日志)	alter database archivelog
	alter database recover(恢复数据库)	只能完全恢复，不能执行不完全恢复
	拥有 restricted session(会话限制) 权限	拥有 restricted session 权限
	可以让用户作为sys用户连接	可以进行一些基本的操作，但不能查看用户数据
	登录之后用户是sys	登录之后用户是public

管理初始化参数

初始化参数用于设置实例或是数据库的特征. oracle 9i提供了200多个初始化参数，并且每个初始化参数都有默认值.

显示初始化参数

(1) Show parameter 命令

如何修改参数

需要说明的如果你希望修改这些初始化的参数, 可以到文件

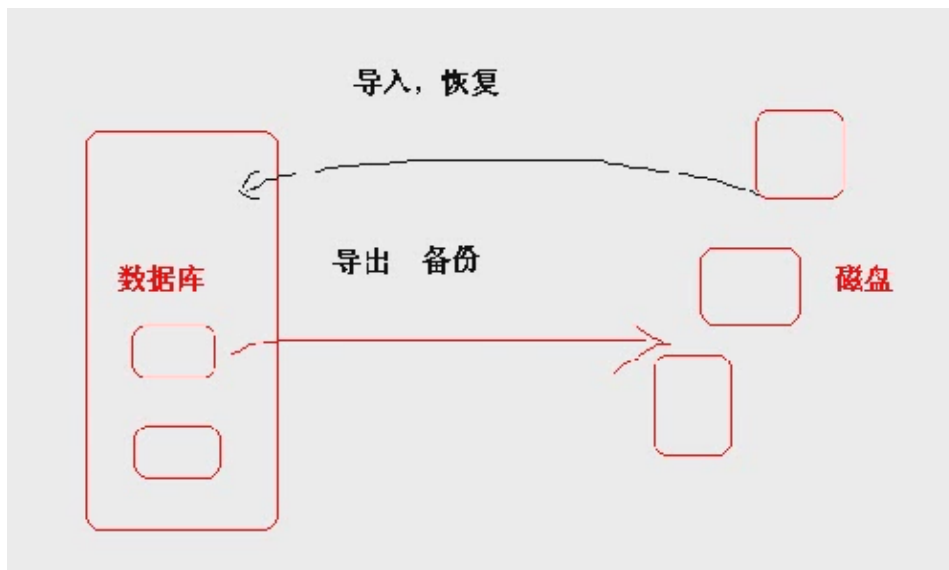
D:\ORACLE\ADMIN\MYORAL\PFIL\INIT.ORA文件中去修改比如修改实例的名字

数据库(表)的逻辑备份与恢复

介绍

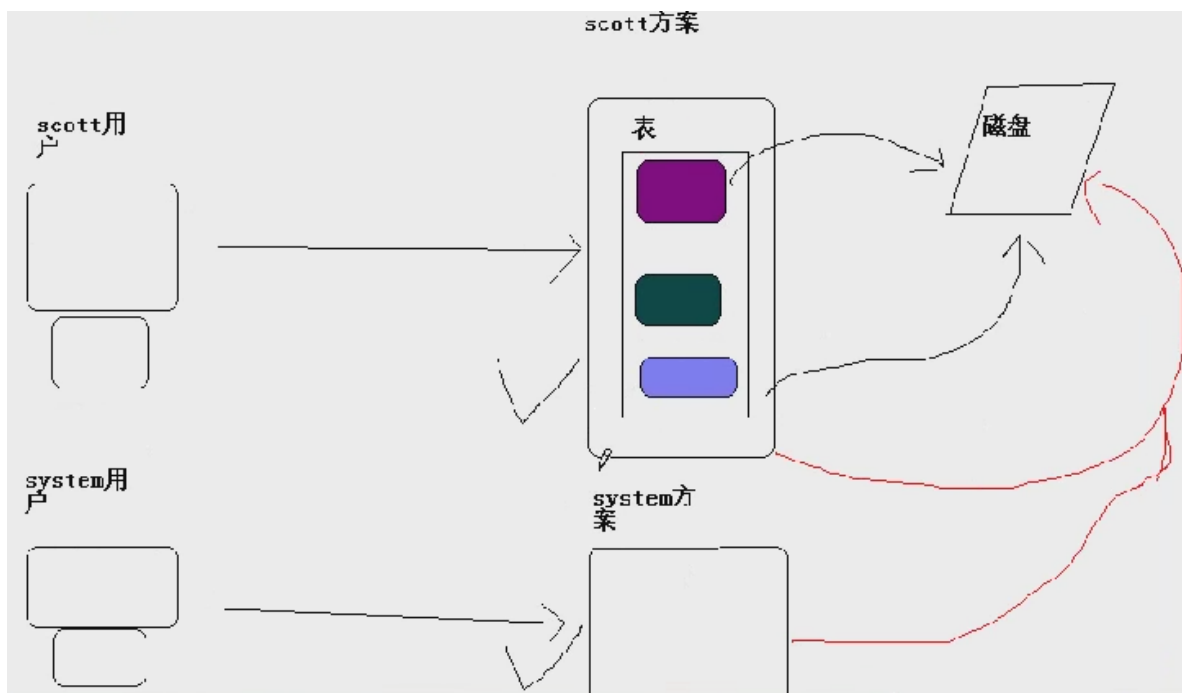
逻辑备份是指使用工具export将数据对象的结构和数据导出到文件的过程，逻辑恢复是指当数据库对象被误操作而损坏后使用工具import利用备份的文件把数据对象导入到数据库的过程. 物

理备份即可在数据库open的状态下进行也可在关闭数据库后进行,但是逻辑备份和恢复只能在open的状态下进行.



导出

导出具体的分为:导出表, 导出方案, 导出数据库三种方式.



导出使用exp命令来完成的, 该命令常用的选项有:

userid:用于指定执行导出操作的用户名, 口令, 连接字符串

tables:用于指定执行导出操作的表

owner:用于指定执行导出操作的方案

full=y:用于指定执行导出操作的数据库

inctype:用于指定执行导出操作的增量类型

rows:用于指定执行导出操作是否要导出表中的数据

file:用于指定导出的文件名

(特别说明:在导出和导入的时候, 要到oracle目录的bin目录下去导。)

导出表

(1) 导出自己的表

```
Exp userid=scott/tiger(密码)@myora(实例)1 tables=(emp) file=d:\e1.dmp
D:\app\JoePotter\product\11.2.0\dbhome_1\BIN>exp userid=joe/manager@ORCL tables=
(emp) file=d:\a1.dmp

Export: Release 11.2.0.1.0 - Production on Sun Aug 12 17:02:50 2012

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - Produc
tion
With the Partitioning, OLAP, Data Mining and Real Application Testing options
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set

About to export specified tables via Conventional Path ...
. . exporting table EMP 17 rows exported
Export terminated successfully without warnings.
D:\app\JoePotter\product\11.2.0\dbhome_1\BIN>exp userid=joe/manager@ORCL tables=
(emp,dept) file=d:\a1.dmp

Export: Release 11.2.0.1.0 - Production on Sun Aug 12 17:04:55 2012

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - Produc
tion
With the Partitioning, OLAP, Data Mining and Real Application Testing options
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set

About to export specified tables via Conventional Path ...
. . exporting table EMP 17 rows exported
. . exporting table DEPT 4 rows exported
Export terminated successfully without warnings.
```

(2) 导出其它方案的表

如果用户要导出其它方案的表，则需要dba的权限或是exp_full_database的权限，比如system就可以导出scott的表

```
D:\app\JoePotter\product\11.2.0\dbhome_1\BIN>exp userid=system/manager@ORCL tabl
es={joe.emp,joe.dept} file=d:\a2.dmp

Export: Release 11.2.0.1.0 - Production on Sun Aug 12 17:08:04 2012

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - Produc
tion
With the Partitioning, OLAP, Data Mining and Real Application Testing options
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set

About to export specified tables via Conventional Path ...
Current user changed to JOE
. . exporting table EMP 17 rows exported
. . exporting table DEPT 4 rows exported
Export terminated successfully without warnings.
```

(3) 导出表的结构

```
Exp userid=scott/tiger@accp tables=(emp) file=d:\e3.dmp rows=n
```

```
D:\program\myoracle\BIN>exp userid=scott/m123@myora1 tables=(emp,dept) file=d:\emp3.dmp rows=n

Export: Release 9.0.1.1.1 - Production on 星期四 10月 15 13:41:49 2009

(c) Copyright 2001 Oracle Corporation. All rights reserved.

连接到: Oracle9i Enterprise Edition Release 9.0.1.1.1 - Production
With the Partitioning option
JServer Release 9.0.1.1.1 - Production
已导出 ZHS16GBK 字符集和 AL16UTF16 NCHAR 字符集
注: 将不会导出表数据 (行)

即将导出指定的表通过常规路径 ...
. . 正在导出表 EMP
. . 正在导出表 DEPT
在没有警告的情况下成功终止导出。
```

(4) 使用直接导出方式

```
D:\program\myoracle\BIN>exp userid=scott/m123@myora1 tables=(emp,dept) file=d:\emp4.dmp direct=y

Export: Release 9.0.1.1.1 - Production on 星期四 10月 15 13:43:36 2009

(c) Copyright 2001 Oracle Corporation. All rights reserved.

连接到: Oracle9i Enterprise Edition Release 9.0.1.1.1 - Production
With the Partitioning option
JServer Release 9.0.1.1.1 - Production
已导出 ZHS16GBK 字符集和 AL16UTF16 NCHAR 字符集

即将导出指定的表通过直接路径 ...
. . 正在导出表 EMP 19 行被导出
. . 正在导出表 DEPT 4 行被导出
在没有警告的情况下成功终止导出。
```

```
Exp userid=scott/tiger@accp table=(emp) file=d:e3.dmp direct=y
```

这种方式比默认的常规方式速度要快，当数据量大时，可以考虑使用这样的方法
这时需要数据库的字符集要客户端字符集完全一致，否则会报错...

导出数据库

导出数据库是指利用export导出所有数据库中的对象及数据, 要求该用户具有dba的权限或是exp_full_database权限

```
Exp userid=system/manger@myor full=y inctype=complete file=D:\A.dmp(数据量不较大, 需要较长的时间)
```

导入表

(1) 导入自己的表

```
Imp userid=scott/manager@myor tables=(emp) file=d:\emp.dmp
```

```

D:\app\JoePotter\product\11.2.0\dbhome_1\BIN>imp userid=joe/manager@orcl tables=
(emp) file=d:\emp.dmp

Import: Release 11.2.0.1.0 - Production on Sun Aug 12 20:37:38 2012

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - Produc
tion
With the Partitioning, OLAP, Data Mining and Real Application Testing options

Export file created by EXPORT:V11.02.00 via conventional path
import done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
. importing JOE's objects into JOE
. importing JOE's objects into JOE
. importing table "EMP" 17 rows imported
Import terminated successfully without warnings.

```

(2) 导入表到其它用户

要求该用户具有dba的权限，或是imp_full_database

```
Imp userid=system/manager@myor tables=(emp) file=d:\emp.dmp touser=scott
```

(3) 导入表的结构

只导入表的结构而不导入数据

```
Imp userid=scott/tiger@myor tables=(emp) file=d:\emp.dmp rows=n
```

(5) 导入数据

如果对象(比如表)已经存在可以只导入表的数据

```
Imp userid=scott/tiger@myor tables=(emp) file=d:\emp.dmp ignore=y
```

导入方案

导入方案是指使用import工具将文件中的对象和数据导入到一个或是多个方案中. 如果要导入其它方案，要求该用户具有dba的权限，或是imp_full_database

(1) 导入自身的方案

```
Imp userid=scott/tiger file=d:\xxx.dmp
```

(2) 导入其它方案

要求该用户具有dba的权限

```
Imp userid=system/manager file=d:\xxx.dmp fromuser=system touser=scott
```

导入数据库

在默认情况下，当导入数据库时，会导入所有对象结构和数据，案例如下：

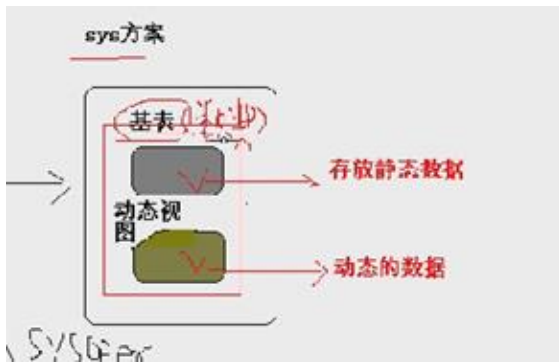
```
Imp userid=system/manager full=y file=d:\xxx.dmp
```

数据字典和动态性能视图

是什么？？

数据字典啊你是oracle数据库中最重要的一部分，它提供了数据库的一些系统信息.

动态性能视图记载了例程启动后的相关信息.



数据字典

数据字典记录了数据库的系统信息，他是只读表和视图的集合，数据字典的所有者为sys用户

用户只能在数据字典上执行查询操作(select语句), 而其维护和修改是由系统自动完成的，我们这里谈谈数据字典的组成：数据字典包括数据字典基表和数据字典视图，其中基表存储数据库的基本信息，普通用户不能直接访问数据字典的基表，数据字典视图是基于数据字典基表所建立的视图，普通用户可以通过查询数据字典视图取得系统信息，数据字典视图主要包括 User_xxx, all_xxx, dba_xxx 三种类型.

User_tables;

用于显示当前用户所拥有的所有表，它只返回用户所对应方案的所有表.

比如：

```
SQL> select table_name from user_tables;
```

```
TABLE_NAME
```

```
-----
```

```
STUDENT
```

```
CLASSES
```

```
DEPT
```

```
SALGRADE
```

```
MYEMP
```

```
USERS
```

```
EMP
```

```
7 rows selected
```

All_tables

用于显示**当前用户可以访问的所有表**, 它不仅会返回当前用户方案的所有表，还会返回当前用户可以访问的其它方案的表；

比如下：

```
SQL> select table_name from all_tables;
```

TABLE_NAME

DUAL

SYSTEM_PRIVILEGE_MAP

TABLE_PRIVILEGE_MAP

STMT_AUDIT_OPTION_MAP

AUDIT_ACTIONS

WRR\$_REPLAY_CALL_FILTER

HS_BULKLOAD_VIEW_OBJ

HS\$_PARALLEL_METADATA

HS_PARTITION_COL_NAME

HS_PARTITION_COL_TYPE

HELP

DR\$OBJECT_ATTRIBUTE

DR\$POLICY_TAB

DR\$THS

DR\$THS_PHRASE

DR\$NUMBER_SEQUENCE

XDB\$XIDX_IMP_T

SRSNAMESPACE_TABLE

SDO_UNITS_OF_MEASURE

SDO_PRIME_MERIDIANS

TABLE_NAME

SDO_ELLIPSOIDS

SDO_DATUMS

SDO_COORD_SYS

SDO_COORD_AXIS_NAMES

SDO_COORD_AXES

SDO_COORD_REF_SYS

SDO_COORD_OP_METHODS

SDO_COORD_OPS

SDO_PREFERRED_OPS_SYSTEM

SDO_PREFERRED_OPS_USER

SDO_COORD_OP_PATHS

SDO_COORD_OP_PARAMS

SDO_COORD_OP_PARAM_USE

SDO_COORD_OP_PARAM_VALS

SDO_CRS_GEOGRAPHIC_PLUS_HEIGHT
SDO_XML_SCHEMAS
AW\$AWCREATE10G
AW\$AWXML
AW\$AWREPORT
MVIEW\$_ADV_INDEX
MVIEW\$_ADV_PARTITION

TABLE_NAME

AW\$EXPRESS
AW\$AWMD
AW\$AWCREATE
SDO_CS_SRS
NTV2_XML_DATA
SDO_PROJECTIONS_OLD_SNAPSHOT
SDO_ELLIPSOIDS_OLD_SNAPSHOT
SDO_DATUMS_OLD_SNAPSHOT
OGIS_SPATIAL_REFERENCE_SYSTEMS
OGIS_GEOMETRY_COLUMNS
SDO_WS_CONFERENC
SDO_WS_CONFERENC_RESULTS
SDO_WS_CONFERENC_PARTICIPANTS
SDO_TIN_PC_SEQ
SDO_TIN_PC_SYSDATA_TABLE
WWV_FLOW_DUAL100
SDO_GEOR_XMLSCHEMA_TABLE
SDO_GEOR_PLUGIN_REGISTRY
STUDENT
CLASSES
DEPT

TABLE_NAME

SALGRADE
MYEMP
USERS
EMP
WWV_FLOW_TEMP_TABLE

WWV_FLOW_LOV_TEMP
SDO_WFS_LOCAL_TXNS
SDO_GR_RDT_1
SDO_GR_MOSAIC_3
SDO_GR_MOSAIC_2
SDO_GR_MOSAIC_1
SDO_GR_MOSAIC_0
SDO_TOPO_DATA\$
SDO_TOPO_RELATION_DATA
SDO_TOPO_TRANSACT_DATA
SDO_CS_CONTEXT_INFORMATION
SDO_TXN_IDX_EXP_UPD_RGN
SDO_TXN_IDX_DELETES
SDO_TXN_IDX_INSERTS
SDO_ST_TOLERANCE
OLAP_SESSION_CUBES

TABLE_NAME

OLAP_SESSION_DIMS
SAM_SPARSITY_ADVICE
MVIEW\$_ADV_OWB
OLAPTABLELEVELTUPLES
OLAPTABLELEVELS
RLM\$PARSEDCOND
KU\$_DATAPUMP_MASTER_10_1
KU\$_DATAPUMP_MASTER_11_1
KU\$_DATAPUMP_MASTER_11_1_0_7
KU\$_DATAPUMP_MASTER_11_2
IMPDP_STATS
ODCI_PMO_ROWIDS\$
ODCI_WARNINGS\$
ODCI_SECOBJ\$
KU\$_LIST_FILTER_TEMP_2
KU\$_LIST_FILTER_TEMP
KU\$NOEXP_TAB
OL\$NODES
OL\$HINTS
OL\$

PLAN_TABLE\$

TABLE_NAME

WRI\$ADV_ASA_RECO_DATA

PSTUBTBL

106 rows selected

Db_tables

它会显示所有方案拥有的数据库，但是查询这种数据库字典视图，要求用户必须是dba角色或是有select any table 系统权限，

例如：当用system用户查询数据字典视图dba_tables时，会返回syetem, sys, scott...方案所对应的数据库表，

SQL> desc dba_users;

Name	Type	Nullable	Default	Comments
-----	-----	-----	-----	-----
-----	-----	-----	-----	-----
USERNAME	VARCHAR2(30)			Name of the user
USER_ID	NUMBER			ID number of the user
PASSWORD	VARCHAR2(30)	Y		Deprecated from 11.2 -- use AUTHENTICATION_TYPE instead
ACCOUNT_STATUS	VARCHAR2(32)			
LOCK_DATE	DATE			Y
EXPIRY_DATE	DATE			Y
DEFAULT_TABLESPACE	VARCHAR2(30)			Default tablespace for data
TEMPORARY_TABLESPACE	VARCHAR2(30)			Default tablespace for temporary tables
CREATED	DATE			User creation date
PROFILE	VARCHAR2(30)			User resource profile name

Parameter	Default Value	Nullable	Description
INITIAL_RSRC_CONSUMER_GROUP	VARCHAR2(30)	Y	User's initial consumer group
EXTERNAL_NAME	VARCHAR2(4000)	Y	User external name
PASSWORD_VERSIONS	VARCHAR2(8)	Y	Versions of encrypted passwords
EDITIONS_ENABLED	VARCHAR2(1)	Y	Whether editions are enabled for this user

用户名，权限，角色

在建立用户时，oracle会把用户的信息存放到数据字典中，当给用户授予权限或是角色时，oracle会将权限和角色的信息存放到数据字典，

通过查询dba_users可以显示所有数据库用户的详细信息；

```
SQL> select username from dba_users;
```

```

USERNAME
-----
MGMT_VIEW
SYS
SYSTEM
DBSNMP
SYSMAN
HR
JOE
OUTLN
FLOWS_FILES
MDSYS
ORDSYS
EXFSYS
WMSYS
APPQOSSYS
APEX_030200
OWBSYS_AUDIT
ORDDATA
CTXSYS
ANONYMOUS
XDB

USERNAME
-----

```

```

ORDPLUGINS
OWBSYS
SI_INFORMTN_SCHEMA
OLAPSYS
SCOTT
ORACLE_OCM
XS$NULL
BI
PM
MDDATA
IX
SH
DIP
OE
APEX_PUBLIC_USER
SPATIAL_CSW_ADMIN_USR
SPATIAL_WFS_ADMIN_USR

```

37 rows selected

通过查询数据字典视图dba_sys_privs,可以显示用户所具有的系统权限;

```
SQL> desc dba_sys_privs;
```

Name	Type	Nullable	Default	Comments
GRANTEE	VARCHAR2(30)			Grantee Name, User or Role receiving the grant
PRIVILEGE	VARCHAR2(40)			System privilege
ADMIN_OPTION	VARCHAR2(3)	Y		Grant was with the ADMIN option

用法:

```
SQL> select * from dba_sys_privs where grantee='HR';
```

GRANTEE	PRIVILEGE
ADMIN_OPTION	
HR	CREATE VIEW
	NO

HR	UNLIMITED TABLESPACE	NO
HR	CREATE DATABASE LINK	NO
HR	CREATE SEQUENCE	NO
HR	CREATE SESSION	NO
HR	ALTER SESSION	NO
HR	CREATE SYNONYM	NO

7 rows selected

通过查询数据字典视图dba_tab_privs可以显示用户具有的对象的权限;

```
SQL> desc dba_tab_privs;
```

Name	Type	Nullable	Default	Comments
GRANTEE	VARCHAR2(30)			User to whom access was granted
OWNER	VARCHAR2(30)			Owner of the object
TABLE_NAME	VARCHAR2(30)			Name of the object
GRANTOR	VARCHAR2(30)			Name of the user who performed the grant
PRIVILEGE	VARCHAR2(40)			Table Privilege
GRANTABLE	VARCHAR2(3)	Y		Privilege is grantable
HIERARCHY	VARCHAR2(3)	Y		Privilege is with hierarchy option

通过查询数据字典视图dba_col_privs可以显示用户具有的列权限;

```
SQL> desc dba_role_privs;
```

Name	Type	Nullable	Default	Comments
GRANTEE	VARCHAR2(30)	Y		Grantee Name, User or Role receiving the grant

GRANTED_ROLE VARCHAR2 (30)	Granted role
name	
ADMIN_OPTION VARCHAR2 (3) Y	Grant was with the ADMIN
option	
DEFAULT_ROLE VARCHAR2 (3) Y	Role is designated as a DEFAULT
ROLE for the user	

通过查询数据字典视图dba_role_privs可以显示用户所有具有的角色,

```
SQL> desc dba_role_privs;
```

Name	Type	Nullable	Default	Comments

GRANTEE	VARCHAR2 (30)	Y		Grantee Name, User or Role
receiving the grant				
GRANTED_ROLE	VARCHAR2 (30)			Granted role
name				
ADMIN_OPTION	VARCHAR2 (3)	Y		Grant was with the ADMIN
option				
DEFAULT_ROLE	VARCHAR2 (3)	Y		Role is designated as a DEFAULT
ROLE for the user				

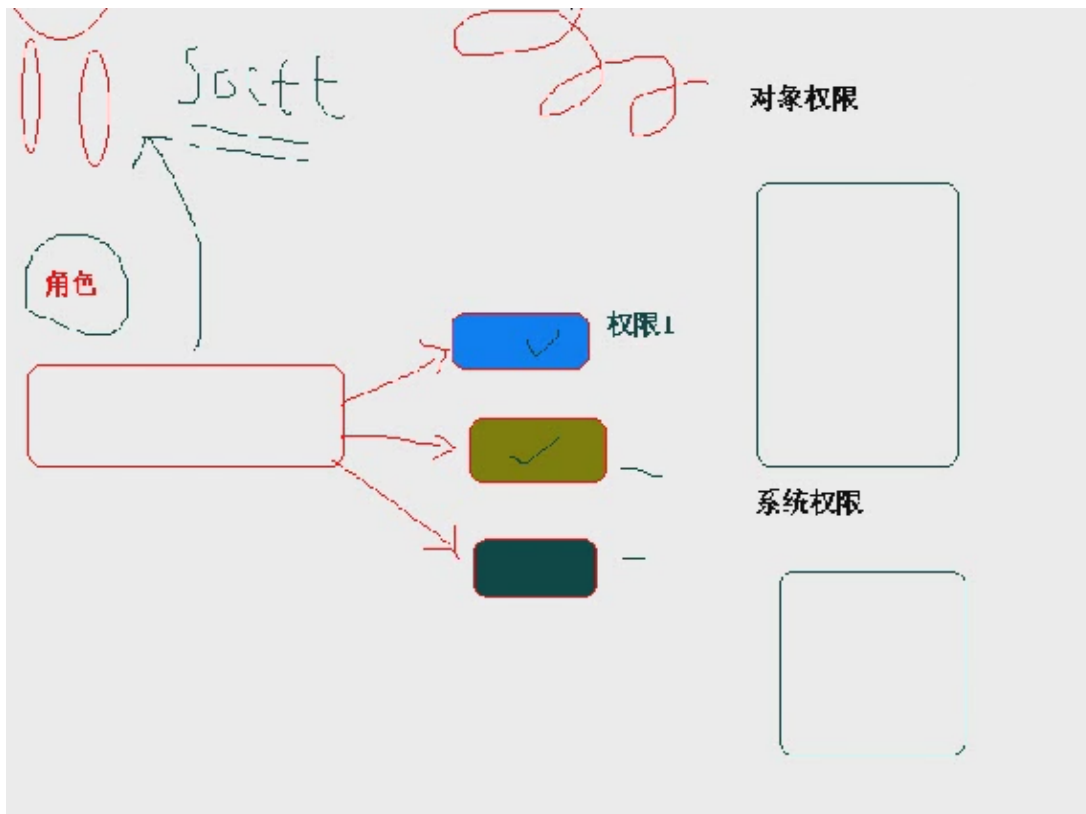
用法:

```
SQL> select * from dba_role_privs where GRANTEE='JOE';
```

GRANTEE	GRANTED_ROLE	ADMIN_OPTION
DEFAULT_ROLE		

JOE	RESOURCE	NO
YES		
JOE	CONNECT	NO
YES		

这里给大家再讲讲角色和权限的关系.



a. 如何查询一个角色包括的权限？

a. 一个角色包括的系统权限

SQL> SELECT * FROM DBA_SYS_PRIVS WHERE GRANTEE='CONNECT';红色可换

GRANTEE	PRIVILEGE	
ADMIN_OPTION		

CONNECT	CREATE SESSION	NO
---------	----------------	----

SQL> SELECT * FROM DBA_SYS_PRIVS WHERE GRANTEE='RESOURCE';

GRANTEE	PRIVILEGE	
ADMIN_OPTION		

RESOURCE	CREATE TRIGGER	NO
RESOURCE	CREATE SEQUENCE	NO
RESOURCE	CREATE TYPE	NO
RESOURCE	CREATE PROCEDURE	NO
RESOURCE	CREATE CLUSTER	NO
RESOURCE	CREATE OPERATOR	NO

RESOURCE	CREATE INDEXTYPE	NO
RESOURCE	CREATE TABLE	NO

另外亦可以这样查看:select * from role_sys_privs where role ='connect';

b. 一个角色包括的对象权限

SELECT * FROM DBA_TAB_PRIVS WHERE GRANTEE=' CONNECT' ;

查询oracle中所有的系统权限，一般是dba

SQL> select * from system_privilege_map order by 'JOE';

PRIVILEGE NAME	PROPERTY

-3 ALTER SYSTEM	0
-4 AUDIT SYSTEM	0
-5 CREATE SESSION	0
-6 ALTER SESSION	0
-7 RESTRICTED SESSION	0
-10 CREATE TABLESPACE	0
-11 ALTER TABLESPACE	0
-12 MANAGE TABLESPACE	0
-13 DROP TABLESPACE	0
-15 UNLIMITED TABLESPACE	0
-20 CREATE USER	0
-21 BECOME USER	0
-22 ALTER USER	0
-23 DROP USER	0
-30 CREATE ROLLBACK SEGMENT	0
-31 ALTER ROLLBACK SEGMENT	0
-32 DROP ROLLBACK SEGMENT	0
-40 CREATE TABLE	0
-41 CREATE ANY TABLE	0
-42 ALTER ANY TABLE	0
PRIVILEGE NAME	PROPERTY

-43 BACKUP ANY TABLE	0
-44 DROP ANY TABLE	0
-45 LOCK ANY TABLE	0

-46	COMMENT ANY TABLE	0
-47	SELECT ANY TABLE	0
-48	INSERT ANY TABLE	0
-49	UPDATE ANY TABLE	0
-50	DELETE ANY TABLE	0
-60	CREATE CLUSTER	0
-61	CREATE ANY CLUSTER	0
-62	ALTER ANY CLUSTER	0
-63	DROP ANY CLUSTER	0
-71	CREATE ANY INDEX	0
-72	ALTER ANY INDEX	0
-73	DROP ANY INDEX	0
-80	CREATE SYNONYM	0
-81	CREATE ANY SYNONYM	0
-82	DROP ANY SYNONYM	0
-83	SYSDBA	0
-84	SYSOPER	0
-85	CREATE PUBLIC SYNONYM	0

PRIVILEGE NAME	PROPERTY	
-86	DROP PUBLIC SYNONYM	0
-90	CREATE VIEW	0
-91	CREATE ANY VIEW	0
-92	DROP ANY VIEW	0
-105	CREATE SEQUENCE	0
-106	CREATE ANY SEQUENCE	0
-107	ALTER ANY SEQUENCE	0
-108	DROP ANY SEQUENCE	0
-109	SELECT ANY SEQUENCE	0
-115	CREATE DATABASE LINK	0
-120	CREATE PUBLIC DATABASE LINK	0
-121	DROP PUBLIC DATABASE LINK	0
-125	CREATE ROLE	0
-126	DROP ANY ROLE	0
-127	GRANT ANY ROLE	0
-128	ALTER ANY ROLE	0
-130	AUDIT ANY	0
-135	ALTER DATABASE	0

-138	FORCE TRANSACTION	0
-139	FORCE ANY TRANSACTION	0
-140	CREATE PROCEDURE	0

PRIVILEGE NAME	PROPERTY	
-141	CREATE ANY PROCEDURE	0
-142	ALTER ANY PROCEDURE	0
-143	DROP ANY PROCEDURE	0
-144	EXECUTE ANY PROCEDURE	0
-151	CREATE TRIGGER	0
-152	CREATE ANY TRIGGER	0
-153	ALTER ANY TRIGGER	0
-154	DROP ANY TRIGGER	0
-160	CREATE PROFILE	0
-161	ALTER PROFILE	0
-162	DROP PROFILE	0
-163	ALTER RESOURCE COST	0
-165	ANALYZE ANY	0
-167	GRANT ANY PRIVILEGE	0
-172	CREATE MATERIALIZED VIEW	0
-173	CREATE ANY MATERIALIZED VIEW	0
-174	ALTER ANY MATERIALIZED VIEW	0
-175	DROP ANY MATERIALIZED VIEW	0
-177	CREATE ANY DIRECTORY	0
-178	DROP ANY DIRECTORY	0
-180	CREATE TYPE	0

PRIVILEGE NAME	PROPERTY	
-181	CREATE ANY TYPE	0
-182	ALTER ANY TYPE	0
-183	DROP ANY TYPE	0
-184	EXECUTE ANY TYPE	0
-186	UNDER ANY TYPE	0
-188	CREATE LIBRARY	0
-189	CREATE ANY LIBRARY	0
-190	ALTER ANY LIBRARY	0
-191	DROP ANY LIBRARY	0

-192 EXECUTE ANY LIBRARY	0
-200 CREATE OPERATOR	0
-201 CREATE ANY OPERATOR	0
-202 ALTER ANY OPERATOR	0
-203 DROP ANY OPERATOR	0
-204 EXECUTE ANY OPERATOR	0
-205 CREATE INDEXTYPE	0
-206 CREATE ANY INDEXTYPE	0
-207 ALTER ANY INDEXTYPE	0
-208 DROP ANY INDEXTYPE	0
-209 UNDER ANY VIEW	0
-210 QUERY REWRITE	0

PRIVILEGE NAME	PROPERTY
-211 GLOBAL QUERY REWRITE	0
-212 EXECUTE ANY INDEXTYPE	0
-213 UNDER ANY TABLE	0
-214 CREATE DIMENSION	0
-215 CREATE ANY DIMENSION	0
-216 ALTER ANY DIMENSION	0
-217 DROP ANY DIMENSION	0
-218 MANAGE ANY QUEUE	1
-219 ENQUEUE ANY QUEUE	1
-220 DEQUEUE ANY QUEUE	1
-222 CREATE ANY CONTEXT	0
-223 DROP ANY CONTEXT	0
-224 CREATE ANY OUTLINE	0
-225 ALTER ANY OUTLINE	0
-226 DROP ANY OUTLINE	0
-227 ADMINISTER RESOURCE MANAGER	1
-228 ADMINISTER DATABASE TRIGGER	0
-233 MERGE ANY VIEW	0
-234 ON COMMIT REFRESH	0
-235 EXEMPT ACCESS POLICY	0
-236 RESUMABLE	0

PRIVILEGE NAME	PROPERTY
----------------	----------

-237	SELECT ANY DICTIONARY	0
-238	DEBUG CONNECT SESSION	0
-241	DEBUG ANY PROCEDURE	0
-243	FLASHBACK ANY TABLE	0
-244	GRANT ANY OBJECT PRIVILEGE	0
-245	CREATE EVALUATION CONTEXT	1
-246	CREATE ANY EVALUATION CONTEXT	1
-247	ALTER ANY EVALUATION CONTEXT	1
-248	DROP ANY EVALUATION CONTEXT	1
-249	EXECUTE ANY EVALUATION CONTEXT	1
-250	CREATE RULE SET	1
-251	CREATE ANY RULE SET	1
-252	ALTER ANY RULE SET	1
-253	DROP ANY RULE SET	1
-254	EXECUTE ANY RULE SET	1
-255	EXPORT FULL DATABASE	0
-256	IMPORT FULL DATABASE	0
-257	CREATE RULE	1
-258	CREATE ANY RULE	1
-259	ALTER ANY RULE	1
-260	DROP ANY RULE	1

PRIVILEGE NAME	PROPERTY	
-261	EXECUTE ANY RULE	1
-262	ANALYZE ANY DICTIONARY	0
-263	ADVISOR	0
-264	CREATE JOB	0
-265	CREATE ANY JOB	0
-266	EXECUTE ANY PROGRAM	0
-267	EXECUTE ANY CLASS	0
-268	MANAGE SCHEDULER	0
-269	SELECT ANY TRANSACTION	0
-270	DROP ANY SQL PROFILE	0
-271	ALTER ANY SQL PROFILE	0
-272	ADMINISTER SQL TUNING SET	0
-273	ADMINISTER ANY SQL TUNING SET	0
-274	CREATE ANY SQL PROFILE	0
-275	EXEMPT IDENTITY POLICY	0

-276	MANAGE FILE GROUP	1
-277	MANAGE ANY FILE GROUP	1
-278	READ ANY FILE GROUP	1
-279	CHANGE NOTIFICATION	0
-280	CREATE EXTERNAL JOB	0
-281	CREATE ANY EDITION	0

PRIVILEGE NAME	PROPERTY	
-282	DROP ANY EDITION	0
-283	ALTER ANY EDITION	0
-284	CREATE ASSEMBLY	0
-285	CREATE ANY ASSEMBLY	0
-286	ALTER ANY ASSEMBLY	0
-287	DROP ANY ASSEMBLY	0
-288	EXECUTE ANY ASSEMBLY	0
-289	EXECUTE ASSEMBLY	0
-290	CREATE MINING MODEL	0
-291	CREATE ANY MINING MODEL	0
-292	DROP ANY MINING MODEL	0
-293	SELECT ANY MINING MODEL	0
-294	ALTER ANY MINING MODEL	0
-295	COMMENT ANY MINING MODEL	0
-301	CREATE CUBE DIMENSION	0
-302	ALTER ANY CUBE DIMENSION	0
-303	CREATE ANY CUBE DIMENSION	0
-304	DELETE ANY CUBE DIMENSION	0
-305	DROP ANY CUBE DIMENSION	0
-306	INSERT ANY CUBE DIMENSION	0
-307	SELECT ANY CUBE DIMENSION	0

PRIVILEGE NAME	PROPERTY	
-308	CREATE CUBE	0
-309	ALTER ANY CUBE	0
-310	CREATE ANY CUBE	0
-311	DROP ANY CUBE	0
-312	SELECT ANY CUBE	0
-313	UPDATE ANY CUBE	0

-314 CREATE MEASURE FOLDER	0
-315 CREATE ANY MEASURE FOLDER	0
-316 DELETE ANY MEASURE FOLDER	0
-317 DROP ANY MEASURE FOLDER	0
-318 INSERT ANY MEASURE FOLDER	0
-319 CREATE CUBE BUILD PROCESS	0
-320 CREATE ANY CUBE BUILD PROCESS	0
-321 DROP ANY CUBE BUILD PROCESS	0
-322 UPDATE ANY CUBE BUILD PROCESS	0
-326 UPDATE ANY CUBE DIMENSION	0
-327 ADMINISTER SQL MANAGEMENT OBJECT	0
-328 ALTER PUBLIC DATABASE LINK	0
-329 ALTER DATABASE LINK	0
-350 FLASHBACK ARCHIVE ADMINISTER	0

208 rows selected

查询oracle中所有对象权限,一般为dba

Select distinct privilege from dba_tab_privs;

SQL> select distinct privilege from dba_tab_privs order by 'JOE';

PRIVILEGE

EXECUTE

FLASHBACK

DEQUEUE

ON COMMIT REFRESH

ALTER

DELETE

UPDATE

DEBUG

QUERY REWRITE

SELECT

READ

USE

WRITE

INSERT

INDEX

REFERENCES

MERGE VIEW

17 rows selected

a. Oracle究竟有多少种角色?

1. Select * from dba_roles;

2. SQL> select * from dba_roles;

3.

4. ROLE	PASSWORD_REQUIRED	AUTHENTICATION_TYPE
---------	-------------------	---------------------

5. -----	-----	-----
-		

6. CONNECT	NO	NONE
7. RESOURCE	NO	NONE
8. DBA	NO	NONE
9. SELECT_CATALOG_ROLE	NO	NONE
10. EXECUTE_CATALOG_ROLE	NO	NONE
11. DELETE_CATALOG_ROLE	NO	NONE
12. EXP_FULL_DATABASE	NO	NONE
13. IMP_FULL_DATABASE	NO	NONE
14. LOGSTDBY_ADMINISTRATOR	NO	NONE
15. DBFS_ROLE	NO	NONE
16. AQ_ADMINISTRATOR_ROLE	NO	NONE
17. AQ_USER_ROLE	NO	NONE
18. DATAPUMP_EXP_FULL_DATABASE	NO	NONE
19. DATAPUMP_IMP_FULL_DATABASE	NO	NONE
20. ADM_PARALLEL_EXECUTE_TASK	NO	NONE
21. GATHER_SYSTEM_STATISTICS	NO	NONE
22. JAVA_DEPLOY	NO	NONE
23. RECOVERY_CATALOG_OWNER	NO	NONE
24. SCHEDULER_ADMIN	NO	NONE
25. HS_ADMIN_SELECT_ROLE	NO	NONE

26.

27. ROLE	PASSWORD_REQUIRED	AUTHENTICATION_TYPE
----------	-------------------	---------------------

28. -----	-----	-----
--		

29. HS_ADMIN_EXECUTE_ROLE	NO	NONE
30. HS_ADMIN_ROLE	NO	NONE

31. GLOBAL_AQ_USER_ROLE	GLOBAL	GLOBAL
32. OEM_ADVISOR	NO	NONE
33. OEM_MONITOR	NO	NONE
34. WM_ADMIN_ROLE	NO	NONE
35. JAVAUSERPRIV	NO	NONE
36. JAVAIDPRIV	NO	NONE
37. JAVASYSPRIV	NO	NONE
38. JAVADEBUGPRIV	NO	NONE
39. EJBCLIENT	NO	NONE
40. JMXSERVER	NO	NONE
41. JAVA_ADMIN	NO	NONE
42. CTXAPP	NO	NONE
43. XDBADMIN	NO	NONE
44. XDB_SET_INVOKER	NO	NONE
45. AUTHENTICATEDUSER	NO	NONE
46. XDB_WEBSERVICES	NO	NONE
47. XDB_WEBSERVICES_WITH_PUBLIC	NO	NONE
48. XDB_WEBSERVICES_OVER_HTTP	NO	NONE
49. ORDADMIN	NO	NONE
50.		
51. ROLE	PASSWORD_REQUIRED	
AUTHENTICATION_TYPE		
52. -----	-----	-----
--		
53. OLAPI_TRACE_USER	NO	NONE
54. OLAP_XS_ADMIN	NO	NONE
55. OWB_USER	NO	NONE
56. OLAP_DBA	NO	NONE
57. CWM_USER	NO	NONE
58. OLAP_USER	NO	NONE
59. SPATIAL_WFS_ADMIN	NO	NONE
60. WFS_USR_ROLE	YES	PASSWORD
61. SPATIAL_CSW_ADMIN	YES	PASSWORD
62. CSW_USR_ROLE	YES	PASSWORD
63. MGMT_USER	NO	NONE
64. APEX_ADMINISTRATOR_ROLE	NO	NONE
65. OWB\$CLIENT	YES	PASSWORD
66. OWB_DESIGNCENTER_VIEW	NO	NONE
67.		

68. 55 rows selected

查询数据库的表空间

Select tablespace_name from dba_tablespaces;

SQL> Select tablespace_name from dba_tablespaces;

TABLESPACE_NAME

SYSTEM

SYSAUX

UNDOTBS1

TEMP

USERS

EXAMPLE

如何查看某个用户，具有什么样的角色？

SELECT * FROM DBA_ROLE_PRIVS WHERE GRANTEE=' 用户名' ;

SQL> SELECT * FROM DBA_ROLE_PRIVS WHERE GRANTEE='JOE';

GRANTEE GRANTED_ROLE ADMIN_OPTION

DEFAULT_ROLE

JOE RESOURCE NO

YES

JOE CONNECT NO

YES

显示当前用户可以访问的所有数据库

Select * from dict where comments like '%grant%' ;

SQL> Select * from dict where comments like '%grant%';

TABLE_NAME COMMENTS

DBA_AUDIT_STATEMENT Audit trail records concerning grant,
revoke, audit, noaudit and alter system

DBA_CONNECT_ROLE_GRANTEES	Information regarding which users are granted CONNECT
DBA_COL_PRIVS	All grants on columns in the database
DBA_ROLE_PRIVS	Roles granted to users and roles
DBA_STREAMS_ADMINISTRATOR	Users granted the privileges to be a streams administrator
DBA_SYS_PRIVS	System privileges granted to users and roles
DBA_TAB_PRIVS	All grants on objects in the database
USER_AUDIT_STATEMENT	Audit trail records concerning grant, revoke, audit, noaudit and alter system
USER_COL_PRIVS	Grants on columns for which the user is the owner, grantor or grantee
USER_COL_PRIVS_MADE	All grants on columns of objects owned by the user
USER_COL_PRIVS_RECD	Grants on columns for which the user is the grantee
USER_ROLE_PRIVS	Roles granted to current user
USER_SYS_PRIVS	System privileges granted to current user
USER_TAB_PRIVS	Grants on objects for which the user is the owner, grantor or grantee
USER_TAB_PRIVS_MADE	All grants on objects owned by the user
USER_TAB_PRIVS_RECD	Grants on objects for which the user is the grantee
ALL_COL_PRIVS	Grants on columns for which the user is the grantor, grantee, owner, or an enabled role or PUBLIC is the grantee
ALL_COL_PRIVS_MADE	Grants on columns for which the user is owner or grantor
ALL_COL_PRIVS_RECD	Grants on columns for which the user, PUBLIC or enabled role is the grantee
ALL_TAB_PRIVS	Grants on objects for which the user is the grantor, grantee, owner, or an enabled role or PUBLIC is the grantee

TABLE_NAME	COMMENTS
------------	----------

-----	-----

ALL_TAB_PRIVS_MADE	User's grants and grants on user's objects
ALL_TAB_PRIVS_RECD	Grants on objects for which the user, PUBLIC or enabled role is the grantee
COLUMN_PRIVILEGES	Grants on columns for which the user is the grantor, grantee, owner, or an enabled role or PUBLIC is the grantee
ROLE_ROLE_PRIVS	Roles which are granted to roles
ROLE_SYS_PRIVS	System privileges granted to roles
ROLE_TAB_PRIVS	Table privileges granted to roles
TABLE_PRIVILEGES	Grants on objects for which the user is the grantor, grantee, owner, or an enabled role or PUBLIC is the grantee

27 rows selected

显示当前数据库的全称

```
SQL> select * from global_name;
```

```
GLOBAL_NAME
```

```
-----
-----
ORCL
```

其它说明

数据字典记录有oracle数据库的所有系统信息，通过查询数据字典可以取得以下系统信息：比如

- a. 对象定义情况
- b. 对象占用空间大小
- c. 列信息
- d. 约束信息
- e. . . .

但是因为这些信息，可以通过pl/sql development工具查询得到，

动态性能视图

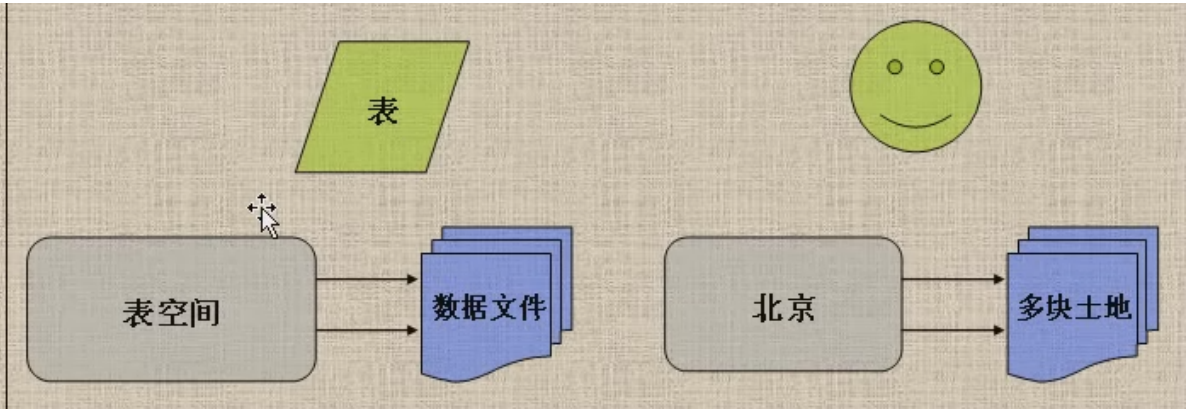
动态性能视图用于记录当前例程的活定信息，当启动oracle server时，系统会建立动态性能视图；当停止oracle server时，系统会删除动态性能视图，oracle为每个动态性能视图都提供了

相应的同义词，并且其同义词是已v\$开始的，
例如v_\$datafile的同义词为\$datafile;动态性能视图的所有者为sys，一般情况下，由dba或是特权用户来查询动态性能视图。
因为这个在实际中用的较少，飞过。

管理表空间和数据文件

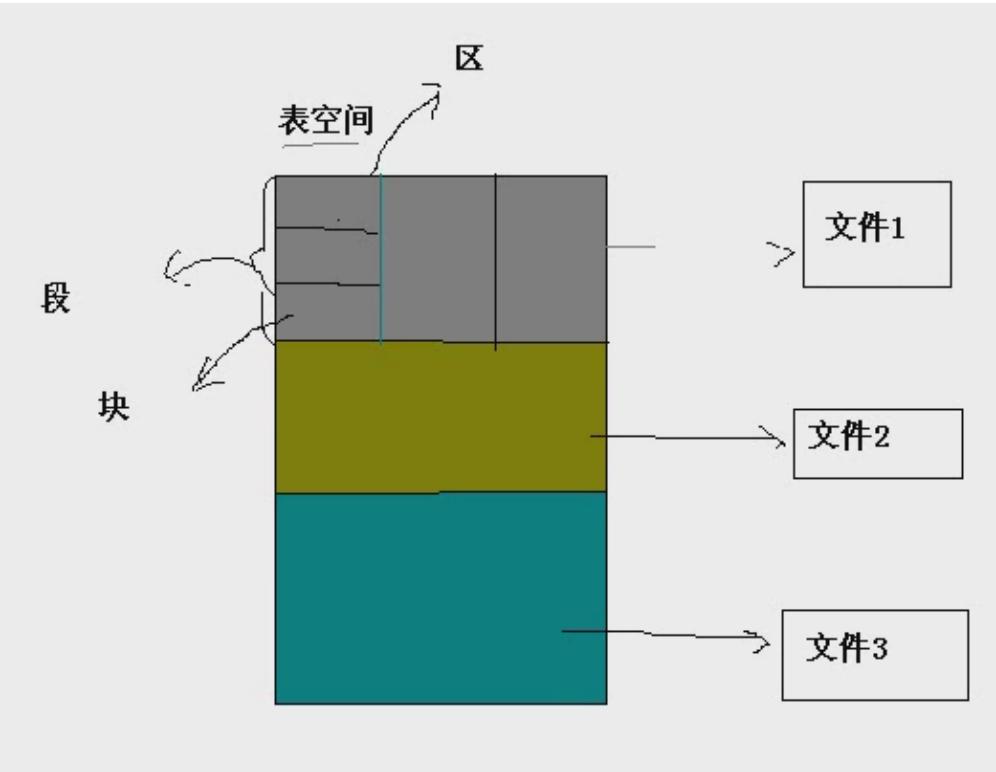
介绍

表空间是数据库的逻辑组成部分，从物理上讲，数据库数据存放在数据文件中；从逻辑上讲，数据库则是存放在表空间中，表空间由一个或是多个数据文件组成。



Oracle中逻辑结构包括表空间、段、区和块。

说明一下数据库由表空间构成，而表空间又是由段构成，而段又是由区构成，而区又是由oracle块构成的的这样的一种结构，可以提高数据库的效率. 画图说明逻辑关系；



表空间用于从逻辑上组织数据库的数据. 数据库逻辑上是由一个或是多个表空间组成的. 通过表空间可以达到一下的作用:

- i. 控制数据库占用的磁盘空间
- ii. **Db**a可以将不同数据类型部署到不同的位置, 这样有利于提高**i/o**性能, 同时利于备份和恢复等管理操作

建立表空间

建立表空间是使用create tablespace 命令完成的, 需要注意的是, 一般情况下, 建立表空间是特权用户或是dba来执行的, 如果用其它用户来创建表空间, 则用户必须要具有create tablespace 的系统权限.

建立数据表空间

在建立数据库后, 为便于管理表, 最好建立自己的表空间

```
Create tablespace sp001 datafile 'd:\sp001' size 20m uniform size 128k;
```

```
SQL> create tablespace sp001 datafile 'd:\sp001.dbf' size 20m uniform size 128k;
```

Tablespace created

说明: 执行完上述命令后, 会建立名称为sp001的表空间, 并且为该表空间建立名称为sp001.dbf的数据文件, 区的大小为128k

使用数据表空间

```
Create table mypart(deptno number(2),dname varchar2(14),loc varchar(13)) tablespace sp001;
```

改变表空间的状态

当建立表空间时, 表空间处于联机的状态, 此时该表空间是可以访问的, 并且该表空间是可以读写的, 即可以查询该表空间的数据, 而且还可以在表空间执行各种语句. 但是在进行系统维护或是数据维护时, 可能需要改变表空间的状态. 一般情况下, 又特权用户或是dba来操作.

1. 使表空间脱机

```
Alter tablespace 表空间名 offline
```

2. 使表空间联机

```
Alter tablespace 表空间名 online
```

3. 只读表空间

当建立表空间时, 表空间可以读写, 如果不希望在该表空间上执行update, delete, insert操作, 那么可以将表空间修改为

```
只读alter tablespace 表空间名 read only
```

```
读写alter tablespace 表空间名 read write
```

说明只读特性:

1. 知道表空间名，显示该表空间包括的所有表

```
Select * from all_tables where tablespace_name=' 表空间名' ;
```

2. 知道表名，查看该表属于那个表空间

```
Select tablespace_name,table_name from user_tables where table_name=' emp' ;
```

```
SQL> SELECT TABLESPACE_NAME, TABLE_NAME FROM USER_TABLES
      2  WHERE TABLE_NAME='EMP';
```

TABLESPACE_NAME	TABLE_NAME
USERS	EMP

```
SQL> SELECT TABLESPACE_NAME, TABLE_NAME FROM USER_TABLES WHERE
TABLE_NAME='MYPART';
```

TABLESPACE_NAME	TABLE_NAME
SP001	MYPART

通过2) 我们可以知道scott.emp是system这个表空间上，现在我们可以将system改为只读的是我们不会成功，因为system是系统表空间，如果是普通表空间，那么我们就可以将其设为只读的，给大家做一个演示，可以加强理解。

删除表空间

一般情况下，由特权用户或是dba来操作，如果是其它用户来操作，那么要求用户具有drop tablespace系统的权限。

```
Drop tablespace '表空间' including contents and datafiles;
```

说明：including content 表示删除表空间时，删除该表空间的所有数据库对象，而datafiles表示将数据库文件也删除。

1、使用drop tablespace including contents;方式直接来删除。

```
SQL>set linesize 132
```

```
SQL>set pagezie 0
```

```
SQL>set timing on
```

```
SQL>drop tablespace TBS_TEST including contents
```

已用时间：03：35：39.10

经过我耐心的等待，花了三个半小时。

2、测一下同样的表空间，把它转换为Local方式管理的删除效率。

a、把TBS_TEST通过恢复回来。

b、把TBS_TEST转化为Local管理的方式。

```
SQL>set timing on
```

```
SQL> exec sys.dbms_space_admin.tablespace_migrate_to_local(' TBS_TEST' ) ;
```

已用时间：00：06：33.25

c、删出这个空间。

```
SQL> drop tablespace TBS_TEST including contents
```

```
2 /
```

已用时间：00：00：45.56

可以看到总共才花费了7分多钟。

3、测一下同样的表空间，先删除其中的对象，然后再删这个表空间的效率如何。

a、把TBS_TEST通过恢复回来。

b、形成删除表的语句

```
SQL>set linesize 132
```

```
SQL>set pagezie 0
```

```
SQL>set timing off
```

```
SQL>spool drop_test_tables.sql
```

```
SQL>SELECT ' Drop table ' ||TABLE_name||' ;' FROM dba_tables WHERE  
tablespace_name=' TBS_TEST' ;
```

```
SQL>spool off
```

c、删除表

```
SQL>@drop_test_tables.sql
```

这一步大约花费20秒。

d、删出这个空间。

```
SQL>set timing on
```

```
SQL> drop tablespace TBS_TEST including contents;
```

已用时间: 00: 07: 35.53

可以看到总共才花费了将近8分钟。

总结：我们在做数据字典管理的表空间的删除时，最好先删除表空间中的对象再进行删除该表空间操作。也可以先把它转换为本地(local)管理的空间再进行删除。不过需要补充的是本地管理的空间在8i以后的版本中才没有简单的方法来删除表空间的数据文件,唯一的方法是删除整个定义的表空间,步骤有下面(前提是这个数据文件上的数据是不需要了):

如果数据库运行在非归档模式:

1. MOUNT数据库 - startup mount
2. 删除数据文件 - alter database datafile xxx offline drop

3. 打开 (OPEN) 数据库 - `alter database open`
4. 查看属于该表空间的所有对象:

```
select owner, segment_name, segment_type
from dba_segments
where tablespace_name=' tbs_name'
```
5. 导出该表空间的所有对象-----用exp命令来做
6. 删除表空间 - `drop tablespace tbs_name including contents`
7. 删除这个表空间的所有物理的数据文件Delete the physical datafiles belonging to the tablespace
8. 重建表空间, 导入前面导出的DMP文件.

如果数据库是运行在归档模式:

1. MOUNT数据库 - `startup mount`
2. 删除数据文件 - `alter database datafile xxx offline`
(Note: offline这个数据文件, 此数据文件还是属于这个数据库的一部分, 只是在控制文件中将它状态标记为offline.)
3. 在操作系统一级删除物理的数据文件
4. 打开 (OPEN) 数据库 - `alter database open`
5. 后面的可以做下面操作:
导出该表空间的对象
删除表空间
重建表空间并导入对象

如果数据库运行在归档模式下, 并且数据文件有备份:

1. MOUNT数据库
2. OFFLINE数据文件:`alter database datafile xxx offline;`
3. 将备份的数据文件拷贝到原来数据文件的位置.
4. 将备份数据文件到目前的所有归档日志放到归档目录.
5. 恢复数据文件:`recover automatic datafile xxx`(要输入全路径名)
6. 然后ONLINE数据文件:`alter database datafile xxx online;`
7. 打开 (OPEN) 数据库:`alter database open;`
8. 做一次数据库的关机全备份.

扩展表空间

1. 增加数据文件

```
Sql>alter tablespace sp001 add datafile 'd:\sp002' size 20m;
```

2. 增加数据文件的大小

```
Sql>alter tablespace 表空间名 'd:\sp001.dbf' resize 20m;
```

这里需要注意的是数据文件的大小不要超过500m

3. 设置文件的自动增长

```
Sql>alter tablespace 表空间名 'd:\sp001.dbf' autoextend on next 10m maxsize 500m;
```

移动数据文件

有时，如果你的数据文件所在的磁盘损坏时，该数据文件将不能在使用，为了我能够重新使用，需要将这些文件的副本移动到其它的磁盘，让后恢复。下面以移动数据文件sp001.dbf为例来说明：

1. 确定数据文件所在的表空间

```
Select tablespace_name from dba_data_files where file_name=' d:\sp001' ;
```

2. 使用空间脱机

确保数据文件的一致性，将表空间转变为offline的状态.

```
Alter tablespace sp001 offline;
```

3. 使用命令移动数据文件到指定的目标位置

```
Sql>host move d:\sp001.dbf c:\sp001.dbf;
```

4. 执行alter tablespace 命令

在物理上移动了数据后，还必须执行alter tablespace 命令对数据库文件进行逻辑修改；

```
Sql>alter tablespace sp001 rename datafile 'd:\sp001.dbf' to 'c:\sp001.dbf' ;
```

5. 使表空间联机

在移动了数据文件后，为了使用户访问该表空间，必须将其转变为online状态：

```
Sql>alter tablespace sp001 online;
```

表空间小结

1. 了解表空间和数据文件的作用

2. 掌握常用表空间，undo表空间和临时表空间的建立方法

3. 了解表空间的各个状态

(online, offline, read write, read only) 的作用，及如何该表表空间的状态的方法

4. 了解移动数据文件的原因，及使用alter tablespace 和alter datatable 命令移动数据文件的方法.

其它表空间

除了常用的数据表空间外，还有其它类型表空间：

1. 索引表空间
2. undo表空间
3. 临时表空间
4. 非标准块的表空间

这几种表空间，大家可以自己参考书籍研究，这里不讲。

维护数据的完整性

介绍

数据的完整性用于确保数据库数据遵从一定的商业和逻辑规则. 在oracle中，数据完整性可以使用约束、触发器、应用程序（过程、函数）三个方法来实现，在这三种方法中，因为约束易于维护，并且具有最好的性能，所以作为维护数据完整性的首选.

约束

约束用于确保数据库数据满足特定的商业规则. 在oracle中，约束包括：not null、unique、primary key, foreign key, 和check五种.

Not null（非空）

如果在列上定义了not null，那么当插入数据时，必须为列提供数据.

Unique(唯一)

当定义了唯一约束后，该列值是不能重复的，但是可以为null。

Primary key（主键）

用于唯一的标示表行的数据，当定义主键约束后，该列不但不能重复而且不能为null。

需要说明的是:一张表中最多只能有一个主键，但是可以有多个unique约束。

Foreign key（外键）

用于定义主表和从表之间的关系，外键约束要定义在从表上，主表则必须具有主键约束或是unique约束，当定义外键约束后，要求外键列数据库必须在主表的主键列存在或是为null，

Check

用于强制行数据必须满足的条件，假定在sal列上定义了check约束，并要求sal列值在1000~2000之间，如果不再1000~2000之间就会提示出错.

商店售货系统设计案例

现有一个商品的数据，记录客户及其购物情况，由下面三个表组成：

商品goods（商品号goodsid，商品名goodsname，单价unitprice，商品类别category，供应商provider）；

客户customer（客户号customerid，姓名name，住址address，电邮email，性别sex，省份证carid）；

购买purchase（客户号customerid，商品号goodsid，购买数量nums）；

请用sql语言完成下列功能：

1. 建表，在定义中要求声明：

每个表的主外键

客户的姓名不能为空值

单价必须大于0，购买数量必须在1到30之间；

电邮不能够重复；

客户的性别必须是 男或者是女， 默认是男

```
SQL> create table goods (goodsid char(8) primary key,  
2 goodsname varchar2(20),  
3 unitprice number(10,2) check (unitprice > 0),  
4 category varchar2(20),  
5 provider varchar2(20));
```

Table created

```
SQL> create table customer (customerid char(8) primary key,  
2 name varchar2(30) not null,  
3 address varchar2(30),  
4 email varchar2(30) unique,  
5 sex char(2) default '男' check (sex in ('女','男')),  
6 carid char(18));
```

Table created

```
SQL> create table purchase (customerid char(8) references  
customer(customerid),  
2 goodsid char(8) references goods(goodsid),  
3 nums number(5) check(nums between 1 and 30));
```

Table created

商品售货系统表设计案例（2）

如果在建表时忘记建立必要的约束，则可以在建立表后使用alter table 命令为表增加约束，但是要注意：增加not null约束时，需要使用motify选项，而增加其它四种约束使用add选项。

每个表的主外键

客户的姓名不能为空值，-增加商品名也不能为空

```
SQL> alter table goods modify goodsname not null;
```

Table altered

单价必须大于0，购买数量必须在1到30之间；

电邮不能够重复；增加省份证也不能重复

```
SQL> alter table customer add constraint cardunique unique (CARID);
```

Table altered

客户的性别必须是 男或者是女， 默认是男

增加客户的地址只能是' 海淀' , ' 朝阳' , ' 东城' , ' 西城' , ' 通州' , ' 崇文' 。

```
SQL> ALTER TABLE CUSTOMER ADD CONSTRAINT ADDRESSCHECK CHECK (ADDRESS IN (
    2  '海淀','东城','西城','通州','崇文'));
```

Table altered

删除约束

当不再需要某个约束，可以删除，alter table 表名drop constraint 约束名称；

特别注意：

在删除主键约束的时候，可能有错误，比如：

Alter table 表名drop primary key;

这是因为如果在两张表存在主从关系，那么在删除主表的主键约束时，必须带上cascade选项
如像

Alter table 表名 drop primary key cascade;

显示约束信息

1显示约束信息

通过查询数据字典视图user_constraints，可以显示当前用户所有的约束的信息，

```
Select constraint_name, constraint_type,status, validated from user_constraints
where table_name=' 表名' ;??????????
```

显示约束列

通过查询数据字典视图user_cons_columns，可以显示约束所对应的表列信息。

```
Select column_name,position from user_cons_columns where constraint_name=' 约束
名' ;
```

当然也有更容易的方法，直接用pl/sql developer 查看即可，

列级定义

列级定义是在定义列的同时定义约束，

如在dcpartment表定义主键约束

```
Create table department4
```

```
(dept_id number(2) constraint pk_department primary key,
```

```
Name varchar2(12),
```

```
Loc varchar2(12));
```

表级定义

表级定义是指在定义了所有列后面，再定义约束，这里需要注意：

Not null约束只能在列级上定义，

以在建立employee2表时定义主键约束和外键约束为例：

```
Create table employee2
```

```
(emp_id number(4),name varchar2(15),dept_id number(2),
```

```
Constraint pk_employee primary key (emp_id),
```

```
Constraint fk_department foreign key (dept_id)
```

```
References department4(dept_id));
```

管理索引

介绍

索引是用于加速数据存取的数据对象，合理的使用索引可以大大降低i/o次数，从而提高数据访问性能. 索引有很多种我们主要介绍常用的几种：

为什么添加了索引后，加快查询速度呢？