数据库实训课程讲义

1、数据库实例创建

数据库软件可以创建多个数据库实例，实例之间通过端口号区分。达梦数据库实例默认的端口号是 5236.

切换到路径 /home/dmdba/dmdbms/tool

[dmdba@localhost 桌面]$ cd /home/dmdba/dmdbms/tool/

查看当前工作目录

[dmdba@localhost tool]$ pwd

列出当前目录下的信息（包含目录和文件）

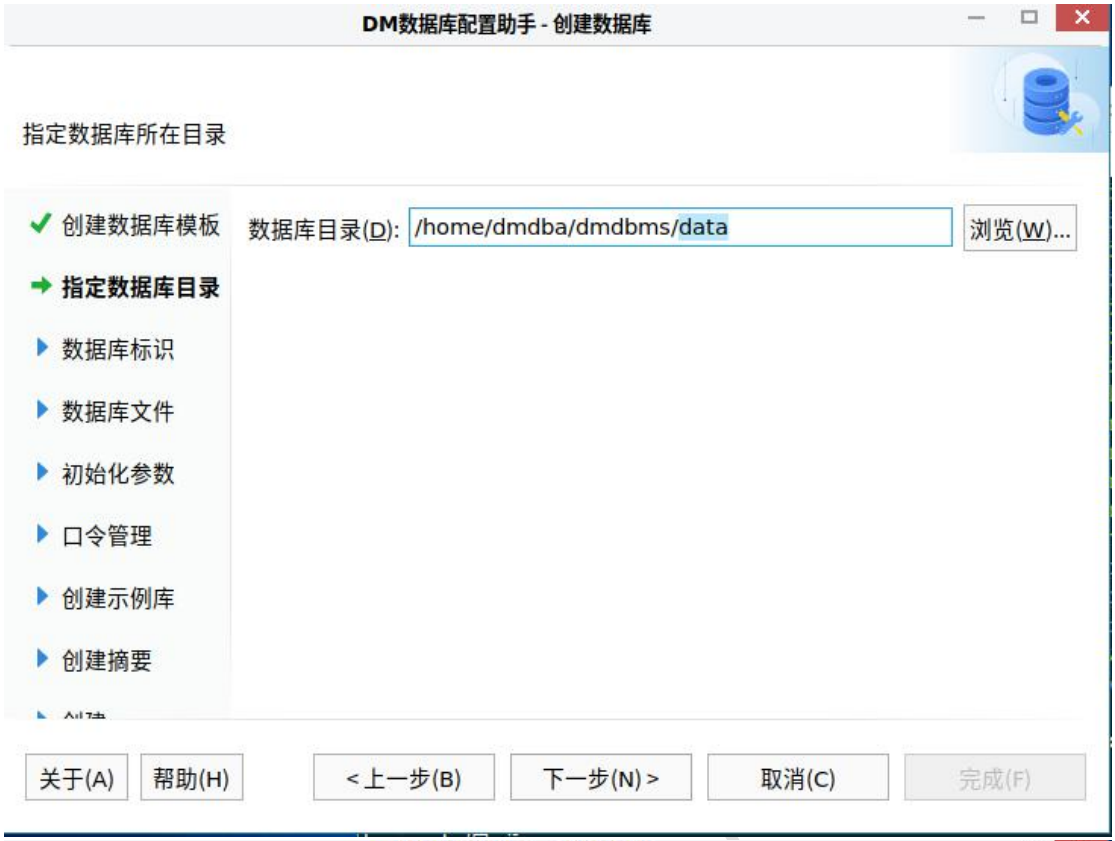
[dmdba@localhost tool]$ ls

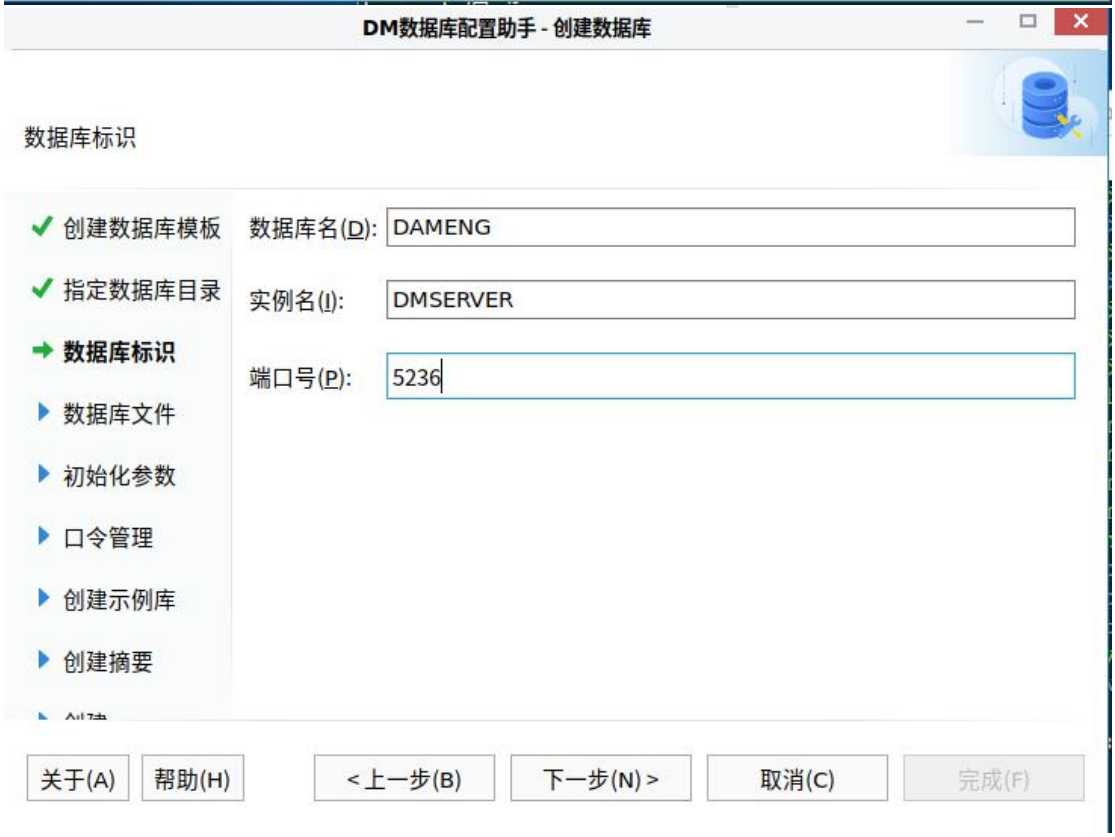
调用配置助手创建数据库实例

[dmdba@localhost tool]$ ./dbca.sh


达梦数据库配置助手
欢迎使用达梦数据库配置助手
这个向导可以指引您创建或删除数据库
创建数据库实例(C)
删除数据库实例(D)
注册数据库服务(R)
删除数据库服务(X)
取消(F)
开始(S)


DM数据库配置助手 - 创建数据库
创建数据库模板
创建数据库模板
指定数据库目录
数据库标识
数据库文件
初始化参数
口令管理
创建示例库
创建摘要
请选择指定的数据库模板:
一般用途(O)
联机分析处理(L)
联机事务处理(T)
自动调整数据库性能参数(X)
高性能(P)
平衡(V)
关于(A)
帮助(H)
下一步(N) >
取消(C)
完成(F)


DM数据库配置助手 - 创建数据库
指定数据库所在目录
创建数据库模板
指定数据库目录
数据库标识
数据库文件
初始化参数
口令管理
创建示例库
创建摘要
数据库目录(D): /home/dmdba/dmdbms/data
浏览(W)...
关于(A)
帮助(H)
< 上一步(B)
下一步(N) >
取消(C)
完成(F)


DM数据库配置助手 - 创建数据库
数据库标识
创建数据库模板
指定数据库目录
数据库标识
数据库文件
初始化参数
口令管理
创建示例库
创建摘要
数据库名(D): DAMENG
实例名(I): DMSERVER
端口号(P): 5236
关于(A)
帮助(H)
< 上一步(B)
下一步(N) >
取消(C)
完成(F)

可能需要设置簇大小、页大小、字符集等参数

考试时一定要按要求设置数据库管理员 SYSDBA 的密码,Dameng123

切换到 root 账号下，执行三个脚本文件

[dmdba@localhost 桌面]$ su -

[root@localhost ~]# mv /home/dmdba/dmdbms/bin/DmServiceDMSERVER.service /usr/lib/systemd/system/DmServiceDMSERVER.service

[root@localhost ~]# systemctl enable DmServiceDMSERVER.service

[root@localhost ~]# systemctl start DmServiceDMSERVER.service

2、数据库实例管理

2.1 连接数据库

[dmdba@localhost 桌面]$ cd /home/dmdba/dmdbms/tool/

[dmdba@localhost tool]$ ./manager

查看当前数据库连接用户

select user;

2.2 启动和关闭数据库

（1）数据库的状态

Shutdown ：数据库的关闭状态

Mount：数据库的配置状态，配置数据库归档模式、集群等

Open：数据库的打开状态，数据库能读写访问

查看当前数据库实例状态：

select status$ from v$instance;

（2）数据库启动和关闭

1）通过后台进程方式启动和关闭数据库：

进入到数据库服务所在目录

[dmdba@localhost 桌面]$ cd /home/dmdba/dmdbms/bin

查看数据库服务名：

[dmdba@localhost bin]$ ll DmS*

查看数据库是否在运行中

[dmdba@localhost bin]$ ./DmServiceDMSERVER status

关闭数据库

[dmdba@localhost bin]$ ./DmServiceDMSERVER stop

启动数据库

[dmdba@localhost bin]$ ./DmServiceDMSERVER start

2）通过服务查看器方式启动和关闭数据库

[root@localhost ~]# cd /home/dmdba/dmdbms/tool/

[root@localhost tool]# ./dmservice.sh

3、DMSQL

SQL 是结构化的查询语言。是应用和数据库交互的接口。DMSQL 是基于 SQL92,SQL99 标准。

3.1DMSQL 分类：

DDL：数据定义语言，包括 create table,alter table,drop table,create view,create index 等。

DML：数据操纵语言，包括 select,insert,update,delete。

DCL：数据控制语言，包括 grant,revoke。

TCL：事务控制语言，包括 commit,rollback。

3.2SQL 语句书写规范

SQL 语言大小写不敏感，字符串大小写敏感

SQL 可以写在一行或者多行

关键字不能被拆分、缩写

SQL 以;结束

3.3 简单查询

语法：

Select () from ();

第一个括号可以接：单列查询、多列查询、全列查询、去重查询、对列起别名、表达式、连接列或字符串

第二个括号就是表

单列查询

select employee_name from dmhr.EMPLOYEE;

多列查询

select employee_name,DEPARTMENT_ID from dmhr.EMPLOYEE;

全列查询

select * from dmhr.EMPLOYEE;

去重查询

select DISTINCT department_id from dmhr.EMPLOYEE;

对列起别名：

select EMPLOYEE_NAME,salary gz from dmhr.EMPLOYEE;

select EMPLOYEE_NAME,salary as gz from dmhr.EMPLOYEE;

接表达式

select EMPLOYEE_NAME,salary,salary+500 gz from dmhr.EMPLOYEE;

连接列 ||

….的工资是….

select employee_name||'的工资是：'|| salary from dmhr.EMPLOYEE;

3.4 排序查询

关键字 Order by，放在 SQL 语句的句尾

升序：asc

数字类型：从小到大

日期时间类型：从远到近

select employee_name,SALARY from dmhr.EMPLOYEE order by salary;

select employee_name,SALARY from dmhr.EMPLOYEE order by salary asc;

select employee_name,SALARY from dmhr.EMPLOYEE order by 2;

select employee_name,SALARY from dmhr.EMPLOYEE order by 2 asc;

降序：desc

数字类型：从大到小排序

日期时间类型：从近到远排序

select employee_name,SALARY from dmhr.EMPLOYEE order by salary desc;

select employee_name,SALARY from dmhr.EMPLOYEE order by 2 desc;

3.5 分组汇总查询

3.5.1 常见的聚合函数

Count：统计计数

Sum：求和

Max：最大值

Min：最小值

Avg：平均值

select count(*) from dmhr.EMPLOYEE;

select sum(salary) from dmhr.EMPLOYEE;

select max(salary) from dmhr.EMPLOYEE;

select min(salary) from dmhr.EMPLOYEE;

select avg(salary) from dmhr.EMPLOYEE;

3.5.2 分组函数

（1）group by

除了聚合函数出现的列，不用跟在 group by 后面，其他查询的列都要

放在 group by 后面。

语法：

Select () from () where () group ();

Select () from () group by ();

统计每个部门有多少个人

```
select department_id,count(*)    from dmhr.EMPLOYEE
group by DEPARTMENT_ID;
```

求每个部门的工资总和

```
select department_id,sum(salary) from dmhr.EMPLOYEE
group by DEPARTMENT_ID;
```

（2）having

Having 是对 group by 进一步的过滤，没有 group by 不能 having，having 不能单独使用。

语法：

Select () from () group by () having ();

统计部门人数超过 20 个人的。

```
select department_id,count(*)    from dmhr.EMPLOYEE
group by DEPARTMENT_ID
having count(*) >20;
```

统计部门人数超过 20 个人，部门人数降序显示。

```
select department_id,count(*)    from dmhr.EMPLOYEE
group by DEPARTMENT_ID
having count(*) >20
ORDER BY COUNT(*) DESC;
```

查找部门平均工资超过 9000 的有哪些部门，按部门平均工资降序显示。

```
select department_id,avg(salary) from dmhr.EMPLOYEE
group by DEPARTMENT_ID
having avg(salary) > 9000
order by avg(salary) desc;
```

3.6 过滤查询

（1）比较运算符

= > >= < <= <>

```
select employee_name,department_id from dmhr.EMPLOYEE where DEPARTMENT_ID=101;

select employee_name,department_id,salary from dmhr.EMPLOYEE where salary>=18000 ;
```

Between … and 例如 between 30 and 50 取值范围就是 30-50 之间的值，并且包含 30 和 50

select employee_name,department_id,salary from dmhr.EMPLOYEE where salary BETWEEN 18000 AND 25000;

等价于

select employee_name,department_id,salary from dmhr.EMPLOYEE where salary>= 18000 AND salary <=25000;

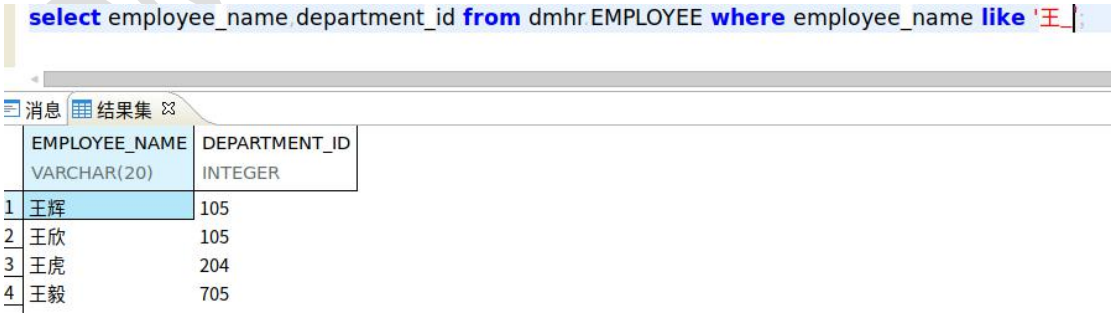
In()举，与列表中的任意一个值进行匹配

select employee_name,department_id,salary from dmhr.EMPLOYEE where department_id in (101,103);

Like 模糊查询

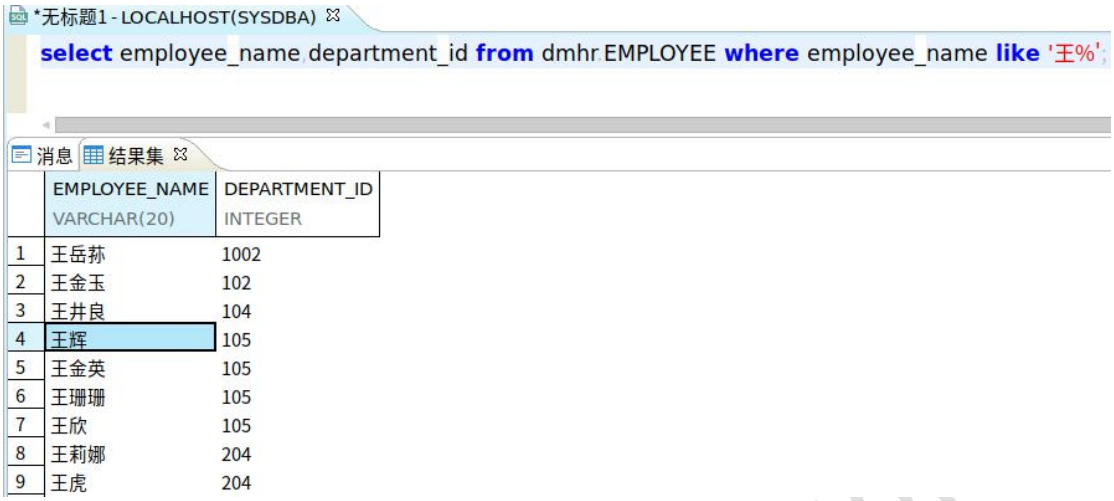
Is null/is not null 判断一个列上的值是否为空，null 就是空，不等于 0 也不等于任何值。

Like_ 匹配 1 个字符



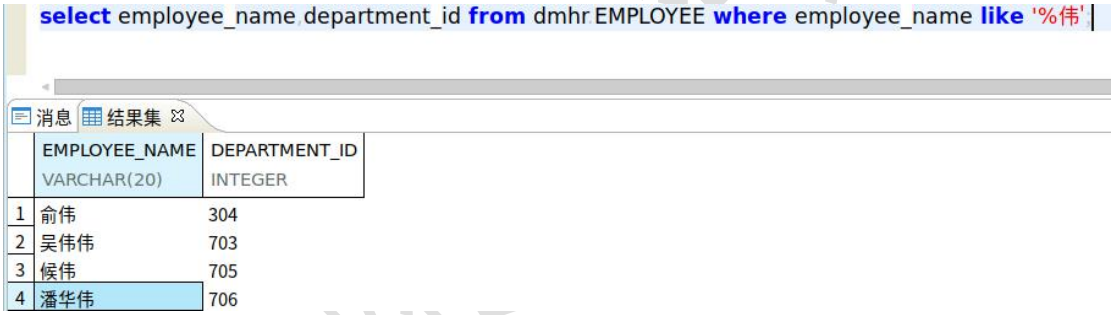
Like % 匹配 0 个或多个字符

查找以“王”开头的



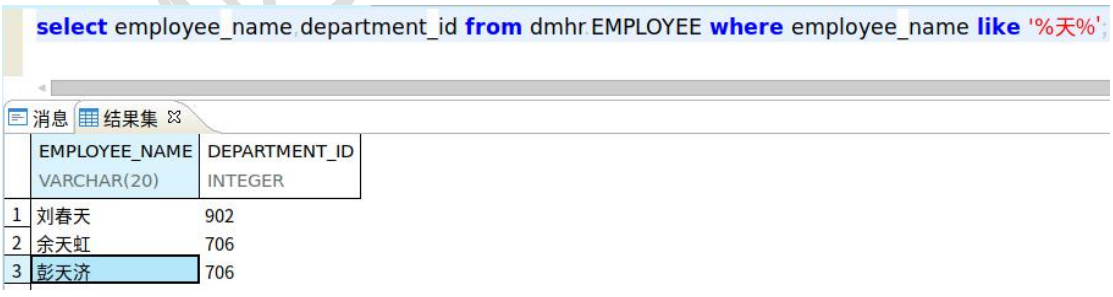
	EMPLOYEE_NAME VARCHAR(20)	DEPARTMENT_ID INTEGER
1	王岳荪	1002
2	王金玉	102
3	王井良	104
4	王辉	105
5	王金英	105
6	王珊珊	105
7	王欣	105
8	王莉娜	204
9	王虎	204

查找以“伟”结尾的



	EMPLOYEE_NAME VARCHAR(20)	DEPARTMENT_ID INTEGER
1	俞伟	304
2	吴伟伟	703
3	候伟	705
4	潘华伟	706

查找包含“天”的



	EMPLOYEE_NAME VARCHAR(20)	DEPARTMENT_ID INTEGER
1	刘春天	902
2	余天虹	706
3	彭天济	706

Is null/is not null

select employee_name,department_id ,job_id from dmhr.EMPLOYEE where job_id is null;

```
select employee_name,department_id ,job_id from dmhr.EMPLOYEE
where job_id is not  null;
```

（2）逻辑运算符

and ：同时满足所有的条件，就返回数据

查找部门编号是 101 的并且员工工资大于等于 30000.

```
select employee_name,salary,department_id from dmhr.EMPLOYEE
where department_id=101 and SALARY >=30000;
```

Or ：满足任意 1 个条件就返回数据

查找部门编号是 101 的或者员工工资大于等于 30000.

```
select employee_name,salary,department_id from dmhr.EMPLOYEE
where department_id=101 or SALARY >=30000;
```

Not：条件为假时就返回数据

3.7 连接查询

连接查询可以分为：内连接查询和外连接查询。

内连接查询：返回满足条件的数据

外连接查询：除了返回满足条件的数据外，不满足的也返回用 null 代替。

3.7.1 内连接查询

语法：

SELECT *table1.column, table2.column*

FROM *table1*

[CROSS JOIN *table2*] |

[NATURAL JOIN *table2*] |

[JOIN *table2* USING (*column_name*)] |

[JOIN *table2*

ON(*table1.column_name* = *table2.column_name*)];

（1）交叉连接

语法：

SELECT *table1.column, table2.column*

FROM *table1*

CROSS JOIN *table2*;

交叉连接也叫笛卡集，是表的乘积，一定要避免

select count(*) from dmhr.EMPLOYEE CROSS JOIN dmhr.DEPARTMENT;

等价于

select count(*) from dmhr.DEPARTMENT ,DMHR.EMPLOYEE;

（2）自然连接

语法：

SELECT *table1.column, table2.column*

FROM *table1*

NATURAL JOIN *table2*;

自然连接查询的表，需要具有相同的列名并且相同的数据类型

select a.EMPLOYEE_NAME,a.SALARY,b.DEPARTMENT_NAME from dmhr.EMPLOYEE a NATURAL JOIN dmhr.DEPARTMENT b;

（3）USING

语法：

SELECT *table1.column, table2.column*

FROM *table1*

JOIN *table2* USING (*column_name*);

select a.EMPLOYEE_NAME,DEPARTMENT_ID,b.DEPARTMENT_NAME from

dmhr.EMPLOYEE a JOIN dmhr.DEPARTMENT b

using (department_id);

（4）ON 必须要掌握

SELECT *table1.column, table2.column*

FROM *table1*

JOIN *table2*

ON(table1.column_name = table2.column_name);

select a.EMPLOYEE_NAME,b.DEPARTMENT_ID,b.DEPARTMENT_NAME
from
dmhr.EMPLOYEE a JOIN dmhr.DEPARTMENT b
on a.DEPARTMENT_ID=b.DEPARTMENT_ID;

等价于

select a.EMPLOYEE_NAME,b.DEPARTMENT_ID,b.DEPARTMENT_NAME
from
dmhr.EMPLOYEE a , dmhr.DEPARTMENT b
where a.DEPARTMENT_ID=b.DEPARTMENT_ID;

3.7.2 外连接查询

语法：

SELECT table1.column, table2.column
FROM table1
[LEFT|RIGHT|FULL OUTER JOIN table2
ON (table1.column_name = table2.column_name)];

构造数据：

create table dmhr.test1 (id int,name varchar(30));
create table dmhr.test2 (name varchar(30),addr varchar(50));

```
insert into dmhr.test1 values(1,'AAA');
insert into dmhr.test1 values(null,'BBB');
insert into dmhr.test1 values(3,'CCC');
insert into dmhr.test1 values(null,'DDD');

insert into dmhr.test2 values('AAA','dkajkdakdkdk');
insert into dmhr.test2 values('BBB',null);
insert into dmhr.test2 values('CCC',null);
insert into dmhr.test2 values('DDD','dakzzzzzz');
insert into dmhr.test2(name) values('ZZZ');
commit;
```

（1）左外连接查询

返回左边表的全部记录，右边表返回满足条件的记录，不满足的就用 null 代替。

```
select a.ID,b.ADDR from dmhr.TEST1 a left OUTER join dmhr.test2 b
on a.NAME=b.NAME;
```

（2）右外连接查询

返回右边表的全部记录，左边边表返回满足条件的记录，不满足的就用 null 代替。

select a.ID,b.ADDR from dmhr.TEST1 a right outer join dmhr.test2 b

on a.NAME=b.NAME;

（3）全外连接查询

左外连接+右外连接

select a.ID,b.ADDR from dmhr.TEST1 a full outer join dmhr.test2 b

on a.NAME=b.NAME;

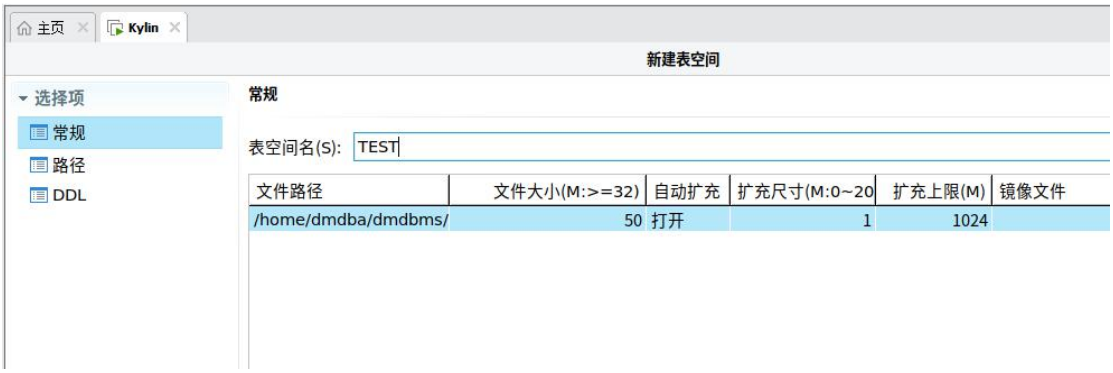
4、表空间管理

4.1 查询表空间

select * from dba_tablespaces;

4.2 创建表空间

创建表空间 TEST，数据文件 /home/dmdba/dmdbms/data/DAMENG/TEST01.DBF,初始大小为 50M，开启自动扩展，每次扩展 1M，最大可扩展至 1G.




```
create tablespace "TEST" datafile '/home/dmdba/dmdbms/data/DAMENG/TEST01.DBF' size 50 autoextend on next 1 maxsize 1024 CACHE = NORMAL;
```

需要注意的地方：表空间名需要大写，数据文件路径及名称，自动扩充要选择打开，扩充上线需要注意，1G=1024M,10G=10240M，达梦数据库单位默认就是 M.

5、用户管理

5.1 查看用户

select

username,ACCOUNT_STATUS,LOCK_DATE,EXPIRY_DATE,DEFAULT_TABLESPACE from dba_users;

USERNAME:用户名

ACCOUNT_STATUS：用户的状态，open 状态才能连接数据库

LOCK_DATE：用户的锁定时间

EXPIRY_DATE：多长时间过期，过期之后，下次登录数据库时需要更改密码

DEFAULT_TABLESPACE：用户的默认表空间

5.2 介绍 DM 用户

DM 用户可以分为：预定义用户和自定义用户。

SYS：系统用户，不能登录，存放数据库字典的信息。

SYSDBA：数据库管理员

SYSAUDITOR：数据库审计员，创建审计规则，查看审计记录

SYSSSO：数据库安全员，创建安全规则。

5.3 权限

权限可以分为：系统权限和对象权限

系统权限

权限名称	说明
CREATE TABLE	创建表
ALTER DATABASE	修改数据库
CREATE TABLESPACE	创建表空间
ALTER TABLESPACE	修改表空间
DROP TABLESPACE	删除表空间
CREATE USER	创建用户
CREATE VIEW	创建视图
CREATE PROCEDURE	创建存储过程/函数
CREATE ROLE	创建角色
CREATE SCHEMA	创建模式

对象权限

对象权限	表	视图	序列	过程
ALTER	√		√	
DELETE	√	√		
EXECUTE				√
INSERT	√	√		
REFERENCES	√	√		
SELECT	√	√	√	
UPDATE	√	√		

CREATE TABLE：创建表

CREATE VIEW：创建视图

CREATE INDEX：创建索引

5.3 角色

角色就是一组权限的集合，使用角色为了方便权限管理。角色可以被授予给角色，也可以被授予给用户。

5.4 创建用户

规划用户时应该考虑到：

用户的命名：以字母开头，长度不超过 128 个字符

密码：

权限：系统权限、对象权限、角色

用户存储位置：表空间

资源配置：失败登录次数，密码锁定，口令有效期等

例 1，创建 TEST 用户，密码是 Dameng123，默认表空间是 TEST，该用户拥有创建表、创建视图的权限。

```
CREATE USER "TEST" IDENTIFIED BY Dameng123 DEFAULT TABLESPACE "TEST";
grant CREATE TABLE,CREATE VIEW  to "TEST";
```

例 2，创建用户 DMTEST，密码是 Dameng123,拥有创建表，创建视图，创建索引的权限，该用户在登录失败 5 次后锁定 3 分钟，密码在 180 天之后自动过期。

```
create user "DMTEST" identified by "Dameng123"
limit FAILED_LOGIN_ATTEMPS 5 PASSWORD_LIFE_TIME 180 PASSWORD_LOCK_TIME 3;
grant CREATE TABLE,CREATE VIEW,CREATE INDEX to "DMTEST";
```

5.5 创建角色

创建角色 ROLE1，拥有创建表和创建视图的权限，可以查看 dmhr.employee.employee_name,dmhr.employee.hire_date,dmhr.employee.email，dmhr.department 可以修改列 dmhr.employee.email

```
create role "ROLE1";
grant CREATE TABLE,CREATE VIEW to "ROLE1";
```

grant SELECT on "DMHR"."DEPARTMENT" to "ROLE1";

grant SELECT("HIRE_DATE") on "DMHR"."EMPLOYEE" to "ROLE1";

grant SELECT("EMAIL"),UPDATE("EMAIL") on "DMHR"."EMPLOYEE" to "ROLE1";

grant SELECT("EMPLOYEE_NAME") on "DMHR"."EMPLOYEE" to "ROLE1";

需要注意：角色名一定要大写

5.6 修改用户

---修改用户的密码

alter user "SYSDBA" identified by "dameng123";

---锁定/解锁用户

alter USER TEST ACCOUNT LOCK;

alter USER TEST ACCOUNT UNLOCK;

---修改用户的默认表空间

alter user "DMTEST" default tablespace "TEST";

---将 ROLE1 角色授予给 TEST 用户

grant "ROLE1" to "TEST";

---删除用户

drop user "DMTEST";

6、模式对象管理

6.1 模式

模式就是一组数据库对象的集合。

模式对象：表、视图、约束、索引、存储过程、函数、包、序列、同义词等。

模式与用户的关系：一个用户可以拥有多个模式，一个模式只能属于一个用户。新建用户时，会自动创建同名的模式。

访问其他模式下面的表，不想添加模式名？

切换模式

set SCHEMA DMHR;

select * from employee;

6.2 表

DM 支持表的类型：索引组织表、堆表、外部表、分区表

DM 数据库默认表的类型是索引组织表。

6.2.1 创建表

规划表应该考虑到：

表的命名:以字母开头，长度不超过 128 个字符

数据类型：int,number,char,varchar,varchar2,clob,text,date,time,datetime,timestamp,blob,binary,varbinary 等。

约束：非空约束、唯一约束、主键约束、外键约束、检查约束

存储位置：表空间

（1）非空约束

```
create table test.TEST1 (id int not null);

insert into test.test1 (id) values(1);

insert into test.test1 values(2);

Commit;---事务提交
```

注意：对表的数据做完 insert,update,delete 后一定要记得提交，不提交等于白做。

（2）唯一约束

唯一约束要求数据唯一，创建唯一约束会自动创建唯一索引

```
CREATE TABLE "TEST"."TEST2"
(
"ID" INT,
"NAME" VARCHAR(50),
UNIQUE("NAME")) STORAGE(ON "MAIN", CLUSTERBTR) ;
```

```
insert into test.test2 values(1,'AAA');
insert into test.test2 values(2,'BBB');
insert into test.test2 values(3,'CCC');
insert into test.test2 values(4,NULL);
insert into test.test2 values(5,NULL);
insert into test.test2 values(6,'AAA');
```

消息

```
insert into test.test2 values(4,NULL);
执行成功, 执行耗时0毫秒. 执行号:1728
影响了1条记录

[执行语句5]:
insert into test.test2 values(5,NULL);
执行成功, 执行耗时1毫秒. 执行号:1729
影响了1条记录

[执行语句6]:
insert into test.test2 values(6,'AAA');
执行失败(语句6)
-6625: 违反表[TEST2]唯一性约束条件[CONS134218871]
```

原因：唯一约束会忽视 null，因为 null 什么都不是，不等于任何值。

（3）主键约束（必考）

主键约束就是非空+唯一，一个表只能有一个主键

```
CREATE TABLE "TEST"."TEST3"
(
"ID" INTEGER NOT NULL,
"NAME" VARCHAR(50),
NOT CLUSTER PRIMARY KEY("ID")) STORAGE(ON "TEST", CLUSTERBTR) ;
```

（4）外键约束（必考）

外键约束也叫参照约束或者完整性约束

```
CREATE TABLE "TEST"."TEST4"
(
"ID" INTEGER,
"NAME" CHAR(10) NOT NULL,
NOT CLUSTER PRIMARY KEY("NAME"),
FOREIGN KEY("ID") REFERENCES "TEST"."TEST3"("ID")) STORAGE(ON
"TEST", CLUSTERBTR) ;

insert into test.test3 values(2,'CCC');
insert into test.test4 values(2,'BBB');

delete from test.test4 where id=1;
delete from test.test3  where id=1;
```

注意：有主外键约束关系的表，在插入数据时，要先插入主键约束所在表的数据，再插入外键约束所在表的数据；在删除数据时，要先删除外键约束所在表的数据，再删除主键约束所在表的数据。

（5）检查约束

```
CREATE TABLE "TEST"."TEST5"
```

(

"ID" INT,

CHECK(ID > 8)) STORAGE(ON "TEST", CLUSTERBTR) ;

更新数据：

update test.test2 set name='FFF' where id=4;

6.2.2 修改表

增加列

alter table "TEST"."TEST1" add column("ADDR" VARCHAR(100));

修改列

alter table "TEST"."TEST1" modify "ADDR" VARCHAR(50);

对列重命名

alter table test.test1 rename COLUMN addr to addr1;

对表重命名

alter table test."test1" rename to TEST1;

删除列

alter table "TEST"."TEST1" drop column "ADDR";

删除表：

drop table "TEST"."TEST1" restrict;

6.3 视图

创建视图语法：

Create view 模式名.视图名 as select .....;

create view test.view_name as select

a.EMPLOYEE_NAME,a.SALARY,b.DEPARTMENT_ID,b.DEPARTMENT_NAM

E from dmhr.EMPLOYEE a ,dmhr.DEPARTMENT b

where a.DEPARTMENT_ID=b.DEPARTMENT_ID

and a.DEPARTMENT_ID=101;

删除视图：

drop view "TEST"."VIEW_NAME" restrict;

6.4 索引

create index "DMHR"."IND_NAME" on "DMHR"."EMPLOYEE"("EMPLOYEE_NAME") ;

创建索引需要注意：索引名大写，索引列是否正确。

7、备份还原

7.1 备份还原的概念

（1）备份：

备份就是数据库在某个时间点做的副本。

备份的目的就是为了防止数据丢失、数据库损坏。

备份可以分为：冷备和热备。冷备也叫脱机备份，是数据库在关闭的状态下做的备份；热备也叫联机备份，是数据库在运行状态下产生的备份。

备份可以分为：完全备份和增量备份。完全备份，备份包含整个库/整个表空间的所有数据；增量备份，在上一次完全备份或上一次增量备份后，以后每次只备份修改过的文件。

备份可以分为：逻辑备份和物理备份。逻辑备份，备份的就是数据库的对象，包括表、视图、约束、索引、存储过程、函数、包、序列、同义词、触发器等；物理备份，备份的是数据库文件。

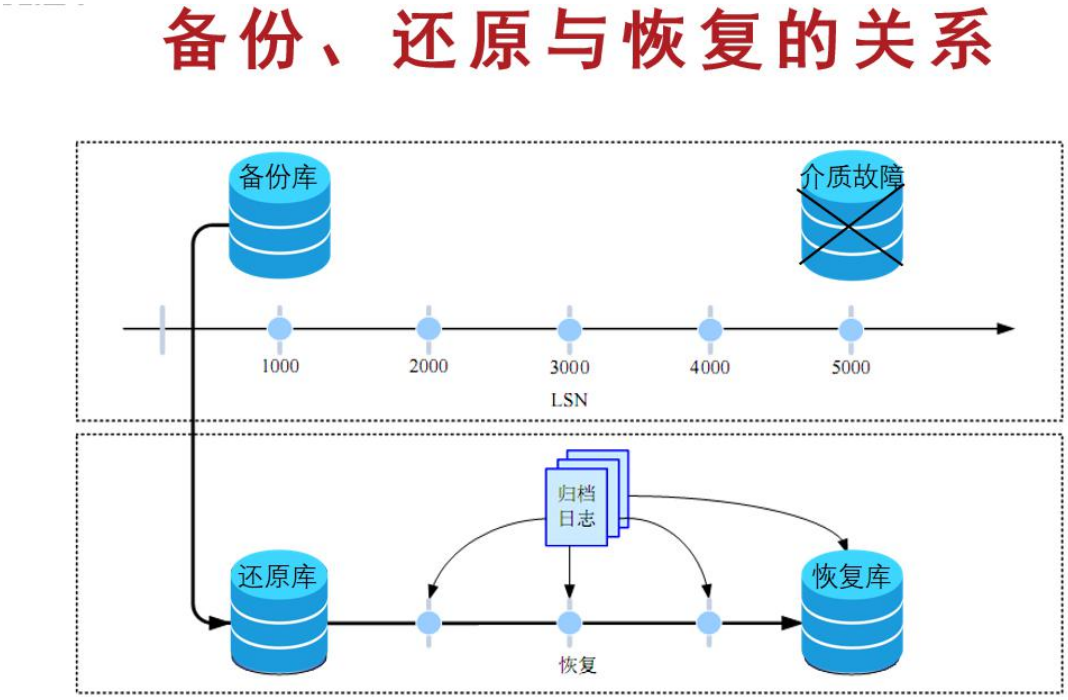
（2）还原

读取备份文件的信息，校验备份数据，解密、解压缩重新生成数据库文件并拷贝到数据库对应的路径。

（3）恢复

利用数据库的归档日志（记录的是数据库发生变更的信息）将数据库

恢复到指定的状态。



7.2 冷备

创建备份路径：

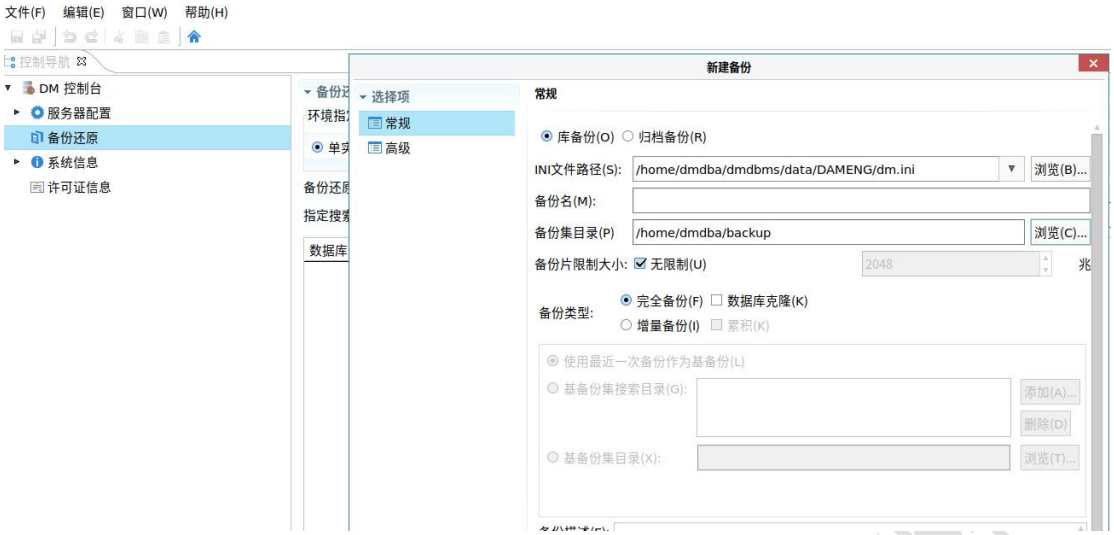
[dmdba@localhost bin]$ mkdir -p /home/dmdba/backup

需要关闭数据库

[dmdba@localhost DAMENG]$ cd /home/dmdba/dmdbms/bin

[dmdba@localhost bin]$ ./DmServiceDMSERVER stop

[dmdba@localhost tool]$ ./console



7.3 热备（必考）

数据库热备的必要条件：

数据库打开

数据库开启归档模式

启动数据库：

[dmdba@localhost backup]$ cd /home/dmdba/dmdbms/bin

[dmdba@localhost bin]$ ./DmServiceDMSERVER start

开启数据库的归档模式：

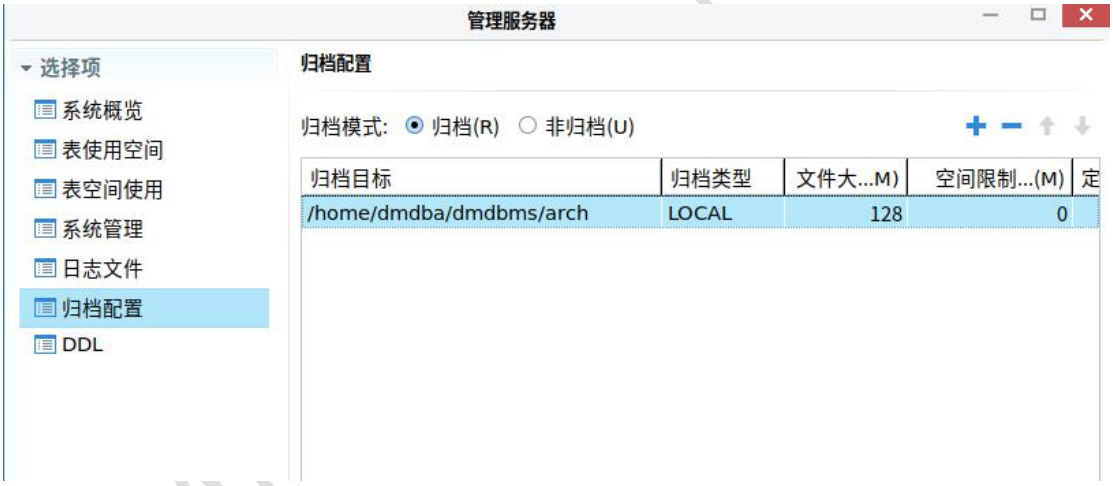
创建归档路径：

[dmdba@localhost bin]$ mkdir -p /home/dmdba/dmdbms/arch

将数据库切换到配置状态：

设置数据库的归档属性：

归档路径，归档文件大小，归档空间限制



将数据库切换到打开状态

查看数据库的状态和归档模式：

将数据库备份到默认路径

/home/dmdba/dmdbms/data/DAMENG/bak

将表 dmhr.employee 备份到默认路径

注意：在做整库备份和表备份时，不需要选择路径，默认路径即可，选择路径会报错：备份目录冲突。

7.4 逻辑备份

逻辑备份一共有 4 个级别：full,owner,schemas,tables

FULL:整库导出

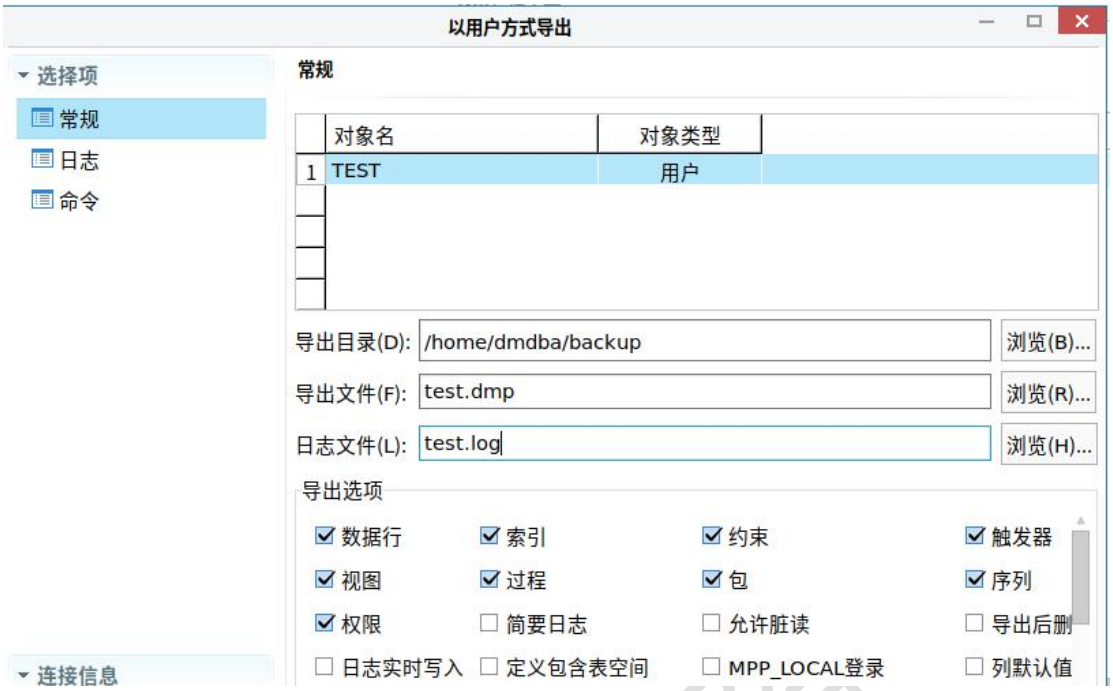
OWNER:用户级别导出，可以一次性导出一个或多个用户

SCHEMAS:模式级别导出，可以一次导出一个或多个模式

TABLES:表级别导出

（1）整库导出

（2）用户级别导出



（3）模式级别导出

（4）表级别导出

8、作业管理

作业就是类似于服务器的计划任务，能定时的去做某些事情。

新建作业 JOB1，每周一、二、三、四、五、六 22：00 对数据库做完全备份。


新建作业
选择项
常规
作业步骤
作业调度
DDL
常规
作业名(J):
JOB1
作业描述(D):
通知:
邮件发送(I):
当作业执行成功时
网络发送(T):
当作业执行成功时
注意:
1.如果使用电子邮件通知请确保服务器上的防火墙设置能够保证电子邮件的发送成功;
2.网络发送功能只有在Windows操作系统下才会执行，同时必须开启服务Messager才能发送成功。


新建作业步骤
选择项
常规
高级
连接信息
常规
作业名称(J):
JOB1
步骤名称(N):
B1
步骤类型(T):
备份数据库
备份路径(P):
打开(O)
备份并行数(R):
(0~9)
备份片大小(S):
(0~9)
读速度上限(U):
(M/s)
写速度上限(W):
(M/s)
备份方式:
完全备份(F)
差异增量备份(I)
归档备份(B)
累计增量备份
压缩级别(E):
(1~9)
压缩(C)
备份日志(L)
基备份目录

注意：作业名一致并大写，作业类型，作业调度时间。

新建作业 JOB2，每周日 22:00 对数据库做增量备份。

9、考点

创建数据库，创建表空间，创建用户及分配权限，创建表，数据增删改，创建视图，创建索引，物理备份（热备），逻辑备份，作业管理。