一、达梦数据库安装

1 创建dmdba用户

root用户执行

```
1. 创建用户组：groupadd dinstall
2. 创建用户：useradd -g dinstall -m -d /home/dmdba -s /bin/bash dmdba
3. 修改用户密码：passwd dmdba
```

2 创建安装目录并赋予目录权限

root用户执行

```
1. mkdir /dm8
2. chown dmdba:dinstall /dm8
```

3 挂载iso

root用户执行

```
1. mkdir /mnt/dm
2. mount XXX.iso /mnt/dm
```

4 设置系统打开文件数

root用户执行

```
vim /etc/security/limit.conf

* soft nofile 102400
* hard nofile 204800
```

5 安装

dmdba用户执行

图形化安装

1. 设置图形化界面：

```
(1) xhost+
(2) echo $DISPLAY
(3) su - dmdba
(4) export DISPLAY=XX
```

2. DMInstall.bin

命令行安装

1. su dmdba
2. DMInstall.bin

静默安装

主要用于自动化部署使用，需要配置XML文件
DMInstall.bin -q [配置文件路径]

常见安装问题

1. 安装时tmp不足：

```
(1) mkdir /opt/tmp
(2) chown dmdba:dinstall /opt/tmp
(3) su -dmdba
(4) export DM_INSTALL_TMPDIR=/opt/tmp
```

2. SWT类的错误：
   （1）首先root执行xhost
   （2）设置DISPLAY环境变量

二、数据库实例创建、注册、连接、启动

数据库实例创建与注册

图形化创建

1. 设置DISPLAY，同上
2. tool/dbca.sh

命令行创建

1. bin/dminit PATH=[初始化数据库存放路径] DB_NAME=[数据库名字] INSTANCE_NAME=[实例名] SYSDBA_PWD=[管理员密码] PORT_NUM=[实例端口] PAGE_SIZE=[页大小]

注意：

- 命令行创建不会自动注册，需要手动注册
  - 通过图形界面注册数据库服务
  - root用户执行script/root/dm_service_install.sh进行注册

创建实例注意事项

1. 数据库簇和页大小的问题：
   （1）簇是最小分配单元，页是最小存储单元
   （2）页的大小会影响VARCHAR的长度（VARCHAR长度不能超过页大小的一半）

数据库实例连接

图形化连接

1. 设置DISPLAY，同上
2. tool/manager

命令行连接

1. bin/disql [用户名]/[密码]:[port]
2. 通过网络配置助手（tool/nca.sh）可以对ip和port进行网络服务配置，方便disql连接：disql [用户名]/[密码]@[网络服务名]

数据库启动

- 达梦数据库有4种状态：shutdown、mount、open、suspend
    - mount：配置状态，可以对数据库的配置进行修改，进行归档、主备等模式的配置，但是不能进行数据文件的读写（只可以读取内存中/控制文件中的表，即V$开头的动态视图）
    - open：打开状态，正常进行服务
    - suspend：只读状态（DML操作一旦commit，数据库实例就会被挂起阻塞）
- 在数据库启动时：
    - 首先寻找配置ini文件，寻找数据库控制文件（.ctl）
    - 基于控制文件，启动进程和线程，启动数据库实例，进入mount状态
    - 再根据配置文件找到数据文件(.DBF)路径和日志文件(.log)路径，打开数据文件和联机日志，进入open状态
- 启动方式：
    - root用户：systemctl启动服务或者使用tool/dmservice.sh
    - dmdba用户：
        - 前台启动：bin/dmserver [dm.ini文件路径]
        - 后台启动：bin/DmServiceXXXX start

三、常用客户端工具

所有的图形化工具都在tool目录下：

1. 管理工具：manager，联机工具
2. 控制台工具：console，是一个脱机工具，主要用于备份欢迎和数据库参数修改
3. 迁移工具：dts，迁移使用
4. 性能监视工具：monitor，监控线程、空间占用等

四、DMSQL

SQL分类

- **DML（Data Manipulation Language）数据操纵语言**：对数据表中的记录行进行操作，SELECT、INSEART、DELETE、UPDATE、MERGE，默认不自动提交，需要手动进行commit和rollback，对数据的操作会产生undo和redo
- **DDL（Data Definition Language）数据定义语言**：对数据表对象进行操作，CREATE、ALTER、DROP、TRUNCATE、COMMENT，默认自动提交
- **DCL（Data Control Language）数据控制语言**：权限的赋予与收回，GRANT、REVOKE
- **TCL（Transactional Control Language）事物控制语言**：维护数据一致性，COMMIT、ROLLBACK、SAVEPOINT

注意：
DELETE和TRUNCATE的区别：

- DELETE是DML语言，需要手动提交，会产生undo和redo，会记录回滚段支持闪回查询（不立即释放空间，DM中系统自动清理回滚页，时间参数由undo_retention指定），可以带where条件，在删除大量数据时较慢
- TRUNCATE是DDL语言，自动提交，只能清空表中所有数据，立即释放空间，可以用于降低表的水位线

常用SQL

见附件培训讲义中SQL章节内容

GROUP BY和HAVING

```
SELECT a.id, count(*) FROM employee a GROUP BY a.id HAVING count(*) > 10
```

CUBE和ROLLUP

- CUBE是全排列，比如CUBE（A,B,C）相当于对（A,B,C）(A,B)(A,C)(B,C)(A)(B)(C)都聚合
- ROLLUP是向前排列，比如ROLLUP（A,B,C）相当于（A,B,C）(A,B)(A)

```
SELECT a.id, a.dnum, count(*) FROM employee a GROUP BY CUBE(a.id,a.dnum) HAVING
count(*) > 10
SELECT a.id, a.dnum, count(*) FROM employee a GROUP BY ROLLUP(a.id,a.dnum) HAVING
count(*) > 10
```

五、DM的体系结构

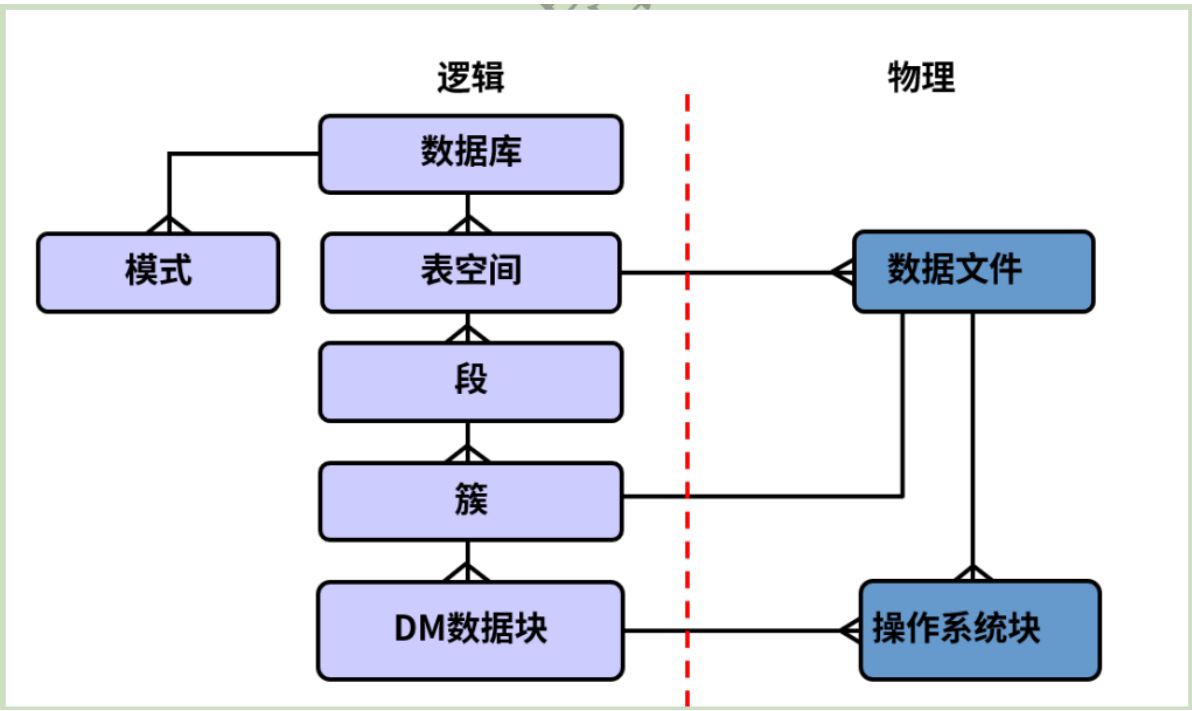
体系结构

DM数据库是由数据库和实例构成的。

- 数据库：理解为存放在磁盘上的数据的集合
- 实例：一组正在运行的DM进程/线程以及一个大型的共享内存组成
  PS：也就是说，DM体系结构由三个部分构成，即：进程/线程、存储、内存

存储结构

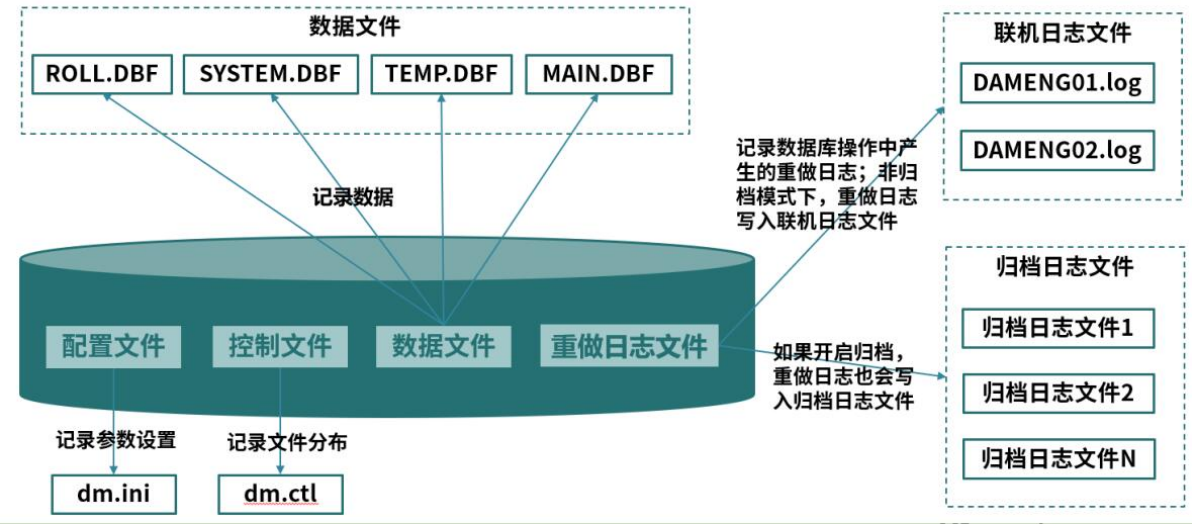
DM中，表空间是最大的存储单元，所有的数据都存储在表空间中，表空间下会划分为段=>簇=>页，分别完成数据表和数据记录的存储



逻辑单元

- 表空间：存储所有的数据，对应物理结构的数据文件（一个表空间可以有多个数据文件，一个数据文件只能归属于一个表空间），数据库实例创建后，默认生成5个表空间（SYS、TEMP、ROLL、MAIN（用户行）、HMAIN（用户列））
- 段：对应的就是数据表，大小不固定，内部就是簇和页
- 簇：实例创建时设置大小，最小的分配单元
- 页：实例创建时设置大小，对应操作系统块，最小的存储单元

DM核心的四个文件



配置文件dm.ini

dm.ini文件是达梦数据库的参数配置文件，在DM中，总共有四类参数：

- READ_ONLY：手动参数，只能通过修改dm.ini文件进行修改
- SESSION：动态参数（会话级），可以在数据库运行中直接修改（内存、文件、会话），只对当前会话生效
- SYS：动态参数（系统级），可以在数据库运行中直接修改（内存、文件）
- IN_FILE：静态参数，不能修改内存中的值，只能修改参数文件中的值（文件），重启后生效

查询参数的方法

```
select PARA_NAME,PARA_VALUE,PARA_TYPE,FILE_VALUE,SESS_VALUE from v$dm_ini where PARA_NAME like '%XXX%'
```

修改参数的方法：

- 通过console控制台工具修改，重启后才可以生效
- 修改dm.ini文件，重启后才可以生效
- 通过sql进行修改，根据情况不同，生效时机不同：

```
alter system set '[参数名]'=值 both|spfile|memory
alter session set '[参数名]'=值 （会话级参数）
```

PS：修改参数可以分为内存（memory）、文件（spfile）、全部（both）

控制文件dm.ctl

dm.ctl是一个二进制文件，可以通过strings/dmctlcvt工具转换查看，主要包含了数据文件信息（表空间及表空间对应的数据文件位置）、联机日志文件信息

数据文件xxx.DBF

数据文件用来存放数据使用，每一个表空间都会对应一个或多个数据文件

日志文件xxx.log

日志文件用于存放各类日志信息

- 联机日志：存放redo重做信息
- 归档日志
- 备份日志

内存结构

数据缓冲区

BUFFER用来缓存数据文件中的数据页，BUFFER用来缓存数据文件中的数据页，分为normal、fast、recycle、keep四种类型，由dm.ini中的BUFFER、FAST_POOL_PAGES、RECYCLE、KEEP确定大小。一般OLTP数据库的数据缓冲区占内存的**40%-60%**，在数据库执行检查点时会刷脏页，将缓冲区的脏页写入磁盘。

- 查询数据缓冲区相关参数

```
select * from v$parameter where name
in('BUFFER','FAST_POOL_SIZE','RECYCLE','KEEP')
```

- 修改数据缓冲区参数

```
alter system set 'BUFFER' = 500 spfile
```

重做日志缓冲区

DML的操作会先写入日志，然后再写入内存，最终再写入磁盘
相当于是数据库和磁盘之间的一道缓冲
对应ini参数是RLOG_BUF_SIZE
PS：commit时或日志缓冲区满时进行日志的刷盘（commit不会触发数据缓冲区的刷盘，只有在数据缓冲区满或执行检查点或系统关闭时时，数据缓冲区才刷盘）

SQL缓冲区

缓存SQL语句，执行计划和对应的结果集（需要开启结果集缓存，默认是不开的）
对应ini参数是：CACHE_POOL_SIZE

字典缓冲区

主要用于缓存系统表中的信息，权限、数据对象等元数据信息
对应ini参数是：DICT_BUF_SIZE

主内存池（共享内存池）

当其他内存池不够用时，会先向主内存池申请空间
对应ini参数是：

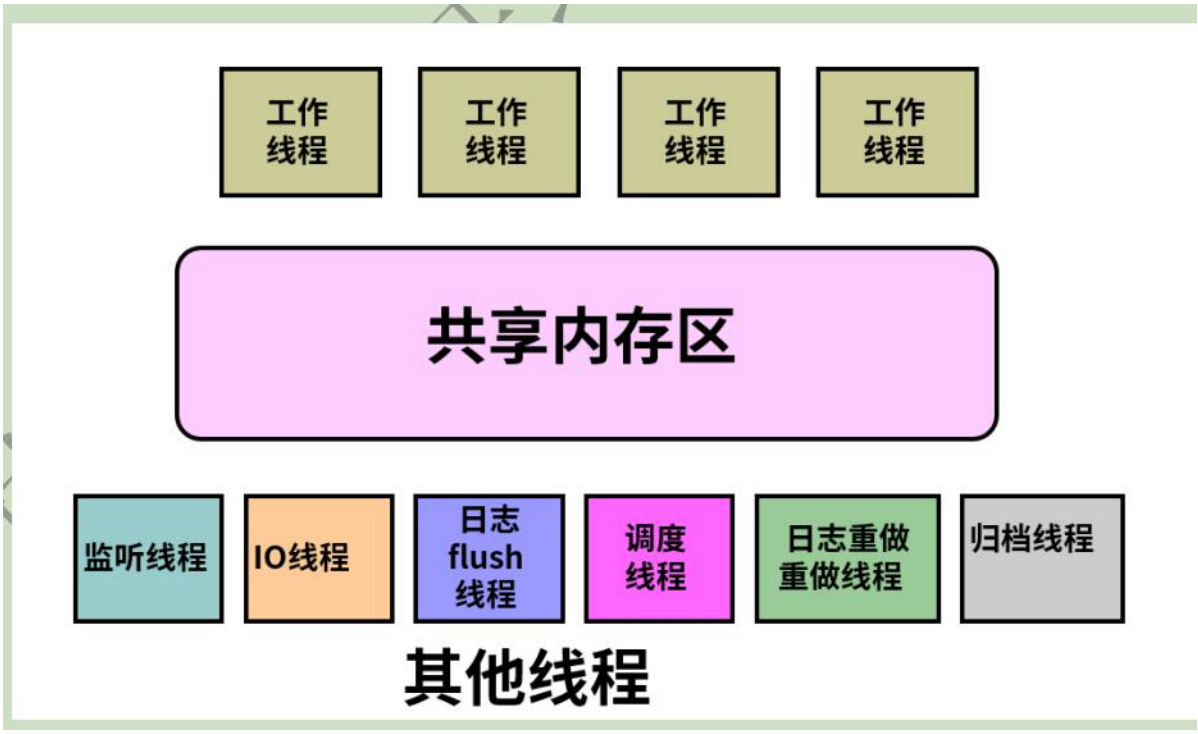
- MEMORY_POOL：初始大小
- MEMORY_EXTENT_SIZE：当需要的内存大于初始大小时，可以进行自动扩充
- MEMORY_TARGET：最大可以扩充的大小
- MAX_OS_MEMORY：内存占操作系统的比例，设置为80-90比较好

运行时内存

使用时申请，用完即释放，主要包含虚拟内存池（sql执行），会话内存池（sql执行占用），排序区（少量数据排序），hash区（HASH连接）等
PS：排序区和hash区都属于虚拟缓冲区，实际申请时使用虚拟内存池或会话内存池。

进程/线程

DM是单进程多线程的对称结构（效率高，但健壮性一般），Oracle是多进程结构（健壮性高）



进程/线程查看

- 查看数据库进程：
  - ps -ef | grep dmserver
  - 对应动态视图：select * from V$process
- 查看数据库对应的线程：
  - ps -T -p [进程pid]
  - 对应动态视图：select * from V$threads

关键线程

- 监听线程（dm_lsnr_thd）：在服务器端口监听客户端连接的，当有连接时，监听线程唤醒并生成会话线程，将申请的任务加入到工作队列中等待工作线程进行处理
- 会话线程（dm_sql_thd）：每一个数据库连接会话都对应了一个会话线程，对应的动态视图为V$sessions
- 工作线程（dm_tskwrk_thd,dm_wrkgrp_thd）：用来执行作业任务（SQL的解析和执行）的，是DM的核心线程，默认16个
- IO线程（dm_io_thd）：物理读和刷脏页
- 日志刷新/归档/重做线程
- 调度线程：定时调度其他线程执行定时任务

一个sql的执行过程：

1. 客户端发起请求，监听线程被唤醒，生成会话线程，并将任务加入到工作线程队列中
2. 对sql进行语法语义权限解析（字典缓冲区、语法语义校验、权限校验等）
   - **软解析：**在sql缓冲区中找到了对应的sql语句，并查找到sql执行计划，根据执行计划执行sql

    - **硬解析：**没有命中缓冲区，首先将sql语句加入缓冲区，并生成sql执行计划（成本很高），执行后返回结果集
3. 执行sql：分为查询和修改两类
    - **查询：**读数据缓冲区，如果存在，直接返回（**逻辑读**）；如果不存在，会话线程调用IO线程从磁盘中读取数据文件并放入缓冲区（**物理读**）。【PS：如果查询涉及连接，会占用HASH区，如果设计排序，会占用排序区，二者均以会话池和虚拟池的方式申请】
    - **修改**：首先将数据读入到数据缓冲区中（过程同上），然后在数据缓冲区中进行修改，写入redo日志（放入日志缓冲区）和undo日志（放入回滚段）；执行commit后，日志刷新线程将日志缓冲区中的redo日志写入联机日志文件；在数据库执行检查点/缓冲区溢出时，IO线程刷脏页至磁盘

六、表空间管理

表空间介绍

表空间是最大的**逻辑存储单元**，表中的数据都存储在表空间中，类似于一个大的容器，里面放着的就是一个个数据文件（由段、簇、页构成）

- SYSTEM：系统表空间，存储系统的元数据信息，默认是自动扩展的（不可以关闭）
- ROLL：存储undo回滚页，用于保持数据一致性
- MAIN：用户的默认表空间
- TEMP：临时表空间，用于大量数据排序，会自动扩展，但是需要手动回收
- HMAIN：列式用户表空间，新版本和MAIN合并

PS:一般而言，会将数据表空间和索引表空间分开，同时根据业务的不同将数据表空间也进行分开设置，方便根据业务的需求，对表空间进行不同的设置。

管理表空间

- 图形化操作：在达梦管理工具中的表空间栏目即可进行表空间的管理操作
- 命令操作：

```
-- 创建表空间并添加数据文件（数据文件大小要大于页的4096倍）
create tablespace tbs DATAFILE 'TBS01.DBF' size 32;
-- 查询表空间
select * from dba_tablespaces;
-- 脱机表空间（一般用于数据文件迁移）
alter tablespace tbs offline;
-- 重命名表空间（系统表空间不可以重命名）
alter tablespace tbs rename to dmtbs;
-- 删除表空间（系统表空间不能删除）
drop tablespace dmtbs;
```

管理数据文件

- 图形化操作：在达梦管理工具中的表空间栏目即可进行数据空间的管理操作
- 命令操作：

```
-- 修改表空间增加数据文件
alter tablespace tbs add DATAFILE 'TBS02.DBF' size 32 AUTOEXTEND ON NEXT 2
MAXSIZE 20480
-- 修改数据文件大小
alter tablespace tbs RESIZE DATAFILE 'TBS01.DBF' TO 128
-- 迁移数据文件
alter tablesapce tas offline;
alter tablespace tbs RENAME DATAFILE 'TBS01.DBF' TO 'Linux路径'
```

管理联机日志文件

创建数据库实例时默认初始化两个联机日志文件（大小均为256MB），用于存放redo日志，两个文件**循环覆盖使用**

- 图形化操作：在达梦管理工具中，右键数据库实例，点击【管理服务器】-【日志文件】中进行日志文件的管理
  注意：联机日志当前只能添加，不能删除

管理归档日志文件

联机日志因为是循环覆盖使用的，所有只会保留最近的redo日志，因此开启归档模式后，就会对redo联机日志进行归档，一般在对数据安全要求很高的情况下需要开启该模式
归档的目的是为了使得数据库可以完全恢复到故障的前一刻，并支撑日志分析和数据挖掘功能

- 图形化操作：
  - 在达梦管理工具中，右键数据库实例，点击【管理服务器】-【系统管理】中设置数据库实例为挂起状态
  - 在【归档配置】中开启归档并设置归档文件路径
  - 在【系统管理】中设置数据库实例为打开状态

七、用户管理

用户管理

使用者在数据库中的身份

- 系统口令策略问题：隐含参数PWD_POLICY
  - 0：无策略
  - 1：禁止与用户名相同
  - 2：口令长度大于等于9（默认策略）
  - 4：至少包含一个大写字母
  - 8：至少包含一个数字
  - 16：至少包含标点符号

```
-- 修改口令策略
alter SYSTEM set 'PWD_POLICY'=6 both
```

注意：密码中如果含有特殊字符，需要用双信号括起来输入

权限管理

数据库系统可以赋予操作者的最小权力单位

- 系统权限：对数据库、管理表、用户、模式等的操作，默认是不能转授的
- 对象权限：对表、视图、过程等等查询、插入等操作

角色管理

一组权限的集合，方便进行用户权限管理，也即RBAC模型

- DBA：除审计和强制访问控制外其他权限都具备，默认赋给SYSDBA用户
- PUBLIC：对当前模式下对象的DML数据操作权限
- RESOURCE：对当前模式下对象的定义权限（创建表、索引、视图等）
- SOI：查询sys开头的系统表
- VTI：查询v$开头动态视图权限
- DB_AUDIT：审计权限，默认赋给SYSAUDITOR用户
- DB_POLICY：安全相关权限，强制访问权限，默认赋给SYSSSO用户

八、模式对象管理

模式管理

- 模式是一组特定对象的集合（表、视图、索引等等），当建立一个用户时，会自动生成一个同名的模式
- 用户可以建立其他模式，一个模式只能属于一个用户
- 建议一个用户只建立一个模式，这样在写sql时不需要带上模式名，只用表名就可以，方便后续的扩展

表管理

- 分为用户表（用户自己创建的）和系统表（DM Server创建，SYS开头的，存储数据库相关信息）
- 创建、修改表的命令和oracle相同，也可以直接用图形化工具做
- DM默认创建的是索引组织表（Oracle创建的是堆表），表数据是按照聚簇索引键排序的（数据有序，插入有序），默认用rowid作为聚簇索引键（rowid是逻辑rowid，会占用存储空间，所以索引组织表占的空间会大一些）
    - 当使用聚簇索引键查询时，效率会更高
    - 堆表数据无序，插入也无序，因此插入效率更高
- **导入数据**：
    - 客户端直接执行sql，完成后commit即可；
    - disql下导入，效率会高一些（start xxx.sql）
      注意：导入数据时默认开启信息回显，会拖慢效率，当数据量大的时候，应该关闭回显：

```
set TIMING off
set FEEDback off
set echo off
```

约束管理

- NOT NULL：非空约束
- UNIQUE：唯一约束，默认索引
- PRIMARY KEY：唯一约束+非空约束，默认索引
- FOREIGN KEY：外键约束（注意，外键列是子表的，参照列是父表的）
- CHECK：检验约束

视图管理

- 视图就是一组特定select查询组成虚拟表，对于用户而言就是一个特定的视角下的数据库
  - 简单视图支持基本的DML，会转换为对基表的操作
  - 复杂视图一般不支持DML
- 达梦管理工具建立视图
- 用命令创建视图

```
CREATE VIEW [view_name] as [查询语句]
```

索引管理

- 优势：让数据变得有序，提高查询效率，索引可以理解为是将索引列值加上rowid和聚簇索引构建的一张小表
- 缺点：索引会占空间，同时会增加DML的代价
- 索引类型：聚簇索引（叶子节点会存数据）、普通索引（叶子节点存rowid）等等
- 通过explain sql可以查看sql的执行计划，看索引的使用情况

九、数据字典与动态性能视图

数据字典

数据字典存储在SYSTEM表空间中，保存对象定义、用户、权限、角色等信息

```
-- 系统中所有对象的信息
select * from SYSOBJECTS;
-- 系统中所有索引信息
select * from SYSINDEXES;
```

动态性能视图

动态性能视图是从**内存**或**控制文件**中读取的数据，以虚拟表的形式展示

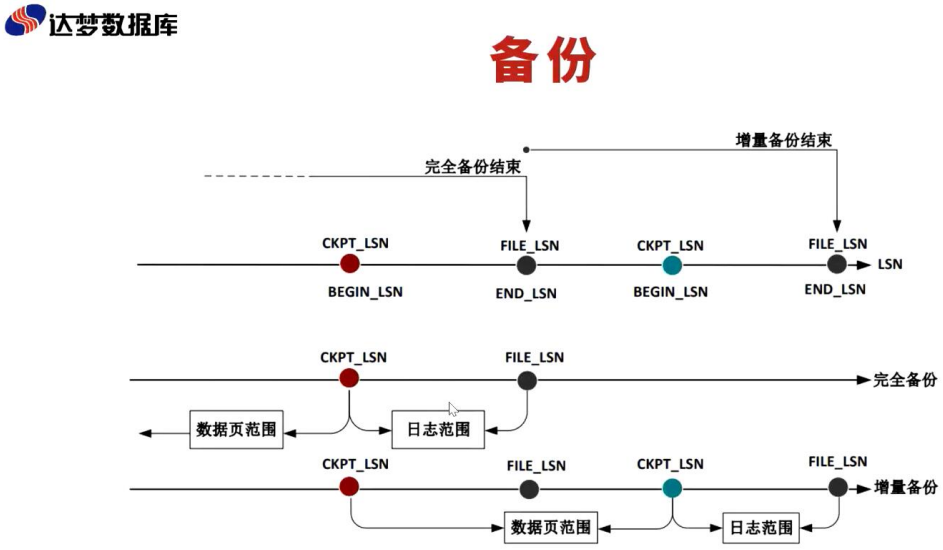
```
-- 数据文件信息
select * from V$DATAFILE;
-- 表空间信息
select * from V$TABLESPACE;
-- 数据缓冲区信息
select * from V$BUFFERPOOL
-- 查询锁相关信息
select * from V$TRXWAIT
select * from V$LOCK where blocked=1
-- 查询dm的函数信息
select * from v$ifun where name like '%xxx%'
```

十、数据库的备份与还原

备份

- 备份分为物理备份和逻辑备份
  - 物理备份：拷贝数据文件中**有效**的数据页（DM会自动结算那些数据页被使用了，只会拷贝这部分数据页）
    - 完全备份：包含所有有效数据页（可以热备，也可以冷备）
    - 增量备份：分为差异增量备份（基础备份集可以是全量也可以是增量）和累计增量备份（基础备份集只可以是全量），增量备份必须热备
  - 逻辑备份：导出的是数据库的逻辑数据，即dmp文件，分为full、owner、schema、tables四种级别

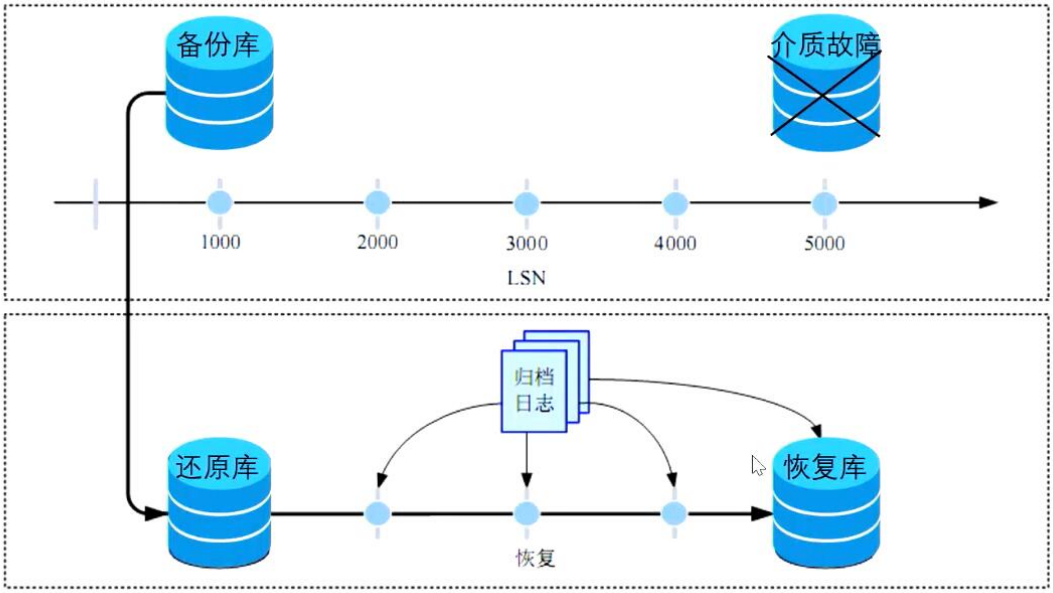
注意：备份不只是是数据页，还会有一部分重做日志信息，保证数据一致性，**所以联机备份首先要开启归档**



还原恢复

- 还原是指基于备份数据将数据库状态还原至备份的时刻

- 恢复是指在还原的基础上进一步基于归档日志将数据库状态恢复至后续的任意一刻



关注两个参数:

```
-- BAK_PATH：备份路径；BAK_USE_AP：【1：使用AP服务；2：使用DM自身服务】
select * from v$parameter t where name in ('BAK_PATH', 'BAK_USE_AP');
```

一般来说，备份、还原、恢复最好使用DmAPService在后台进行，提高并行度，避免拖垮线上服务，注意必须启动dmap进程

联机备份常见问题

建议采用图形化界面进行备份

- 报错备份目录创建失败：大概率是因为dmdba用户没有该目录的写权限
- 报错备份目录冲突：备份集和归档设为了同一个目录
- 归档日志不连续：一般是由于刚开启归档/关了又开的时候出现
    - 执行完全检查点操作：checkpoint(100)
    - 等待几分钟，数据库自行执行检查点操作后
    - 删除之前断裂的归档日志文件

```
select * from V$ARCH_FILE
sf_archivelog_delete_before_lsn([id])
```

脱机备份与还原常见问题

建议采用console控制台工具进行备份与还原

- 需要指定待备份数据库的dm.ini配置文件
- 还原分为库还原、表空间还原两大类：
    - 库还原：正常关闭数据库=>还原=>恢复=>**更新数据库模数**
    - 表空间还原：正常关闭数据库=>还原=>恢复

逻辑导入与导出

建议采用达梦管理工具导入导出即可
如果表已经存在，只想导数据，记得勾选【忽略创建错误】

十一、作业管理

先创建代理，然后通过图形化设置JOB即可。

```
-- 查询当前数据库作业
select * from sysjob.sysjobs;
-- 直接运行作业
SP_DBMS_JOB_RUN([jobid])
```

十二、DM8开发

JDBC配置

```
private static String jdbcString = "dm.jdbc.driver.DmDriver";
private static String urlString = "jdbc:dm://localhost:5236";
private static String user= "SYSDBA";
private static String password = "SYSDBA";
```

ODBC配置

```
-- 1.解压
tar -zxvf unixODBC-2.3.0.tar.gz
-- 2.配置
cd unixODBC-2.3.0/
./configure
-- 3.编译
make
-- 4.安装
make install
-- 5.配置驱动信息 odbcinst.ini 和数据源信息 odbc.ini
odbcinst -j
vim /usr/local/etc/odbcinst.ini
    Description = ODBC DRIVER FOR DM8
    Driver = /dm8/bin/libdodbc.so

vim /usr/local/etc/odbc.ini
    [数据库名称]
    Description = DM ODBC DSN
    Driver = DM8 ODBC DRIVER
    SERVER = localhost
    UID = SYSDBA
    PWD = Dameng123
    TCP_PORT = 5236
```