数据库经典笔试题总结

1，范式

7大范式：1NF，2NF,3NF,BCNF,4NF,5NF,6NF

什么叫normalization？Denormalization?

Normalization是数据库规范化，denormalization是数据库逆规范化。

在设计和操作维护数据库时，关键的步骤就是要确保数据正确地分布到数据库的表中。使用正确的数据结构，不仅便于对数据库进行相应的存取操作，而且可以极大地简化应用程序的其他内容（查询、窗体、报表、代码等）。正确进行表设计的正式名称就是"数据库规范化"。目的：减少数据库中数据冗余，增进数据的一致性。

范式概念：

1）1NF：目标就是表中每列都不可分割；

2）2NF：目标就是表中的每行都是有标识的。前提是满足了1NF．当关键字为单field时，一定满足2NF。当关键字为组合field时（即超过一个field）,不能存在组合关键字中有某个字段能够决定非关键字段的某部分。非主field非部分依赖于主field，即非关键字段必须完全依赖于一组 组合关键字，而不是组合关键字的某一部分。

3）3NF：目标是一个table里面所有的列不依赖于另外一个table里面非关键的列。前提是满足了2NF，不存在某个非关键字段决定另外一个非关键字段。即：不存在传递依赖（关键字x->非关键属性y->非关键属性z）

4）BCNF：前提是满足了2NF，不存在某个非关键字段决定另外一个非关键字段。也不存在某个关键字段决定另外一个关键字段。即：在3NF基础上，加上约束：不存在某个关键字段决定另外一个关键字段。

1 第一范式（1NF）

在任何一个关系数据库中，第一范式（1NF）是对关系模式的基本要求，不满足第一范式（1NF）的数据库就不是关系数据库。所谓第一范式（1NF）是指数据库表的每一列都是不可分割的基本数据项，同一列中不能有多个值，即实体中的某个属性不能有多个值或者不能有重复的属性。如果出现重复的属性，就可能需要定义一个新的实体，新的实体由重复的属性构成，新实体与原实体之间为一对多关系。在第一范式（1NF）中表的每一行只包含一个实例的信息。例如，对于图3-2 中的员工信息表，不能将员工信息都放在一列中显示，也不能将其中的两列或多列在一列中显示；员工信息表的每一行只表示一个员工的信息，一个员工的信息在表中只出现一次。简而言之，第一范式就是无重复的列。

2 第二范式（2NF）

第二范式（2NF）是在第一范式（1NF）的基础上建立起来的，即满足第二范式（2NF）必须先满足第一范式（1NF）。第二范式（2NF）要求数据库表中的每个实例或行必须可以被惟一地区分。为实现区分通常需要为表加上一个列，以存储各个实例的惟一标识。如图3-2 员工信息表中加上了员工编号（emp_id）列，因为每个员工的员工编号是惟一的，因此每个员工可以被惟一区分。这个惟一属性列被称为主关键字或主键、主码。第二范式（2NF）要求实体的属性完全依赖于主关键字。所谓完全依赖是指不能存在仅依赖主关键字一部分的属性，如果存在，那么这个属性和主关键字的这一部分应该分离出来形成一个新的实体，新实体与原实体之间是一对多的关系。为实现区分通常需要为表加上一个列，以存储各个实例的惟一标识。简而言之，第二范式就是非主属性非部分依赖于主关键字。

3 第三范式（3NF）

满足第三范式（3NF）必须先满足第二范式（2NF）。简而言之，第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。例如，存在一个部门信息表，其中每个部门有部门编号（dept_id）、部门名称、部门简介等信息。那么在图3-2的员工信息表中列出部门编号后就不能再将部门名称、部门简介等与部门有关的信息再加入员工信息表中。如果不存在部门信息表，则根据第三范式（3NF）也应该构建它，否则就会有大量的数据冗余。简而言之，第三范式就是属性不依赖于其它非主属性。

例子：

第一范式（1NF）：数据库表中的字段都是单一属性的，不可再分。这个单一属性由基本类型构成，包括整型、实数、字符型、逻辑型、日期型等。

例如，如下的数据库表是符合第一范式的：字段1 字段2 字段3 字段4

而这样的数据库表是不符合第一范式的：字段1 字段2 字段3 字段4 字段31字段32

很显然，在当前的任何关系数据库管理系统（S）中，傻瓜也不可能做出不符合第一范式的数据库，因为这些S不允许你把数据库表的一列再分成二列或多列。因此，你想在现有的S中设计出不符合第一范式的数据库都是不可能的。

第二范式（2NF）：数据库表中不存在非关键字段对任一候选关键字段的部分函数依赖（部分函数依赖指的是存在组合关键字中的某些字段决定非关键字段的情况），也即所有非关键字段都完全依赖于任意一组候选关键字。

假定选课关系表为Ss(学号，姓名，年龄，课程名称，成绩，学分)，关键字为组合关键字(学号，课程名称)，因为存在如下决定关系：

(学号，课程名称) → (姓名，年龄，成绩，学分)

这个数据库表不满足第二范式，因为存在如下决定关系：

(课程名称) → (学分)

(学号) → (姓名，年龄)

即存在组合关键字中的字段决定非关键字的情况。

由于不符合2NF，这个选课关系表会存在如下问题：1) 数据冗余：同一门课程由n个学生选修，"学分"就重复n-1次；同一个学生选修了门课程，姓名和年龄就重复了-1次。2) 更新异常：若调整了某门课程的学分，数据表中所有行的"学分"值都要更新，否则会出现同一门课程学分不同的情况。3) 插入异常：假设要开设一门新的课程，暂时还没有人选修。由于还没有"学号"关键字，课程名称和学分也无法记录入数据库。4) 删除异常：假设一批学生已经完成课程的选修，这些选修记录就应该从数据库表中删除。但是，与此同时，课程名称和学分信息也被删除了。很显然，这也会导致插入异常。

把选课关系表Ss改为如下三个表：

学生：Sn(学号，姓名，年龄)；

课程：s(课程名称，学分)；

选课关系：Ss(学号，课程名称，成绩)。

这样的数据库表是符合第二范式的，消除了数据冗余、更新异常、插入异常和删除异常。

另外，所有单关键字的数据库表都符合第二范式，因为不可能存在组合关键字。

第三范式（3NF）：在第二范式的基础上，数据表中如果不存在非关键字段对任一候选关键字段的传递函数依赖则符合第三范式。所谓传递函数依赖，指的是如果存在"A → → "的决定关系，则传递函数依赖于A。因此，满足第三范式的数据库表应该不存在如下依赖关系：关键字段 → 非关键字段x → 非关键字段y

假定学生关系表为Sn(学号，姓名，年龄，所在[]学院[]，学院地点，学院电话)，关键字为单一关键字"学号"，因为存在如下决定关系：

(学号) → (姓名，年龄，所在[]学院[]，学院[]地点，[]学院[]电话)

这个数据库是符合2NF的，但是不符合3NF，因为存在如下决定关系：

(学号) → (所在[]学院[]) → ([]学院[]地点，[]学院[]电话)

即存在非关键字段"[]学院[]地点"、"[]学院[]电话"对关键字段"学号"的传递函数依赖。

它也会存在数据冗余、更新异常、插入异常和删除异常的情况，读者可自行分析得知。

把学生关系表分为如下两个表：

学生：(学号，姓名，年龄，所在[]学院[])；

[]学院[]：([]学院[]，地点，电话)。

这样的数据库表是符合第三范式的，消除了数据冗余、更新异常、插入异常和删除异常。

鲍依斯-科得范式（BCNF）：在第三范式的基础上，数据库表中如果不存在任何字段对任一候选关键字段的传递函数依赖则符合BCNF.

假设仓库管理关系表为Ssanag(仓库，存储物品，管理员，数量)，且有一个管理员只在一个仓库工作；一个仓库可以存储多种物品。这个数据库表中存在如下决定关系：

(仓库，存储物品) →(管理员，数量)

(管理员，存储物品) → (仓库，数量)

所以，(仓库，存储物品)和(管理员，存储物品)都是Ssanag的候选关键字，表中的唯一非关键字段为数量，它是符合第三范式的。但是，由于存在如下决定关系：

(仓库) → (管理员)

(管理员) → (仓库)

即存在关键字段决定关键字段的情况，所以其不符合BCNF范式。它会出现如下异常情况：1) 删除异常：当仓库被清空

后，所有"存储物品"和"数量"信息被删除的同时，"仓库"和"管理员"信息也被删除了。2) 插入异常：当仓库没有存储任何物品时，无法给仓库分配管理员。3) 更新异常：如果仓库换了管理员，则表中所有行的管理员都要修改。

把仓库管理关系表分解为二个关系表：

仓库管理：Ssanag(仓库，管理员)；

仓库：Ss(仓库，存储物品，数量)。

这样的数据库表是符合BCNF范式的，消除了删除异常、插入异常和更新异常。

简言之数据库五大范式：

第一范式：对于表中的每一行，必须且仅仅有唯一的行值.在一行中的每一列仅有唯一的值并且具有原子性.

（第一范式是通过把重复的组放到每个独立的表中，把这些表通过一对多关联联系起来这种方式来消除重复组的）

第二范式：第二范式要求非主键列是主键的子集，非主键列活动必须完全依赖整个主键。主键必须有唯一性的元素,一个主键可以由一个或更多的组成唯一值的列组成。一旦创建，主键无法改变，外键关联一个表的主键。主外键关联意味着一对多的关系.（第二范式处理冗余数据的删除问题。当某张表中的信息依赖于该表中其它的不是主键部分的列的时候，通常会违反第二范式）

第三范式：第三范式要求非主键列互不依赖.（第三范式规则查找以消除没有直接依赖于第一范式和第二范式形成的表的主键的属性。我们为没有与表的主键关联的所有信息建立了一张新表。每张新表保存了来自源表的信息和它们所依赖的主键）

第四范式：第四范式禁止主键列和非主键列一对多关系不受约束

第五范式：第五范式将表分割成尽可能小的块，为了排除在表中所有的冗余。

2，索引：

什么叫 revised key index?

反键索引是B*Tree索引的一个分支，它的设计是为了运用在某些特定的环境下的。Oracle推出它的主要目的就是为了降低在并行服务器（Oracle Parallel Server）环境下索引叶块的争用。当B*Tree索引中有一列是由递增的序列号产生的话，那么这些索引信息基本上分布在同一个叶块，当用户修改或访问相似的列时，索引块很容易产生争用。反向索引中的索引码将会被分布到各个索引块中，减少了争用.

例子：有一个字段id，他的值落在一个很小的区间，比如从9000-9999，如果建b-tree索引，那么值过于紧密，反键的原理是把值取反，那么id的区间就从0009-9999，区间就被放大，这个时候通过索引来查找数据效率会比较高（oracle这么说的）。

好处是:解决了树的倾斜问题，而且可以解决在大量IO操作的情况下,防止硬盘在某个区域操作过于频繁,引起"热点"问题。

树的分支：因为索引一般是按树这个数据结构来组织，所以有很多分支，把不同类别或范围的数据存放在分支里，在符合条件的分支里查询比在全表查询效率高很多。

树的倾斜：树的某个分支过与庞大,而其他分支内容却很少，这样的索引非常不健康的，查询速度也很慢，如上面的示例数据，都在10000-20000 的分支，而20000-30000或者以上的分支是空的。反转后把这些数据均匀分布到不同的分支，可以使索引更加健康，也更有效率。

热点问题：由于系统在表数据的增删改查的同时，同时要承担索引开支，而这主要是硬盘的IO操作，如果树是倾斜的，而且数据的增加是按一定顺序增长的，这种情况会导致硬盘对某一固定区域操作频繁，会出现热点问题，而且出现瓶颈。

Oracle五种索引：

1）b*tree index：几乎所有的关系型数据库中都有b*tree类型索引，也是被最多使用的。其树结构与二叉树比较类似，根据rid快速定位所访问的行。 B-Tree索引是基于二叉树的，由分支块（branch block）和叶块（leaf block）组成。在树结构中，位于最底层底块被称为叶块，包含每个被索引列的值和行所对应的rowid。在叶节点的上面是分支块，用来导航结构，包含了索引列（关键字）范围和另一索引块的地址。

2）反向索引：反转了b*tree索引码中的字节，是索引条目分配更均匀，多用于并行服务器环境下，用于减少索引叶的竞争。反向索引又一个缺点就是不能在所有使用常规索引的地方使用。在范围搜索中其不能被使用。

3）降序索引：8i中新出现的索引类型，针对逆向排序的查询。

4）位图索引：使用位图来管理与数据行的对应关系，多用于OLAP系统。位图索引最好用于低cardinality列（即列的唯一值除以行数为一个很小的值，接近零），例如又一个“性别”列，列值有“Male”，“Female”，“Null”等3种，但一共有

300万条记录，那么3/3000000约等于0，这种情况下最适合用位图索引。位图以一种压缩格式存放，因此占用的磁盘空间比B-Tree索引要小得多。

5）函数索引：这种索引中保存了数据列基于function返回的值，在select * from table where function(column)=value这种类型的语句中起作用。基于函数的索引也是8i以来的新产物，它有索引计算列的能力，它易于使用并且提供计算好的值，在不修改应用程序的逻辑上提高了查询性能。使用基于函数的索引有几个先决条件：

（1）必须拥有QUERY REWRITE（本模式下）或GLOBAL QUERY REWRITE（其他模式下）权限。

（2）必须使用基于成本的优化器，基于规则的优化器将被忽略。

（3）必须设置以下两个系统参数：

QUERY_REWRITE_ENABLED=TRUE

QUERY_REWRITE_INTEGRITY=TRUSTED

可以通过alter system set,alter session set在系统级或线程级设置，也可以通过在init.ora添 加实现。

五种索引的创建：

（1）*Tree索引。

```
Create index indexname on tablename(columnname[columnname...])
```

（2）反向索引。

```
Create index indexname on tablename(columnname[columnname...]) reverse
```

（3）降序索引。

```
Create index indexname on tablename(columnname DESC[columnname...])
```

（4）位图索引。

```
Create BITMAP index indexname on tablename(columnname[columnname...])
```

（5）函数索引。

```
Create index indexname on tablename(functionname(columnname))
```

注意：创建索引后分析要索引才能起作用。

五种索引的使用场所：

（1）B*Tree索引。

常规索引，多用于oltp系统，快速定位行，应建立于高cardinality列（即列的唯一值除以行数为一个很大的值，存在很少的相同值）。

（2）反向索引。

B*Tree的衍生产物，应用于特殊场合，在ops环境加序列增加的列上建立，不适合做区域扫描。

（3）降序索引。

B*Tree的衍生产物，应用于有降序排列的搜索语句中，索引中储存了降序排列的索引码，提供了快速的降序搜索。

（4）位图索引。

位图方式管理的索引，适用于OLAP（在线分析）和DSS（决策处理）系统，应建立于低cardinality列，适合集中读取，不适合插入和修改，提供比B*Tree索引更节省的空间。

（5）函数索引。

B*Tree的衍生产物，应用于查询语句条件列上包含函数的情况，索引中储存了经过函数计算的索引码值。可以在不修改应用程序的基础上能提高查询效率。

索引不管用的时候：

（1）RBO&CBO。

Oracle有**两种执行优化器，一种是**RBO(Rule Based Optimizer)**基于规则的优化器，这种优化器是基于sql语句写法选择执行路径的；另一种是**CBO（Cost Based Optimizer）**基于规则的优化器**，这种优化器是Oracle根据统计分析信息来选择执行路径，如果表和索引没有进行分析，Oracle将会使用RBO代替CBO；如果表和索引很久未分析，CBO也有可能选择错误执行路径，不过CBO是Oracle发展的方向，自8i版本来已经逐渐取代RBO.

（2）AUTOTRACE。

要看索引是否被使用我们要借助Oracle的一个叫做AUTOTRACE功能,它显示了sql语句的执行路径，我们能看到Oracle内部是怎么执行sql的，这是一个非常好的辅助工具，在sql调优里广泛被运用。我们来看一下怎么运用AUTOTRACE：

① 由于AUTOTRACE自动为用户指定了Execution Plan,因此该用户使用AUTOTRACE前必须已经建立了PLAN_TABLE。如果没有的

话，请运行utlxplan.sql脚本（它在$ORACLE_HOME/rdbms/admin目录中）。

② AUTOTRACE可以通过运行plustrce.sql脚本(它在$ORACLE_HOME/sqlplus/admin目录中)来设置，用sys用户登陆然后运行plustrce.sql后会建立一个PLUSTRACE角色，然后给相关用户授予PLUSTRACE角色，然后这些用户就可以使用AUTOTRACE功能了。

③ AUTOTRACE的默认使用方法是set autotrace on，但是这方法不总是适合各种场合，特别当返回行数很多的时候。Set autotrace traceonly提供了只查看统计信息而不查询数据的功能。

3，死锁

是指两个或两个以上的进程在执行过程中,因争夺资源而造成的一种互相等待的现象,若无外力作用,它们都将无法推进下去.此时称系统处于死锁状态或系统产生了死锁,这些永远在互相等待的进程称为死锁进程.由于资源占用是互斥的，当某个进程提出申请资源后，使得有关进程在无外力协助下，永远分配不到必需的资源而无法继续运行，这就产生了一种特殊现象死锁。

产生死锁的原因主要是：

（1） 因为系统资源不足。

（2） 进程运行推进的顺序不合适。

（3） 资源分配不当等。

如果系统资源充足，进程的资源请求都能够得到满足，死锁出现的可能性就很低，否则就会因争夺有限的资源而陷入死锁。其次，进程运行推进顺序与速度不同，也可能产生死锁。

产生死锁的四个必要条件：

（1） 互斥条件：一个资源每次只能被一个进程使用。

（2） 请求与保持条件：一个进程因请求资源而阻塞时，对已获得的资源保持不放。

（3） 不剥夺条件:进程已获得的资源，在末使用完之前，不能强行剥夺。

（4） 循环等待条件:若干进程之间形成一种头尾相接的循环等待资源关系。

这四个条件是死锁的必要条件，只要系统发生死锁，这些条件必然成立，而只要上述条件之一不满足，就不会发生死锁。

例子：

运行事务 1 的线程 T1 具有学生基本信息表上的排它锁。运行事务2的线程 T2 具有系部表上的排它锁，并且之后需要学生基本信息表上的锁。事务2 无法获得这一锁，因为事务 1 已拥有它。事务2 被阻塞，等待事务 1。然后，事务1 需要系部表的锁，但无法获得锁，因为事务 2 将它锁定了。事务在提交或回滚之前不能释放持有的锁。因为事务需要对方控制的锁才能继续操作，所以它们不能提交或回滚。

4，BYTE[] buf = BYTE[1024];in.read(buf);

in是一个接收图像数据的网络IO流，请指出这段代码有什么问题，并请用java代码改进它。

答：流操作都可能会跑出IOException，应该对该异常进行捕获处理。且当buf没有被初始化的时候使用会抛出NullPointerException。

```
byte [] buf = new byte[1024];
try {
        System.in.read(buf);
} catch (IOException e) {
        e.printStackTrace();
}
```

5，**设计模式：**Facade

你正在分析一个子系统的接口，发现接口很多。然后你同事劝你用Fecade，问你用Fecade有什么好处？

Facade（外观）模式为子系统中的各类（或结构与方法）提供一个简明一致的界面，隐藏子系统的复杂性，使子系统更加容

易使用。Facade模式正是这样一个“门面”：我们本来需要与后台的多个类或者接口打交道，而Facade模式是客户端和后台之间插入一个中间层——门面，这个门面跟后台的多个类或接口打交道，而客户端只需要跟门面打交道即可。使用Facade模式可以说是后台设计和编码人员的一个必备素质。我不止碰到过一个这样的后台开发人员，他们认为只要把后台功能完成了就万事大吉，而没有站在后台使用者的角度来看一看自己写出来的代码。其实，我们写出来的后台代码是要给别人使用的，所以我们提供给使用者的接口要越简单越好，这不单是对使用者好，同时对开发者也是好处多多的，至少你的接口简单了，你和使用者的交流就容易了。

区分Façade模式、Adapter模式、Bridge模式与Decorator模式。Façade模式注重简化接口，Adapter模式注重转换接口，Bridge模式注重分离接口（抽象）与其实现，Decorator模式注重稳定接口的前提下为对象扩展功能

在遇到以下情况使用Facade模式：

1）当你要为一个复杂子系统提供一个简单接口时。子系统往往因为不断演化而变得越来越复杂。大多数模式使用时都会产生更多更小的类。这使得子系统更具可重用性，也更容易对子系统进行定制，但这也给那些不需要定制子系统的用户带来一些使用上的困难。 Facade可以提供一个简单的缺省视图，这一视图对大多数用户来说已经足够，而那些需要更多的可定制性的用户可以越过Facade层。

2）客户程序与抽象类的实现部分之间存在着很大的依赖性。引入Facade将这个子系统与客户以及其他的子系统分离，可以提高子系统的独立性和可移植性。

3）当你需要构建一个层次结构的子系统时，使用Facade模式定义子系统中每层的入口点，如果子系统之间是相互依赖的，你可以让它们仅通过Facade进行通讯，从而简化了它们之间的依赖关系。

优缺点：

1）它对客户屏蔽子系统组件，因而减少了客户处理的对象的数目并使得子系统使用起来更加方便。

2）它实现了子系统与客户之间的松耦合关系，而子系统内部的功能组件往往是紧耦合的。

松耦合关系使得子系统的组件变化不会影响到它的客户。Facade模式有助于建立层次结构系统，也有助于对对象之间的依赖关系分层。Facade模式可以消除复杂的循环依赖关系。这一点在客户程序与子系统是分别实现的时候尤为重要。在大型软件系统中降低编译依赖性至关重要。在子系统类改变时，希望尽量减少重编译工作以节省时间。用Facade可以降低编译依赖性，限制重要系统中较小的变化所需的重编译工作。Facade模式同样也有利于简化系统在不同平台之间的移植过程，因为编译一个子系统一般不需要编译所有其他的子系统。

6，冷备份与热备份

冷备份：

冷备份发生在数据库已经正常关闭的情况下，当正常关闭时会提供给我们一个完整的数据库。冷备份是将关键性文件拷贝到另外位置的一种说法。对于备份Oracle信息而言，冷备份是最快和最安全的方法。

冷备份的优点是：

1．是非常快速的备份方法（只需拷贝文件）

2．容易归档（简单拷贝即可）

3．容易恢复到某个时间点上（只需将文件再拷贝回去）

4．能与归档方法相结合，作数据库“最新状态”的恢复。

5．低度维护，高度安全。

冷备份也有如下不足：

1．单独使用时，只能提供到“某一时间点上”的恢复。

2．在实施备份的全过程中，数据库必须要作备份而不能作其它工作。也就是说，在冷备份过程中，数据库必须是关闭状态。

3．若磁盘空间有限，只能拷贝到磁带等其它外部存储设备上，速度会很慢。

4．不能按表或按用户恢复。

如果可能的话（主要看效率），应将信息备份到磁盘上，然后启动数据库（使用户可以工作）并将所备份的信息拷贝到磁带上（拷贝的同时，数据库也可以工作）。

冷备份中必须拷贝的文件包括：

1．所有数据文件

2．所有控制文件

3．所有联机REDO LOG文件

4．Init.ora文件（可选）。

下面是做冷备份的完整例子：

（1） 关闭数据库＄sqldba lmode=y

SQLDBA >connect internal;

SQLDBA >shutdown normal;

（2） 用拷贝命令备份全部的时间文件、重做日志文件、控制文件、初始化参数文件

SQLDBA >! cp < file > < backup directory >

（3） 重启Oracle数据库

＄sqldba lmode=y

SQLDBA >connect internal;

SQLDBA >startup;

热备份

热备份是在数据库运行的情况下，采用archivelog mode方式备份数据的方法。所以，如果你有昨天夜里的一个冷备份而且又有今天的热备份文件，在发生问题时，就可以利用这些资料恢复更多的信息。

热备份的要求是：

1．热备份工作必需要求数据库在Archivelog 方式下操作,在SQLDBA状态下用alter database archivelog|noarchivelog命令可改变备份的模式。

2．热备份只能在数据库不使用或使用率低的情况下进行。

3．热备份需要大量的档案空间。

一般情况，Oracle 以循环的方式写入Online redo log 文件，当填满第一个redo log文件后写第二个，直至最后一个，最后一个被填满后，后台进程LGWR就覆盖第一个，在Archivelog方式下，后台进程ARCH在每一个redo log 文件被覆盖前，给它作一个拷贝，一般，这些文档的redo log 文件被写入磁盘或磁带中。如果磁盘空间够用，建议使用磁盘，这样可大大减少完成备份所需的时间。

在作热备份之前，要将config.ora文件中的log_archive_start 设为true 将log_archive_dest一旦数据库运行在archivelog状态下，就可以做备份了。

热备份的命令文件由三部分组成：

1．数据文件一个表空间一个表空间地备份。

（1）设置表空间为备份状态

（2）备份表空间的数据文件

（3）恢复表空间为正常状态

2．备份归档log文件。

（1）临时停止归档进程

（2）log下那些在archive redo log目标目录中的文件

（3）重新启动archive进程

（4）备份归档的redo log 文件

3．用alter database backup controlfile命令来备份拷贝文件

热备份的优点是：

1．可在表空间或数据文件级备份，备份时间短。

2．备份时数据库仍可使用。

3．可达到秒级恢复（恢复到某一时间点上）。

4．可对几乎所有数据库实体作恢复。

5．恢复是快速的，在大多数情况下在数据库仍工作时恢复。

热备份的不足是：

1．不能出错，否则后果严重。

2．若热备份不成功，所得结果不可用于时间点的恢复。
3．因难于维护，所以要特别仔细小心，不允许“以失败而告终”。

7，你必须利用备份恢复数据库，但是你没有控制文件，该如何解决问题呢?
重建控制文件，用带backup control file 子句的recover 命令恢复数据库。
recover dbname using backup controlfile;

8，如何转换init.ora到spfile?
create spfile from pfile='init.ora';

9，oracle体系结构：
Oracle物理结构
由控制文件、数据文件、重做日志文件、参数文件、归档文件、密码文件组成
控制文件：包含维护和验证数据库完整性的必要信息、例如，控制文件用于识别数据文件和重做日志文件，一个数据库至少需要一个控制文件
数据文件：存储数据的文件
重做日志文件：含对数据库所做的更改记录，这样万一出现故障可以启用数据恢复。一个数据库至少需要两个重做日志文件
参数文件：定义Oracle 例程的特性，例如它包含调整SGA 中一些内存结构大小的参数
归档文件：是重做日志文件的脱机副本，这些副本可能对于从介质失败中进行恢复很必要。
密码文件：认证哪些用户有权限启动和关闭Oracle例程
oracle逻辑结构
表空间、段、区、块
表空间：是数据库中的基本逻辑结构，一系列数据文件的集合。
段：是对象在数据库中占用的空间
区：是为数据一次性预留的一个较大的存储空间
块：ORACLE最基本的存储单位，在建立数据库的时候指定
oracle内存分配
SGA和PGA
SGA：是用于存储数据库信息的内存区，该信息为数据库进程所共享。它包含Oracle 服务器的数据和控制信息，它是在Oracle 服务器所驻留的计算机的实际内存中得以分配，如果实际内存不够再往虚拟内存中写。
PGA：包含单个服务器进程或单个后台进程的数据和控制信息，与几个进程共享的SGA 正相反PGA 是只被一个进程使用的区域，PGA 在创建进程时分配在终止进程时回收
oracle后台进程
(数据写进程、日志写进程、系统监控、进程监控、检查点进程、归档进程、服务进程、用户进程)
数据写进程：负责将更改的数据从数据库缓冲区高速缓存写入数据文件
日志写进程：将重做日志缓冲区中的更改写入在线重做日志文件
系统监控：检查数据库的一致性如有必要还会在数据库打开时启动数据库的恢复
进程监控：负责在一个Oracle 进程失败时清理资源
检查点进程：负责在每当缓冲区高速缓存中的更改永久地记录在数据库中时,更新控制文件和数据文件中的数据库状态信息。
归档进程：在每次日志切换时把已满的日志组进行备份或归档
服务进程：用户进程服务。
用户进程：在客户端，负责将用户的SQL 语句传递给服务进程，并从服务器段拿回查询数据。
Oracle Instance

Oracle 例程由SGA 内存结构和用于管理数据库的后台进程组成。例程一次只能打开和使用一个数据库。

SCN(System Change Number)：**系统改变号**

一个由系统内部维护的序列号。当系统需要更新的时候自动增加，他是系统中维持数据的一致性和顺序恢复的重要标志。

实例和SID的关系是什么？

经常有人问SID 是什么？在Oracle 系统中SID 是一个经常出现的变量，如环境变量ORACLE_SID， 初始化文件initSID.ora，那究竟什么是SID 呢？其实SID 就是Oracle 实例的标识，不同的SID 对应不同的内存缓冲(SGA)和不同的后台进程。这样一来我们就可以得当在一台物理的服务器上可以有多个SID 的数据库实例

Oracle数据库和实例的关系是什么？

数据库是由物理文件和存取数据文件的实例组成，当存取数据文件的实例是一个的时候，数据库被称做单节点数据库。这是我们看到的最多的数据库形式。当然还有一种多节点数据库，就是一个以上的实例共同访问一个数据库(或者说共同访问一组数据文件)， 更好的提供稳定性和并行处理能力。这在8i中被称为OPS(Oracle Parallel Server)，在Oracle9i 中被称为RAC(real application cluster)。在这种数据库中。两个/多个实例分别在不同服务器上，所有Oracle 数据文件在共享的磁盘阵列上，多个服务器上的实例可以同时工作，他们通过一个内部的网络进行通信。如果一台服务器不能提供服务的话，另一台会接管它的工作，特别是在关键的业务有很大的潜力。

在运行的数据库中数据文件中是不是可能存在没有被提交的数据？

这是可能存在的，因为用户数据文件的数据是由DBWR写入的，DBWR是一个很底层的后台进程，不负责与用户交互。用户的交互是由LGWR完成的。

在问题10中，如果存在没有写入的数据，那么机器突然断电，数据完整性会不会损坏？

不会的，因为数据库的完整性是LGWR来保证的，而且ORACLE保证了DBWR写入数据文件的任何修改已经被记录在重做日志文件中。当系统再次启动的时候，通过读取重做日志文件就可以知道那些数据没有被提交。这时候ORACLE 会自动回滚那些数据。所以说联机日志的损坏，特别是当前联机日志的损坏，对数据库的影响是巨大的，可能会导致数据库的不完整。

Oracle**安装完成后的初始用户名与密码**?

sys/change_on_install

system/manager

scott/tiger

sysman/oem_temp

创建Sequence

```
create sequence zxk increment by 10  start with 10 maxvalue 1000  cycle ;
```

数据字典

数据字典是ORACLE数据库的最重要的部分之一，是由一组只读的表及其视图所组成。这些表和视图是数据库被建立同时由数据库系统建立起来的，起着系统状态的目录表的作用。数据字典描述表、列、索引、用户、访问权以及数据库中的其它实体，当其中的一个实体被建立、修改或取消时，数据库将自动修改数据字典。因此，数据字典总是包含着数据库的当前描述，可用SQL存取数据字典，由于数据字典为只读，只允许查询。数据字典中全部基本表和用户可存取视图为ORACLE用户SYS所持有，所有对象包含在SYS模式中。当ORACLE数据库系统启动后，数据字典总是可用，它驻留在SYSTEM表空间中。数据字典包含视图集，在许多情况下，每一视图集有三种视图包含有类似信息，彼此以前缀相区别，前缀为USER、ALL和DBA。

10，给出两个检查表结构的方法

1）DESCRIBE命令

```
sql>desc dbname;
```

2）DBMS_METADATA.GET_DDL 包

11，怎样查看数据库引擎的报错

alert log.

12，比较truncate和delete 命令

1、在功能上，truncate是清空一个表的内容，它相当于delete from table_name。

2、 delete是dml操作，truncate是ddl操作；因此，用delete删除整个表的数据时，会产生大量的roolback，占用很多的rollback segments， 而truncate不会。

3、在内存中，用delete删除数据，表空间中其被删除数据的表占用的空间还在，便于以后的使用，另外它是“假相”的删除，相当于windows中用delete删除数据是把数据放到回收站中，还可以恢复，当然如果这个时候重新启动系统（OS或者RDBMS），它也就不能恢复了！而用truncate清除数据，内存中表空间中其被删除数据的表占用的空间会被立即释放，相当于windows中用shift+delete删除数据，不能够恢复！

4、truncate 调整high water mark 而delete不；truncate之后，TABLE的HWM退回到 INITIAL和NEXT的位置（默认）delete则不可以。

5、truncate 只能对TABLE，delete 可以是table,view,synonym。

6、TRUNCATE TABLE 的对象必须是本模式下的，或者有drop any table的权限 而 DELETE 则是对象必须是本模式下的，或被授予 DELETE ON SCHEMA.TABLE 或DELETE ANY TABLE的权限。

7、在外层中，truncate或者delete后，其占用的空间都将释放。

8、truncate和delete只删除数据，而drop则删除整个表（结构和数据）。

在删除大数据量时(一个表中大部分数据时),可以采用以下步骤：

1、先将不需要删除的数据复制到一个临时表中

2、trunc table 表

3、将不需要删除的数据复制回来。

不同：DELETE语句执行删除的过程是每次从表中删除一行，并且同时将该行的的删除操作作为事务记录在日志中保存以便进行进行回滚操作。TRUNCATE TABLE 则一次性地从表中删除所有的数据页并不把单独的删除操作记录记入日志保存，删除行是不能恢复的。并且在删除的过程中不会激活与表有关的删除触发器。执行速度快。

truncate的功能可以拆分为:dropping + re-creating a table，但truncate的效率要高。因为truncate无须考虑表索引、约束、触发器等的重建。另外truncate执行后，无法回滚（roll back）

HWM（high water mark，**最高水位线**）：The high water mark is divides a segment into used blocks and free blocks。Blocks below the high water mark (used blocks) have at least once contained data. This data might have been deleted. Since Oracle knows that blocks beyond the high water mark don't have data, it only reads blocks up to the high water mark in a full table scan（FTS）.在oracle10g中增加了调整HWM的命令：alter table xxx shrink space。

13，归档模式与非归档模式：

非归档模式：只能做冷备份,并且恢复时只能做完全备份.最近一次完全备份到系统出错期间的数据不能恢复。非归档模式只能离线备份，而且必须备份所有的数据文件，控制文件，日志文件！

归档模式：可以做热备份,并且可以做增量备份,可以做部分恢复.归档模式，可以在线或者离线备份数据库，可以是全备份或者是部分备份（单个表空间，数据文件）.归档模式能够做到零数据丢失。当然归档会消耗一些存储和性能资源。归档模式，数据库的日志可以长时间保存，有了归档日志，可以随时恢复归档日期内任何时间点的数据，便于数据库数据安全保护。

RMAN**备份：**必须使用archivelog,对于是否使用ARCHIVELOG,得看应用具体情况的.数据库归档模式，可以使用rman或手工在线备份数据。

用ARCHIVE LOG LIST 可以查看期模式状态时归档模式还是非归档模式.

```
sql>archive log list;
sql>alter dbname archivelog;
sql>alter dbname noarchivelog;
```

14，如何建立备份控制文件？

SQL> alter database backup controlfile to trace;

15，数据库的状态：

四种状态：open,close,mount,nomount

打开数据库：

STARTUP NOMOUNT – 启动instance，但是不装载数据库

STARTUP MOUNT - 启动instance，并装载数据库，但是不打开数据库

STARTUP OPEN – 启动instance，并装载数据库，然后打开数据库

1）启动&装载&打开&限制仅DBA访问：STARTUP RESTRICT;

取消：ALTER SYSTEM DISABLE RESTRICTED SESSION;

2）强制启动：STARTUP FORCE;

3）只读方式打开数据库：STARTUP OPEN READ ONLY;

关闭数据库：

SHUTDOWN NORMAL;

SHUTDOWN IMMEDIATE;

SHUTDOWN TRANSACTIONAL;

SHUTDOWN ABORT;

16，解释$ORACLE_HOME**和**$ORACLE_BASE**的区别**?

ORACLE_BASE是oracle的根目录，ORACLE_HOME是oracle产品的目录。

17，如何判断数据库的时区?

SQL> select dbtimezone from dual;

SQL> select sessiontimezone from dual;

SQL> select sysdate from dual;

SQL> select systimestamp from dual;

SQL> select current_date from dual;

SQL> select current_timestamp from dual;

18，解释GLOBAL_NAMES

GLOBAL_NAME:Global name of the database,是数据库的一个持久性设置.

GLOBAL_NAMES:用于设定数据库链接名称是否必须与远程数据库的GLOBAL_NAME匹配,默认为false。如果这个参数设置为TRUE,在建立数据库链接时就必须用相同的名字连结远程数据库。

19，物化视图：materialized view

Oracle数据库中有两种view，一种是存储为pure SQL，一种是维护一张表。是一种特殊的物理表，“物化”(Materialized)视图是相对普通视图而言的。普通视图是虚拟表，应用的局限性大，任何对视图的查询，Oracle都实际上转换为视图SQL语句的查询。这样对整体查询性能的提高，并没有实质上的好处。物化视图是包括一个查询结果的数据库对像，它是远程数据的的本地副本，或者用来生成基于数据表求和的汇总表。物化视图存储基于远程表的数据，也可以称为**快照**。

物化视图对于前台数据库使用者来说如同一个实际的表,具有和表相通的一般select操作,而其实际上是一个视图,一个定期刷新数据的视图(具体刷新时间在定义物化视图的时候已有定义),使用物化视图可以实现视图的所有功能,而物化视图确不是在使用时才读取,大大提高了读取速度,特别适用抽取大数据量表某些信息以及数据链连接表使用.

具体语法如下:

```
CREATE MATERIALIZED VIEW an_user_base_file_no_charge
             REFRESH COMPLETE START WITH SYSDATE
            NEXT TRUNC(SYSDATE+29)+5.5/24
```

```
   as
select distinct user_no
from cw_arrearage t
where (t.mon = dbms_tianjin.getLastMonth or
       t.mon = add_months(dbms_tianjin.getLastMonth, -1))
drop materialized view an_user_base_file_no_charge;
```

补充：

Oracle的物化视图提供了强大的功能，可以用于预先计算并保存表连接或聚集等耗时较多的操作的结果，这样，在执行查询时，就可以避免进行这些耗时的操作，而从快速的得到结果。物化视图有很多方面和索引很相似：使用物化视图的目的是为了提高查询性能；物化视图对应用透明，增加和删除物化视图不会影响应用程序中SQL语句的正确性和有效性；物化视图需要占用存储空间；当基表发生变化时，物化视图也应当刷新。

物化视图可以分为以下三种类型：包含聚集的物化视图；只包含连接的物化视图；嵌套物化视图。三种物化视图的快速刷新的限制条件有很大区别，而对于其他方面则区别不大。创建物化视图时可以指定多种选项，下面对几种主要的选择进行简单说明：

创建方式（Build Methods）：包括BUILD IMMEDIATE和BUILD DEFERRED两种。BUILD IMMEDIATE是在创建物化视图的时候就生成数据，而BUILD DEFERRED则在创建时不生成数据，以后根据需要在生成数据。默认为BUILD IMMEDIATE。

查询重写（Query Rewrite）：包括ENABLE QUERY REWRITE和DISABLE QUERY REWRITE两种。分别指出创建的物化视图是否支持查询重写。查询重写是指当对物化视图的基表进行查询时，Oracle会自动判断能否通过查询物化视图来得到结果，如果可以，则避免了聚集或连接操作，而直接从已经计算好的物化视图中读取数据。默认为DISABLE QUERY REWRITE。

刷新（Refresh）：指当基表发生了DML操作后，物化视图何时采用哪种方式和基表进行同步。刷新的模式有两种：ON DEMAND和ON COMMIT。ON DEMAND指物化视图在用户需要的时候进行刷新，可以手工通过DBMS_MVIEW.REFRESH等方法来进行刷新，也可以通过JOB定时进行刷新。ON COMMIT指出物化视图在对基表的DML操作提交的同时进行刷新。刷新的方法有四种：FAST、COMPLETE、FORCE和NEVE*。**ST刷新采用增量刷新，只刷新自上次刷新以后进行的修改。COMPLETE刷新对整个物化视图进行完全的刷新。如果选择FORCE方式，则Oracle在刷新时会去判断是否可以进行快速刷新，如果可以则采用FAST方式，否则采用COMPLETE的方式。NEVER指物化视图不进行任何刷新。默认值是FORCE ON DEMAND。

在建立物化视图的时候可以指定ORDER BY语句，使生成的数据按照一定的顺序进行保存。不过这个语句不会写入物化视图的定义中，而且对以后的刷新也无效。

物化视图日志：如果需要进行快速刷新，则需要建立物化视图日志。物化视图日志根据不同物化视图的快速刷新的需要，可以建立为ROWID或PRIMARY KEY类型的。还可以选择是否包括SEQUENCE、INCLUDING NEW VALUES以及指定列的列表。

可以指明ON PREBUILD TABLE语句将物化视图建立在一个已经存在的表上。这种情况下，物化视图和表必须同名。当删除物化视图时，不会删除同名的表。这种物化视图的查询重写要求参数QUERY_REWRITE_INTEGERITY必须设置为trusted或者stale_tolerated。

物化视图可以进行分区。而且基于分区的物化视图可以支持分区变化跟踪（PCT）。具有这种特性的物化视图，当基表进行了分区维护操作后，仍然可以进行快速刷新操作。对于聚集物化视图，可以在GROUP BY列表中使用CUBE或ROLLUP，来建立不同等级的聚集物化视图。

20，**当用户进程出错，哪个后台进程负责清理它？**PMON：进程监控进程

21，**哪个后台进程刷新**materialized views?The Job Queue Processes：作业队列进程

22，**如何判断哪个**session**正在连结以及它们等待的资源**?V$SESSION / V$SESSION_WAIT

23，**如何分辨某个用户是从哪台机器登陆**ORACLE**的：**SELECT machine , terminal FROM V$SESSION

24，**如何查看系统被锁的事务时间**：select * from v$locked_object ;

25．**如何以**archivelog**的方式运行**oracle：在init.Ora中，log_archive_start = true，重启。

26．**怎么获取有哪些用户在使用数据库：**select username from v$session;

27．**数据表中的字段最大数是多少：**表或视图中的最大列数为 1000

28．**怎样查得数据库的**SID ：select name from v$database; 也可以直接查看 init.ora文件

29，**创建**Sequence：create sequence zxk increment by 10 start with 10 maxvalue 1000 cycle ;

30，oracle日志

Oracle9i中有2种日志，一种称为Redo Log（重做日志），另一种叫做Archive Log（归档日志）。重做日志redo log file是LGWR进程（日志写入进程）从Oracle实例中的redo log buffer写入的，是循环利用的。就是说一个redo log file (group) 写满后，才写下一个。归档日志archive log是当数据库运行在归档模式下时，一个redo log file(group)写满后，由ARCn进程（归档进程）将重做日志的内容备份到归档日志文件下，然后这个redo log file(group)才能被下一次使用。不管数据库是否是归档模式，重做日志是肯定要写的。而只有数据库在归档模式下，重做日志才会备份，形成归档日志。归档日志结合全备份，用于数据库出现问题后的恢复使用。

31，如何强制进行 Log Switch?

当一个redo log file被写满了之后要换另外一个redo log file，这个时候要用到的操作叫log switch。

SQL>ALTER SYSTEM SWITCH LOGFILE;

32，Coalescing做了什么?

Coalescing针对于字典管理的tablespace进行碎片整理，将临近的小extents合并成单个的大extent.

33，oracle表空间

Oracle数据库（tablespace）是由若干个表空间构成的。任何数据库对象在存储时都必须存储在某个表空间中。表空间对应于若干个磁盘文件，即表空间是由一个或多个磁盘文件构成的。**表空间相当于操作系统中的文件夹**，也是数据库逻辑结构与物理文件之间的一个映射。每个数据库至少有一个表空间，表空间的大小等于所有从属于它的数据文件大小的总和。

在Oracle 10g中有以下几种比较特殊的表空间：

（1）系统表空间

系统表空间（system tablespace）是每个Oracle数据库都必须具备的。其功能是在系统表空间中存放诸如表空间名称、表空间所含数据文件等数据库管理所需的信息。系统表空间的名称是不可更改的。系统表空间必须在任何时候都可以用，也是数据库运行的必要条件。因此，系统表空间是不能脱机的。**系统表空间包括数据字典、存储过程、触发器和系统回滚段**。为避免系统表空间产生存储碎片以及争用系统资源的问题，应创建一个独立的表空间用来单独存储用户数据。

（2）SYSAUX表空间

SYSAUX表空间是随着数据库的创建而创建的，它充当SYSTEM的辅助表空间，主要存储除数据字典以外的其他对象。SYSAUX也是许多Oracle 数据库的默认表空间，它减少了由数据库和DBA管理的表空间数量，降低了SYSTEM表空间的负荷。

（3）临时表空间

相对于其他表空间而言，临时表空间（temp tablespace）主要用于存储Oracle数据库运行期间所产生的临时数据。数据库可以建立多个临时表空间。当数据库关闭后，临时表空间中所有数据将全部被清除。除临时表空间外，其他表空间都属于永久性表空间。

（4）撤销表空间

用于保存Oracle数据库撤销信息，即保存用户回滚段的表空间称之为回滚表空间（或简称为RBS撤销表空间（undo tablespace））。在Oracle8i中是rollback tablespace，从Oracle9i开始改为undo tablespace。**在Oracle 10g中初始创建的只有6个表空间**sysaux、system、temp、undotbs1、example**和**users**。**其中temp是临时表空间，undotbs1是undo撤销表空间。

34，**创建用户时，需要赋予新用户什么权限才能使它联上数据库。**CONNECT

35，**如何在**tablespace**里增加数据文件**?ALTER TABLESPACE ADD DATAFILE SIZE

36，**如何变动数据文件的大小**?ALTER DATABASE DATAFILE RESIZE ;

37，**哪个**VIEW**用来检查数据文件的大小**? DBA_DATA_FILES

38，**哪个**VIEW**用来判断**tablespace**的剩余空间？**DBA_FREE_SPACE

39，如何判断谁往数据库里面插入了一条记录？

三种方法：事先打开审计功能或者在表上建立触发器，事后可以通过logmnr查看。

1）数据库审计功能：

先设置audit_trail参数，决定审计结果保存地点； 然后执行audit insert on schema.table_name whenever successful;这样就行了，在有人做insert操作后，根据audit_trail参数到相应位置去看审计结果就行了。

2）触发器：

```
create or replace trigger tgname
after insert
on tbname -->判断此表是否被插入记录
for each row
begin
     insert into ta(日期) values(sysdate) ;
     commit;
end ;
```

建触发器其实也是审计，是基于值的审计，比数据库审计慢，不过审计的内容可以更详细。

3）logmnr查看日志

35，如何重构索引？

```
SQL>ALTER INDEX indexname REBUILD;
```

快速重建索引：

```
alter session set workarea_size_policy=manual;
          oracle 10G以上默认auto,修改成manual才能设置排序区大小。
alter session set sort_area_size=1073741824;
alter session set sort_area_retained_size=1073741824;
          设置排序区大小为1G
alter session set db_file_multiblock_read_count=128;
alter index IDX_POI_ID_NAME rebuild online parallel 4 nologging---4个并行操作。
```

需要注意的是在并行重构完成以后，一定要取消索引的并行度，否则，在OLTP环境中，可能会因为意外的使用并行而出现严重性能问题。alter index IDX_POI_ID_NAME noparallel;

36，oracle partitioning

Oracle 分区是 Oracle9i 企业版的一个选项，可以增强各种应用程序的可管理性、性能和可用性。**分区允许将表、索引以及索引编排表细分为更小的段**，从而能在更细的粒度级管理和访问这些数据库对象。Oracle 提供丰富的分区模式来满足每一种商务需求。而且，由于它在 SQL 语句中是完全透明的，分区几乎可应用于任何应用程序。

分区的优点

分区能改善各种应用程序的可管理性、性能和可用性，具有非常多的优点。分区往往根据数量级提高某些查询或维护操作的性能。另外，分区可以极大地简化常见的管理任务。分区还可以使数据库设计人员和管理人员能够处理一些最前沿的应用程序而引发的最棘手的难题。分区在构建 TB 级的系统或对可用性要求极高的系统时非常关键。

分区的基本知识

分区允许将表、索引或索引编排表细分为更小的段。数据库对象的每个段就叫一个区。每个区有自己的名称，也可以具备自己的存储特征。从数据库管理员的角度看，分区的对象有多个段，既可以一起管理也可单独管理。这就赋予管理人员相当大的灵活性来管理分区的对象。然而，从应用的角度看，分区的表与不分区的表是一致的；在通过 SQL DML 命令访问分区表时不需要做任何的修改。

表通过使用‘分区键’分区；分区键是确定某个行所在区的一组列。

Oracle9i 提供四种表的分区技术：

1）范围分区：每个区由一系列分区键的值来指定（对于将日期列作为分区键的表，‘1 月-2001年’区包含分区键值为‘01-1 月-2001’至‘31-1 月-2001’的所有行）

2）列表分区：每个区由一列分区键值指定（对于将区域列作为分区键值的表，‘北美州’区可能包含的值有‘加拿大’、‘美国’以及‘墨西哥’）

3）哈希分区：哈希算法应用于分区键值，来确定某个行的区

4）范围-哈希组合分区：即范围和哈希分区技术的结合。表首先进行范围分区，然后每个范围区再单独通过哈希分区技术进行分区。

索引编排表既可做范围分区也可做哈希分区。

Oracle9i 还提供三种类型的分区索引：

1）本地索引：本地索引是分区表中的一种索引，分区方法与基本分区表的完全一样。本地索引的每个区只对应于基表的一个区。

2）全局分区索引：全局分区索引是分区或非分区表中的索引，通过该表中的不同分区键分区。全局分区索引只能采用范围分区法。例如，表可根据月份进行范围分区，这样就有 12 个区，而该表的索引使用不同的分区键进行范围分区，就会有不同数量的区。

3）全局非分区索引：全局非分区索引基本上与非分区表的索引一致。索引结构未被分区。

Oracle 提供一套强健的技术用于表、索引和索引编排表的分区，因此分区功能可最优地应用于任何商务环境中的任何应用程序。 Oracle 还另外提供了一组完整的 SQL 命令用于管理分区表。其中的命令包括添加新区、删除区、拆分分区以及合并分区。

通过限制要检查或操作的数据量和启用并行执行，Oracle 分区选项提供许多性能的优点。这些特性包括：

1）分区修剪 (Partitioning Pruning)：分区修剪是最简单也是最有效的通过分区改善性能的方法。分区修剪通常根据几个数量级改善查询性能。例如，假定一个应用程序中有一个包含订单历史记录的订单表，而且该表已经按周进行了分区。请求一周内定单的查询只会访问订单表的一个区。如果订单表有 2 年的历史数据，该查询将访问一个区而不是 104 个区。仅仅由于分区修剪功能的作用，该查询的执行速度实际上快了 100 倍。分区修剪与 Oracle 其它所有的性能特性一起发挥作用。Oracle 将分区修剪功能与任何的索引技术、连接技术或并行访问方法一起使用。

2）智能化分区连接 (Partition-wise Join)：分区也可通过使用被称为智能化分区连接的技术改善多表连接的性能。智能化分区连接可用于将两个表连接在一起，而且这两个表都在连接键上分区。智能化分区连接将大型的连接拆分为小的连接，在每个区间执行，减少了整个连接的时间。这显著地提高了串行和并行执行的性能。

3）更新和删除的并行执行：分区能够并行执行 UPDATE、DELETE 和 MERGE 语句。当访问分区和非分区的数据库对象时，Oracle 会并行化 SELECT 语句和 INSERT 语句。不过，为了并行化 UPDATE、DELETE 和 MERGE 语句，目标表必须进行分区。并行执行这些 SQL 操作可极大地改善性能，尤其是对涉及大量数据的 UPDATE、DELETE 或 MERGE 操作。

37，刚编译了一个PL/SQL Package但是有错误报道，如何显示出错信息?

SHOW ERRORS

38，如何搜集表的各种状态数据?

ANALYZE：用于分析表或者索引结构的一致性,判断索引与表间是否匹配<cascade>,有没有坏块,数据是不是正确分布在正确的分区中,索引压缩效率等。DBMS_UTILITY.ANALYZE_SCHEMA是等同于analyze的，只是DBMS_UTILITY是在PLSQL中调用了analyze命令。另外ORACLE提供了dbms_stats，该包提供了分析优化相关的信息的更强的功能，但是他不能分析非优化相关的信息。

analyze命令分两种，一种是分析优化相关的信息，语句为analyze table [index] compute[estimate] statistics [for 语句];如果要分析非优化相关的信息，语句如analyze index ... validate structure; analyze table ... validate structure [cascade]等。

39，oracle trace

SQL Trace主要是对数据库进行SQL监测，可以随时监测和调整作用于数据的应用程序。比如ERP系统它的应用界面很多，涉及的底层操作也很多，如果想知道在某个界面的操作在底层数据库执行了哪些SQL语句，就需要开启Trace功能记录下这些SQL操作，方便开发人员了解上层应用程序对数据库做了哪些动作。

```
alter session set sql_trace = true; /*开启*/
alter session set sql_trace = false; /*关闭*/
```

DBMS_SESSION.SET_SQL_TRACE也可以启动session级别的sql trace。

40，IMPORT和SQL*LOADER 这2个工具的不同点?

这两个ORACLE工具都是用来将数据导入数据库的。区别是：IMPORT工具只能处理由另一个ORACLE工具EXPORT生成的数据。而SQL*LOADER可以导入不同的ASCII格式的数据源。

41，用于网络连接的2个文件?

TNSNAMES.ORA ， SQLNET.ORA

42，有一个A 数据库，分别复制到B和C。 B 要求每次A数据更新B也同时更新，C 每天更新一次就行，如何制定复制策略!

a->b

1)、如果使用SQL Server复制功能，那么让a->b使用事务性复制方式(同步复制)。

2)、如果表不多，也可以自己写触发器，利用linkserver+distribute transaction。

a->c

1)、如果使用SQL Server复制功能，那么让a->b使用快照复制方式，在某一时间点进行一次性复制。

2)、也可以自己写bat，将a备份后，通过ftp传输备份介质，恢复c。(比较麻烦，不推荐)

43， 有一个数据库200G大小，每天增加50M 允许用户随时访问，制定备份策略(详细说明)。

这种情况可以采用增量备份方式。每周日做一次全备份，周一到周六作增量备份(由于数据量较少，可以考虑每30分钟增量备份一次)。这样可以尽量减少性能消耗，而且如果transaction log丢失的情况下，可以保证最多丢失30分钟数据。

44，管理50台数据库，日常工作是检查数据库作业是否完成，你该如何完成这项检查工作?

在每台机器上建立linkserver，然后在DBA管理服务器上做个分布式视图，每次查询该视图，各个机器上的作业情况一目了然。分布式视图写法：

```
create view vw_job
as
select '机器一' as MName,* from linkserver1..sysjobactivity
union all
select '机器二' as MName,* from linkserver2..sysjobactivity
union all
select '机器三' as MName,* from linkserver3..sysjobactivity
。。。
```

45，数据库三级模式：

数据库结构分为3级：面向用户或应用程序员的用户级、面向建立和维护数据库人员的概念级、面向系统程序员的物理级。用户级对应外模式，概念级对应模式，物理级对应内模式，使不同级别的用户对数据库形成不同的视图。所谓视图，就是指观察、认识和理解数据的范围、角度和方法，是数据库在用户“眼中"的反映，很显然，不同层次(级别)用户所“看到’’的数据库是不相同的。

1）模式.

模式又称概念模式或逻辑模式，对应于概念级。它是由数据库设计者综合所有用户的数据，按照统一的观点构造的全局逻辑结构，是对数据库中全部数据的逻辑结构和特征的总体描述，是所有用户的公共数据视图(全局视图)。它是由数据库管理系统提供的数据模式描述语言(Data Description Language，DDL)来描述、定义的，体现、反映了数据库系统的整体观。

2）外模式

外模式又称子模式，对应于用户级。它是某个或某几个用户所看到的数据库的数据视图，是与某一应用有关的数据的逻辑表示。外模式是从模式导出的一个子集，包含模式中允许特定用户使用的那部分数据。用户可以通过外模式描述语言来描述、定义对应于用户的数据记录(外模式)，也可以利用数据操纵语言(Data Manipulation Lang uage，DML)对这些数据记录进行。外模式反映了数据库的用户观。

3）内模式

内模式又称存储模式，对应于物理级，它是数据库中全体数据的内部表示或底层描述，是数据库最低一级的逻辑描述，它描述了数据在存储介质上的存储方式翱物理结构，对应着实际存储在外存储介质上的数据库。内模式由内模式描述语言来描述、定义，它是数据库的存储观。在一个数据库系统中，只有唯一的数据库， 因而作为定义 、描述数据库存储结构的内模式和定义、描述数据库逻辑结构的模式，也是惟一的，但建立在数据库系统之上的应用则是非常广泛、多样的，所以对应的外模式不是惟一的，也不可能是惟一的。

三级模式间的映射

数据库的三级模式是数据库在三个级别 (层次)上的抽象，使用户能够逻辑地、抽象地处理数据而不必关心数据在计算机中的物理表示和存储。实际上，对于一个数据库系统而言一有物理级数据库是客观存在的，它是进行数据库操作的基础，概念级数据库中不过是物理数据库的一种逻辑的、抽象的描述(即模式)，用户级数据库则是用户与数据库的接口，它是概念级数据库的一个子集(外模式)。 用户应用程序根据外模式进行数据操作，通过外模式一模式映射，定义和建立某个外模式与模式间的对应关系，将外模式与模式联系起来，当模式发生改变时，只要改变其映射，就可以使外模式保持不变，对应的应用程序也可保持不变；另一方面，通过模式一内模式映射，定义建立数据的逻辑结构(模式)与存储结构(内模式)间的对应关系，当数据的存储结构发生变化时，只需改变模式一内模式映射，就能保持模式不变，因此应用程序也可以保持不变。

46， 关系数据库管理系统能实现的专门关系运算？

选择、连接和投影

47，oracle RBO，CBO

ORACLE 提供了CBO、RBO两种SQL优化器。RBO是Rule Based Optimizer。CBO是Cost Based Optimizer。CBO在ORACLE7 引入，但在ORACLE8i 中才成熟。ORACLE 已经明确声明在ORACLE9i之后的版本中(ORACLE 10G)，RBO将不再支持。因此选择CBO 是必然的趋势。CBO和 RBO作为不同的SQL优化器，对SQL语句的执行计划产生重大影响，如果要对现有的应用程序从RBO向CBO移植，则必须充分考虑这些影响，避免SQL语句性能急剧下降；但是，对新的应用系统，则可以考虑直接使用CBO，在CBO模式下进行SQL语句编写、分析执行计划、性能测试等工作，这需要开发者对CBO的特性比较熟悉。

在CBO下写SQL语句的注意事项：

1）RBO自ORACLE 6版以来被采用，有着一套严格的使用规则，只要你按照它去写SQL语句，无论数据表中的内容怎样，也不会影响到你的“执行计划”，也就是说对数据不“敏感”；CBO计算各种可能“执行计划”的“代价”，即cost，从中选用cost最低的方案，作为实际运行方案。各“执行计划”的cost的计算根据，依赖于数据表中数据的统计分布，ORACLE数据库本身对该统计分布并不清楚，必须要分析表和相关的索引（使用ANALYZE 命令），才能搜集到CBO所需的数据。

2）使用CBO 时，编写SQL语句时，不必考虑"FROM" 子句后面的表或视图的顺序和"WHERE" 子句后面的条件顺序；ORACLE自7版以来采用的许多新技术都是基于CBO的，如星型连接排列查询，哈希连接查询，函数索引，和并行查询等。

3）一般而言，CBO所选择的“执行计划”都不会比RBO的“执行计划”差，而且相对而言，CBO对程序员的要求没有RBO那么苛刻，节省了程序员为了从多个可能的“执行计划”中选择一个最优的方案而花费的调试时间，但在某些场合下也会存在问题。较典型的问题有：有时，表明明建有索引，但查询过程显然没有用到相关的索引，导致查询过程耗时漫长，占用资源巨大，这时就需要仔细分析执行计划，找出原因。例如，可以看连接顺序是否允许使用相关索引。假设表emp的deptno列上有索引，表dept的列deptno上无索引，WHERE语句有emp.deptno=dept.deptno条件。在做NL连接时，emp做为外表，先被访问，由于连接机制原因，外表的数据访问方式是全表扫描，emp.deptno上的索引显然是用不上，最多在其上做索引全扫描或索引快速全扫描。

4）如果一个语句使用 RBO的执行计划确实比CBO 好，则可以通过加 " rule" 提示，强制使用RBO。

5）使用CBO 时，SQL语句 "FROM" 子句后面的表，必须全部使用ANALYZE 命令分析过，如果"FROM" 子句后面的是视图，则此视图的基础表，也必须全部使用ANALYZE 命令分析过；否则，ORACLE 会在执行此SQL语句之前，自动进行ANALYZE 命令分析，这会极大导致SQL语句执行极其缓慢。

6）使用CBO 时，SQL语句 "FROM" 子句后面的表的个数不宜太多，因为CBO在选择表连接顺序时，会对"FROM" 子句后面的表进行阶乘运算，选择最好的一个连接顺序。假如"FROM" 子句后有6个表，则其可选择的连接顺序就是6*5*4*3*2*1 = 720

种，CBO 选择其中一种，而如果"FROM" 子句后有12个表，则其可选择的连接顺序就是12*11*10*9*8*7*6*5*4*3*2*1=479001600 种，可以想象从中选择一种，会消耗多少CPU 时间？如果实在是要访问很多表，则最好使用 ORDER 提示，强制使用"FROM" 子句表固定的访问顺序。

7）使用CBO 时，SQL语句中不能引用系统数据字典表或视图，因为系统数据字典表都未被分析过，可能导致极差的“执行计划”。但是不要擅自对数据字典表做分析，否则可能导致死锁，或系统性能严重下降。

8）使用CBO 时，要注意看采用了哪种类型的表连接方式。ORACLE**的共有**Sort Merge Join（SMJ）、Hash Join（HJ）**和** Nested Loop Join（NL）。CBO有时会偏重于SMJ 和 HJ，但在OLTP 系统中，NL 一般会更好，因为它高效的使用了索引。在两张表连接，且内表的目标列上建有索引时，只有Nested Loop才能有效地利用到该索引。SMJ即使相关列上建有索引，最多只能因索引的存在，避免数据排序过程。HJ由于须做HASH运算，索引的存在对数据查询速度几乎没有影响。

9）使用CBO 时，必须保证为表和相关的索引搜集足够的统计数据。对数据经常有增、删、改的表最好定期对表和索引进行分析，可用SQL语句“analyze table xxx compute statistics for all indexes;"ORACLE掌握了充分反映实际的统计数据，才有可能做出正确的选择。

10）使用CBO 时，要注意被索引的字段的值的数据分布，会影响SQL语句的执行计划。例如：表emp，共有一百万行数据，但其中的emp.deptno列，数据只有4种不同的值，如10、20、30、40。虽然emp数据行有很多，ORACLE缺省认定表中列的值是在所有数据行均匀分布的，也就是说每种deptno值各有25万数据行与之对应。假设SQL搜索条件DEPTNO=10，利用deptno列上的索引进行数据搜索效率，往往不比全表扫描的高，ORACLE理所当然对索引“视而不见”，认为该索引的选择性不高。

优化模式包括Rule、Choose、First rows、All rows**四种方式：**

1）Rule：基于规则的方式。

2）Choose：默认的情况下Oracle用的便是这种方式。指的是当一个表或或索引有统计信息，则走CBO的方式，如果表或索引没统计信息，表又不是特别的小，而且相应的列有索引时，那么就走索引，走RBO的方式。

3）First Rows：它与Choose方式是类似的，所不同的是当一个表有统计信息时，它将是以最快的方式返回查询的最先的几行，从总体上减少了响应时间。

4）All Rows:也就是我们所说的Cost的方式，当一个表有统计信息时，它将以最快的方式返回表的所有的行，从总体上提高查询的吞吐量。没有统计信息则走RBO的方式。

设定选用哪种优化模式：

1）Instance级别我们可以通过在initSID.ora文件中设定

OPTIMIZER_MODE=RULE/CHOOSE/FIRST_ROWS/ALL_ROWS

如果没设定OPTIMIZER_MODE参数则默认用的是Choose方式。

2）Sessions级别通过ALTER SESSION SET OPTIMIZER_MODE=RULE/CHOOSE/FIRST_ROWS/ALL_ROWS来设定。

3）语句级别用Hint（/*+ ... */）来设定

为什么表的某个字段明明有索引，但执行计划却不走索引？

1）优化模式是all_rows的方式

2）表作过analyze，有统计信息(最可能的就是统计信息有误)

3）表很小，上文提到过的，Oracle的优化器认为不值得走索引。

48，oracle访问数据库的存取方式：

1）全表扫描：（Full Table Scans, FTS）

为实现全表扫描，Oracle读取表中所有的行，并检查每一行是否满足语句的WHERE限制条件。一个多块读操作可以使一次I/O能读取多块数据块 (db_block_multiblock_read_count参数设定)，而不是只读取一个数据块，这极大的减少了I/O总次数，提高了系统的吞吐量，所以利用多块读的方法可以十分高效地实现全表扫描，而且只有在全表扫描的情况下才能使用多块读操作。在这种访问模式下，每个数据块只被读一次。使用FTS的前提条件：在较大的表上不建议使用全表扫描，除非取出数据的比较多，超过总量的5% -- 10%，或你想使用并行查询功能时。

2）通过ROWID的表存取（Table Access by ROWID或rowid lookup）

行的ROWID指出了该行所在的数据文件、数据块以及行在该块中的位置，所以通过ROWID来存取数据可以快速定位到目标数据上，是Oracle存取单行数据的最快方法。这种存取方法不会用到多块读操作，一次I/O只能读取一个数据块。我们会经常在执行计划中看到该存取方法，如通过索引查询数据。

3）索引扫描（Index Scan或index lookup）

我们先通过index查找到数据对应的rowid值(对于非唯一索引可能返回多个rowid值)，然后根据rowid直接从表中得到具体的数据，这种查找方式称为索引扫描或索引查找(index lookup)。一个rowid唯一的表示一行数据，该行对应的数据块是通过一次i/o得到的，在此情况下该次i/o只会读取一个数据库块。

在索引中，除了存储每个索引的值外，索引还存储具有此值的行对应的ROWID值。索引扫描可以由2步组成：(1) 扫描索引得到对应的rowid值。 (2) 通过找到的rowid从表中读出具体的数据。每步都是单独的一次I/O，但是对于索引，由于经常使用，绝大多数都已经CACHE到内存中，所以第1步的 I/O经常是逻辑I/O，即数据可以从内存中得到。但是对于第2步来说，如果表比较大，则其数据不可能全在内存中，所以其I/O很有可能是物理I/O，这是一个机械操作，相对逻辑I/O来说，是极其费时间的。所以如果多大表进行索引扫描，取出的数据如果大于总量的5% -- 10%，使用索引扫描会效率下降很多。

根据索引的类型与where限制条件的不同，有4种类型的索引扫描：

1）索引唯一扫描(index unique scan)

2）索引范围扫描(index range scan)

3）索引全扫描(index full scan)

4）索引快速扫描(index fast full scan)

49，查看Oracle sql语句执行计划：

SQL>EXPLAIN PLAN FOR select * from dual;

50，pctused与pctfree

pctused（percent used）与pctfree（percent free）是Oracle的两个与性能相关的块级存储参数。

pctused：已用百分比，一个块的使用水位的百分比，这个水位将使该块返回到可用列表中去等待更多的插入操作。

pctfree：空闲百分比，用来为一个块保留的空间百分比，以防止在今后的更新操作中增加一列或多列值的长度。

freelist：可用列表是表中的一组可插入数据的可用块。

行连接：指一行存储在多个块中的情况，这是因为该行的长度超过了一个块的可用空间大小，即行链接是跨越多块的行。

行迁移：指一个数据行不适合放入当前块而被重新定位到另一个块（那里有充足的空间）中，但在原始块中保留一个指针的情形。原始块中的指针是必需的，因为索引的ROWID项仍然指向原始位置。

计算公式：

PCTFREE=(Average Row Size-Initial Row Size)*100/Average Row Size

PCTUSED=(100-PCTFREE) -Average Row Size * 100/Availabe Data Space

51，oracle表空间管理

表空间（Tablespace）——为数据库提供使用空间的逻辑结构，其对应物理结构是数据文件，一个表空间可以包含多个数据文件。本地管理表空间（Locally Managed Tablespace简称LMT）——8i以后出现的一种新的表空间的管理模式，通过本地位图来管理表空间的空间使用。 字典管理表空间（Dictionary-Managed Tablespace简称DMT）——8i以前包括以后都还可以使用的一种表空间管理模式，通过数据字典管理表空间的空间。

本地化管理：

就是指Oracle不再利用数据字典表来记录Oracle表空间里面的区的使用状况，而是在每个表空间的数据文件的头部加入了一个位图区，在其中记录每个区的使用状况。每当一个区被使用，或者被释放以供重新使用时，Oracle都会更新数据文件头部的这个记录，反映这个变化。

本地化管理的表空间的创建过程：

```
语法：CREATE TABLESPACE 表空间名字
            DATAFILE ’数据文件详细信息’
            [EXTENT MANAGEMENT { LOCAL
            {AUTOALLOCATE | UNIFORM [SIZE INTETER [K|M] ] } } ]
```

关键字EXTENT MANAGEMENT LOCAL **指定这是一个本地化管理的表空间**。对于系统表空间，只能在创建数据库的时候指定EXTENT MANGEMENT LOCAL，因为它是数据库创建时建立的第一个表空间。 在8i中，字典管理还是默认的管理方式，当选择了LOCAL关键字，即表明这是一个本地管理的表空间。当然还可以继续选择更细的管理方式：是 AUTOALLOCATE 还是

UNIFORM.。若为AUTOALLOCATE，则表明让Oracle来决定区块的使用办法；若选择了UNIFORM，则还可以详细指定每个区块的大小，若不加指定，则为每个区使用1M大小。

本地管理表空间与字典管理表空间相比大大提高了管理效率和数据库性能，其优点如下：

1）减少了递归空间管理

本地管理表空间是自己管理分配，而不是象字典管理表空间需要系统来管理空间分配，本地表空间是通过在表空间的每个数据文件中维持一个位图来跟踪在此文件中块的剩余空间及使用情况。并及时做更新。这种更新只对表空间的额度情况做修改而不对其他数据字典表做任何update操作，所以不会产生任何回退信息，从而大大减少了空间管理，提高了管理效率。同时由于本地管理表空间可以采用统一大小分配方式(UNIFORM),因此也大大减小了空间管理，提高了数据库性能。

2）系统自动管理extents大小或采用统一extents大小

本地管理表空间有自动分配(AUTOALLOCATE)和统一大小分配(UNIFORM)两种空间分配方式，自动分配方式(AUTOALLOCATE)是由系统来自动决定extents大小，而统一大小分配(UNIFORM)则是由用户指定extents大小。这两种分配方式都提高了空间管理效率。

3）减少了数据字典之间的竞争

因为本地管理表空间通过维持每个数据文件的一个位图来跟踪在此文件中块的空间情况并做更新，这种更新只修改表空间的额度情况，而不涉及到其他数据字典表，从而大大减少了数据字典表之间的竞争，提高了数据库性能。

4）不产生回退信息

因为本地管理表空间的空间管理除对表空间的额度情况做更新之外不修改其它任何数据字典表，因此不产生回退信息，从而大大提高了数据库的运行速度。

5）不需合并相邻的剩余空间

因为本地管理表空间的extents空间管理会自动跟踪相邻的剩余空间并由系统自动管理，因而不需要去合并相邻的剩余空间。同时，本地管理表空间的所有extents还可以具有相同的大小，从而也减少了空间碎片。

6）减少了空间碎片

7）对临时表空间提供了更好的管理

表空间管理方式转换：

字典管理表空间每当表或其他对象需要扩大的时候都检查其数据字典以确保有可用的空间分配给对象，然后给对象分配一个新区段并更新其可用空间信息。本地管理表空间保存数据文件本身的空间管理信息，而且表空间自动跟踪每个数据文件块的可用或已用状态。 在事务比较多的数据库中显然字典管理每次插入数据时都会检查数据字典，这就使得数据库性能有所损耗。

1）命令方式转移。

首先你要新建一个oracle表空间，在oracle 10g以后默认都是采用本地管理表空间的。

对于表空间的转移使用命令： ALTER TABLE temp MOVE TABLESPACE new_temp;

对于索引你需要重建： ALTER TABLE index REBUILD TABLESPACE new_index;

显然上面的方法并不适用于对system表进行转换，因为你不能建立2个同名的system表。

2）采用oracle提供的PL/SQL数据包中的DBMS_SPACE_ADMIN.

在转换system表前，你必须把所有的其他表空间转换为本地管理。

EXECUTE dbms_space_admin.tablespace_migrate_to_local（"tablespace"）； ——转行表空间

然后同样的方法将system表空间也进行转换。

EXECUTE dbms_space_admin.tablespace_migrate_to_local（"system"）；

使用这种方法很好，但是它建立的表空间没有automatic segment space managerment选项，所有字典管理表空间都是用默认手动段空间管理，而且在转换为本地管理的表空间是不能进行修改。还有一个缺点，就是表空间如果存在空间碎片的话，此方法也不能解决碎片问题。

=======》oracle建议采用第一种方法。

52，Oracle ASSM

Oracle 自动段空间管理。从10g开始，oracle开始提供Shrink的命令，假如我们的表空间中支持自动段空间管理 (ASSM),就可以使用这个特性缩小段，即降低HWM。

创建一个本地管理的表空间，采用段自动管理方式

```
create tablespace demo
datafile /ora01/oem/demo01.dbf
size 50m
EXTENT MANAGEMENT LOCAL           --一定是本地管理
SEGMENT SPACE MANAGEMENT AUTO.  --ASSM管理的标志
```

53，如果一个表在2004-08-04 10:30:00 被drop，在有完善的归档和备份的情况下，如何恢复?

手工拷贝回所有备份的数据文件

```
startup mount;
alter database recover automatic until time '2004-08-04:10:30:00';
alter database open resetlogs;
```

54，rman是什么,有何特点?

RMAN(Recovery Manager)是DBA的一个重要工具，用于备份、还原和恢复oracle数据库，RMAN 可以用来备份和恢复数据库文件、归档日志、控制文件、系统参数文件,也可以用来执行完全或不完全的数据库恢复。RMAN有三种不同的用户接口：COMMAND LINE方式、GUI 方式(集成在OEM 中的备份管理器)、API 方式(用于集成到第三方的备份软件中)。

具有如下特点：

1)功能类似物理备份，但比物理备份强大N倍;
2)可以压缩空块;
3)可以在块水平上实现增量;
4)可以把备份的输出打包成备份集，也可以按固定大小分割备份集;
5)备份与恢复的过程可以自动管理;
6)可以使用脚本(存在Recovery catalog 中)
7)可以做坏块监测

55，standby的特点

备用数据库(standby database)：ORACLE推出的一种高可用性(HIGH AVAILABLE)数据库方案，在主节点与备用节点间通过日志同步来保证数据的同步，备用节点作为主节点的备份，可以实现快速切换与灾难性恢复,从920开始，还开始支持物理与逻辑备用服务器。

Oracle 9i中的三种数据保护模式分别是：

1)、MAXIMIZE PROTECTION ：最大数据保护与无数据分歧，LGWR将同时传送到备用节点，在主节点事务确认之前，备用节点也必须完全收到日志数据。如果网络不好，引起LGWR不能传送数据，将引起严重的性能问题，导致主节点DOWN机。

2)、MAXIMIZE AVAILABILITY ：无数据丢失模式，允许数据分歧，允许异步传送。正常情况下运行在最大保护模式，在主节点与备用节点的网络断开或连接不正常时，自动切换到最大性能模式，主节点的操作还是可以继续的。在网络不好的情况下有较大的性能影响。

3)、MAXIMIZE PERFORMANCE：这种模式应当可以说是从8i继承过来的备用服务器模式，异步传送，无数据同步检查，可能丢失数据，但是能获得主节点的最大性能。9i在配置DATA GUARD的时候默认就是MAXIMIZE PERFORMANCE。

56，SGA主要有那些部分，主要作用是什么

db_cache:

数据库缓存（Block Buffer）对于Oracle数据库的运转和性能起着非常关键的作用，它占据Oracle数据库SGA（系统共享内存区）的主要部分。Oracle数据库通过使用LRU 算法，将最近访问的数据块存放到缓存中，从而优化对磁盘数据的访问.

shared_pool:

共享池的大小对于Oracle 性能来说都是很重要的。共享池中保存数据字典高速缓冲和完全解析或编译的的PL/SQL 块和SQL

语句及控制结构

large_pool:

使用MTS配置时，因为要在SGA中分配UGA来保持用户的会话，就是用Large_pool来保持这个会话内存。 使用RMAN做备份的时候，要使用Large_pool这个内存结构来做磁盘I/O缓存器

java_pool:

为java procedure预备的内存区域,如果没有使用java proc,java_pool不是必须的

57，statspack有何认识

Oracle有提供一个用于诊断性能问题的工具包(statspack).当然，可能也有一些第三方的用于oracle性能诊断的工具，但是我想没有其它工具比ORACLE本身提供的工具更全面，更准确了。STATSPACK需要安装。安装后我们需要设置一下检查的时间间隔，实际上就是创建一个oracle的JOB。Statspack或根据我们设定的时间间隔来从oracle的动态性能视图中捕捉一些与性能相关的数据，然后根据一定的公式进行计算，生成一个有关于oracle各项性能指标的报告。报告罗列了一些实际的活动状况(负载，内存命中率等等)，最高等待事件，以及TOP SQL等内容，是我们进行性能诊断的一个比较综合的一个工具。

58，如果系统现在需要在一个很大的表上创建一个索引，你会考虑哪些因素，如何做以尽量减小对应用的影响。

在系统比较空闲时： 使用nologging选项(如果有dataguard则不可以使用nologging)，建索引时，Oracle会对数据按照索引进行排序,所以要增大的sort_ared_size(workarea_size_policy=manual) 或 pga_aggregate_target (workarea_size_policy=auto)

59，关于oracle自带的表

emp：

empno：员工编号； ename：员工名字； job：员工工种； mgr: 上司； hiredate：入职时间；sal： 基本工资；

comm： 补贴； deptno：所属部门编号；

dept：

deptno：部门编号； dname：部门名称； loc：地理位置；

salgrade：

grade： 工资等级； losal：最低限额； hisal：最高限额；

dual：

系统自带的一张空表； 可用于计算数据：select 2*3 from dual;

60， 回滚段的作用是什么

事务回滚：当事务修改表中数据的时候，该数据修改前的值(即前影像)会存放在回滚段中，当用户回滚事务(ROLLBACK)时，ORACLE将会利用回滚段中的数据前影像来将修改的数据恢复到原来的值。

事务恢复：当事务正在处理的时候，例程失败，回滚段的信息保存在undo表空间中，ORACLE将在下次打开数据库时利用回滚来恢复未提交的数据。

读一致性：当一个会话正在修改数据时，其他的会话将看不到该会话未提交的修改。当一个语句正在执行时，该语句将看不到从该语句开始执行后的未提交的修改(语句级读一致性)。当ORACLE执行Select语句时，ORACLE依照当前的系统改变号(SYSTEM CHANGE NUMBER-SCN)来保证任何前于当前SCN的未提交的改变不被该语句处理。可以想象：当一个长时间的查询正在执行时，若其他会话改变了该查询要查询的某个数据块，ORACLE将利用回滚段的数据前影像来构造一个读一致性视图。

61， 绑定变量是什么?绑定变量有什么优缺点?

绑定变量是相对文本变量来讲的,所谓文本变量是指在SQL直接书写查询条件，这样的SQL在不同条件下需要反复解析,绑定变量是指使用变量来代替直接书写条件，查询bind value在运行时传递，然后绑定执行。优点是减少硬解析,降低CPU的争用,节省shared_pool ;缺点是不能使用histogram,sql优化比较困难。

62，不借助第三方工具，怎样查看sql的执行计划

1）使用Explain Plan,查询PLAN_TABLE;

```
SQL> EXPLAIN   PLAN
      SET STATEMENT_ID='QUERY1'
      FOR
      SELECT *
      FROM a
      WHERE aa=1;
SQL> SELECT    operation, options, object_name, object_type, ID, parent_id
       FROM plan_table
      WHERE STATEMENT_ID = 'QUERY1'
   ORDER BY ID;
```

2）SQLPLUS中的SET TRACE 即可看到Execution Plan Statistics

```
   SET AUTOTRACE ON;
```

63，如何定位重要(消耗资源多)的SQL

使用CPU多的用户session

```
   SELECT    a.SID, spid, status, SUBSTR (a.program, 1, 40) prog, a.terminal,a.SQL_TEXT,
             osuser, VALUE / 60 / 100 VALUE
        FROM v$session a, v$process b, v$sesstat c
       WHERE c.statistic# = 12 AND c.SID = a.SID AND a.paddr = b.addr
   ORDER BY VALUE DESC;
```

64，如何跟踪某个session的SQL

利用TRACE 跟踪

```
   ALTER SESSION SET SQLTRACE ON;
   COLUMN SQL format a200;
   SELECT    machine, sql_text SQL
       FROM v$sqltext a, v$session b
      WHERE address = sql_address
        AND machine = '&A'
   ORDER BY hash_value, piece;
```

65，Oracle学习两大块：

一块是开发，一块是管理。开发主要是写写存储过程、触发器什么的，还有就是用Oracle的Develop工具做form。有点类似于程序员，需要有较强的逻辑思维和创造能力，是青春饭J；管理则需要对oracle数据库的原理有深刻的认识，有全局操纵的能力和紧密的思维，责任较大，因为一个小的失误就会down掉整个数据库，相对前者来说，后者更看重经验。

===>IF 走数据库路线：数据库管理的责任重大，很少公司愿意请一个刚刚接触oracle的人去管理数据库。对于刚刚毕业的年轻人来说，可以先选择做开发，有一定经验后转型，去做数据库的管理

===>深入学习的方向：

管理：可以考OCP（Oracle9i Certified Professional）证书，对oracle先有一个系统的学习，然后看Oracle Concepts、oracle online document,对oracle的原理会有更深入的了解，同时可以开始进行一些专题的研究如：RMAN、RAS、STATSPACT、DATAGUARD、TUNING、BACKUP&RECOVER等等。

开发：对于想做Oracle开发的，在了解完Oracle基本的体系结构之后，可以重点关注PL/SQL及Oracle的开发工具这一部分。PL/SQL主要是包括怎么写SQL语句，怎么使用Oracle本身的函数，怎么写存储过程、存储函数、触发器等。 Oracle的开发工具主要就是Oracle自己的Developer Suite（Oracle Forms Developer and Reports Developer这些），学会如何熟练使用这些工具。

66，oracle中的动态性能表：

表名	说明

V$ACCESS	显示数据库中的对象信息
V$ARCHIVE	数据库系统中每个索引的归档日志方面的信息
V$BACKUP	所有在线数据文件的状态
V$BGPROCESS	描述后台进程
V$CIRCUIT	有关虚拟电路信息
V$DATABASE	控制文件中的数据库信息
V$DATAFILE	控制文件中的数据文件信息
V$DBFILE	构成数据库所有数据文件
V$DB_OBJECT_CACHE	表示库高速缓存中被缓存的数据库对象
V$DISPATCHER	调度进程信息
V$ENABLEDPRIVS	那些特权接通
V$FILESTAT	文件读/写统计信息
V$FIXED_TABLE	显示数据库中所有固定表、视图和派生表
V$INSTANCE	当前实例状态
V$LATCH	每类闩锁的信息
V$LATCHHOLDER	当前闩锁占有者的信息
V$LATCHNAME	在V$LATCH表中表示的闩锁的译码闩锁名
V$LIBRARYCACHE	库高速缓冲存储管理统计
V$LICENSE	许可限制信息
V$LOADCSTAT	SQL*Loader在直接装入执行过程中的编译统计
V$LOCK	有关封锁和资源信息，不包含DDL封锁
V$LOG	控制文件中的日志文件信息
V$LOGFILE	有关日志文件信息
V$LOGHIST	控制文件中的日志历史信息
V$LOGHISTORY	日志历史中所有日志的归档日志名
V$NLS_PARAMETERS	NLS参数的当前值
V$OPEN_CURSOR	每一个用户会话期当前已打开和分析的光标
V$PARAMETER	当前参数值的信息
V$PROCESS	当前活动进程的信息
V$QUEUE	多线索信息队列的信息
V$REVOVERY_LOG	需要完成介质恢复的归档日志
V$RECOVERY_FILE	需要介质恢复的文件状态
V$REQDIST	请求时间直方图，分为12个范围
V$RESOURCE	有关资源信息
V$ROLLNAME	所有在线回滚段的名字
V$ROLLSTAT	所有在线回滚段的统计信息
V$ROWCACHE	数据字典活动的统计信息（每一个包含一个数据字典高速缓存的统计信息）
V$SESSION	每一个当前会话期的会话信息
V$SESSION_WAIT	列出活动会话等待的资源或事件
V$SESSTAT	对于每一个当前会话的当前统计值
V$SESS_IO	每一个用户会话的I/O统计
V$SGA	系统全局区统计信息
V$SGASTAT	系统全局区的详细信息
V$SHARED_SERVER	共享服务器进程信息
V$SQLAREA	共享光标高速缓存区的统计信息，每一个有一个共享光标的统计信息
V$SQLTEXT	属于SGA中的共享SQL光标的SQL语句文本
V$STATNAME	在V$SESSTAT表中表示的统计信息的译码统计名
V$SYSSTAT	表V$SESSETA中当前每个统计的全面的系统值
V$THREAD	从控制文件中得到线索信息
V$TIMER	以百分之一秒为单位的当前时间
V$TRANSACTION	有关事务的信息
V$TYPE_SIZE	各种数据库成分的大小
V$VERSION	ORACLE Server中核心库成员的版本号，每个成员一行
V$WAITSTAT	块竞争统计，当时间统计可能时，才能更新该表

67，SQL **常用命令：**

数据检索： Select

数据维护(DML)： insert、update、delete

数据定义(DDL) ：create、drop、alert、rename、truncate

事务处理控制：commit、rollback 、savepoint

数据控制(DCL) ：Grant、revoke

68，SQL*Plus **的使用：**

1）文件命令

a) SAVE filename 把当前SQL缓冲区的内容存储在文件filename 中
b) GET filename 把文件filename 中的内容写入当前SQL缓冲区
c) START filename 执行存储在filename中的内容
d) @ filename 执行存储在 filename 中的内容
e) EDIT filename 打开文本编辑器，把当前SQL 缓冲区的内容写入文件afiedt.buf
f) SPOOL filename 把查询的数据结果存储在filename 中
g) EXIT 退出SQL*Plus

2）文本编辑命令

a) A[PPEND] text
b) C[HANGE]/old/new
c) CL[EAR]buff[ER]
d) DEL
e) I[NPUT] text
f) L[IST] n
g) N text

69，oracle数据字典的四大视图类型：

- user 用户拥有的对象
- all 用户可访问对象
- DBA 所有数据对象
- v$ 服务器性能对象

70，oracle通用数据类型：

1）BINARY_INTEGER： 基本数值整型，-2147483647 -----2147483647
2）NUMBER[(precision,scale)：] 基本浮点数值型
3）CHAR[(maximun_length)]： 固定长度的字符型，最大值为32760
4）LONG ：可变常字符型，最大长度为32760
5）LONG RAW：二进制型，最大长度为32760
6）VARCHAR2(maximum_length)： 可变长字符型，最大长度为32767
7）DATE ：日期和时间类型
8）BOOLEAN： 逻辑型（TRUE、FALSE 或NULL）

71，数据库引擎

数据库引擎是用于存储、处理和保护数据的核心服务。利用数据库引擎可控制访问权限并快速处理事务，从而满足企业内大多数需要处理大量数据的应用程序的要求。使用数据库引擎创建用于联机事务处理或联机分析处理数据的关系数据库。这包括创建用于存储数据的表和用于查看、管理和保护数据安全的数据库对象（如索引、视图和存储过程）。

78，Mysql引擎：

在缺省情况下，MySQL支持三个引擎：ISAM、MyISAM和HEAP。另外两种类型InnoDB和Berkley（BDB），也常常可以使用。

ISAM

ISAM是一个定义明确且历经时间考验的数据表格管理方法，它在设计之时就考虑到数据库被查询的次数要远大于更新的次数。因此，ISAM执行读取操作的速度很快，而且不占用大量的内存和存储资源。ISAM的两个主要不足之处在于，它不支 持事务处理，也不能够容错：如果你的硬盘崩溃了，那么数据文件就无法恢复了。如果你正在把ISAM用在关键任务应用程序

里，那就必须经常备份你所有的实时 数据，通过其复制特性，MySQL能够支持这样的备份应用程序。

MyISAM

MyISAM是MySQL的ISAM扩展格式和缺省的数据库引擎。除了提供ISAM 里所没有的索引和字段管理的大量功能，MyISAM还使用一种表格锁定的机制，来优化多个并发的读写操作。其代价是你需要经常运行OPTIMIZE TABLE命令，来恢复被更新机制所浪费的空间。MyISAM还有一些有用的扩展，例如用来修复数据库文件的MyISAMChk工具和用来恢复浪费空间的 MyISAMPack工具。MyISAM强调了快速读取操作，这可能就是为什么MySQL受到了Web开发如 此青睐的主要原因：在Web开发中你所进行的大量数据操作都是读取操作。所以，大多数虚拟主机提供商和Internet平台提供商（Internet Presence Provider，IPP）只允许使用MyISAM格式。

HEAP

HEAP允许只驻留在内存里的临时表格。驻留在内存里让HEAP要比ISAM和 MyISAM都快，但是它所管理的数据是不稳定的，而且如果在关机之前没有进行保存，那么所有的数据都会丢失。在数据行被删除的时候，HEAP也不会浪费 大量的空间。HEAP表格在你需要使用SELECT表达式来选择和操控数据的时候非常有用。要记住，在用完表格之后就删除表格。让我再重复一遍：在你用完 表格之后，不要忘记删除表格。

InnoDB和Berkley DB

InnoDB和Berkley DB（BDB）数据库引擎都是造就MySQL灵活性的技术的直接产品，这项技术就是MySQL++ API。在使用MySQL的时候，你所面对的每一个挑战几乎都源于ISAM和MyISAM数据库引擎不支持事务处理也不支持外来键。尽管要比ISAM和MyISAM引擎慢很多，但是InnoDB和BDB包括了对事务处理和外来键的支持，这两点都是前两个引擎所没有的。如前所述，如果你的设计需要这些特性 中的一者或者两者，那你就要被迫使用后两个引擎中的一个了。

数据库引擎设定与切换：

1）CREATE TABLE tblMyISAM (

```
    id INT NOT NULL AUTO_INCREMENT,
    PRIMARY KEY (id),
    value_a TINYINT
```

)TYPE=MyISAM;

2）ALTER TABLE tblMyISAM CHANGE TYPE=InnoDB;

3）SHOW TABLE STATUS FROM tblInnoDB;

79，Mysql6的新特性：（与mysql5比较）

1）新Falcon事务存储引擎（“Falcon存储引擎” ） 。

2）支持更多的Unicode字符集： utf16 ， utf32 ，和4字节utf8 。 这些字符集支持这些附加的Unicode字符集，也就是那些在基础多语言基础之外的字符 。

3）增加了 BACKUP DATABASE 和 RESTORE 语句来进行备份和还原操作. 见第6.3节， “使用MySQL备份” 。

4）改进INFORMATION_SCHEMA数据库，并增加了INFORMATION_SCHEMA.PARAMETERS 表，INFORMATION_SCHEMA.ROUTINES 增加了新列

5）对子查询和Join进行了优化,包括对MyISAD和InnoB存储引擎分散范围内的批量索引访问。

6）RESET SLAVE不再更改复制连接的参数；以前，它重置他们到命令行指定的数值

7）LOCK TABLES **语法已经扩展，支持不会自动事务提交的事务表锁**。在后面的 LOCK TABLES ... IN SHARE MODE 或者 LOCK TABLES ... IN EXCLUSIVE MODE 你可以使用未提及的表级锁，你也可以确保 LOCK TABLES 语句可以连续的得到多次的事务锁，增加额外的表格到锁集合，而无需解锁以前已经锁住的表格。当使用LOCK TABLES with IN SHARE MODE 或者在EXCLUSIVE MODE ,表级锁在事务结束前不会解锁。 使用LOCK TABLE获得的事务锁在事务结束时释放,包括显示的提交或者回滚，或者由于语句引起的隐式提交，或者由于链接关闭. LOCK TABLES的行为在 READ和WRITE锁时报出不变(也就是当不使用 IN SHARE MODE 或者 IN EXCLUSIVE MODE) 。

8）增强的XML功能，包括一个新的LOAD XML 语法

9）支持扩展的注释，包括表，列和索引。

80，oracle版本间比较：

Oracle 11g： g= grid computing

1.**数据库管理部分**

◆数据库重演(Database Replay)

这一特性可以捕捉整个数据的负载，并且传递到一个从备份或者standby数据库中创建的测试数据库上，然后重演负责以测试系统调优后的效果。

◆SQL重演(SQL Replay) 和前一特性类似。但是只是捕捉SQL负载部分，而不是全部负载。

◆计划管理（Plan Management） 这一特性允许你将某一特定语句的查询计划固定下来，无论统计数据变化还是数据库版本变化都不会改变她的查询计划。

◆自动诊断知识库（Automatic Diagnostic Repository ADR） 当Oracle探测到重要错误时，会自动创纪一个事件（incident），并且捕捉到和这一事件相关的信息，同时自动进行数据库健康检查并通知DBA。此外，这些信息还可以打包发送给Oracle支持团队。

◆事件打包服务(Incident Packaging Service) 如果你需要进一步测试或者保留相关信息，这一特性可以将与某一事件相关的信息打包。并且你还可以将打包信息发给oracle支持团队。

◆基于特性打补丁（Feature Based Patching） 在打补丁包时，这一特性可以使你很容易区分出补丁包中的那些特性是你正在使用而必须打的。企业管理器（EM）使你能订阅一个基于特性的补丁服务，因此企业管理器可以自动扫描那些你正在使用的特性有补丁可以打。

◆自动SQL优化(Auto SQL Tuning) 10g的自动优化建议器可以将优化建议写在SQL profile中。而在11g中，你可以让oracle自动将能3倍于原有性能的profile应用到SQL语句上。性能比较由维护窗口中一个新管理任务来完成。

◆访问建议器（Access Advisor） 11g的访问建议器可以给出分区建议，包括对新的间隔分区（interval partitioning）的建议。间隔分区相当于范围分区（range partitioning）的自动化版本，她可以在必要时自动创建一个相同大小的分区。范围分区和间隔分区可以同时存在于一张表中，并且范围分区可以转换为间隔分区。

◆自动内存优化（Auto Memory Tuning） 在9i中，引入了自动PGA优化；10g中，又引入了自动SGA优化。到了 11g，所有内存可以通过只设定一个参数来实现全表自动优化。你只要告诉oracle有多少内存可用，她就可以自动指定多少内存分配给PGA、多少内存分 配给SGA和多少内存分配给操作系统进程。当然也可以设定最大、最小阈值。

◆资源管理器（Resource Manager） 11g的资源管理器不仅可以管理CPU，还可以管理IO。你可以设置特定文件的优先级、文件类型和ASM磁盘组。

◆ADDM ，ADDM在10g被引入。11g中，ADDM不仅可以给单个实例建议，还可以对整个RAC（即数据库级别）给出建议。另外，还可以将一些指示（directive）加入ADDM，使之忽略一些你不关心的信息。

◆AWR 基线（AWR Baselines） AWR基线得到了扩展。可以为一些其他使用到的特性自动创建基线。默认会创建周基线。

2.PLSQL**部分**

◆结果集缓存（Result Set Caching） 这一特性能大大提高很多程序的性能。在一些MIS系统或者OLAP系统中，需要使 用到很多"select count(*)"这样的查询。在之前，我们如果要提高这样的查询的性能，可能需要使用物化视图或者查询重写的技术。在11g，我们就只需要加一个 /*+result_cache*/的提示就可以将结果集缓存住，这样就能大大提高查询性能。当然，在这种情况下，我们可能还要关心另外一个问题：完整性。因为在oracle中是通过一致性读来保证数据的完整性的。而显然，在这种新特性下，为提高性能，是从缓存中的结果集中读取数据，而不会从回滚段中读取数据的。关于这个问题，答案是完全能保证完整性。因为结果集是被独立缓存的，在查询期间，任何其他DML语句都不会影响结果集中的内容，因而可以保证数据的完整性。

◆对象依赖性改进 在11g之前，如果有函数或者视图依赖于某张表，一旦这张表发生结构变化，无论是否涉及到函数或视图所依赖的属性，都会使函数或视图变为invalid。在11g中，对这种情况进行了调整：如果表改变的属性与相关的函数或视图无关，则相关对象状态不会发生变化。

◆正则表达式的改进 在10g中，引入了正则表达式。这一特性大大方便了开发人员。11g，oracle再次对这一特性进行了改进。其中，增加了一个名为regexp_count的函数。另外，其他的正则表达式函数也得到了改进。

◆新SQL语法 => 我们在调用某一函数时，可以通过=>来为特定的函数参数指定数据。而在11g中，这一语法也同样可以出现在sql语句中了。例如，你可以写这样的语句：select f(x=>6) from dual;

◆对TCP包（utl_tcp、utl_smtp…）支持FGAC(Fine Grained Access Control)安全控制

◆增加了只读表（read-only table） 在以前，我们是通过触发器或者约束来实现对表的只读控制。11g中不需要这么麻烦了，可以直接指定表为只读表。

◆触发器执行效率提高了

◆设置触发器顺序 可能在一张表上存在多个触发器。在11g中，你可以指定它们的触发顺序，而不必担心顺序混乱导致数据混乱。

◆混合触发器（compound trigger） 这是11g中新出现的一种触发器。她可以让你在同一触发器中同时具有申明部分、before过程部分、after each row过程部分和after过程部分。

◆创建无效触发器(Disabled Trigger) 11g中，开发人员可以可以闲创建一个invalid触发器，需要时再编译她。

◆在非DML语句中使用序列（sequence） 在之前版本，如果要将sequence的值赋给变量，需要通过类似以下语句实现：select seq_x.next_val into v_x from dual;在11g中，不需要这么麻烦了，下面语句就可以实现：v_x := seq_x.next_val;

◆PLSQL_Warning 11g中，可以通过设置PLSQL_Warning=enable all，如果在"when others"没有错误爆出就发警告信息。

◆PLSQL的可继承性 可以在oracle对象类型中通过super（和java中类似）关键字来实现继承性。

◆编译速度提高 因为不在使用外部C编译器了，因此编译速度提高了。

◆改进了DBMS_SQL包 其中的改进之一就是DBMS_SQL可以接收大于32k的CLOB了。另外还能支持用户自定义类型和bulk操作。

◆增加了continue关键字 在PLSQL的循环语句中可以使用continue关键字了（功能和其他高级语言中的continue关键字相同）。

◆新的PLSQL数据类型——simple_integer 这是一个比pls_integer效率更高的整数数据类型。

3.**其他部分**

◆增强的压缩技术 可以最多压缩2/3的空间。

◆高速推进技术 可以大大提高对文件系统的数据读取速度。

◆增强了DATA Guard 可以创建standby数据库的快照，用于测试。结合数据库重演技术，可以实现模拟生成系统负载的压力测试。

◆在线应用升级 也就是热补丁——安装升级或打补丁不需要重启数据库。

◆数据库修复建议器 可以在错误诊断和解决方案实施过程中指导DBA。

◆逻辑对象分区 可以对逻辑对象进行分区，并且可以自动创建分区以方便管理超大数据库（Very Large Databases VLDBs）。

◆新的高性能的LOB基础结构

◆新的PHP驱动

Oracle 10g：

1）对新的架构支持

对 Intel 64 位平台的支持。支持 infiniband 。极大地改进了多层开发架构下的性能和可扩展能力。新的版本也借用了 Windows 操作系统对 Fiber 支持的优势。

2）高速数据处理能力

在这个版本中，一个新类型的表对象被引入。该表结构对大量插入和解析数据很有益处。这个表结构对 FIFO 的数据处理应用有着很好的支持。这样的应用在电信、生产应用中常常能够用到。通过使用这种优化的表结构能够对电信级的应用起到巨大的性能改进作用。

3）RAC workload 管理

一个新的服务框架。使得管理员作为服务来设置、管理监视应用负载。

4）针对 OLAP 的分区

通过对哈希分区的全局索引的支持可以提供大量的并发插入的能力

5）新的改进的调度器(Scheduler)

引入了一个新的数据库调度器，提供企业级调度功能。这个调度器可以使得管理员有能力在特定日期、特定时间调度 Job 。还有能力创建调度对象的库能够和既有的对象被其他的用户共享。

6）简化的数据库配置与升级

提供了预升级检查能力，有效地减少升级错误。去除了了很多和数据库配置有关的任务或者对其加以自动化。在初始安装的时候，所有数据库都被预配置包括在 OEM 环境中而无需建立一个管理资料库。补丁程序可以自动标记并自动从 Oracle Metalink 下载。

7）自动存储管理

新版本的数据库能够配置成使用 Oracle 提供的存储虚拟层（ Storage Virtualization Layer) 。自动并简化数据库

的存储。管理员现在可以管理少数的磁盘组而无需管理数千个文件--自动存储管理功能可以自动配置磁盘组，提供数据冗余和数据的优化分配。

8）自动的基于磁盘备份与恢复

10G也极大的简化了备份与恢复操作。这个改进被称作Disk based Recovery Area ，可以被一个联机Disk Cache 用来进行备份与恢复操作。备份可以调度成自动化操作，自动化优化调整。备份失败的时候，可以自动重启，以确保 Oracle 能够有一个一致的环境使用。

9）应用优化

以前的版本中， DBA 更多时候要手工对 SQL 语句进行优化调整。这里引入了一些新的工具，从此 DBA 无需手工做这些累人的事情。(这样的说法似乎有些太绝对）

10）自动化统计收集

为对象自动化收集优化统计。

11）自动化实例调整

DBA需要干预的越来越少么? 好消息还是坏消息?

12）自动化内存调整

上一个版本对 UGA 能够进行自动化 Tuning ，这版本能够对 SGA 相关的参数进行调整。这意味着 DBA 只需要对2 个内存参数进行配置：用户可用的总的内存数量和共享区的大小。

13）缩短应用和数据库升级的宕机时间

通过使用 standby 数据库。允许在不同版本的 standby 和产品数据库间切换。现有的联机重定义功能能够支持一步克隆所有相关的数据库对象。

14）回闪 (Flashback) 任何错误

该版本的 Oracle 也扩展了 Flashback 的能力。加了一个新类型的 Log 文件，该文件记录了数据库块的变化。这个新的 Log 文件也被自动磁盘备份和恢复功能所管理。如果有错误发生，例如针对不成功的批处理操作，DBA 可以运行 FlashBack。用这些 before Images 快速恢复整个数据库到先前的时间点--无须进行恢复操作，这个新功能也可以用到 Standby 数据库中。 Flashback 是数据库级别的操作，也能回闪整个表。既有的 FlashBack 查询的能力也已经加强。在这个版本中，管理员能够快速查看特定事务导致的变化。

15）Enhanced Data Guard Infrastructure

16）超大数据库的支持

可支持到 8E 的数据量。改进的存储、备份、恢复管理也对超大数据库有着很好的支持。分区可以支持索引组织表。

17）缩短信息周转时间

新版本的 Oracle 提供了加强的 ETL 功能。可以方便的构建大型数据仓库和多个数据集市。一个新的变化数据捕捉的框架允许管理员能够轻易的捕捉并发布数据的变化。新的 CDC 功能利用的是 Oracle 的 Stream 技术架构。 对于大数据量的转移，新版本提供了对可传输表的跨平台的支持，允许大批量数据快速从数据库上的脱离并附接到第二个数据库上。

18）增强的外部表功能

19）SQL Loader 的功能加强

20）增强的 SQL 分析能力

21）增强的 OLAP 分析功能

Oracle 内建的分析功能得到增强。提供了新的基于 PL/SQL 和 XML 的接口。提供了新的并行能力，以便于进行聚合和 SQL IMPERT 操作。一些算法得到改进。同时 OEM 能够用来监视并管理数据挖掘环境。

22）BIOINFORMATICS 的支持

这个版本包含对 BIOINFORMATICS 技术的特定支持。包括对 Double 和 Float 数据类型的 Native 支持。内建的统计函数支持常见的 ANOVA 分析等。

23）改进的数据挖掘的能力

24）XML 方面的增强，多媒体，文档与文本管理，

25）SQL语言，PL/SQL语言，JDBC支持更好

81，**数据库课本：**

1）**常用数据模型：**层次，网状，关系，面向对象

2）**关系模型允许三类完整性约束：**实体完整性，参照完整性，用户自定义完整性。

3）**数据库设计的基本步骤：**

需求分析；概念结构设计；逻辑结构设计；物理结构设计；数据库实施；数据库运行与维护。

4）**常用的三类存储方法：**

索引：目前主要为B+树。

聚类：cluster

哈希：hash

5）**事务特性：**ACID：原子性，一致性，隔离性，持续性。

6）**数据不一致：**丢失修改，不可重复读，读脏数据

7）**并发控制：封锁**　　排他锁（写锁），　共享锁（读锁）

8）**三级封锁协议：**

一级：修改前加写锁，事务结束后释放写锁。

二级：一级的基础上，读取前加读锁，读完释放读锁。

三级：一级的基础上，读取前加读锁，事务结束才释放读锁。

9）**两段锁协议：**

一段：对任何数据进行读写之前，首先申请获得全部需要的锁。

二段：在释放一个锁之后，在该事务中只能释放锁了，不能再继续申请锁。

10）oracle**三个级别的安全性：**表级，行级，列级。