001、CPU占用率高，大量sql语句的parse操作。

alter system set cursor\_sharing = ‘SIMILAR’

原因在于应用程序大量拼接sql，应尽量使用变量绑定，提高sql共享。

002、SQL\*Net more data from client，数据库吞吐量低。

01：网络瓶颈；02：应用服务器资源瓶颈，无法及时向数据库发送服务请求。

003、常见问题和观点。

系统参数一定要调，还要合理地调，但调好了并不一定能解决所有问题。

应用系统的问题无法交给DBA处理。

开发过程中就应该考虑到性能问题。

优化sql语句的性能，并不都是改语句。

多表连接性能太差是个错误的观点。

CPU占率用越低越好是错误的，可能是因为人为造成的堵塞造成的。

内存操作不一定最快，CPU占用无法忽视。

业务规则优化、数据库体系结构优化、数据库逻辑结构设计、数据库物理结构设计、应用系统设计等。合理的索引策略，合理地访问路径，优化SQL语句的分析数，充分使用PL/SQL，减少锁冲突等。



先拿最耗资源的SQL和模块动手。

IT系统一般分为OLTP(联机事务处理)和DSS(Decision Support System，决策分析系统，即数据仓库)。前者优化应当以响应速度为主，后者则应当以整体吞吐量为主。随着报表的增多，现在已逐渐都是混合型的系统，白天OLTP，晚上DSS。

004、sql执行计划分析：

1、表的访问方式；2、表的索引类型和策略；3、表的连接类型和过程；4、排序过程；

5、汇总过程；6、并行处理过程

量化指标：

1、消耗时间：包括Elapsed Time、CPU Time等；

2、内存消耗：包括Buffer Gets、Consistant Gets等；

3、I/O消耗：包括Physical Reads、Physical Writes等；

4、语句分析次数：包括Parses、Hard Parses、Soft Parses等

005、常用工具

explain，这个不解释了。

sql\*trace+TKPROF，前者生成的文件交由后者解析成可读文件。设置方法：

alter session set sql\_trace = true;

execute dbms\_session.set\_sql\_trace(true);

execute dbms\_system.set\_sql\_trace\_in\_session(session\_id,serial\_id,true);session\_id和serial\_id通过其他视图查询得到。

AutoTrace：

打开：set autotrace on; autotrace的执行计划是真正执行后分析的，而explain并没有执行sql。

统计信息重点看consistent gets和physical reads，分别代表内存消耗和磁盘I/O消耗，单位是db\_block\_size大小。

以上都在命令行模式下进行。

006、基本索引

基本概念：(1) rowid，索引的门牌号；(2) 索引就是目录

B\*树索引单字段设计：

1、分析sql语句中的约束条件

2、如果约束条件不固定，则建议创建针对单字段的B\*树索引

3、选择可选性最高的字段建立索引(即不同值最多的字段)

4、如果是多表连接的sql语句，注意被驱动表的连接字段是否需要创建索引

5、通过分析工具分析执行计划，量化执行效果

驱动表：在RBO(基于规则的优化器)中，from后的最后一张表是驱动表；CBO中，输出量最少的是驱动表。

索引被抑制：尽可能不对查询字段进行函数、表达式处理，自定义函数也是函数！函数索引应尽量少用，原因有二，一是影响DML效率，每次执行时都会重新计算；二是增加了索引存储空间。

复合索引(作用在多字段上):

1、前缀性(Prefixing)。比如索引建立在(empNo,empName)上，则查询条件必须包含empNo才能使用索引。这是因为索引是经过排序的，并非智能，此处就使用了empNo+empName排序。

关于skip scan index。这种是因为对索引进行了拆分。如(gender,ename,job)被拆分为(‘男’,ename,job)和(‘女’,ename,job)。这种索引违反了索引的可选性规则。

2、可选性(Selectivity)。索引字段的排列顺序，按值的多少排列。原因：值越多越容易定位。

设计原则：

如果查询条件涉及的字段不固定，组合十分灵活，则分别建立单字段索引；反之，则考虑建立复合索引，要注意前缀性和可选性。

索引监控：

alter index <indexname> monitoring usage; --打开

alter index <indexname> nomonitoring usage; --关闭

通过查询v$Object\_Usage视图判断索引是否使用。

索引碎片：

视图index\_stat中包含索引信息。索引重建、索引压缩，建议使用索引重建。

alter index <indexname> rebuild;

007、语句共享(变量绑定)

sql语句的主要执行过程：

1、Parse(解析)。在share pool(共享池)中查找该sql，如找不到，则进行硬解析(hard parse)，否则进行软解析(soft parse)，要尽可能地让oracle进行软解析。解析结果包括执行计划等。

2、Bind(绑定)。如果该语句中有变量，则通过赋值或传参进行变量绑定。

3、Execute(执行)。根据步骤1执行语句。

4、Fetch(抓取)。只针对select语句，进行查询记录的读取和排序。

OLTP和OLAP：

OLTP优化parse阶段，OLAP优化Execute和Fetch阶段，这跟它们的应用特性有关。

parse优化：(70%以上)

参见001.但是这项工作更应该交给开发人员，在应用程序中进行。如果数据畸形分布，001的解决方式不会最优。

查找可共享的sql语句：

select sql\_text from v$sqlarea where executions = 1;

008、排序、表连接

排序：在PGA(属于内存)中排序、能不排序就不排序。

除了显式的order by，distinct、union都隐含了排序。对于group by，10G之后，由于采用了新的算法，不再保证结果集排序。

执行计划里，sort \*\*\*表示进行了排序；另外就是利用索引，如果查询条件全在索引里，则会使用索引的排序。

Top-N：select \* from (select \* from emp order by salary desc) where rownum <11;直排了前10个。

表连接：(不管有多少表，一次都只连接两个表，因此掌握两表连接再扩展即可)

Nested Loop(嵌套循环)：

select e.\*,d.\* from emp e,dept d where e.deptId = d.id

执行：先循环dept表(假设它被作为驱动表，外表)，再按dept每条记录去查询emp表(被驱动表、内表)。

page 121