**--------------------------------------------**

**前海产险**—是前海人寿下面的公司 属于宝能集团

地点：草铺地铁站附近 中设广场B座 1101~1103

---------------------------------------------

1. 抽取数据的频率（这个问题 主要是为了看后面的处理，他认为我的贷款项目应该有实时抽取的方式）

【你的项目最终是为了报表展示，这一方向对数据实时性要求不高，所以抽取频率，你定一天一次或者一天几次都可以，有的月度报表 一个月一次也行】

1. 说到实时抽取 说ORACLE自带GOLDEN GATE （OGG）你们为什么还用KETTLE?

【首先到底用什么工具抽取，这是系统架构决定的，架构设计时采用的就是KETTLE，另外从业务层面来说，做报表的数据对实时性要求不高，不需要实时抽取】

1. KETTLE 能实时抽取吗？

【可以 10S一同步 见：https://blog.csdn.net/dora\_310/article/details/80511793】

1. DW层与DM层的数据有什么不同？举例说下都有哪些维度？

【DW层的数据是将ODS层的数据做一个初步筛选处理，DM层的数据是按不同维度将DW层数据再做统计；结合你的项目说】

1. 存储过程和FUCTION的区别？

【存储过程主要实现的是数据写入更新不返回结果，FUNCTION是一个函数，调用可以返回一个结果】

1. 物化视图和原表有什么差别？视图有什么作用？如果只是视图只是隐藏了字段，那为什么不用接口只提供所需的字段？

【物化视图是当时建立视图时固定下来的数据，如果原表数据有更新，物化视图中的数据是不变的。】

1. 说一下如何优化？

【作为开发我们优化SQL是我们的主要任务，执行SQL如果觉得慢，会通过执行计划去查看哪一步慢、花费多，对相应的部位进行调整，可以去改扫描方式：比如是想走索引扫描的，但是却走了全表扫描，看是否建了索引或者什么原因导致索引失效，将其完善；尽量减少表的查询次数（行转列就可以减少表的查询次数）；exists替换in 少用union\minus\intersect\ 这类需要排序 消耗比较大的的函数；小表放在后面，容易过滤的条件放在后面】--更多方法见下面

1. 存储程序中为什么不建议用 GO TO
2. 有数据备份吗？

【有备库的（一般数据量都是千万以上的数据 它的备库会保存近3个月的数据，之前的数据会压缩成文件进行保存）】

1. 虽然上面问了优化方法 但是又问了 你处理过亿级的数据 应该有很多性能方面的问题 你都是怎么处理的

【有张4亿数据量的表 查询起来很慢 建了张新的一样的表，将老的表改成其他名字，新进来的数据 写入到新表中】

1. 实际业务情况 具体都有哪些渠道名字（因为我说了以渠道维度去分析）

【结合你设计项目指标来讲】

新增用户量 老用户活跃量 新增用户比例（用户注册的来源比例数据）

每日用户PV（page view:页面的点击数量）数量 每日用户UV（unit view:ip数量）数量

每日运营毛收入统计 每日运营的利润统计 每月成本统计 每月部门收入统计

运营利息收入统计 手续费收入统计

1. DW层有多少张表？ 200张左右

整个项目的表格：800-1000左右

ODS DW DM DA

1. 你有去源系统看过数据吗？

【看过 上游给的用于抽数的账号 就可以登录他们的库】

1. 项目组多少人？

4-10个左右的大数据开发

1-2产品经理

0-1个软件测试

最后写了4个SQL

1. 去除重复的的数据

delete from 表 where rowid not in

(select min(rowid) from 表 group by 重复列);

1. DELETE TRUNCATE DROP 的区别？哪个有缓存
2. 将数据分页 rownum
3. 使用游标写一个for循环

for i in (select 语句) loop

执行语法

end loop;

----------------------

赛意面试题

----------------------

1. 介绍一下做过的项目

2.你的工作中有哪些指标？

3.怎么理解数据的维度和粒度

【维度：是分析数据的角度，如时间，位置；粒度是维度的详细度：比如分析地理时是按省划分、城市划分，还是区】

1. 举例说明一下同比环比的粒度？

【比如按地区算的销售额：

某省同比是指（今年该省某月销售额-去年某月销售额）/去年该月销售额

某市同比是指（今年该市某月销售额-去年某月销售额）/去年该月销售额

某省环比是指（今年该省某月销售额-上一个该省月销售额）/上一个月销售额

某市环比是指（今年该市某月销售额-上一个月销售额）/上一个月销售额

】

1. 说一下什么是三范式？

【三范式是规范如何建表的：

第一范式（1NF）：所有字段值都是不可分解的原子值，即不能同行同列出现两个值。

第二范式（2NF）：满足第一范式为前提，一定要有主属性键，且每一列都和主键相关，而不能只与主键的某一部分相关。

【例】成绩表SC中有字段SNO,CNO,SCORE,SNAME，其中SNO和CNO为组合主键，SNAME只依赖于SNO，因此违反二范式。

第三范式（3NF）：满足第二范式为前提，每一列数据都和主键直接相关，而不能间接相关。

【例】 EMP表中同时存在DEPTNO和DNAME，EMP表中主属性为EMPNO，DNAME直接依赖于DEPTNO,间接依赖于EMPNO，因此违反三范式。】

1. 什么是星型模型和雪花模型？

【行星模型是维度表都和事实表直接关联，雪花模型，有些维度表是和事实表简介间接关联】

1. 怎么对数据进行去重？

【利用rowid】

1. 怎么去除重复代码？

将需要重复使用的代码，写成代码块，例如函数

1. 知道分析函数吗？

聚合/排名/平移函数() over(partition by 分组 order by 排序)

1. 你怎么理解函数和存储过程和触发器？

一个固定的数据计算的过程就是函数；

一段固定的有逻辑的sql语句的操作，就是过程；

当完成a操作的时候，系统自动的完成b操作，就是触发器。

【函数是就像数学中的函数一样 是可以通过逻辑返回一个值得，存储过程主要是用于数据的写入更改等，触发器是在特定事件时 相应的完成一定的操作，比如审计字段值得写入，或者删除日志的记录】

1. 存储过程中有哪些硬代码？

create or replace procedure 过程名(参数名 in/out 数据类型)

as

begin

end;

1. 同步数据的时候，你们用的是增量还是全量？

数据量不大的表格就用全量（部门表或者财务报表之类的），大表格就用增量（销售数据之类的）

1. 写存储过程是遇到过哪些问题

输入数据的计算预期结果和实际结果不一致。

1. 怎么解决这个问题的？

先做数据的抽样检查，例如先拿100条数据等等进行数据的检查；

在每个有逻辑的部分，例如if判断或者循环结构的外面和里面，都对需要的变量进行打印和查看，查询数据在哪个地方偏离的计算。

1. 工作中怎么决重跑问题（重新跑脚本）

先使用调度工具马上运行一次，检查是否有错误；

如果有错误，使用脚本自己在数据库中去运行一次检查是否有错；

如果还有错误，说明数据表有可能发生了表结构上的修改，这个时候就要修改脚本，然后重新配置调度任务，重新运行检查是否有错误。

15.工作中怎么进行优化，做过哪些优化，举例说明一下

复试：

1.请详细介绍一下你做过的项目

2.你做的项目中有那些指标

3.你同步数据中取的是哪些数据，举例说一下，大概有那些字段？

4.你有没有做过多张表关联，分别有哪些表呢？她们通过哪些字段关联的。

5.知道维键吗，你理解的维键是什么？

6.你们的操作系统是什么系统？ centos7 hadoop2.8 hive2.1 kettle7.1

7.你的项目有几个人，开发的有几个呢？你主要做什么？

8.你对自己的定位是什么。或者对未来的发展计划？

9.经过电话面试之后，你有没有对我们公司进行了解过？

---------------------

华为面试题

---------------------

电话面试：

1.讲讲你做过的项目，主要了解一下我做的是什么工作？

2.用过哪些工具

3.主要用kettel做些什么？实现些什么功能呢？

调度工具：oozie control-m

4.还用过其他什么工具呢？

5.知道ds的哪些控件 datastage

6.知道关联的look up和join的区别吗？

7.调度工具的话用的是什么？

8.大概讲一下存储过程的一个框架构成。

9.有用过循环控制吗？ for while loop

10.执行存储过程遇到过什么问题？

11.你们同步数据时候多久更新一次？

12.跑数过程出错了怎么解决，如何监控到出错？

13.知道分析函数吗？

14.工作过程中做过哪些优化？

15.知道有哪几种分区呢？

列表 范围 散列 组合

静态 动态

复试：

1.做一下自我介绍

2.介绍一下你做过的一个项目

3.说一下你处理过的最大数据量是多少，它的字段大概有多少个?

八九千万 字段在三四十个左右

1. 以你处理过的这个数据量最大的这个表为例说一下你怎么把数据提取到这张表的？

以时间为事件进行数据的增量更新

5.你用过的etl工具？

6.具体说一下你是怎么使用kettle去取数的？

csv excel 接口 数据库

1. 怎么从不同的数据库去取数？

添加不同的数据库连接，使用表输入的控件

1. 你知道的函数有哪些？

聚合 单行（数字 字符串 日期 转换） 分析

9.分析函数有哪些？

10.工作中遇到过哪些问题？

11.举例说一下你做的开发工作

12.如果有一个物料编码，这个物料编码有许多的业务意义，如供应商用s表示等等这些规则，

如何把它提取到一张表里面从而展现这些业务属性？

码表

001 已发货

002 未发货

003 退款中

13.你怎么看待加班的？

14.你的薪资期望是多少？

20年 6000-8000 19年 7000-9000 18年 10000-11000

17年 12000-14000 16年 13000-15000 ...

艾德赛 富德生命人寿

1. 之前工作得一些建议或者觉得做的比较好的方面（从架构上或者从模型上）
2. 有涉及到过分析吗
3. 优化经验
4. 执行计划怎么看，看什么指标
5. 对出现异常的程序怎么处理（怎么确定那句语句出现问题）
6. 存储过程中异常处理部分有用过吗
7. 抛异常部分放在哪里（调度工具还是存储过程里）？？？？
8. 报表有做过吗
9. 分析函数
10. 三范式
11. 拉链表怎么设计
12. 缓慢变化维
13. 未来的职业规划

华为

1. 介绍项目  
   2、同比环比怎么算  
   3、表结构  
   4、数据流向是怎样  
   5、异常处理怎么做  
   6、那异常处理之后程序能继续运行，那怎么发现这个错误  
   7、拉链表怎么做  
   8、有效增量怎么取  
   9、数据全量还是增量  
   10、全量怎么做  
   11、那如果你先删除数据之后，业务查数据查不到怎么办  
   12、这么做会有有重复数据啊，怎么解决  
   13、参与需求分析，你主要做什么  
   14、给你表结构，你需要确认什么  
   15、架构是怎样  
   16、你们公司DW和DM层的区别  
   17、kettle熟吗，主要怎么用
2. 关联怎么用
3. 哈希关联和嵌套循环关联
4. 性能调优怎么做  
   21、索引怎么用  
   22、怎么在分区上建立索引  
   23、你每个项目的时间都不长，为什么  
   24、为什么离开上一家公司  
   25、你在上家公司学到了什么

-------------------------------------

平安:

---------------------------------------

1、自我介绍  
2、介绍项目  
3、数仓架构  
4、你做哪一层  
5、常用函数  
6、分析函数  
7、按年对每月数据累计求和  
8、数据字典怎么用  
9、kettle主要是用于抽数吗  
10、是否了解前端的一些工具  
11、BIEE用过吗  
12、专业是什么  
13、拉链表怎么用  
14、怎么取有效增量  
15、索引不能用了，为什么，不是因为失效  
16、hints怎么用

----------------------------

华为

--------------------------

1.DS常用的控件有哪些？

2.lookup与join的区别？

【lookup 故名思义就是 一个表在另一个表中找，处理过程都在内存进行性，大事实表和小纬表用这种方式关联效率高。  
merge 和 join 的处理过程不需用占大量内存。不同在于 merge要先把key值排序在做join，而且Merge的输出集可以设为多个

merge 方式对于key值由重复的，重复记录中不能关联到，所以merge方式需要key值（也就是关联的字段）无重复

】

3.join不排序可不可以？

4.DS版本是什么？

5.DS里面的project，en什么的是什么意思？作用是什么？

6.DS怎么从取数？

7.表的扫描方式是什么？

8.全表扫描和索引扫描的区别是什么？

9.为什么建了索引在单表扫描时还是走的全表扫描？

10.索引为什么会失效？

11.索引的种类有哪些？

12.位图索引什么时候用？

13.知道粒度是什么吗？

14.国 省 市 区 金额

按省统计每个省的总金额有哪些方法？

15.接上题，在统计好的表内容的最下方显示多一行汇总数据，如汇总总金额，省的总数等，如何实现？

16.一个表数据转移到另一张表，之前的表不要了，应该如何实现最快？

17.说说游标是什么？

18.说一下数仓是什么？

19.说说数仓的架构？

20.数仓每一层是干嘛的？

21.知不知道4层架构的数仓？

22.之前有没有做过华为项目？

23.知道control-m吗？

24.你们用的什么报表工具？

25.水晶报表的原理是什么？

26.知道缓慢变化的表吗？

27.知道表结构吗？

28.期望薪资是多少？

29.未来规划是什么？

30.年底还离职，离职原因是什么？

31.知道truncate和delete的区别吗？

32.知道trunc怎么用吗？

33.知道varchar与varchar2的区别吗？

34.rank，dense\_rank，row\_number的区别？

35.空值怎么转换？varchar与varchar2在转空值时的区别？

36.知道怎么授权吗？取消授权呢？

------------------

华为

------------------

1、介绍自己

2、介绍项目

3、公司架构

4、你做哪一层

5、最大的数据量有多少

6、一个表的数据量很大，怎么优化

7、你觉得多大的数据量才需要进行分区

8、你们各个项目组的工作是按什么分的，是每一组做一个主题吗

9、人力配备是怎样的

10、报表是对接的你的表还是ABI

11、你们是直接跟业务对接吗

12、产品经理需要输出什么文档吗

13、你们开发完需要输出什么文档吗

14、说一下星型模型和雪花模型，你们用哪个用得多

15、你们有针对什么维度做过拉链表吗

16、为什么要做

17、你用过DS吗

18、你的优点

19、你有缺点吗

20、公司是甲方还是乙方

21、为什么要离开上一家公司

22、很多人在乙方待不久，你觉得你能待多久

23、你们之前加班多吗

24、怎么看待996

-----------------------------

交通银行

--------------------------

1自我介绍

2专业不是这个怎么会学这个

3什么时候开始工作的

4 ncp全称是什么

5你们对数据进行了什么样的处理

6你主要负责什么

7处理的数据量有多大

8有设计过表吗

9有用过建模工具吗

10数据库优化接触过吗

11数据的导入导出玩过吗

12你们存储过程对异常怎么处理的

13数据库的调度用过吗

14万一半夜报错了怎么办

15异常定位有总结过吗

16数据的备份与恢复

17星型模型和雪花模型

18 sql性能调优有总结过吗

19 linux熟悉吗

20 shell命令熟悉吗

21 有学习过大数据相关的知识吗

22为什么离职

23 你想往什么方向发展

------------------------

软通

--------------------------

1. 自我介绍

2. 介绍项目

3. 用的什么库？

4. 数仓架构是怎么样的？

5. ncp是什么意思？

6. 你们除了数据库定义的异常，自己还定义了什么异常？

7. 你遇到的常见的异常有哪些？

8. 出现了异常怎么处理？

9. 都是人去处理吗？没有程序去处理吗？这条线出问题了没有另一条线去处理吗？

10. 如何优化sql语句？

11. 讲个经典的优化案例

-------------------------

软通永隆银行

------------------------------

1. Oracle where 子句中多个and和or时，执行顺序是怎么样的？、
2. Union和union all 的区别
3. Oracle between是否包含边界值
4. 函数和存储过程有什么区别
5. 嵌套循环和哈希链接各自在什么情况下效率最高
6. 简单说一下做过的项目，然后工作的内容（指标）以及怎么实现、逻辑？
7. 项目有哪些人员
8. 数仓架构
9. 同步数据时同步过来的数据有垃圾数据或者同步过来的数据有问题的数据怎么处理
10. 处理过的最大数据量
11. 优化思路
12. 怎么使用的表分区
13. 知道的有哪些Hints，怎么使用，hints原理
14. Oracle内部执行机制
15. 调度工具用的什么
16. 误删了一张表，怎么恢复
17. 星型模型/雪花模型
18. 自己设计过的星型模型
19. Oracle中的分析函数
20. Shell脚本开头（#!/bin/bash）是什么意思
21. Linux用过哪些命令
22. 调用shell脚本的三种命令
23. Shell的子进程跟路进程
24. 用过的Etl工具以及怎么使用的，跨库（不同类型的数据库，不同文件格式：xml）
25. 有使用过ds吗？
26. 索引的种类
27. 离职的原因
28. 简述下个人优势

----------------------

赛意

--------------------------

1. 介绍最近的项目
2. 指标都是你计算的吗
3. 同比环比怎么算
4. 数据直接从ODS到DM吗，没有中间层吗
5. 全量更新还是增量
6. 依赖配置，调度工具
7. 源系统有哪些
8. 测试出现问题，怎么定位
9. 视图和表的区别
10. 笔试题：

源表字段C1,C2,C3,AMT

目标表字段C1,C2,AMT（按C1,C2统计）,TOTAL\_AMT(不分维度进行计算)，按AMT排序

1. 星型模型和雪花模型
2. SQL优化怎么做
3. 为什么离职，期望薪资
4. 有什么想要问我的

-------------------------

深交所笔试

---------------------------

**深交所笔试题**

**一、设有三个关系表：**

学生表（学号、姓名、年纪、性别）

选课表（学号、课号、成绩）

课程表（课号、课名、老师）

使用SQL语言完成下列查询功能：

（1）检索“WANG”同学不学的课程号；

（2）检索全部学生都选修的课程号与课程名；

（3）检索选修少于三门课程并且没有选修LIU老师所授课程的学生姓名。

**二、数据表T含三个字段（年、季度、增长率）数据记录如下：**

|  |  |  |
| --- | --- | --- |
| **年** | **季度** | **增长率** |
| 2010 | 1 | 0.10 |
| 2010 | 2 | 0.11 |
| 2010 | 3 | 0.09 |
| 2011 | 2 | 0.10 |
| 2011 | 3 | 0.12 |
| 2011 | 4 | 0.14 |
| 2012 | 1 | 0.11 |
| 2012 | 2 | 0.07 |
| 2012 | 4 | 0.06 |

要求编写SQL语句，并简要描述基本思路，实现以下的查询结果输出：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **年** | **一季度** | **二季度** | **三季度** | **四季度** |
| 2010 | 0.10 | 0.11 | 0.09 | - |
| 2011 | - | 0.10 | 0.12 | 0.14 |
| 2012 | 0.11 | 0.07 | - | 0.06 |

**三、用如下SQL建立两张表**

CREATE TABLE TABLE1 (

CALNDR\_DT DATE, *--日期*

CURRENCY\_ID CHAR(6), *--存款币种*

DEPOSTOR\_ID CHAR(10), *--存款人*

D\_TRNOVR DEC(12,0)); *--存款金额*

CREATE TABLE TABLE2(

CURRENCY\_ID CHAR(6), *--存款币种*

DEPOSTOR\_ID CHAR(10), *--存款人*

F\_TRNOVR DEC(12,0)); *--存款金额*

其中TABLE1存放的是存款人每日存款余额数据，该库中有多天的数据。TABLE2存放的是今天支取存款数据，只存当天数据。请用SQL写出今日没有支取存款的储户最早的余额数据。

**四、数据表T有两个字段，分别为编号（INTEGER类型）、姓名（CHAR类型）**

其记录示意如下：

|  |  |
| --- | --- |
| **编号** | **姓名** |
| 1 | 分隔符 |
| 2 | 张三 |
| 3 | 李四 |
| 4 | 分隔符 |
| 5 | 王五 |
| 6 | 赵六 |
| 7 | 钱七 |
| 8 | 孙八 |
| 9 | 分隔符 |
| 10 | 李八 |
| 11 | 周一 |
| 12 | 吴九 |

姓名为“分隔符”的记录是作为分组定位使用，在两个分隔符中的人为一组，每组有一个组号，用于唯一标识为一组，例如：

|  |  |  |
| --- | --- | --- |
| **编号** | **姓名** | **组号** |
| 2 | 张三 | 1 |
| 3 | 李四 | 1 |
| 5 | 王五 | 2 |
| 6 | 赵六 | 2 |
| 7 | 钱七 | 2 |
| 8 | 孙八 | 2 |
| 10 | 李八 | 3 |
| 11 | 周一 | 3 |
| 12 | 吴九 | 3 |

请编写SQL语句，并简要描述基本思路，实现以上查询结果输出。

**五、表TABLE1有两个字段，分别为编号（CHAR类型）、姓名（CHAR类型）。要求：执行一句删除语句，当姓名列上有相同时，只保留编号最小的记录。**

TABLE1示例：

|  |  |
| --- | --- |
| **编号** | **姓名** |
| 1 | a |
| 2 | b |
| 3 | b |
| 6 | b |
| 8 | c |
| 3 | a |
| 3 | c |
| 5 | c |

删除后的结果应如下：

|  |  |
| --- | --- |
| **编号** | **姓名** |
| 1 | a |
| 2 | b |
| 3 | c |

请写出SQL语句。

**六、****数据表TB（****发布日期、存款利率、贷款利率）存有银行分布的历次一年期存贷利率。**

例如（非真实数据）：

|  |  |  |
| --- | --- | --- |
| **发布日期** | **存款利率** | **贷款利率** |
| 2010-01-01 | 8% | 10% |
| 2011-01-01 | 5% | 7% |
| 2012-01-01 | 5% | 7% |
| 2012-10-31 | 5% | 6% |
| 2013-01-01 | 5% | 6% |
| 2013-03-31 | 8% | 9% |
| 2013-09-01 | 8% | 10% |
| 2014-01-01 | 8% | 9% |
| 2015-01-01 | 6% | 9% |

为方便获得每天的一年期银行存贷款利率，我们设计了如下数据表TD（起始日期、截至日期、存款利率、贷款利率），存放各日期段内的银行存贷利率。数据表TD的数据记录示意如下：

|  |  |  |  |
| --- | --- | --- | --- |
| **起始日期** | **截至日期** | **存款利率** | **贷款利率** |
| 0000-01-01 | 2010-12-31 | 8% | 10% |
| 2011-01-01 | 2012-10-30 | 5% | 7% |
| 2012-10-31 | 2013-03-30 | 5% | 6% |
| 2013-03-31 | 2013-08-31 | 8% | 9% |
| 2013-09-01 | 2013-12-31 | 8% | 10% |
| 2014-01-01 | 2014-12-31 | 8% | 9% |
| 2015-01-01 | 9999-12-31 | 6% | 9% |

请编写SQL查询语句，并简要描述基本思路，实现如下功能：根据TB表已维护的数据记录，生成插入TD表所需的数据集。

假设：函数DateOffset（dt，n）返回指定日期dt之前或之后N天的日期，N > 0表示后N天，N < 0表示向前 | N | 天。

例如：调用DateOffset（ ‘ 2012-01-01 ’，-1 ），返回 ’ 2011-12-31 ’。

**七、请简要描述一下SQL查询语句的功能，并给出查询结果。**

数据表T1（fa integer , fb integer）中有如下4条记录

（1，1）

（1，1）

（2，2）

（3，3）

数据表T2（fa integer , fb integer）中有3条记录

（1，2）

（2，2）

（4，4）

SQL查询语句：

SELECT MAX(T) AS T, fa, fb

FROM (SELECT fa, fb, 'T1' AS T FROM T1

UNION

SELECT fa, fb, 'T2' AS T FROM T2) AS CT

GROUP BY fa, fb

HAVING MAX(T) = MIN(T)

ORDER BY T, fa, fb

**八、关系数据表T\_Date ( root int , str varchar(1024) )中，只存有一条记录：**

**( root , str ) = ( 1 , ‘ ab ; cd ; cefg ; 123 ; abcdde ’ )。**

现有下列的SQL递归语法及示例，可以计算出2的0到N次方（假设N = 10）值及对应次方数：

|  |
| --- |
| WITH T\_Query ( root , times ) AS  ( *--WITH表示一个子查询结果集*  *--T\_Query是子查询结果集的名称，作为递归的中间结果集*  *--root和times是T\_Query的查询结果字段*  *--root表示2的times次方值*  SELECT root, 0 AS times  *--root=1（存放在数据表T\_Date中），表示2的0次方值*  FROM T\_Date *--注意：递归的终止条件只能有一条记录，此处用关系数据表T\_Date构造（只有一条记录）*  UNION ALL *--每次递归调用的结果集将被合并在一起*  SELECT root \* 2 AS root, times + 1 AS times  *--递归公式root(K+1)=root(K)\*2*  FROM T\_Query *--递归的中间结果集*  WHERE times < 10 *--最多计算到2的10次方*  )  SELECT times, root *--root=2的times次方值，times表示次方数*  FROM T\_Query *--查询结果集T\_Query中存在2的10次方的值*  ORDER BY times; *--按次方数从小到大顺序排列* |

查询结果如下：

|  |  |
| --- | --- |
| **times** | **root** |
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |
| 8 | 256 |
| 9 | 512 |
| 10 | 1024 |

基于上面对SQL递归语法及示例的理解，请描述下面SQL脚本所实现的功能，并写出该查询输出的结果：

|  |
| --- |
| WITH T\_Query ( str , times ) AS  (SELECT str , 0 AS times  FROM QTEMP / T\_date  UNION ALL  SELECT SUBSTR ( str , POSSTR ( str, ' ; ' ) + 1 ) AS str ,  times + 1 AS times  FROM T\_Query  WHERE POSITION ( str , ' ; ' ) > 0 )  SELECT str , times  FROM T\_Query  WHERE POSSTR( str , ' ; ' ) = 0  ORDER BY str; |
| 附相关SQL函数说明：   * SUBSTR ( STRING , START [ , LENGTH ] )：从制定起始位置START（基于1）开始取长度为LENGTH的子字符串，如果没有提供LENGTH参数，则直到字符串结束； * POSSTR ( SOURCE - STRING , SEARCH - STRING )：定位待查找字符或字符串在源字符串中的位置，如果不存在，则返回0。 |

**九、要往一个已经很大的数据表T（无主键约束）插入一批数据，由于某种误操作，这些数据被重复插入两次，为了补救这个失误，你会怎么做？请写出主要的SQL语句。**

1. **请你谈谈对数据库索引技术的理解和认识，并分享过往在查询效率优化方面的工作经验。**

**六．**

**select min(start\_dt ) as start\_dt, end\_2 ,**

**debit, loan from**

**(**

**select a.debit,a.loan, a.dt, a.start\_dt , case when lead(debit)over(order by dt) = debit and lead(loan)over(order by dt ) = loan then lead(end\_dt) over(order by dt ) else end\_dt end as end\_2**

**from (**

**SELECT dt,**

**CASE**

**WHEN DT = MIN(DT) OVER() THEN**

**DATE '0001-01-01'**

**ELSE**

**DT**

**END as start\_dt, -- 算出开始时间**

**LEAD(DT - 1, 1, DATE '9999-12-31') OVER(ORDER BY DT) as end\_dt, -- 算出结束时间**

**debit,**

**loan**

**FROM XXXX**

**) a -- 查询每一条记录的起始时间**

**) c**

**group by debit, loan,end\_2**

**order by start\_dt**

---------------------

顺丰

-------------------------

1. 有A、B两张表，如何实现在A表中存在，在B表中不存在。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| NAME （姓名） | DEPART-ID（部门） | FEE（费用） |  | DEPART-ID （部门） | DEPART-NAME |
| 张三 | 80 | 10000 |  | 80 | 财务部 |
| 李四 | 80 | 130000 |  | 90 | 人资部 |
| 王五 | 90 | 150000 |  | 100 | 电脑部 |
| 赵六 | 90 | 11000 |  | 110 | 综合部 |
| 周七 | 100 | 120000 |  | 表B | |
| 吴八 | 100 | 180000 |  |  |  |
|  | 表A |  |  |  |  |

1. 员工表EMP的记录如下：

|  |  |  |
| --- | --- | --- |
| NAME | SALARY-TYPE | SALARY |
| 张三 | BASE | 1000 |
| 张三 | JOB | 2000 |
| 张三 | EFF | 200 |
| 李四 | BASE | 1000 |
| 李四 | EFF | 1000 |

请用SQL语句实现以下结果：

|  |  |  |  |
| --- | --- | --- | --- |
| NAME | BASE | JOB | EFF |
| 张三 | 1000 | 2000 | 200 |
| 李四 | 1000 | 0 | 1000 |

1. 员工表EMP的记录如下：

|  |  |  |
| --- | --- | --- |
| NAME（姓名） | DEPT-ID（部门） | FEE（费用） |
| 张三 | 80 | 10000 |
| 李四 | 80 | 130000 |
| 王五 | 90 | 150000 |
| 赵六 | 90 | NULL |
| 周七 | 100 | 120000 |
| 吴八 | 110 | 180000 |

请通过SQL语句实现：个人费用大于部门平均费用的所有员工记录。

1. 系统中对于客户给出的以下需求，如果请你来做需求分析，你会提出哪几个问题或疑问？

|  |  |  |
| --- | --- | --- |
| WAYBILL-NO | VARCHAR2(30) | 运单号 |
| SOURCE-ZONE-CODE | VARCHAR2(30) | 寄件方地点 |
| BEST-ZONE-CODE | VARCHAR2(30) | 收件方地点 |
| CONSIGNEE-EMP-CODE | VARCHAR2(20) | 寄件方员工工号 |
| CONSIGNEE-TM | DATE | 寄件时间 |
| DELIVER-EMP-CODE | VARCHAR2(30) | 收件方员工工号 |
| SUBSCRIBER-NAME | VARCHAR2(100) | 收件方联系人 |
| SIGNIN-TM | DATE | 签收时间 |
| INPUT-TM | DATE | 录入时间 |
| PROD-TYPE | VARCHAR2(30) | 产品类型 |

|  |  |
| --- | --- |
| 调研对象 | 抽样规则 |
| 收件方 | 1、分时段抽取：在系统中按照时间段整数设置，如8-9,9-12 2、每个时间点开始抽取，每个派件方分点部抽取Y（可配置化）个。——针对标准产品，**收派员不重复**  3、每个时间点开始抽取，设置（可配置化）派方区部抽取总数Y个，每个分点都均匀抽取，——针对电商特惠、顺丰特惠、即日到、物流普运 |

------------------------

----------------------------

自我介绍

介绍项目

你主要负责什么

左关联右关联的区别

数据倾斜是什么

A join b和 b join a，a大b小，哪个快？

怎样更新数据，代码是怎么样的

怎么对数据进行分析，知道次日留存，次月留存吗

Delete和truncate的区别，和drop又有什么区别？

什么是索引？

为什么不做成分区？

视图怎么用的

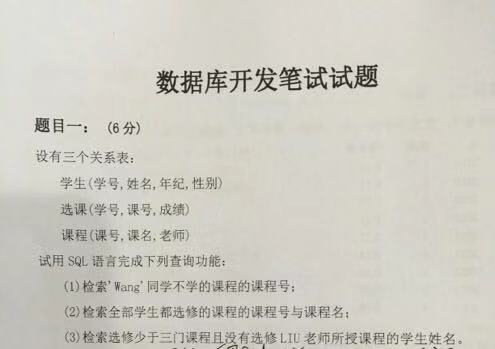
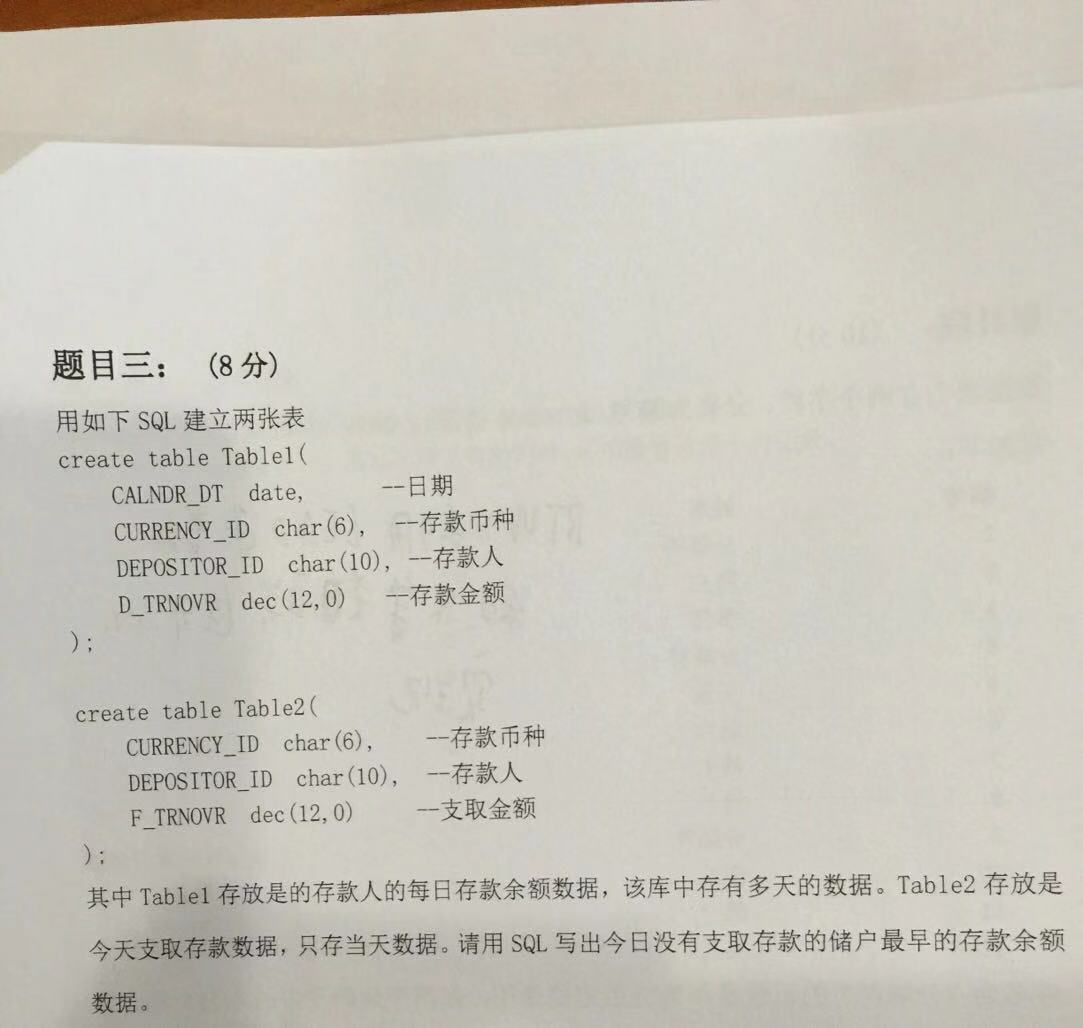
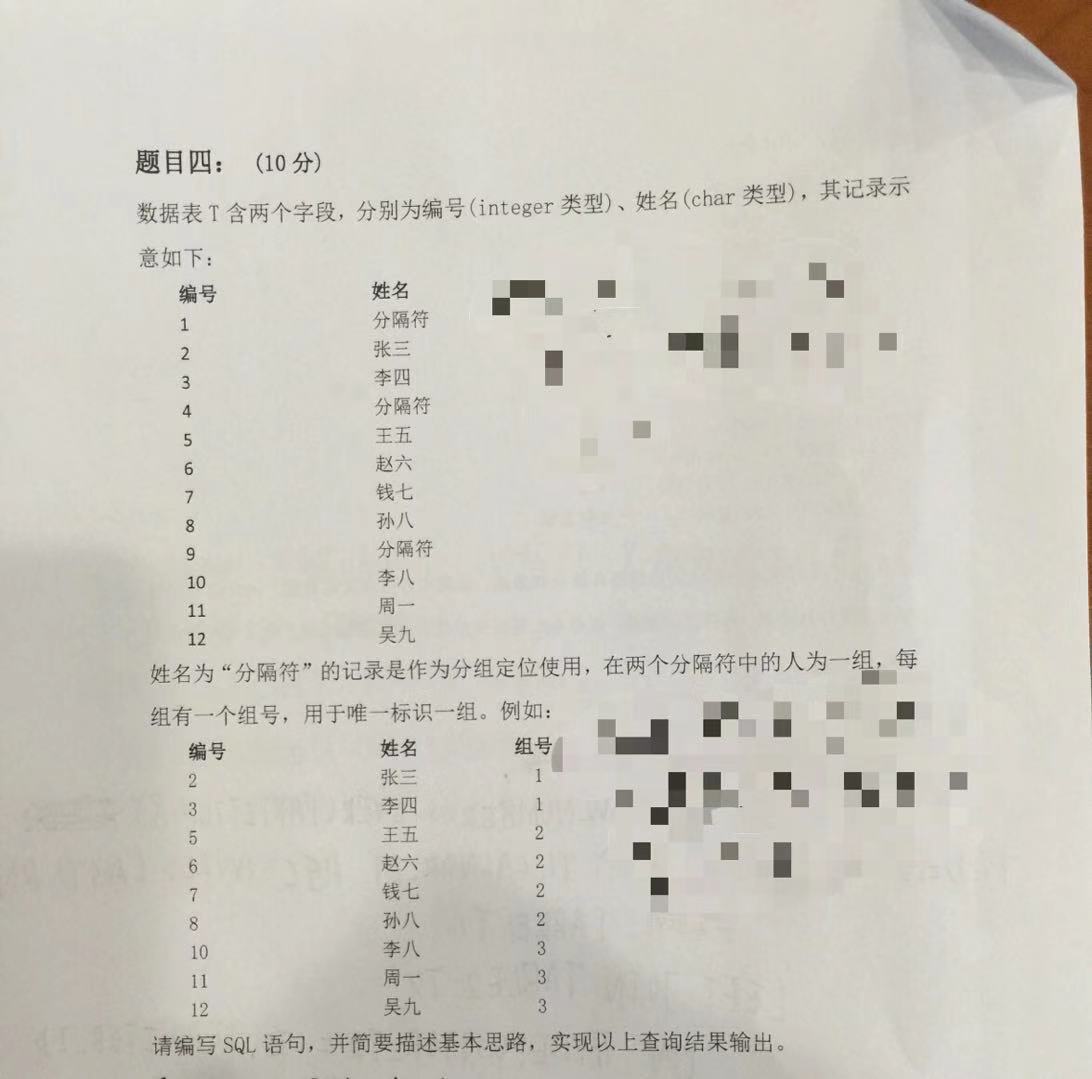
怎么做的etl

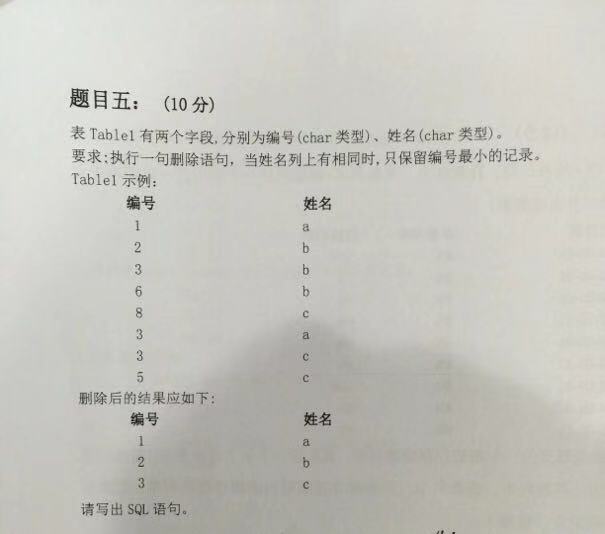
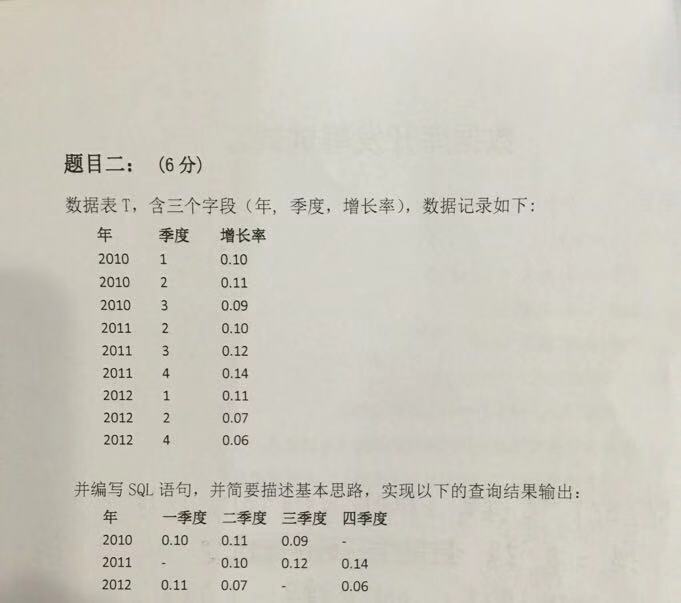
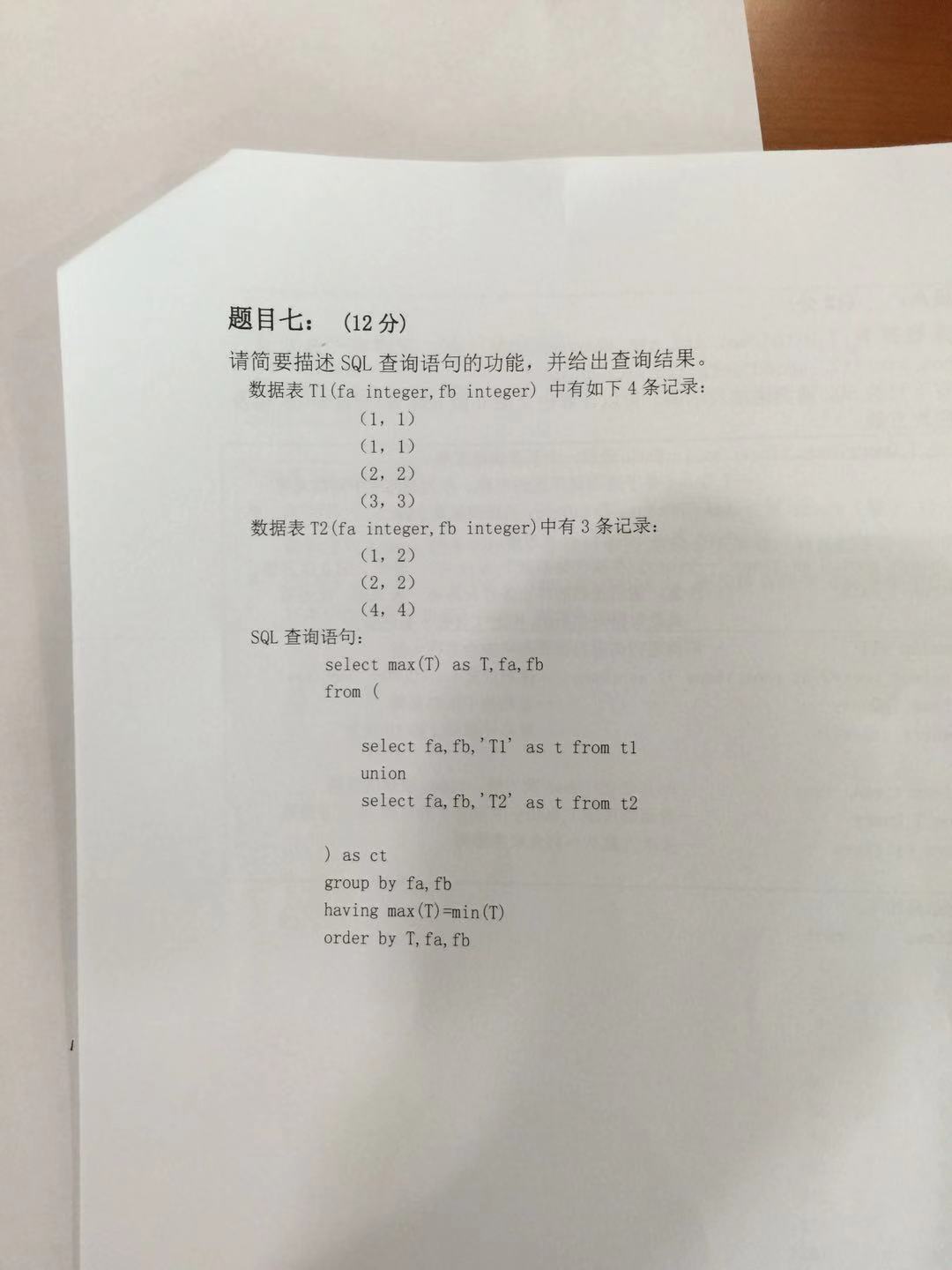
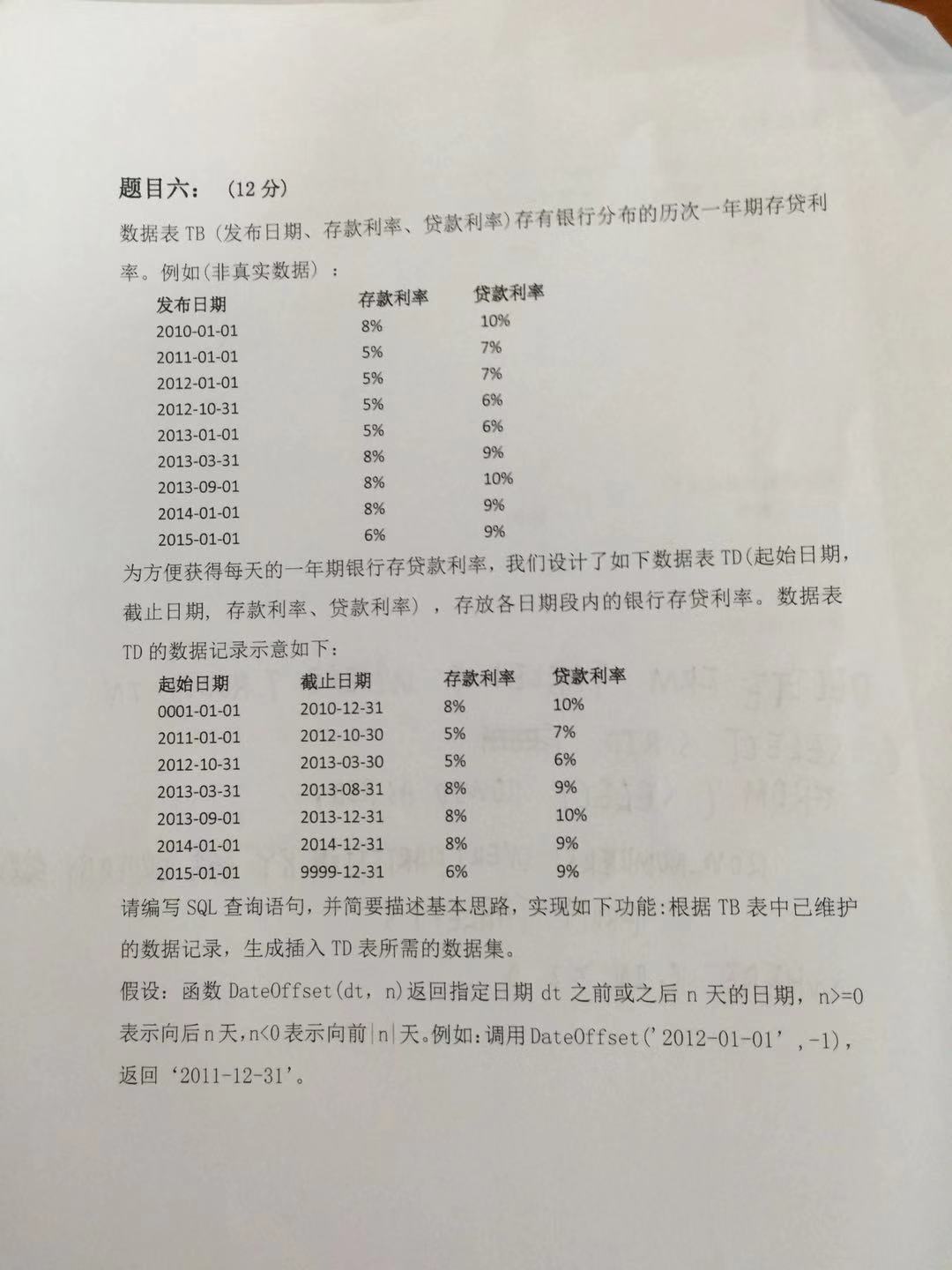
怎么做的调度？

会做报表吗？

Ods是什么意思？数据整合是什么意思？

为什么离职





**下面是优化的内容**

-------------------------------------------------------------------------------

**1. SQL语句执行步骤**

语法分析> 语义分析> 视图转换 >表达式转换> 选择优化器 >选择连接方式 >选择连接顺序 >选择数据的搜索路径 >运行“执行计划”

**2. 选用适合的Oracle优化器**

RULE（基于规则）  COST（基于成本）  CHOOSE（选择性）

**3. 访问Table的方式**

**全表扫描**

  全表扫描就是顺序地访问表中每条记录，ORACLE采用一次读入多个数据块(database block)的方式优化全表扫描。

**通过ROWID访问表**

  ROWID包含了表中记录的物理位置信息，ORACLE采用索引实现了数据和存放数据的物理位置(ROWID)之间的联系，通常索引提供了快速访问ROWID的方法，因此那些基于索引列的查询就可以得到性能上的提高。

**4. 共享 SQL 语句**

* Oracle提供对执行过的SQL语句进行高速缓冲的机制。被解析过并且确定了执行路径的SQL语句存放在SGA的共享池中。
* Oracle执行一个SQL语句之前每次先从SGA共享池中查找是否有缓冲的SQL语句，如果有则直接执行该SQL语句。
* 可以通过适当调整SGA共享池大小来达到提高Oracle执行性能的目的。

**5. 选择最有效率的表名顺序**

* ORACLE的解析器按照从右到左的顺序处理FROM子句中的表名，因此FROM子句中写在最后的表(基础表 driving table)将被最先处理。
* 当ORACLE处理多个表时，会运用排序及合并的方式连接它们。首先，扫描第一个表(FROM子句中最后的那个表)并对记录进行派序，然后扫描第二个表(FROM子句中最后第二个表)，最后将所有从第二个表中检索出的记录与第一个表中合适记录进行合并。
* 只在基于规则的优化器中有效。

举例：

表 TAB1 16,384 条记录

表 TAB2 1 条记录

/\*选择TAB2作为基础表 (最好的方法)\*/

select count(\*) from tab1,tab2 执行时间0.96秒

/\*选择TAB2作为基础表 (不佳的方法)\*/

select count(\*) from tab2,tab1 执行时间26.09秒

如果有3个以上的表连接查询, 那就需要选择交叉表(intersection table)作为基础表, 交叉表是指那个被其他表所引用的表。

SELECT \* FROM LOCATION L, CATEGORY C, EMP E

WHERE E.EMP\_NO BETWEEN **1000** AND **2000**

AND E.CAT\_NO = C.CAT\_NO

AND E.LOCN = L.LOCN

将比下列SQL更有效率

SELECT \* FROM EMP E, LOCATION L, CATEGORY C

WHERE E.CAT\_NO = C.CAT\_NO

AND E.LOCN = L.LOCN

AND E.EMP\_NO BETWEEN **1000** AND **2000**

**6. Where子句中的连接顺序**

Oracle采用自下而上的顺序解析WHERE子句。 根据这个原理,表之间的连接必须写在其他WHERE条件之前，那些可以过滤掉最大数量记录的条件必须写在WHERE子句的末尾。

[复制代码](javascript:void(0);)

/\*低效,执行时间156.3秒\*/

SELECT …

FROM EMP E

WHERE SAL > **50000**

AND JOB = ‘MANAGER’

AND **25** < (SELECT COUNT(\*) FROM EMP

WHERE MGR = E.EMPNO)

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

/\*高效,执行时间10.6秒\*/

SELECT …

FROM EMP E

WHERE **25** < (SELECT COUNT(\*) FROM EMP

WHERE MGR=E.EMPNO)

AND SAL > **50000**

AND JOB = ‘MANAGER’

[复制代码](javascript:void(0);)

**7. SELECT子句中避免使用“\*”**

* Oracle在解析SQL语句的时候，对于“\*”将通过查询数据库字典来将其转换成对应的列名。
* 如果在Select子句中需要列出所有的Column时，建议列出所有的Column名称，而不是简单的用“\*”来替代，这样可以减少多于的数据库查询开销。

**8. 减少访问数据库的次数**

当执行每条SQL语句时, ORACLE在内部执行了许多工作：  解析SQL语句 > 估算索引的利用率 > 绑定变量 > 读数据块等等

由此可见, 减少访问数据库的次数 , 就能实际上减少ORACLE的工作量。

**9. 整个简单无关联的数据库访问**

如果有几个简单的数据库查询语句，你可以把它们整合到一个查询中（即使它们之间没有关系），以减少多于的数据库IO开销。

虽然采取这种方法，效率得到提高，但是程序的可读性大大降低，所以还是要权衡之间的利弊。

**10. 使用Truncate而非Delete**

* Delete表中记录的时候，Oracle会在Rollback段中保存删除信息以备恢复。Truncate删除表中记录的时候不保存删除信息，不能恢复。因此Truncate删除记录比Delete快，而且占用资源少。
* 删除表中记录的时候，如果不需要恢复的情况之下应该尽量使用Truncate而不是Delete。
* Truncate仅适用于删除全表的记录。

**11. 尽量多使用COMMIT**

只要有可能,在程序中尽量多使用COMMIT, 这样程序的性能得到提高,需求也会因为COMMIT所释放的资源而减少。

COMMIT所释放的资源：

* 回滚段上用于恢复数据的信息.
* 被程序语句获得的锁
* redo log buffer 中的空间
* ORACLE为管理上述3种资源中的内部花费

**12. 计算记录条数**

Select count(\*) from tablename;

Select count(**1**) from tablename;

Select max(rownum) from tablename;

 一般认为，在没有索引的情况之下，第一种方式最快。 如果有索引列，使用索引列当然最快。

**13. 用Where子句替换Having子句**

避免使用HAVING子句，HAVING 只会在检索出所有记录之后才对结果集进行过滤。这个处理需要排序、总计等操作。 如果能通过WHERE子句限制记录的数目，就能减少这方面的开销。

**14. 减少对表的查询操作**

在含有子查询的SQL语句中，要注意减少对表的查询操作。

低效：

[复制代码](javascript:void(0);)

SELECT TAB\_NAME FROM TABLES

WHERE TAB\_NAME =（SELECT TAB\_NAME

FROM TAB\_COLUMNS

WHERE VERSION = **604**）

AND DB\_VER =（SELECT DB\_VER

FROM TAB\_COLUMNS

WHERE VERSION = **604**）

[复制代码](javascript:void(0);)

高效：

SELECT TAB\_NAME FROM TABLES

WHERE （TAB\_NAME，DB\_VER）=

（SELECT TAB\_NAME，DB\_VER

FROM TAB\_COLUMNS

WHERE VERSION = **604**）

**15. 使用表的别名（Alias）**

当在SQL语句中连接多个表时, 请使用表的别名并把别名前缀于每个Column上.这样一来,就可以减少解析的时间并减少那些由Column歧义引起的语法错误。

Column歧义指的是由于SQL中不同的表具有相同的Column名,当SQL语句中出现这个Column时,SQL解析器无法判断这个Column的归属。

**16. 用EXISTS替代IN**

在许多基于基础表的查询中，为了满足一个条件 ，往往需要对另一个表进行联接。在这种情况下，使用EXISTS(或NOT EXISTS)通常将提高查询的效率。

低效：

SELECT \* FROM EMP (基础表)

WHERE EMPNO > **0**

AND DEPTNO IN (SELECT DEPTNO

FROM DEPT

WHERE LOC = ‘MELB’)

高效：

[复制代码](javascript:void(0);)

SELECT \* FROM EMP (基础表)

WHERE EMPNO > **0**

AND EXISTS (SELECT ‘X’

FROM DEPT

WHERE DEPT.DEPTNO = EMP.DEPTNO

AND LOC = ‘MELB’)

[复制代码](javascript:void(0);)

**17. 用NOT EXISTS替代NOT IN**

在子查询中，NOT IN子句将执行一个内部的排序和合并，对子查询中的表执行一个全表遍历，因此是非常低效的。

为了避免使用NOT IN，可以把它改写成外连接（Outer Joins）或者NOT EXISTS。

低效：

SELECT …

FROM EMP

WHERE DEPT\_NO NOT IN （SELECT DEPT\_NO

FROM DEPT

WHERE DEPT\_CAT=’A’）

高效：

[复制代码](javascript:void(0);)

SELECT ….

FROM EMP E

WHERE NOT EXISTS （SELECT ‘X’

FROM DEPT D

WHERE D.DEPT\_NO = E.DEPT\_NO

AND DEPT\_CAT = ‘A’）

[复制代码](javascript:void(0);)

**18. 用表连接替换EXISTS**

通常来说 ，采用表连接的方式比EXISTS更有效率 。

低效：

[复制代码](javascript:void(0);)

SELECT ENAME

FROM EMP E

WHERE EXISTS （SELECT ‘X’

FROM DEPT

WHERE DEPT\_NO = E.DEPT\_NO

AND DEPT\_CAT = ‘A’）

[复制代码](javascript:void(0);)

高效：

SELECT ENAME

FROM DEPT D，EMP E

WHERE E.DEPT\_NO = D.DEPT\_NO

AND DEPT\_CAT = ‘A’

**19. 用EXISTS替换DISTINCT**

当提交一个包含对多表信息（比如部门表和雇员表）的查询时，避免在SELECT子句中使用DISTINCT。 一般可以考虑用EXIST替换。

EXISTS 使查询更为迅速，因为RDBMS核心模块将在子查询的条件一旦满足后，立刻返回结果。

低效：

SELECT DISTINCT DEPT\_NO，DEPT\_NAME

FROM DEPT D，EMP E

WHERE D.DEPT\_NO = E.DEPT\_NO

高效：

SELECT DEPT\_NO，DEPT\_NAME

FROM DEPT D

WHERE EXISTS （SELECT ‘X’

FROM EMP E

WHERE E.DEPT\_NO = D.DEPT\_NO

**20. 识别低效的SQL语句**

下面的SQL工具可以找出低效SQL ：

[复制代码](javascript:void(0);)

SELECT EXECUTIONS, DISK\_READS, BUFFER\_GETS,

ROUND ((BUFFER\_GETS-DISK\_READS)/BUFFER\_GETS, **2**) Hit\_radio,

ROUND (DISK\_READS/EXECUTIONS, **2**) Reads\_per\_run,

SQL\_TEXT

FROM V$SQLAREA

WHERE EXECUTIONS>**0**

AND BUFFER\_GETS > **0**

AND (BUFFER\_GETS-DISK\_READS)/BUFFER\_GETS < **0.8**

ORDER BY **4** DESC

[复制代码](javascript:void(0);)

另外也可以使用SQL Trace工具来收集正在执行的SQL的性能状态数据，包括解析次数，执行次数，CPU使用时间等 。

**21. 用Explain Plan分析SQL语句**

EXPLAIN PLAN 是一个很好的分析SQL语句的工具, 它甚至可以在不执行SQL的情况下分析语句. 通过分析, 我们就可以知道ORACLE是怎么样连接表, 使用什么方式扫描表(索引扫描或全表扫描)以及使用到的索引名称。

**22. SQL PLUS的TRACE**

[复制代码](javascript:void(0);)

SQL> list

**1** SELECT \*

**2** FROM dept, emp

**3**\* WHERE emp.deptno = dept.deptno

SQL> set autotrace traceonly /\*traceonly 可以不显示执行结果\*/

SQL> /

**14** rows selected.

Execution Plan

----------------------------------------------------------

**0** SELECT STATEMENT Optimizer=CHOOSE

**1** **0** NESTED LOOPS

**2** **1** TABLE ACCESS (FULL) OF 'EMP'

**3** **1** TABLE ACCESS (BY INDEX ROWID) OF 'DEPT'

**4** **3** INDEX (UNIQUE SCAN) OF 'PK\_DEPT' (UNIQUE)

[复制代码](javascript:void(0);)

**23. 用索引提高效率**

**（1）特点**

优点： 提高效率 主键的唯一性验证

代价： 需要空间存储 定期维护

重构索引：

ALTER INDEX <INDEXNAME> REBUILD <TABLESPACENAME>

**（2）Oracle对索引有两种访问模式**

* 索引唯一扫描 (Index Unique Scan)
* 索引范围扫描 (Index Range Scan)

**（3）基础表的选择**

* 基础表(Driving Table)是指被最先访问的表(通常以全表扫描的方式被访问)。 根据优化器的不同，SQL语句中基础表的选择是不一样的。
* 如果你使用的是CBO (COST BASED OPTIMIZER)，优化器会检查SQL语句中的每个表的物理大小，索引的状态，然后选用花费最低的执行路径。
* 如果你用RBO (RULE BASED OPTIMIZER)， 并且所有的连接条件都有索引对应，在这种情况下，基础表就是FROM 子句中列在最后的那个表。

**（4）多个平等的索引**

* 当SQL语句的执行路径可以使用分布在多个表上的多个索引时，ORACLE会同时使用多个索引并在运行时对它们的记录进行合并，检索出仅对全部索引有效的记录。
* 在ORACLE选择执行路径时，唯一性索引的等级高于非唯一性索引。然而这个规则只有当WHERE子句中索引列和常量比较才有效。如果索引列和其他表的索引类相比较。这种子句在优化器中的等级是非常低的。
* 如果不同表中两个相同等级的索引将被引用，FROM子句中表的顺序将决定哪个会被率先使用。 FROM子句中最后的表的索引将有最高的优先级。
* 如果相同表中两个相同等级的索引将被引用，WHERE子句中最先被引用的索引将有最高的优先级。

**（5）等式比较优先于范围比较**

DEPTNO上有一个非唯一性索引，EMP\_CAT也有一个非唯一性索引。

SELECT ENAME

FROM EMP

WHERE DEPTNO > **20**

AND EMP\_CAT = ‘A’;

这里只有EMP\_CAT索引被用到,然后所有的记录将逐条与DEPTNO条件进行比较. 执行路径如下:

TABLE ACCESS BY ROWID ON EMP

INDEX RANGE SCAN ON CAT\_IDX

即使是唯一性索引，如果做范围比较，其优先级也低于非唯一性索引的等式比较。

**（6）不明确的索引等级**

当ORACLE无法判断索引的等级高低差别，优化器将只使用一个索引,它就是在WHERE子句中被列在最前面的。

DEPTNO上有一个非唯一性索引，EMP\_CAT也有一个非唯一性索引。

SELECT ENAME

FROM EMP

WHERE DEPTNO > **20**

AND EMP\_CAT > ‘A’;

这里, ORACLE只用到了DEPT\_NO索引. 执行路径如下:

TABLE ACCESS BY ROWID ON EMP

INDEX RANGE SCAN ON DEPT\_IDX

**（7）强制索引失效**

如果两个或以上索引具有相同的等级，你可以强制命令ORACLE优化器使用其中的一个(通过它,检索出的记录数量少) 。

SELECT ENAME

FROM EMP

WHERE EMPNO = **7935**

AND DEPTNO + **0** = **10** /\*DEPTNO上的索引将失效\*/

AND EMP\_TYPE || ‘’ = ‘A’ /\*EMP\_TYPE上的索引将失效\*/

**（8）避免在索引列上使用计算**

WHERE子句中，如果索引列是函数的一部分。优化器将不使用索引而使用全表扫描。

低效：

SELECT …

FROM DEPT

WHERE SAL \* **12** > **25000**;

高效：

SELECT …

FROM DEPT

WHERE SAL > **25000**/**12**;

**（9）自动选择索引**

如果表中有两个以上（包括两个）索引，其中有一个唯一性索引，而其他是非唯一性索引。在这种情况下，ORACLE将使用唯一性索引而完全忽略非唯一性索引。

SELECT ENAME

FROM EMP

WHERE EMPNO = **2326**

AND DEPTNO = **20** ;

这里，只有EMPNO上的索引是唯一性的，所以EMPNO索引将用来检索记录。

TABLE ACCESS BY ROWID ON EMP

INDEX UNIQUE SCAN ON EMP\_NO\_IDX

**（10）避免在索引列上使用NOT**

通常，我们要避免在索引列上使用NOT，NOT会产生在和在索引列上使用函数相同的影响。当ORACLE遇到NOT，它就会停止使用索引转而执行全表扫描。

低效: (这里，不使用索引)

SELECT …

FROM DEPT

WHERE NOT DEPT\_CODE = **0**

高效：(这里，使用了索引)

SELECT …

FROM DEPT

WHERE DEPT\_CODE > **0**

**24. 用 >= 替代 >**

如果DEPTNO上有一个索引

高效:

SELECT \*

FROM EMP

WHERE DEPTNO >=**4**

低效：

SELECT \*

FROM EMP

WHERE DEPTNO >**3**

两者的区别在于，前者DBMS将直接跳到第一个DEPT等于4的记录，而后者将首先定位到DEPTNO等于3的记录并且向前扫描到第一个DEPT大于3的记录.

**25. 用Union替换OR（适用于索引列）**

通常情况下，用UNION替换WHERE子句中的OR将会起到较好的效果。对索引列使用OR将造成全表扫描。 注意，以上规则只针对多个索引列有效。

高效：

[复制代码](javascript:void(0);)

SELECT LOC\_ID , LOC\_DESC , REGION

FROM LOCATION

WHERE LOC\_ID = **10**

UNION

SELECT LOC\_ID , LOC\_DESC , REGION

FROM LOCATION

WHERE REGION = “MELBOURNE”

[复制代码](javascript:void(0);)

低效：

SELECT LOC\_ID , LOC\_DESC , REGION

FROM LOCATION

WHERE LOC\_ID = **10** OR REGION = “MELBOURNE”

**26. 用IN替换OR**

低效：

SELECT….

FROM LOCATION

WHERE LOC\_ID = **10**

OR LOC\_ID = **20**

OR LOC\_ID = **30**

高效：

SELECT…

FROM LOCATION

WHERE LOC\_IN IN （**10**，**20**，**30**）

实际的执行效果还须检验，在ORACLE8i下， 两者的执行路径似乎是相同的。

**27. 避免在索引列上使用is null和is not null**

避免在索引中使用任何可以为空的列，ORACLE将无法使用该索引。

低效：（索引失效）

SELECT …

FROM DEPARTMENT

WHERE DEPT\_CODE IS NOT NULL;

高效：（索引有效）

SELECT …

FROM DEPARTMENT

WHERE DEPT\_CODE >=**0**;

**28. 总是使用索引的第一个列**

如果索引是建立在多个列上， 只有在它的第一个列(leading column)被where子句引用时， 优化器才会选择使用该索引。

[复制代码](javascript:void(0);)

SQL> create index multindex on multiindexusage(inda,indb);

Index created.

SQL> select \* from multiindexusage where indb = **1**;

Execution Plan

----------------------------------------------------------

**0** SELECT STATEMENT Optimizer=CHOOSE

**1** **0** TABLE ACCESS (FULL) OF 'MULTIINDEXUSAGE‘

[复制代码](javascript:void(0);)

很明显, 当仅引用索引的第二个列时,优化器使用了全表扫描而忽略了索引。

**29. 使用UNION ALL替代UNION**

当SQL语句需要UNION两个查询结果集合时，这两个结果集合会以UNION-ALL的方式被合并，然后在输出最终结果前进行排序。如果用UNION ALL替代UNION，这样排序就不是必要了，效率就会因此得到提高。

由于UNION ALL的结果没有经过排序，而且不过滤重复的记录，因此是否进行替换需要根据业务需求而定。

**30. 对UNION的优化**

由于UNION会对查询结果进行排序，而且过滤重复记录，因此其执行效率没有UNION ALL高。 UNION操作会使用到SORT\_AREA\_SIZE内存块，因此对这块内存的优化也非常重要。

可以使用下面的SQL来查询排序的消耗量 ：

select substr（name，**1**，**25**） "Sort Area Name"，

substr（value，**1**，**15**） "Value"

from v$sysstat

where name like 'sort%'

**31. 避免改变索引列的类型**

当比较不同数据类型的数据时， ORACLE自动对列进行简单的类型转换。

[复制代码](javascript:void(0);)

/\*假设EMP\_TYPE是一个字符类型的索引列.\*/

SELECT …

FROM EMP

WHERE EMP\_TYPE = **123**

/\*这个语句被ORACLE转换为:\*/

SELECT …

FROM EMP

WHERE TO\_NUMBER(EMP\_TYPE)=**123**

[复制代码](javascript:void(0);)

因为内部发生的类型转换，这个索引将不会被用到。

几点注意：

* 当比较不同数据类型的数据时，ORACLE自动对列进行简单的类型转换。
* 如果在索引列上面进行了隐式类型转换，在查询的时候将不会用到索引。
* 注意当字符和数值比较时，ORACLE会优先转换数值类型到字符类型。
* 为了避免ORACLE对SQL进行隐式的类型转换，最好把类型转换用显式表现出来。

**32. 使用提示（Hints）**

* FULL hint 告诉ORACLE使用全表扫描的方式访问指定表。
* ROWID hint 告诉ORACLE使用TABLE ACCESS BY ROWID的操作访问表。
* CACHE hint 来告诉优化器把查询结果数据保留在SGA中。
* INDEX Hint 告诉ORACLE使用基于索引的扫描方式。

**其他的Oracle Hints**

* ALL\_ROWS
* FIRST\_ROWS
* RULE
* USE\_NL
* USE\_MERGE
* USE\_HASH 等等。

这是一个很有技巧性的工作。建议只针对特定的，少数的SQL进行hint的优化。

**33. 几种不能使用索引的WHERE子句**

（1）下面的例子中，‘!=’ 将不使用索引 ，索引只能告诉你什么存在于表中，而不能告诉你什么不存在于表中。

不使用索引：

SELECT ACCOUNT\_NAME

FROM TRANSACTION

WHERE AMOUNT !=**0**；

使用索引：

SELECT ACCOUNT\_NAME

FROM TRANSACTION

WHERE AMOUNT > **0**；

（2）下面的例子中，‘||’是字符连接函数。就象其他函数那样，停用了索引。

不使用索引：

SELECT ACCOUNT\_NAME，AMOUNT

FROM TRANSACTION

WHERE ACCOUNT\_NAME||ACCOUNT\_TYPE=’AMEXA’；

使用索引：

SELECT ACCOUNT\_NAME，AMOUNT

FROM TRANSACTION

WHERE ACCOUNT\_NAME = ‘AMEX’

AND ACCOUNT\_TYPE=’ A’；

（3）下面的例子中，‘+’是数学函数。就象其他数学函数那样，停用了索引。

不使用索引：

SELECT ACCOUNT\_NAME，AMOUNT

FROM TRANSACTION

WHERE AMOUNT + **3000** >**5000**；

使用索引：

SELECT ACCOUNT\_NAME，AMOUNT

FROM TRANSACTION

WHERE AMOUNT > **2000** ；

（4）下面的例子中，相同的索引列不能互相比较，这将会启用全表扫描。

不使用索引：

SELECT ACCOUNT\_NAME, AMOUNT

FROM TRANSACTION

WHERE ACCOUNT\_NAME = NVL(:ACC\_NAME, ACCOUNT\_NAME)

使用索引：

SELECT ACCOUNT\_NAME，AMOUNT

FROM TRANSACTION

WHERE ACCOUNT\_NAME LIKE NVL(:ACC\_NAME, ’%’)

**34. 连接多个扫描**

如果对一个列和一组有限的值进行比较，优化器可能执行多次扫描并对结果进行合并连接。

举例：

SELECT \*

FROM LODGING

WHERE MANAGER IN (‘BILL GATES’, ’KEN MULLER’)

优化器可能将它转换成以下形式：

SELECT \*

FROM LODGING

WHERE MANAGER = ‘BILL GATES’

OR MANAGER = ’KEN MULLER’

**35. CBO下使用更具选择性的索引**

* 基于成本的优化器（CBO，Cost-Based Optimizer）对索引的选择性进行判断来决定索引的使用是否能提高效率。
* 如果检索数据量超过30%的表中记录数，使用索引将没有显著的效率提高。
* 在特定情况下，使用索引也许会比全表扫描慢。而通常情况下，使用索引比全表扫描要块几倍乃至几千倍！

**36. 避免使用耗费资源的操作**

* 带有DISTINCT，UNION，MINUS，INTERSECT，ORDER BY的SQL语句会启动SQL引擎执行耗费资源的排序（SORT）功能。DISTINCT需要一次排序操作，而其他的至少需要执行两次排序。
* 通常，带有UNION，MINUS，INTERSECT的SQL语句都可以用其他方式重写。

**37. 优化GROUP BY**

提高GROUP BY语句的效率，可以通过将不需要的记录在GROUP BY之前过滤掉。

低效：

SELECT JOB ，AVG（SAL）

FROM EMP

GROUP BY JOB

HAVING JOB = ‘PRESIDENT’

OR JOB = ‘MANAGER’

高效：

SELECT JOB，AVG（SAL）

FROM EMP

WHERE JOB = ‘PRESIDENT’

OR JOB = ‘MANAGER’

GROUP BY JOB

**38. 使用日期**

当使用日期时，需要注意如果有超过5位小数加到日期上，这个日期会进到下一天!

[复制代码](javascript:void(0);)

SELECT TO\_DATE（‘**01**-JAN-**93**’+.**99999**）

FROM DUAL

Returns：

’**01**-JAN-**93** **23**:**59**:**59**’

SELECT TO\_DATE（‘**01**-JAN-**93**’+.**999999**）

FROM DUAL

Returns：

’**02**-JAN-**93** **00**:**00**:**00**’

[复制代码](javascript:void(0);)

**39. 使用显示游标(CURSORS)**

使用隐式的游标，将会执行两次操作。第一次检索记录，第二次检查TOO MANY ROWS 这个exception。而显式游标不执行第二次操作。

**40. 分离表和索引**

* 总是将你的表和索引建立在不同的表空间内（TABLESPACES）。
* 决不要将不属于ORACLE内部系统的对象存放到SYSTEM表空间里。
* 确保数据表空间和索引表空间置于不同的硬盘上。