# 第五章

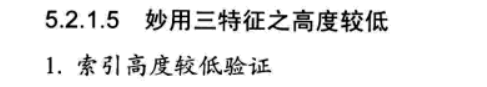
## 5.2.1-Btree树索引

### 5.2.1.3索引结构三大特点

1.索引高度较低

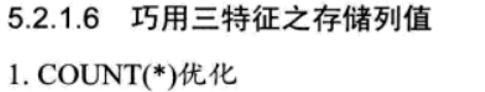
2.索引存储列值

3.索引本身有序











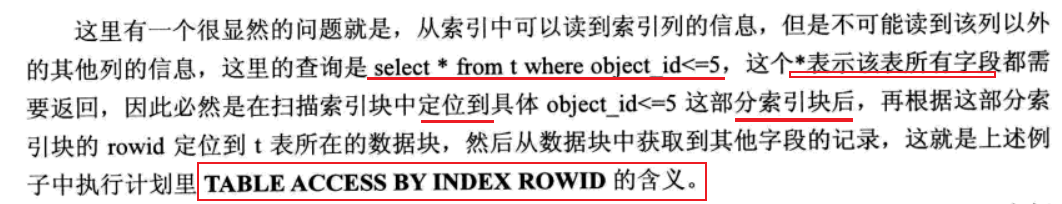


---

### 5.2.1.6

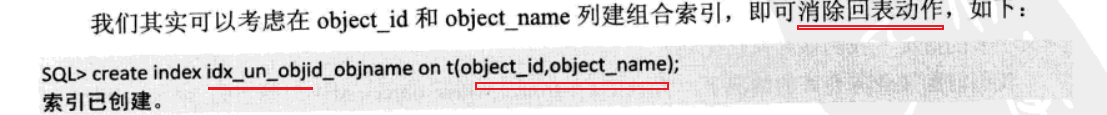
#### 4.索引回表与优化





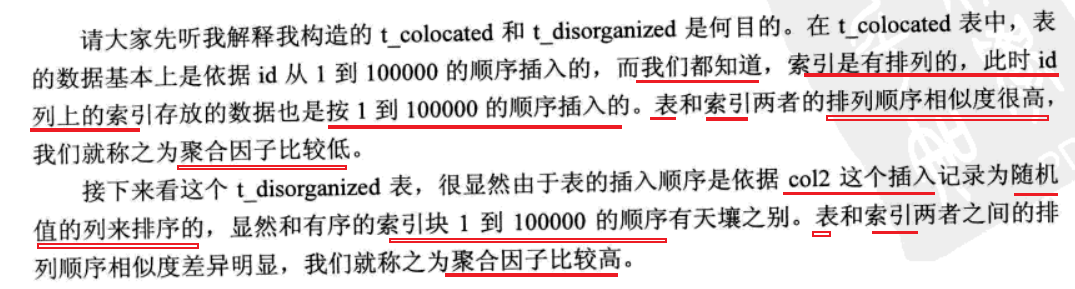
只返回索引字段可避免回表。

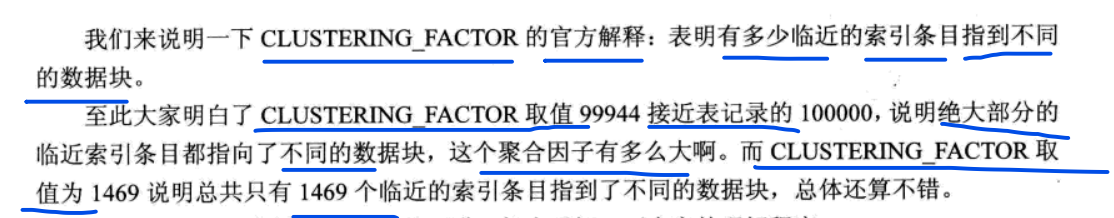
##### 联合索引

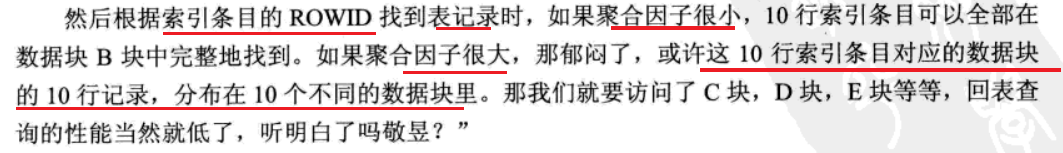


##### 聚合因子

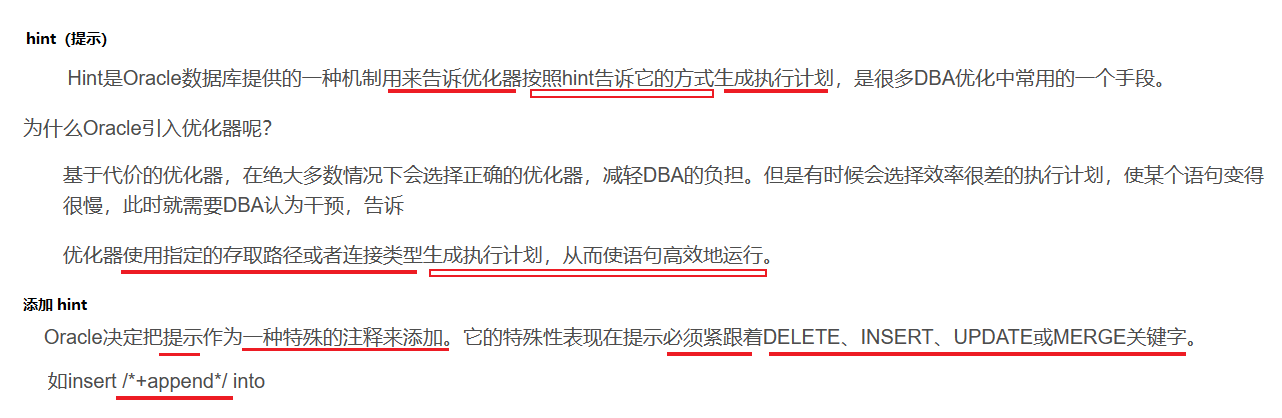




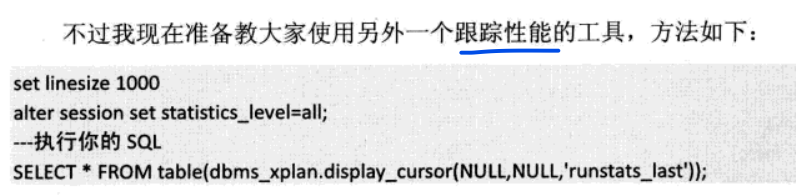


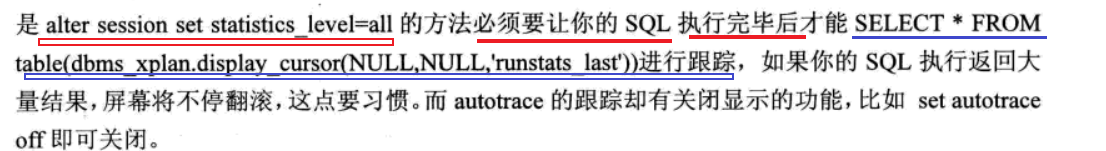


##### Hint

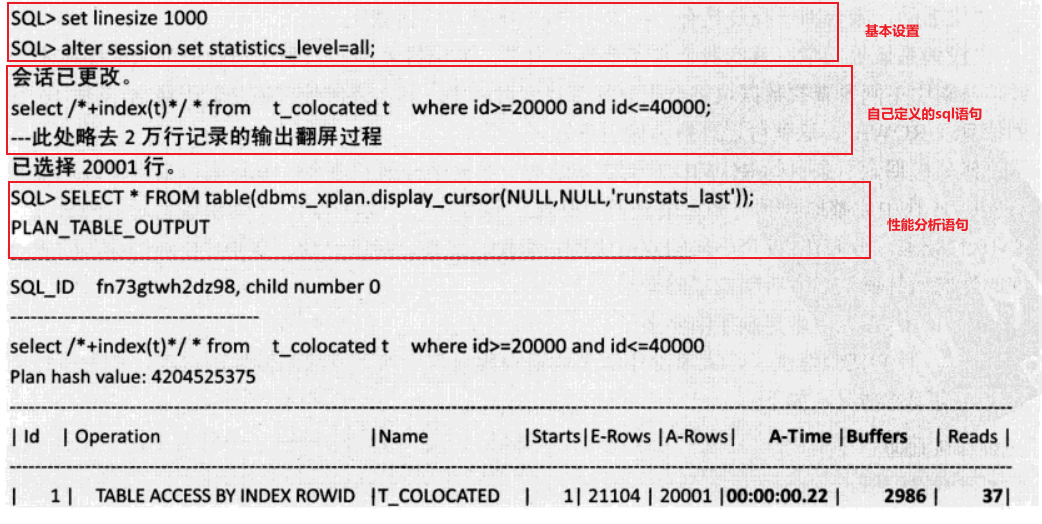


##### 性能跟踪工具

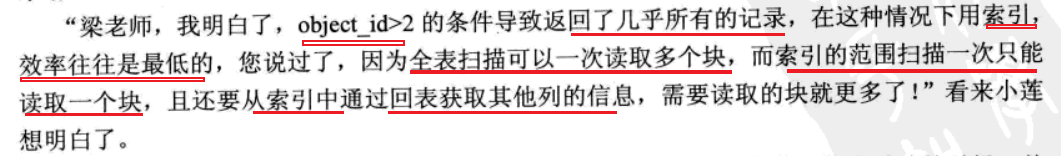




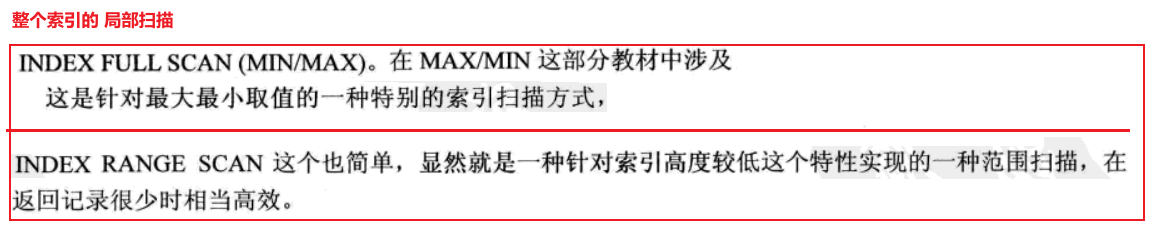
案例



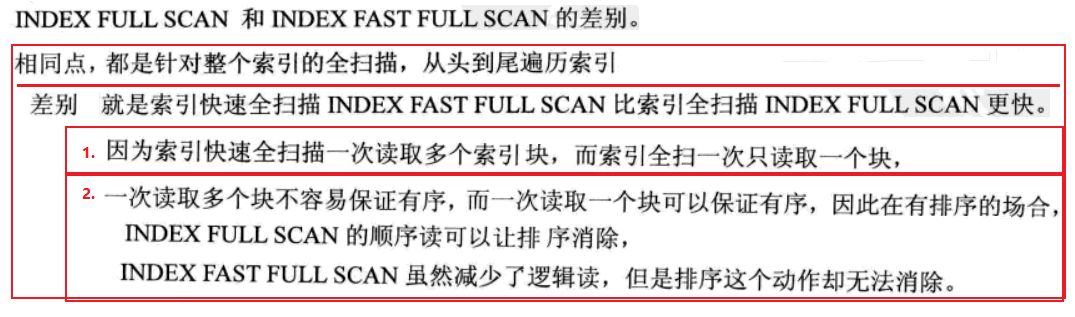
### 5.2.1.7-三特征之索引有序



#### 执行计划总结



整个索引全扫描



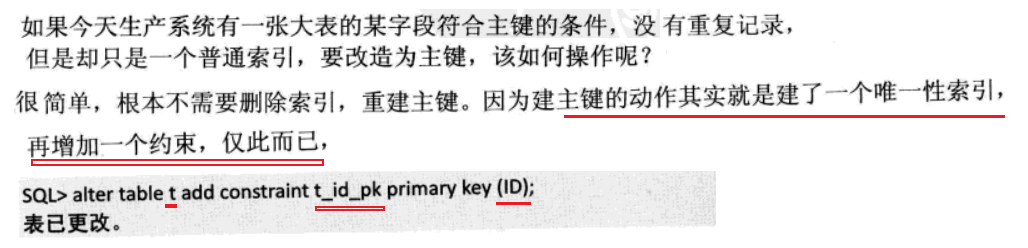
Union 始终会进行排序的， 如果两张表都不重复 使用 union all 进行优化

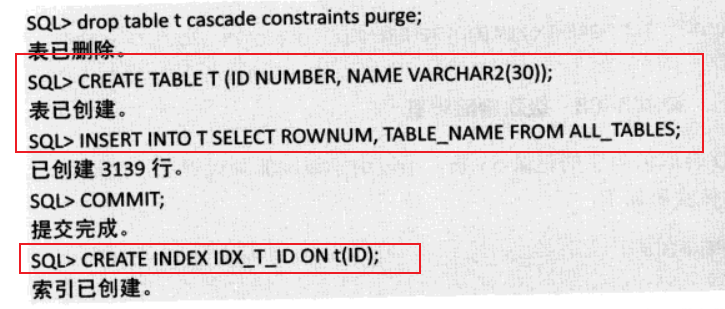
### 5.2.1.8

#### 级联删除



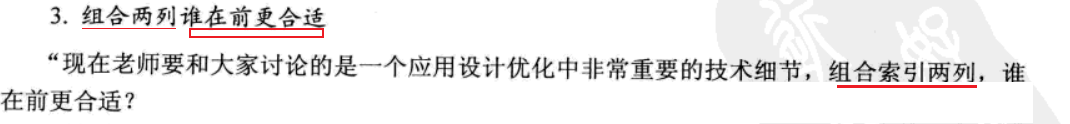
#### 索引变主键



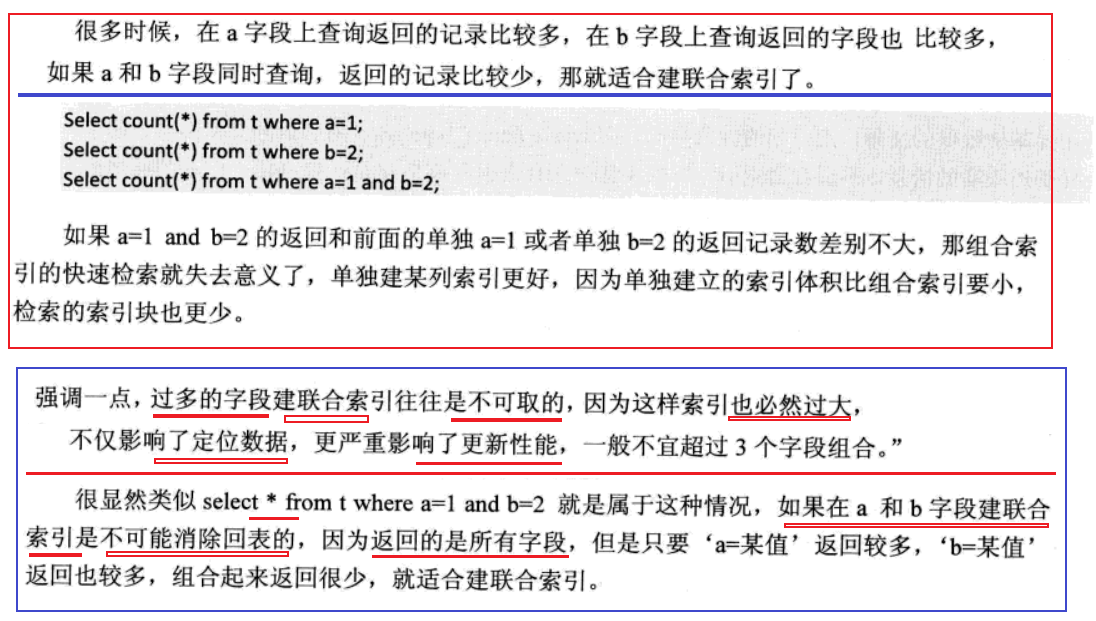


### 5.2.1.9组合索引设计要领

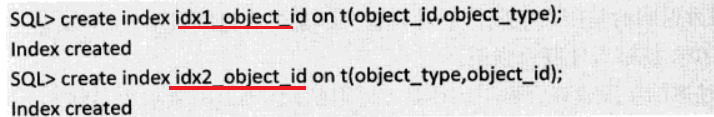
##### 问题



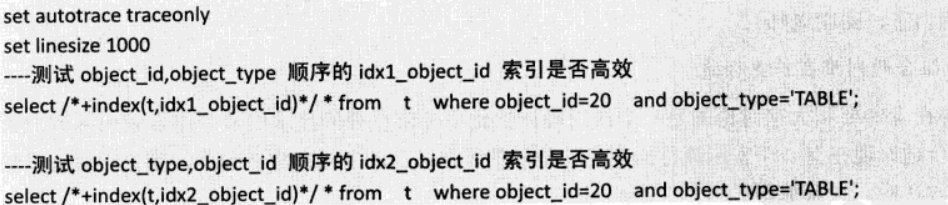
##### 组合索引使用条件

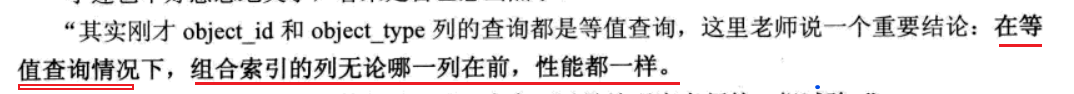


##### 组合索引

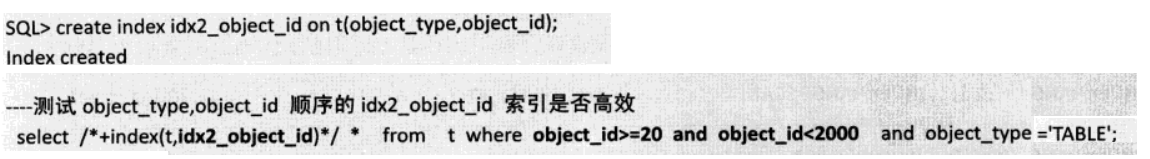


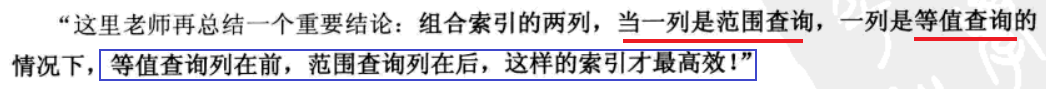
##### 等值查询效率一致

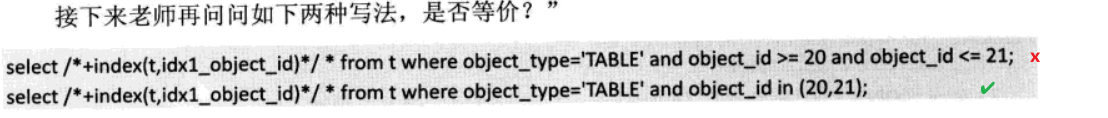


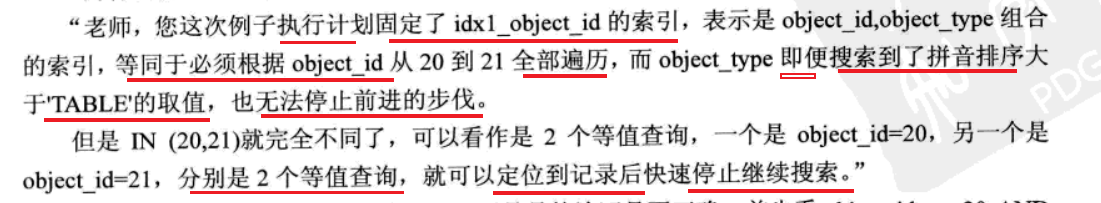


##### 范围查询效率差异

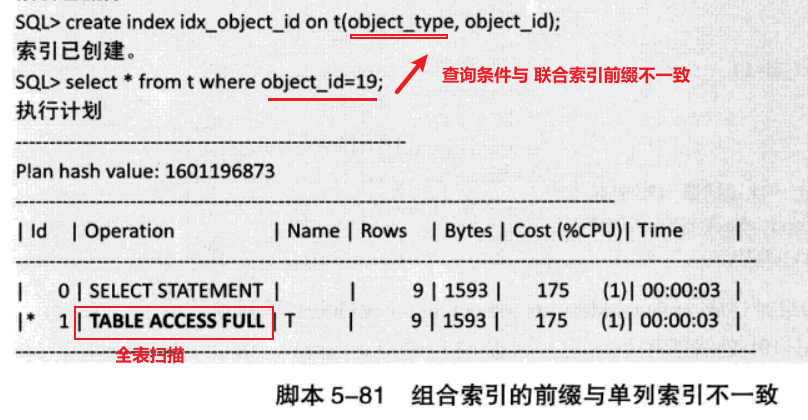
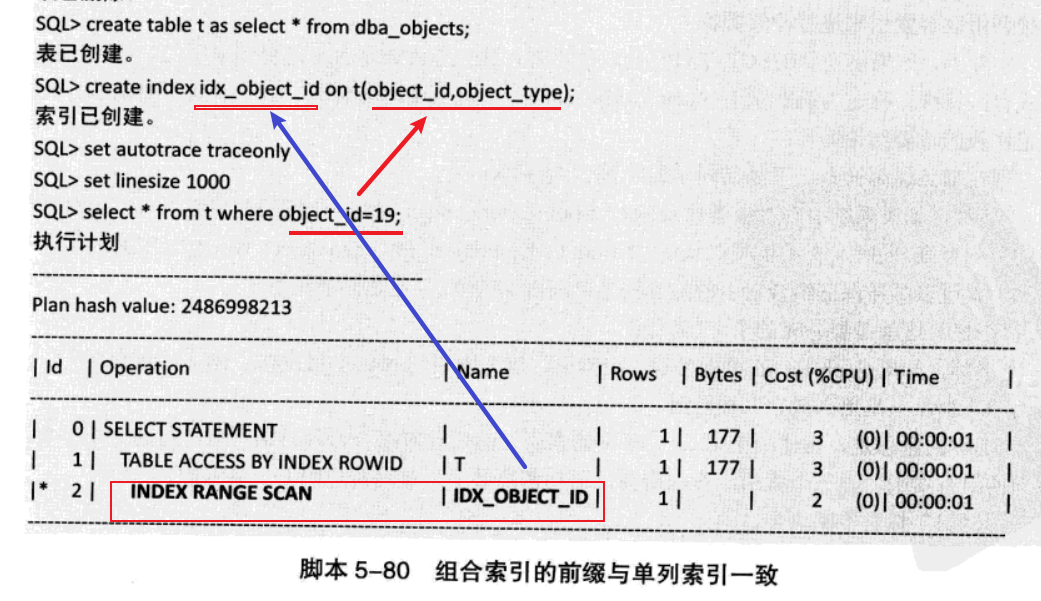


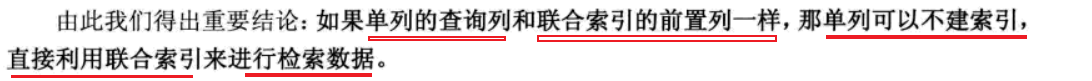






##### 联合索引-单列查询





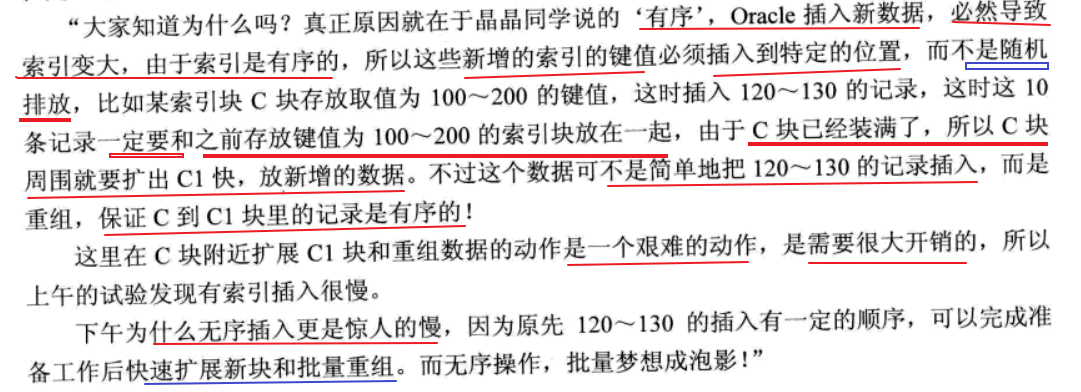
**索引创建的多了后，会严重影响 表的更新速度**

### 5.2.1.10-索引的危害



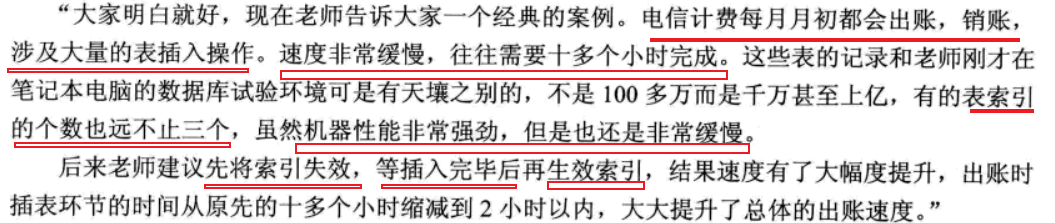






**由于索引导致数据插入速度很慢，解决方案是，先把索引删掉，待数据插入完毕后，重新创建索引。**

##### 案例

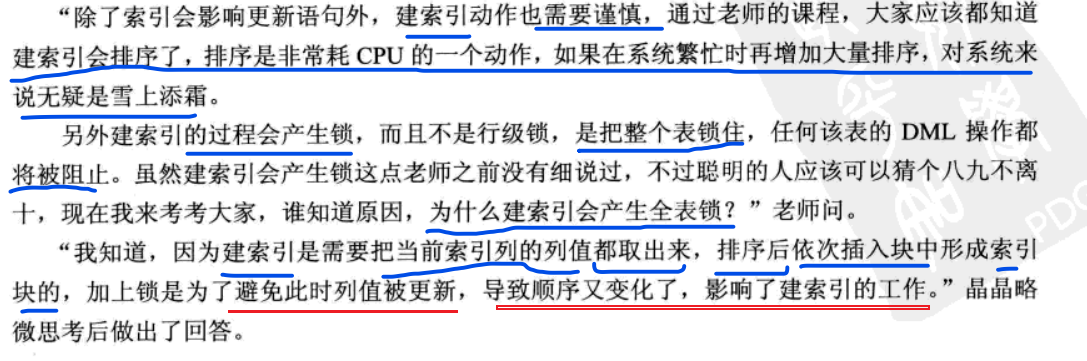




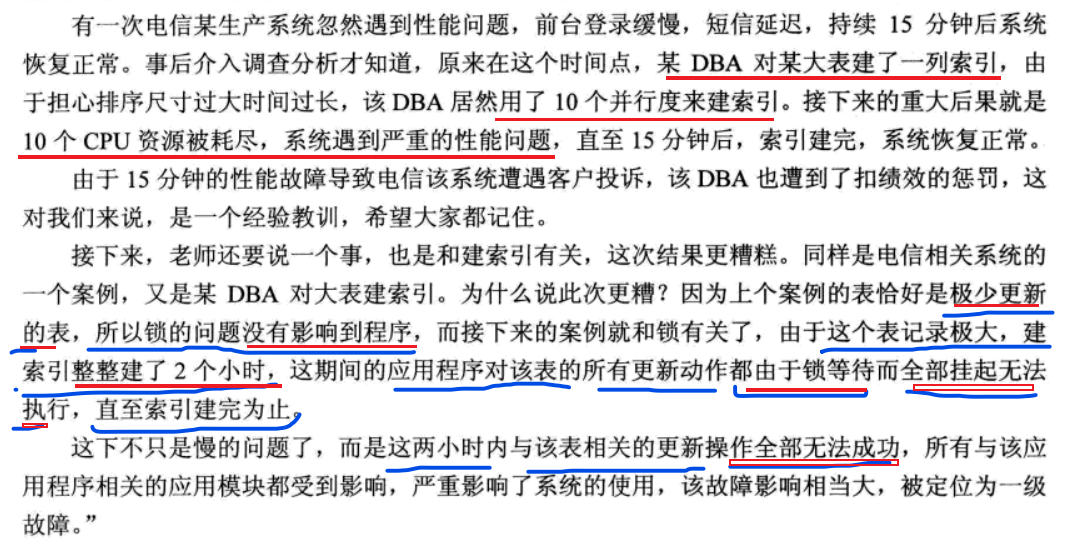




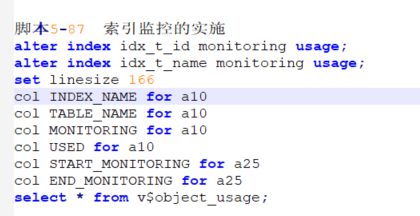
**原因-锁**



**案例-锁**

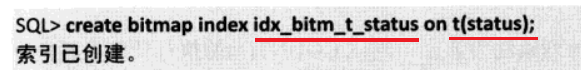


### 5.2.1.11索引监控

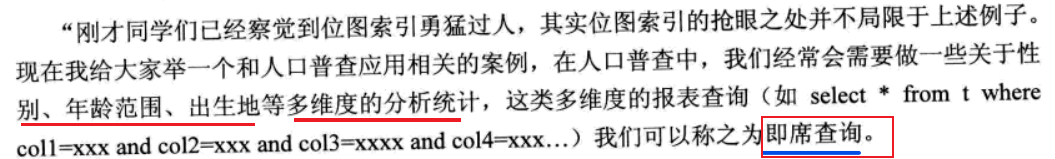


## 5.2.2-位图索引

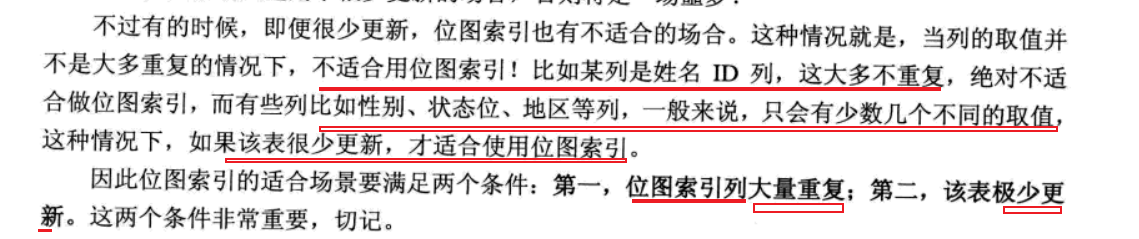
### 1.创建位图索引



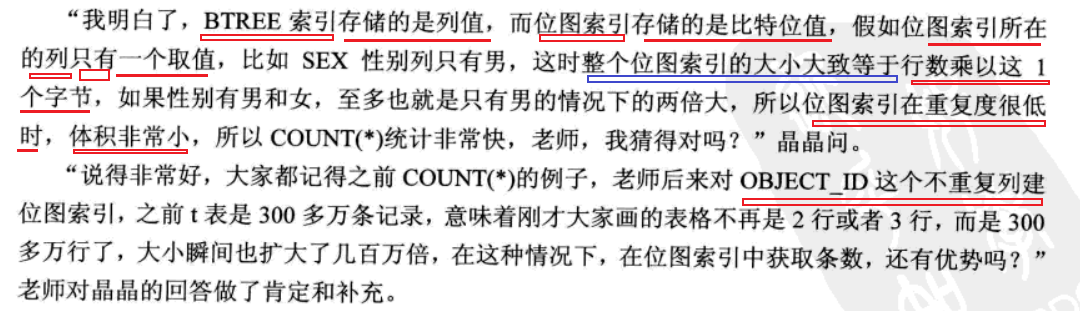
### 2.既席查询

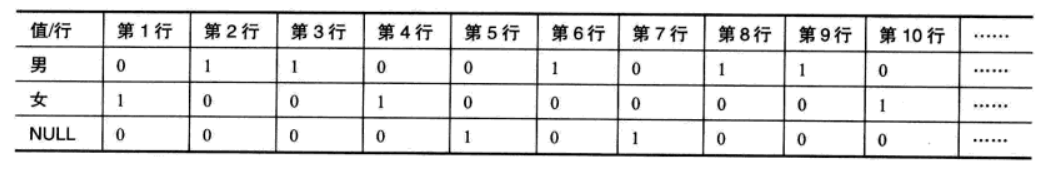


### 3.使用条件

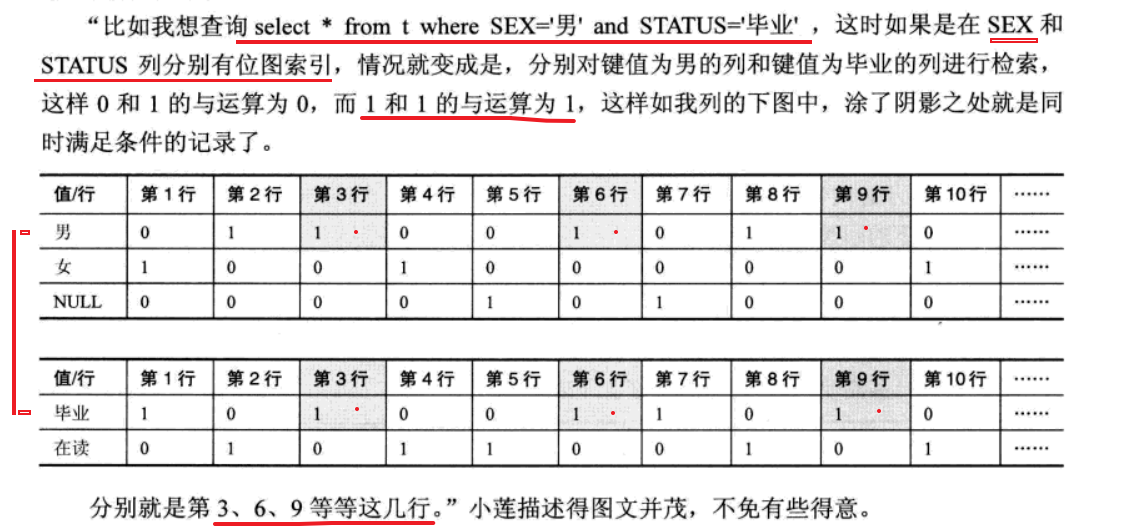


### 4.原理



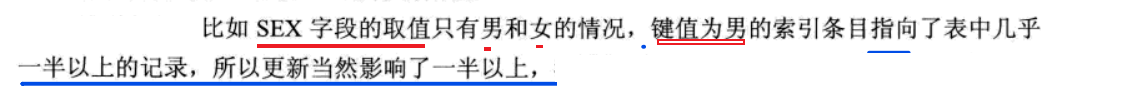


### 既席查询为啥快

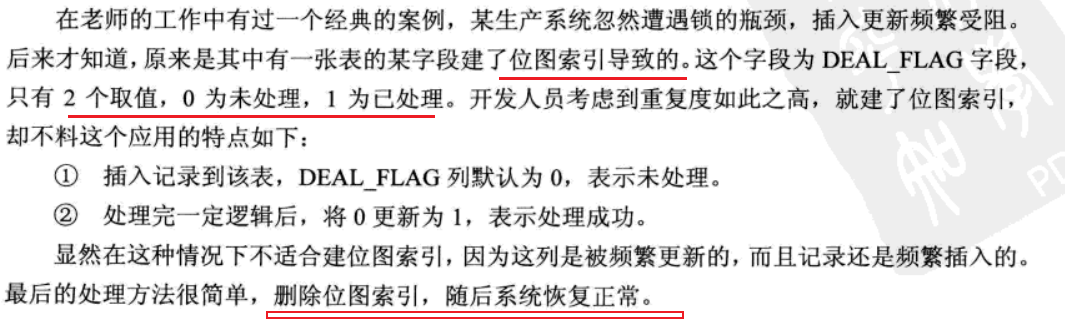


### 为啥位图索引的数据更新很痛苦？

**位图索引字段 唯一值越少 在更新数据的时候 越容易被锁。 因为添加数据后，索引需要更新。**



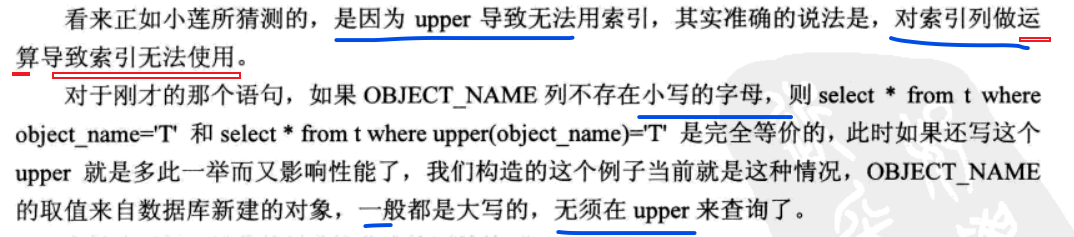
**案例**



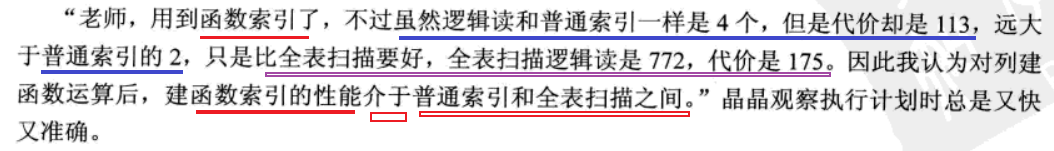
## 5.2.3-函数与索引失效

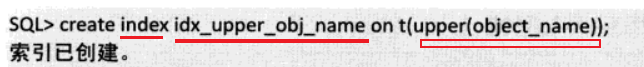
### 函数导致索引失效





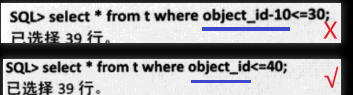
### 创建函数索引

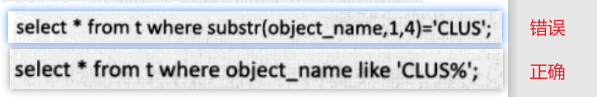






### 避免列运算案例





# 第六章