

SQL语句简单优化

www.huawei.com





前言

- 随着数据库中数据的增加，系统的响应速度就成为目前系统需要解决的最主要的问题之一。系统优化中一个很重要的方面就是SQL语句的优化。对于大量数据，劣质的SQL语句和优质SQL语句之间的速度差别可以达到上百倍。
- 在编写SQL语句时，如果清楚SQL优化规则，有助于写出高性能的SQL语句。



培训目标

- 学完本课程后，您应该能：
 - 了解SQL语句的简单优化规则





目 录

1. SQL语句简单优化

1.1 避免使用 ‘*’

1.2 使用表的别名

1.3 WHERE子句中的连接顺序

1.4 避免使用NOT命令

1.5 用TRUNCATE代替DELETE

1.6 尽量多使用COMMIT

1.7 避免在索引列上使用函数

1.1 避免使用 ‘*’

- 当你能在SELECT子句中列出所有的列时，使用‘*’是一个方便的方法，不幸的是，这是一个非常低效的方法。

```
SELECT * FROM emp;
```

在解析过程中，会将 ‘*’
一次转换成所有的列名，
并通过查询数据字典完成，
意味着耗费更多的时间。



1.2 使用表的别名

- 当在SQL语句中连接多个表时，使用表的别名并把别名前缀于每个column上。这样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。

```
SELECT e.ename, e.sal, d.deptno FROM emp e, dept d WHERE  
d.deptno=e.deptno;
```

1.3 WHERE子句中的连接顺序

```
SELECT * FROM emp e, dept d WHERE d.deptno>10 and e.deptno=30;
```

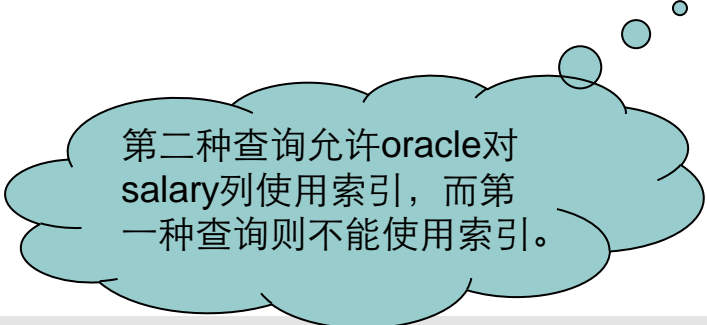
满足条件的数据较少，能大大缩小查询范围，则将最具有选择性部分放到where子句中的最后，响应越快。

1.4 避免使用NOT命令

- 通过使用>=,<=等，避免使用NOT命令

```
SELECT * FROM emp WHERE salary<>3000;
```

```
SELECT * FROM emp WHERE salary<3000 or salary>3000;
```



第二种查询允许oracle对salary列使用索引，而第一种查询则不能使用索引。

1.5 用TRUNCATE代替DELETE

DELETE

如果没有commit操作，回滚段中会存放删除操作恢复的信息。

TRUNCATE

回滚段不再存放任何可被恢复的信息，因此很少的资源被调用，执行时间也会很短。

1.6 尽量多使用COMMIT

- 事务是消耗资源的，大事务还容易引起死锁。
- COMMIT所释放的资源：

回滚段上用户恢复数据的信息

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

1.7 避免在索引列上使用函数

- 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*2>25000;
```

- 高效：

```
SELECT * FROM dept WHERE sal>25000/2;
```

小测试

- 下列描述中不属于SQL语句优化的是（C）
 - SQL语句的优化是以开发者交互进行的操作
 - 提升系统SQL语句执行效率
 - 提交数据操作SQL语句
 - 劣质的SQL语句和优质SQL语句之间的速度差可以达到上百倍
- COMMIT所释放的资源包括（ABCD）
 - 回滚段上用户恢复数据的信息
 - 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

小测试

- 下列哪个不是SQL语句优化规则（D）
 - 尽量多使用COMMIT
 - 避免在索引列上使用函数
 - 避免使用 ‘*’
 - 在索引列上使用函数
- 下列哪个语句执行速度最快（C）
 - SELECT * FROM emp WHERE salary<>3000;
 - SELECT * FROM emp WHERE salary<3000 or salary>3000;
 - SELECT ename,job,sal FROM emp WHERE salary<3000 or salary>3000;
 - SELECT ename,job,sal FROM emp WHERE salary<>3000;

谢谢

www.huawei.com