# 一、SQL执行顺序

1、from子句组装来自不同数据源的数据；

2、where子句基于指定的条件对记录行进行筛选；

3、group by子句将数据划分为多个分组；

4、使用聚集函数进行计算；

5、使用having子句筛选分组；

6、计算所有的表达式；

7、select 的字段；

8、使用order by对结果集进行排序。

例子：

SELECT NAME,AGE FROM PEOPLE WHERE ROWNUM =1 ORDER BY AGE.

输出结果是什么？

# 二、where执行顺序

1.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

/\* 低效 \*/

SELECT \* FROM EMP E WHERE SEX = MALE AND AGE = 20;

/\* 高效 \*/

SELECT \* FROM EMP E WHERE AGE = 20 AND SEX = MALE;

# 三、oracle特定执行方式

1.rownum、rowid

rownum和rowid都是伪列，但是两者的根本是不同的，rownum是根据sql查询出的结果给每行分配一个逻辑编号，所以你的sql不同也就会导致最终rownum不同，

但是rowid是物理结构上的，在每条记录insert到数据库中时，都会有一个唯一的物理记录

通常在sql分页时或是查找某一范围内的记录时，我们会使用rownum。

select \*

from (select rownum rn, a.\* from emp a) t

where t.rn between 2 and 10;

我们在处理一张表中重复记录时经常用到rowid

delete from test t where rowid not in

(

select max(rowid) from test

group by a,b

)

2.（+）用法

用left join 的SQL code

select a.id,b.id

form a left jion b on a.id=b.id

用(+)的SQL code

select a.id,b.id

form a ,b

where a.id=b.id(+)

同理，right join也是一样

# 四、oracle比较基本的函数讲解

1.in 和 exists

select \* from A

where id in(select id from B)

以上查询使用了in语句,in()只执行一次,它查出B表中的所有id字段并缓存起来.之后,检查A表的id是否与B表中的id相等,如果相等则将A表的记录加入结果集中,直到遍历完A表的所有记录.

它的查询过程类似于以下过程

List resultSet=[];

Array A=(select \* from A);

Array B=(select id from B);

for(int i=0;i<A.length;i++) {

for(int j=0;j<B.length;j++) {

if(A[i].id==B[j].id) {

resultSet.add(A[i]);

break;

}

}

}

return resultSet;

可以看出,当B表数据较大时不适合使用in(),因为它会B表数据全部遍历一次.

如:A表有10000条记录,B表有1000000条记录,那么最多有可能遍历10000\*1000000次,效率很差.

再如:A表有10000条记录,B表有100条记录,那么最多有可能遍历10000\*100次,遍历次数大大减少,效率大大提升.

结论:in()适合B表比A表数据小的情况

select a.\* from A a

where exists(select 1 from B b where a.id=b.id)

以上查询使用了exists语句,exists()会执行A.length次,它并不缓存exists()结果集,因为exists()结果集的内容并不重要,重要的是结果集中是否有记录,如果有则返回true,没有则返回false.

它的查询过程类似于以下过程

List resultSet=[];

Array A=(select \* from A)

for(int i=0;i<A.length;i++) {

if(exists(A[i].id) { //执行select 1 from B b where b.id=a.id是否有记录返回

resultSet.add(A[i]);

}

}

return resultSet;

当B表比A表数据大时适合使用exists(),因为它没有那么遍历操作,只需要再执行一次查询就行.

如:A表有10000条记录,B表有1000000条记录,那么exists()会执行10000次去判断A表中的id是否与B表中的id相等.

如:A表有10000条记录,B表有100000000条记录,那么exists()还是执行10000次,因为它只执行A.length次,可见B表数据越多,越适合exists()发挥效果.

再如:A表有10000条记录,B表有100条记录,那么exists()还是执行10000次,还不如使用in()遍历10000\*100次,因为in()是在内存里遍历比较,而exists()需要查询数据库,我们都知道查询数据库所消耗的性能更高,而内存比较很快.

结论:exists()适合B表比A表数据大的情况

当A表数据与B表数据一样大时,in与exists效率差不多,可任选一个使用.

2.case when&decode（decode为oracle专用）

CASE WHEN 几种用法：

CASE sex

WHEN '1' THEN '男'

WHEN '2' THEN '女'

ELSE '其他' END

CASE WHEN sex = '1' THEN '男'

WHEN sex = '2' THEN '女'

ELSE '其他' END

CASE WHEN col\_1 IN ( 'a', 'b') THEN '第一类'

WHEN col\_1 IN ('a') THEN '第二类'

ELSE'其他' END;

DECODE用法：

decode(条件,值1,返回值1,值2,返回值2,...值n,返回值n,缺省值)

该函数的含义如下：

IF 条件=值1 THEN

　　　　RETURN(翻译值1)

ELSIF 条件=值2 THEN

　　　　RETURN(翻译值2)

　　　　......

ELSIF 条件=值n THEN

　　　　RETURN(翻译值n)

ELSE

　　　　RETURN(缺省值)

END IF

自学函数：

MINUS(减去),INTERSECT(交集)和UNION ALL(并集);

# 五、简单SQL查询优化\_索引的正确使用

尽量避免在WHERE子句中对字段进行函数或表达式操作，这将导致引擎放弃使用索引而进行全表扫描。如：

SELECT \* FROM T1 WHERE F1/2=100

应改为:

SELECT \* FROM T1 WHERE F1=100\*2

SELECT \* FROM RECORD WHERE SUBSTRING(CARD\_NO,1,4)=’5378’

应改为:

SELECT \* FROM RECORD WHERE CARD\_NO LIKE ‘5378%’

SELECT member\_number, first\_name, last\_name FROM members

WHERE DATEDIFF(yy,datofbirth,GETDATE()) > 21

应改为:

SELECT member\_number, first\_name, last\_name FROM members

WHERE dateofbirth < DATEADD(yy,-21,GETDATE())

SELECT STARTDATE FROM EMP WHERE TO\_CHAR(STARTDATE,'YYYYMMDD') <= '20160101'

应改为

SELECT STARTDATE FROM EMP WHERE STARTDATE <= TO\_CHAR('20160101','YYYYMMDD')

即：任何对列的操作都将导致表扫描，它包括数据库函数、计算表达式等等，查询时要尽可能将操作移至等号右边

避免使用!=或＜＞、IS NULL或IS NOT NULL、IN ，NOT IN等这样的操作符,因为这会使系统无法使用索引,而只能直接搜索表中的数据。例如:

SELECT id FROM employee WHERE id != 'B%'

优化器将无法通过索引来确定将要命中的行数,因此需要搜索该表的所有行。