# 单表使用索引常见的索引失效

## 1. 全值匹配我最爱

### 1.1 有以下SQL语句

EXPLAIN SELECT SQL\_NO\_CACHE \* FROM emp WHERE emp.age=30

EXPLAIN SELECT SQL\_NO\_CACHE \* FROM emp WHERE emp.age=30 and deptid=4

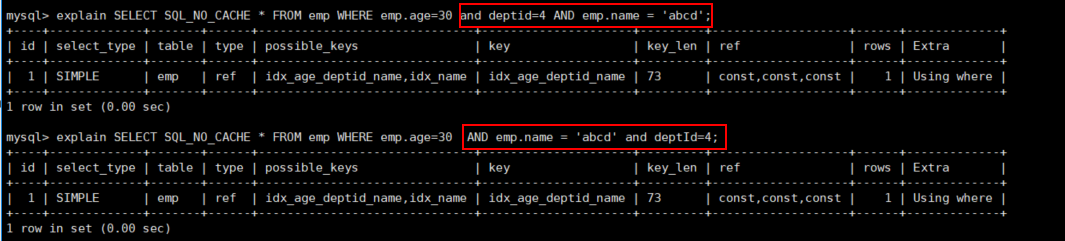
EXPLAIN SELECT SQL\_NO\_CACHE \* FROM emp WHERE emp.age=30 and deptid=4 AND emp.name = 'abcd'

### 1.2 建立索引

CREATE INDEX idx\_age\_deptid\_name ON emp(age,deptid,NAME);

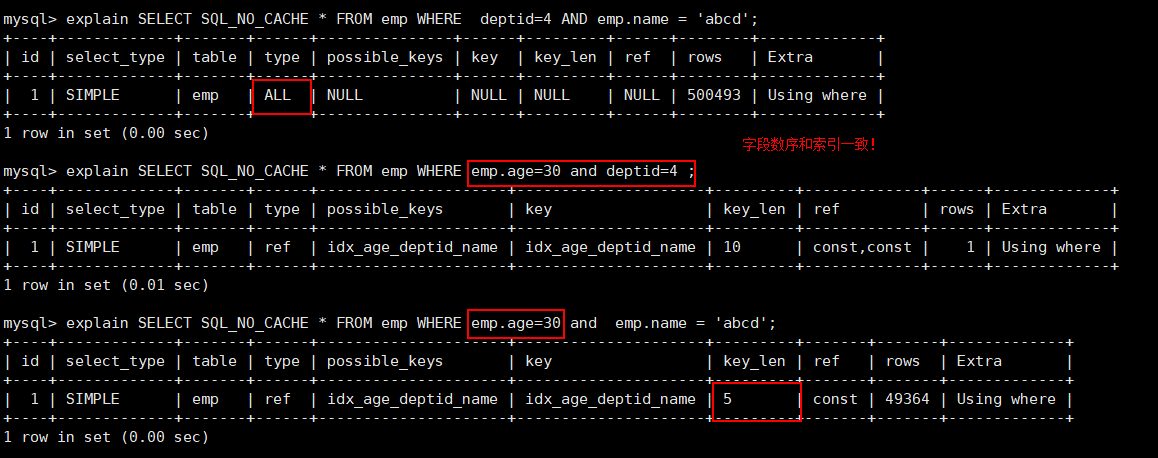


结论：全职匹配我最爱指的是，查询的字段按照顺序在索引中都可以匹配到！



SQL中查询字段的顺序，跟使用索引中字段的顺序，没有关系。优化器会在不影响SQL执行结果的前提下，给你自动地优化。

## 2. 最佳左前缀法则



查询字段与索引字段顺序的不同会导致，索引无法充分使用，甚至索引失效！

原因：使用复合索引，需要遵循最佳左前缀法则，即如果索引了多列，要遵守最左前缀法则。指的是查询从索引的最左前列开始并且不跳过索引中的列。

**结论：过滤条件要使用索引必须按照索引建立时的顺序，依次满足，一旦跳过某个字段，索引后面的字段都无法被使用。**

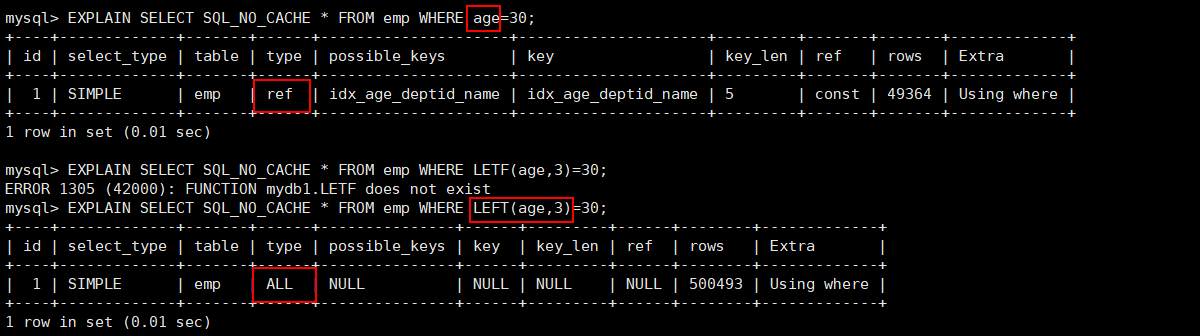
## 3. 不要在索引列上做任何计算

不在索引列上做任何操作（计算、函数、(自动or手动)类型转换），会导致索引失效而转向全表扫描。

### 3.1 在查询列上使用了函数

EXPLAIN SELECT SQL\_NO\_CACHE \* FROM emp WHERE age=30;

EXPLAIN SELECT SQL\_NO\_CACHE \* FROM emp WHERE LEFT(age,3)=30;

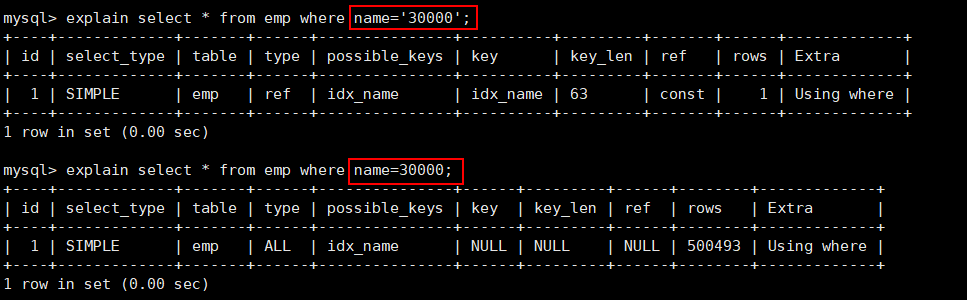


结论：等号左边无计算！

### 3.2 在查询列上做了转换

|  |
| --- |
| create index idx\_name on emp(name); |
| explain select sql\_no\_cache \* from emp where name='30000'; |
| explain select sql\_no\_cache \* from emp where name=30000; |

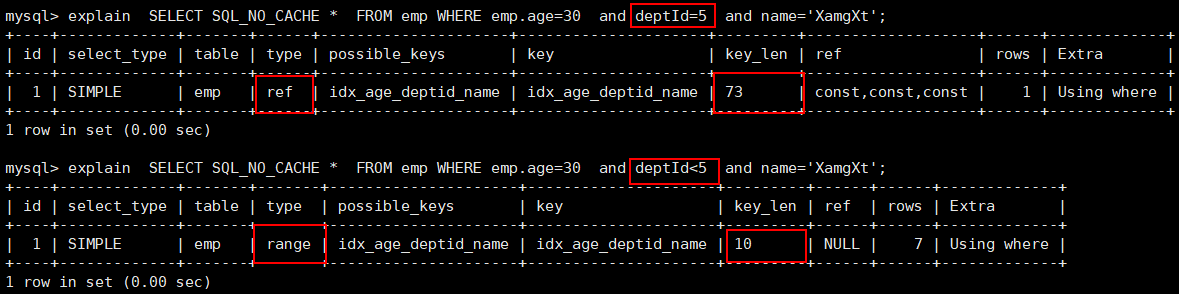
字符串不加单引号，则会在name列上做一次转换！



结论：等号右边无转换！

## 4. 索引列上不能有范围查询

|  |
| --- |
| explain SELECT SQL\_NO\_CACHE \* FROM emp WHERE emp.age=30 and deptid=5 AND emp.name = 'abcd'; |
| explain SELECT SQL\_NO\_CACHE \* FROM emp WHERE emp.age=30 and deptid<5 AND emp.name = 'abcd'; |

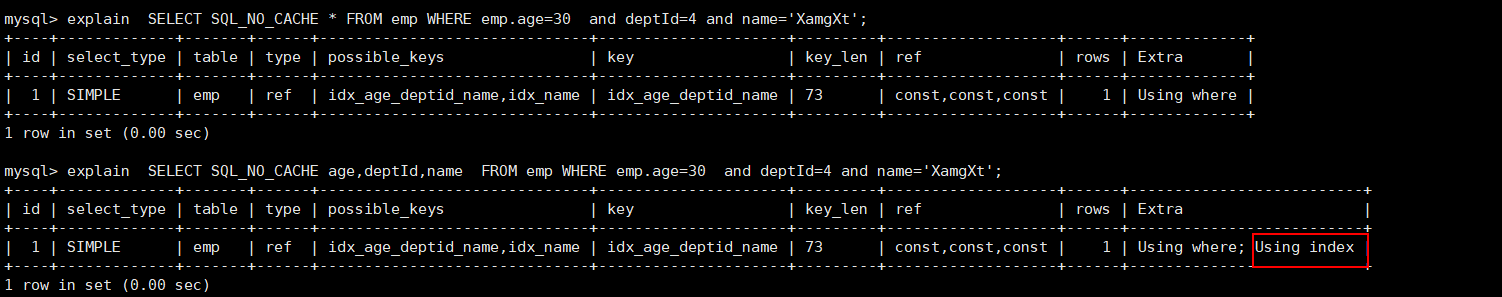


建议：将可能做范围查询的字段的索引顺序放在最后

## 5. 尽量使用覆盖索引

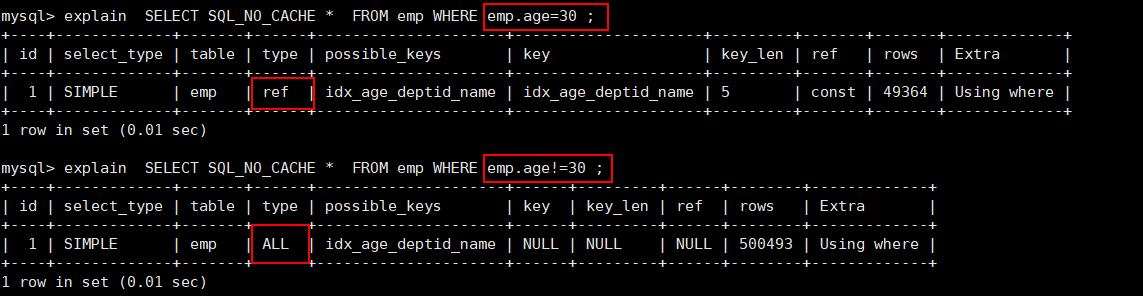
即查询列和索引列一直，不要写select \*!

|  |
| --- |
| explain SELECT SQL\_NO\_CACHE \* FROM emp WHERE emp.age=30 and deptId=4 and name='XamgXt'; |
| explain SELECT SQL\_NO\_CACHE age,deptId,name FROM emp WHERE emp.age=30 and deptId=4 and name='XamgXt'; |

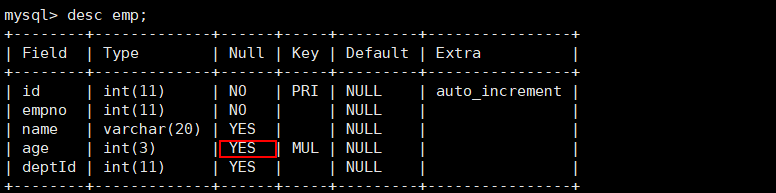


## 6. 使用不等于(!= 或者<>)的时候

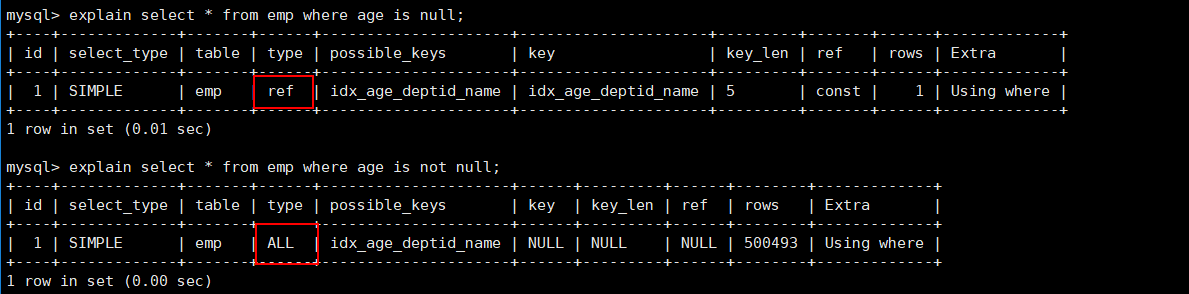
mysql 在使用不等于(!= 或者<>)时，有时会无法使用索引会导致全表扫描。



## 7. 字段的is not null 和 is null

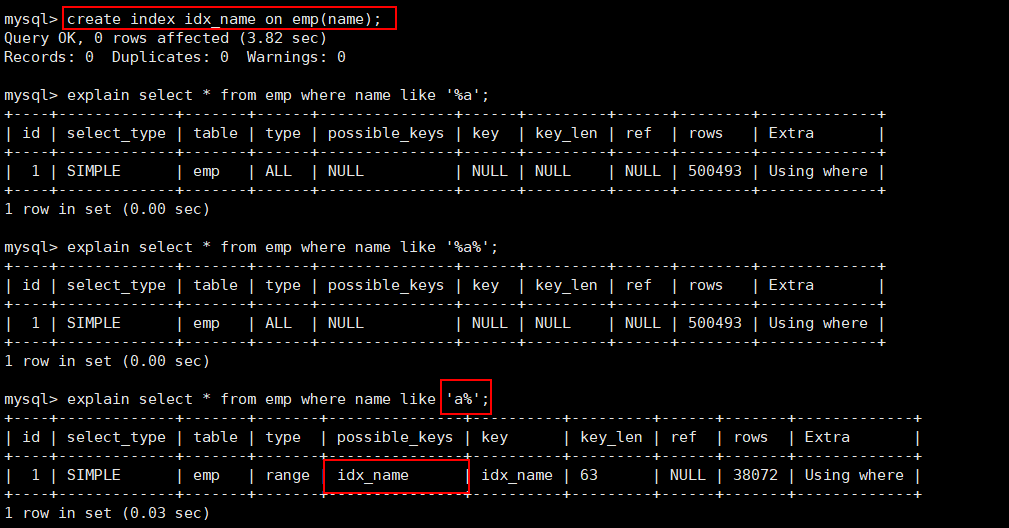


当字段允许为Null的条件下：



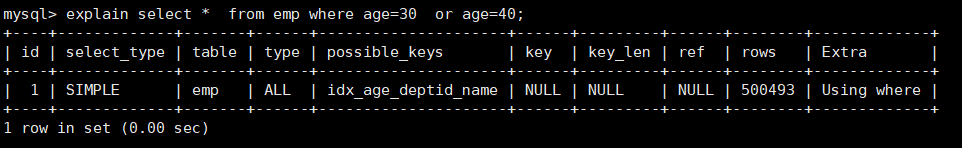
is not null用不到索引，is null可以用到索引。

## 8. like的前后模糊匹配

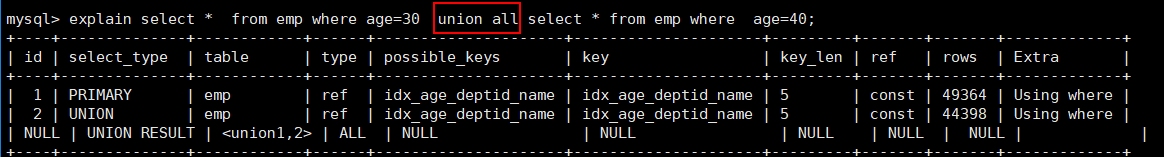


前缀不能出现模糊匹配！

## 9. 减少使用or



使用union all或者union来替代：



## 10. 练习

假设index(a,b,c)；

|  |  |
| --- | --- |
| **Where语句** | **索引是否被使用** |
| where a = 3 | Y,使用到a |
| where a = 3 and b = 5 | Y,使用到a，b |
| where a = 3 and b = 5 and c = 4 | Y,使用到a,b,c |
| where b = 3 或者 where b = 3 and c = 4 或者 where c = 4 | N |
| where a = 3 and c = 5 | 使用到a， 但是c不可以，b中间断了 |
| where a = 3 and b > 4 and c = 5 | 使用到a和b， c不能用在范围之后，b断了 |
| where a is null and b is not null | is null 支持索引 但是is not null 不支持,所以 a 可以使用索引,但是 b不可以使用 |
| where a <> 3 | 不能使用索引 |
| where abs(a) =3 | 不能使用 索引 |
| where a = 3 and b like 'kk%' and c = 4 | Y,使用到a,b,c |
| where a = 3 and b like '%kk' and c = 4 | Y,只用到a |
| where a = 3 and b like '%kk%' and c = 4 | Y,只用到a |
| where a = 3 and b like 'k%kk%' and c = 4 | Y,使用到a,b,c |

## 11. 口诀

全值匹配我最爱，最左前缀要遵守；

带头大哥不能死，中间兄弟不能断；

索引列上少计算，范围之后全失效；

LIKE百分写最右，覆盖索引不写\*；

不等空值还有OR，索引影响要注意；

VAR引号不可丢，SQL优化有诀窍。