

设备的速度

CPU——cache——内存——ssd——网络——硬盘

CPU、内存的操作：缓存数据访问、比较、排序、事务检测、sql语句解析、函数、逻 辑运算。

网络：结果数据传输，sql请求，远程数据访问

硬盘：数据访问，数据写入，日志记录，大数据排序，大表连接

优化原则：

1. 减少数据访问（减少磁盘访问）
2. 返回更少数据（减少网络传输或者磁盘访问）
3. 减少交互次数（减少网络传输次数）
4. 减少CPU开销
5. 利用更多资源

索引

使用索引，select的速度变快，但是update、insert的速度回变慢，因为会构建索引。

建索引的条件：

主键、外键，作为查询条件执行频率高，且用得上索引

复合索引——多个字段组成的索引，索引中字段顺序有讲究

比如：索引——姓名、年龄、性别

通过 [姓名]、[年龄、姓名]、[姓名、年龄、性别]查询，会用的这个复合索引。

不能使用索引的情况：

1、不等号 <> !=

2、使用运算或者数据库函数

3、前导模糊查询的like——like ‘%ssss’类似这种

4、B-TREE索引里不保存字段为NULL值记录，因此IS NULL不能使用索引

5、如果两边数据类型不同时会对字段值隐式转换，相当于加了一层函数处理，所以不能使用索引数据类型转换不能使用索引

6、索引查询应该是已知值，不能是未知参数 a.INDEX\_COLUMN=a.COLUMN\_1

变量绑定（好）

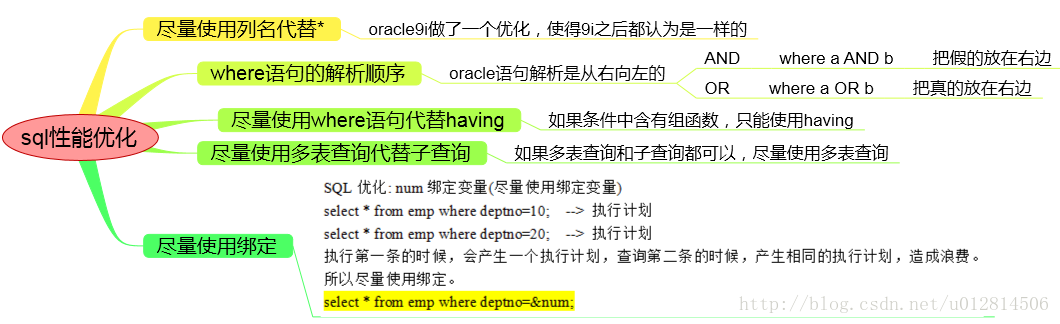
select \* from employee where id=？

PreparedStatement.setInt(1, 123456)

好处：1、防止sql注入。2、提高sql可读性。3、提高sql解析性能

不使用变量绑定

select \* from employee where id=123456



****不能使用索引定位的大量In List，如下所示：****

a in (:1,:2,:3,…,:n)   ----n>20

Sql优化

1、速度太慢，建索引

2、使用参数化sql（可避免sql注入）。数据库会执行预编译，查询优化。后续查询可直接使用预编译结果

3、调整where条件的位置。将能过滤掉大部分数据的条件放在前面。

4、能放在一起查询的，就不要分开查询。这样可以减少网络连接，权限校验，sql优化，发送执行结果。

5、查询条件尽量写在where子句中，having子句主要放聚合函数。

6、使用表的别名可以减少解析时间。

7、in 和 exist都可用的时候，用exist较好。

8、不要在索引的字段上进行计算，这样会放弃索引

效率高 salary > 2500/12

效率低 salary \* 12 > 2500

9、用union all 替换 union （没有重复记录时，union会进行排序去重）。

1、尽量避免全表扫描，首先在where 及 order by 涉及的列上建立索引

2、尽量不要用 is null 判断

3、尽量不要使用不等操作符 != <> ，这样会导致放弃索引

4、or 连接的条件如果有一个没有索引，将放弃索引。

select id from t where num=10 or name=’admin’ 改为

select id from t where num=10

union all

select id from t where name=’admin’

4、能用between and， exist，not exist 就不要用 in ，not in

5、like ‘%dfdfd’ 前置like条件不会使用索引

6、条件中使用参数，会导致全表扫描

7、在where子句中不要使用字段表达式

效率高 salary > 2500/12

效率低 salary \* 12 > 2500

8、where 子句中不要在=左边使用函数、算数运算、表达式运算

9、update语句 如果只修改部分字段，就不要修改全部字段。

10、索引不要太多，一个表的索引不要超过6个。

11、不要更新clustered索引，因为clustered索引数据列顺序就是记录的物理存储顺序，值修改了会导致数据的调整。

12、字段能用数字类型，就不要使用字符类型。

13、varchar 要好一些

14、尽量只返回需要的字段，不要返回所有的字段

15、select into 替代 create table，可以避免大量log，可以提高速度。