**数据库学习**

1.innodb的特点

行锁设计

支持MVCC（多版本并发控制）

支持外键

提供一致性非锁定读

最有效地利用以及使用内存和CPU

2.sql优化

SQL优化方向：执行计划、索引、SQL改写。

架构优化方向：高可用架构、高性能架构、分库分表。

执行计划

当我们的系统上线后数据库的记录不断增加，之前写的一些SQL语句或者一些ORM操作效率变得非常低。我们不得不考虑SQL优化，SQL优化大概是这样一个流程：1.定位执行效率低的SQL语句（定位）,2.分析为什么这段SQL执行的效率比较低（分析），3.最后根据第二步分析的结构采取优化措施（解决）。而EXPLAIN命令的作用就是帮助我们分析SQL的执行情况，属于第二步。说的规范一点就是:EXPLAIN命令是查看查询优化器如何决定执行查询的主要的方法。学会解释EXPLAIN将帮助我们了解SQL优化器是如何工作的。执行计划可以告诉我们SQL如何使用索引，连接查询的执行顺序，查询的数据行数。

要使用EXPLAIN（执行计划）,只需要在查询的SELECT关键字之前增加EXPLAIN这个词。

1.在表中建立索引，优先考虑where.group by使用到的字段。

2.查询条件中，一定不要使用select \*，因为会返回过多无用的字段会降低查询效率。应该使用具体的字段代替\*，只返回使用到的字段。

3.不要在where条件中使用左右两边都是%的like模糊查询，如：

SELECT \* FROM t\_order WHERE customer LIKE '%zhang%'

这样会导致数据库引擎放弃索引进行全表扫描。

优化：尽量在字段后面使用模糊查询。如下：

SELECT \* FROM t\_order WHERE customer LIKE 'zhang%'

4.尽量不要使用in 和not in，会造成全表扫描。如下：

SELECT \* FROM t\_order WHERE id IN (2,3)

SELECT \* FROM t\_order1 WHERE customer IN (SELECT customer FROM t\_order2)

优化：

**对于连续的数值，能用 between 就不要用 in** ，如下：SELECT \* FROM t\_order WHERE id BETWEEN 2 AND 3

对于子查询，可以用exists代替。如下：SELECT \* FROM t\_order1 WHERE EXISTS (SELECT \* FROM t\_order2 WHERE t1.customer = t2.customer)

**5.尽量不要使用or，会造成全表扫描。**如下：

SELECT \* FROM t\_order WHERE id = 1 OR id = 3

优化：可以用union代替or。如下：

SELECT \* FROM t\_order WHERE id = 1

UNION

SELECT \* FROM t\_order WHERE id = 3

**6.尽量不要在 where 子句中对字段进行表达式操作**，这样也会造成全表扫描。如：

select id FROM t\_order where num/2=100

应改为:

select id FROM t\_order where num=100\*2

**7.where条件里尽量不要进行null值的判断**，null的判断也会造成全表扫描。如下：

SELECT \* FROM t\_order WHERE score IS NULL

优化：

给字段添加默认值，对默认值进行判断。如：

SELECT \* FROM t\_order WHERE score = 0

**8.尽量不要在where条件中等号的左侧进行表达式.函数操作**，会导致全表扫描。如下：

SELECT \* FROM t\_order2 WHERE score/10 = 10

SELECT \* FROM t\_order2 WHERE SUBSTR(customer,1,5) = 'zhang'

优化：

将表达式.函数操作移动到等号右侧。如下：

SELECT \* FROM t\_order2 WHERE score = 10\*10

SELECT \* FROM t\_order2 WHERE customer LIKE 'zhang%'

**9.尽量不要使用where 1=1的条件**

有时候，在开发过程中，为了方便拼装查询条件，我们会加上该条件，这样，会造成进行全表扫描。如下：

SELECT \* FROM t\_order WHERE 1=1

优化：

如果用代码拼装sql，则由代码进行判断，没where加where，有where加and

如果用mybatis，请用mybatis的where语法。

**10.程序要尽量避免大事务操作，提高系统并发能力。**

**11.一个表的索引数最好不要超过6个**，如果索引太多的话，就需要考虑一下那些不常使用到的列上建的索引是否有必要。