利用MySQL 5.7新特性做SQL优化

吴炳锡 - 知数堂培训联合创始人

2017.3

关于我

- 知数堂培训联合创始人， MySQL中国用户组发起人之一及主席团成员
- QQ/微信：82565387
- 10年+MySQL从业经验，专注MySQL架构，多IDC设计，高可用，高性能相关技术分析及实践，目前从事MySQL培训及企业MySQL相关服务。

agenda

- 怎么开展SQL优化
- 传统模型下做SQL优化需要做的工作
- 在MySQL 5.7下如何开展SQL优化
  - 利用Sys库发现需要优化的SQL
  - 利用Query Rewrite Plugin进行优化测试
  - 利用HINT改写优化及SQL执行超时保护
- 总结

- 事实上，MySQL 5.7已经不新了，2016年已发布8.0版本
- 2013.4.23发布MySQL 5.7.1，已经将近4年了
- 每年至少3个版本，2016年突然发飙，发布了7个版本
- 官方号称比5.6快3倍以上
- MySQL 5.7 让MySQL DBA有了更多的调优机会
- 截止目前，最新版本5.7.17，仍在持续完善中.....

如何开展SQL优化

占用时间最长的SQL

执长时间最多的SQL

业务高峰期的SQL

如何开展SQL优化

- **SQL优化的目标：**
  - 让系统跑的平稳
  - 有更高的业务处理能力
  - 更短的业务响应能力

传统模型下做SQL优化需要做的工作

存在问题

1. 开发需要提供SQL List列表
2. 对DBA对SQL有没有Where条件，索引使用合理不合理审查
3. 慢日志收集分析，一个月后，看惯了那个老脸也没新鲜了

传统模型下做SQL优化需要做的工作

获取SQL List优化

- 使用general log
- 把long_query_time=0
- 利用tcpdump类工具进行抓包

- 等等方法

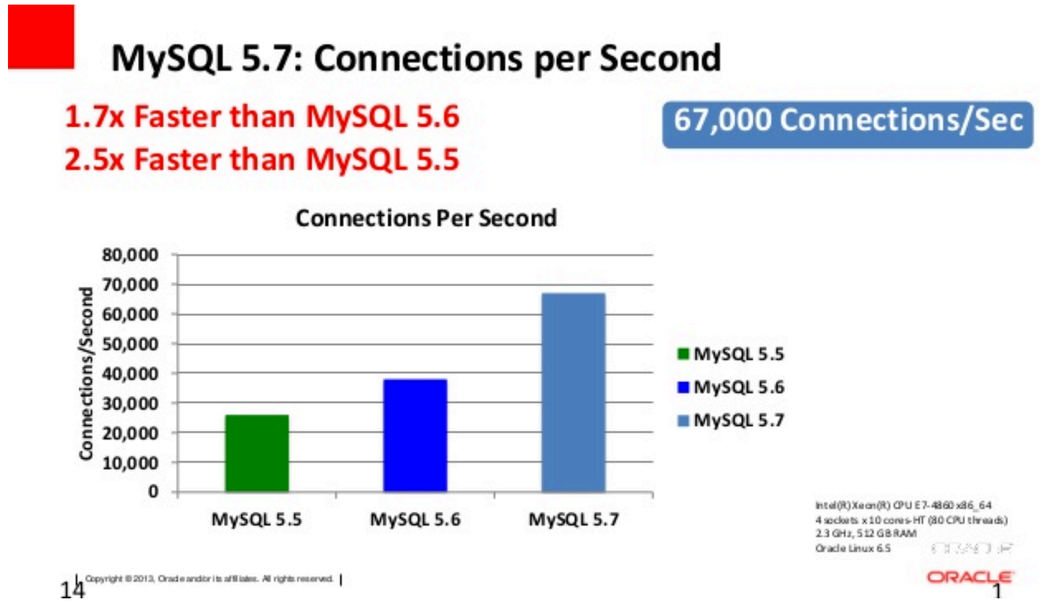
需要一个前题：

能登录上这个服务器才行！

RDS，云上的DB怎么办？

MySQL 5.7下开展SQL优化

一个真正的DB，有DBA出手的机会



MySQL 5.7下开展SQL优化

- 针对SQL优化上改善
  - 更严谨的SQL束约
  - explain for connetion 1024
  - josn数据类型
  - 多源复制，支持多个Triger
  - 优化器HINT提示
  - ServerSide执行超时

MySQL 5.7

更完善

更高效SQL支持

等待着你来发现

Sys Schema

- 从SYS Schema可以快速获取很多
  - 锁等待
  - 内存分配
  - SQL统计

参考： MySQL 5.7系列之sys schema

```
+--------------------+----------+
| statement          | total    |
+--------------------+----------+
| Execute            | 16334741 |
| commit             |   642667 |
| rollback           |     2077 |
| Prepare            |     1050 |
| select             |       18 |
| Close stmt         |      350 |
| stmt               |       12 |
| set                |        4 |
| call_procedure     |        5 |
| show_variables     |        1 |
| jump_if_not        |     4266 |
| show_tables        |        5 |
| freturn            |     1040 |
| set_option         |       30 |
| show_table_status  |        1 |
| Init DB            |        3 |
| Quit               |       11 |
| show_processlist   |        1 |
+--------------------+----------+
18 rows in set (0.00 sec)
```

sys.user_summary_by_statement_type

利用performance_schema

- sys.statement_analysis
  - CALL sys.ps_setup_enable_instrument('statement');

```
root@localhost:mysql3306.sock [sys]>select * from statement_analysis limit 10\G;
*************************** 1. row ***************************
            query: SELECT `c` FROM `sbtest1` WHERE `id` = ?
               db: bench
        full_scan:
       exec_count: 23785
        err_count: 0
       warn_count: 0
    total_latency: 8.15 s
      max_latency: 43.95 ms
      avg_latency: 342.57 us
     lock_latency: 602.96 ms
        rows_sent: 23785
    rows_sent_avg: 1
    rows_examined: 23785
rows_examined_avg: 1
    rows_affected: 0
rows_affected_avg: 0
       tmp_tables: 0
  tmp_disk_tables: 0
      rows_sorted: 0
sort_merge_passes: 0
           digest: 80295d1d2720d4515b05d648e8caa82f
       first_seen: 2017-02-21 17:21:54
        last_seen: 2017-02-21 17:25:53
```

利用performance_schema获取

```
SELECT CONCAT_WS( '','# Time: ', date_format(CURDATE(),'%y%m%d'),'
',TIME_FORMAT(NOW(6),'%H:%i:%s.%f'),'\n' ,'# User@Host:
',t.PROCESSLIST_USER,'[',t.PROCESSLIST_USER,'] @ ',PROCESSLIST_HOST,' []  Id:
',t.PROCESSLIST_ID,'\n' ,'# Schema: ',CURRENT_SCHEMA,'  Last_errno: ',MYSQL_ERRNO,'
','\n' ,'# Query_time: ',ROUND(s.TIMER_WAIT / 1000000000000, 6),' Lock_time:
',ROUND(s.LOCK_TIME / 1000000000000, 6),'  Rows_sent: ',ROWS_SENT,'
Rows_examined: ',ROWS_EXAMINED,'  Rows_affected: ',ROWS_AFFECTED,'\n' ,'#
Tmp_tables: ',CREATED_TMP_TABLES,'  Tmp_disk_tables: ',CREATED_TMP_DISK_TABLES,'
','\n' ,'# Full_scan: ',IF(SELECT_SCAN=0,'No','Yes'),'  Full_join:
',IF(SELECT_FULL_JOIN=0,'No','Yes'),'  Tmp_table:
',IF(CREATED_TMP_TABLES=0,'No','Yes'),'  Tmp_table_on_disk:
',IF(CREATED_TMP_DISK_TABLES=0,'No','Yes'),'\n' , t.PROCESSLIST_INFO,';') FROM
performance_schema.events_statements_history s JOIN performance_schema.threads t
using(thread_id) WHERE t.TYPE = 'FOREGROUND' AND t.PROCESSLIST_INFO IS NOT
NULL AND t.PROCESSLIST_ID != connection_id() ORDER BY t.PROCESSLIST_TIME desc;
```

利用performance_schema获取

```
# User@Host: wubx[wubx] @ 192.168.11.100 []  Id: 24
# Schema: bench  Last_errno: 0
# Query_time: 0.000130 Lock_time: 0.000021  Rows_sent: 1  Rows_examined: 1  Rows_affected: 0
# Tmp_tables: 0  Tmp_disk_tables: 0
# Full_scan: No  Full_join: No  Tmp_table: No  Tmp_table_on_disk: No
SELECT c FROM sbtest5 WHERE id=53561;                                        |
| # Time: 170221 17:44:50.414066
# User@Host: wubx[wubx] @ 192.168.11.100 []  Id: 24
# Schema: bench  Last_errno: 0
# Query_time: 0.000114 Lock_time: 0.000019  Rows_sent: 1  Rows_examined: 1  Rows_affected: 0
# Tmp_tables: 0  Tmp_disk_tables: 0
# Full_scan: No  Full_join: No  Tmp_table: No  Tmp_table_on_disk: No
SELECT c FROM sbtest5 WHERE id=53561;                                        |
+-------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------
-------------------------------------------------------------+
40 rows in set (0.02 sec)
```

MySQL 5.7下开展SQL优化

- 数据就在那里，也没额外开销 MySQL 5.6.6 后默认打开
- 生产环境中也可以放心使用
- 对于Sys库的一个统计计算就是Triger有视图的使用，开销很小，如果不需要可以禁掉。

Query Rewirte Plugin

从容面对

Query Rewrite Plugin介绍

- 在MySQL 5.7.6以后官方引入一个Query Rewrite plugin，可以在Server端收到SQL后，进行改写并执行。
  - 只针对标准的SELECT语句工作，不能对视图定义及存储过程中SELECT语句改写
  - 改写规则记录在内存中，实际对应到：query_rewrite库下的rewrite_rules这个表
  - 利用query_rewrite下的存储过程：load_rewrite_rules() / flush_rewrite_rules() 及DML语句来加载更改规则

Query Rewrite Plugin使用

- 启用Query Rewrite Plugin

```
mysql -u root -p < $basedir/share/install_rewriter.sql
SHOW GLOBAL VARIABLES LIKE 'rewriter_enabled';
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| rewriter_enabled | ON    |
+-----------------+-------+
```

- 在配置文件中添加

```
[mysqld]
rewriter_enabled=ON
```

Query Rewrite Plugin使用

- 利用子查询新特性优化Join

```
root@localhost [employees]>root@localhost [employees]>SELECT count(distinct emp_no) FROM employees.employees INNER JOIN employees.sa
DATEDIFF(to_date, from_date) < 2000;
+----------------------+
| count(distinct emp_no) |
+----------------------+
|                 300024 |
+----------------------+
1 row in set (3.09 sec)
```

- 优化后样子

```
root@localhost [employees]>SELECT count(emp_no) FROM employees.employees WHERE emp_no IN ( SELECT emp_no FROM employees.salaries WHE
RE DATEDIFF(to_date, from_date)<2000);
+--------------+
| count(emp_no) |
+--------------+
|        300024 |
+--------------+
1 row in set (1.37 sec)
```

Query Rewrite Plugin使用

- 利用Query Rewirte固化

```
INSERT INTO query_rewrite.rewrite_rules
 (
pattern,
Replacement
)
VALUES
(
'SELECT count(distinct emp_no) FROM employees.employees INNER JOIN
employees.salaries USING(emp_no) WHERE DATEDIFF(to_date, from_date) < ?',
 'SELECT count(emp_no) FROM employees.employees WHERE emp_no IN ( SELECT emp_no
FROM employees.salaries WHERE DATEDIFF(to_date, from_date) < ?)’
);
CALL query_rewrite.flush_rewrite_rules();
```

ServerSide SQL执行超时

避免对DB过载

max_execution_time

- MySQL 5.7.8后引入的参数
- 默认为0，不限制。如果在全局设置了这个参数，相当于引入全局的ServerSide超时
- 同时可以作用于Session中的/*+ max_execution_time(N) */ #N单位毫秒
- **只对Select语句起作用，但对存储过程中的Select语句不起作用**

max_execution_time

例子：

```
root@localhost [employees]>root@localhost [employees]>SELECT count(distinct emp_no) FROM employees.employees INNER JOIN employees.sa
DATEDIFF(to_date, from_date) < 2000;
+----------------------+
| count(distinct emp_no) |
+----------------------+
|                 300024 |
+----------------------+
1 row in set (3.09 sec)
```

业务高峰期，某些SQL处理上，我希望给降级或是引入过载保护，让DB能撑过业务高峰，对于这个问题我们怎么处理?

pt-kill?

max_execution_time

- 结合Query Rewrite Plugin 和HINT改写

```
INSERT INTO query_rewrite.rewrite_rules
 (
pattern,
Replacement
)
VALUES
(
'SELECT count(distinct emp_no) FROM employees.employees INNER JOIN
employees.salaries USING(emp_no) WHERE DATEDIFF(to_date, from_date) < ?',
‘SELECT /*+ MAX_EXECUTION_TIME(1000)*/ count(distinct emp_no) FROM
employees.employees INNER JOIN employees.salaries USING(emp_no) WHERE
DATEDIFF(to_date, from_date) < ?',
);

CALL query_rewrite.flush_rewrite_rules();
```

max_execution_time

执行效果

```
root@localhost [employees]>SELECT count(distinct emp_no) FROM employees.employees INNER JOIN employees.salaries USING(emp_no) WHERE
DATEDIFF(to_date, from_date) < 2000;
ERROR 3024 (HY000): Query execution was interrupted, maximum statement execution time exceeded
root@localhost [employees]>SELECT count(distinct emp_no) FROM employees.employees INNER JOIN employees.salaries USING(emp_no) WHERE
DATEDIFF(to_date, from_date) < 2000;
ERROR 3024 (HY000): Query execution was interrupted, maximum statement execution time exceeded
```

HINT增强

SELECT /*+ ... */ ...

INSERT /*+ ... */ ...

REPLACE /*+ ... */ ...

UPDATE /*+ ... */ ...

DELETE /*+ ... */ ...

SELECT /*+ NO_RANGE_OPTIMIZATION(t3 PRIMARY, f2_idx) */ * FROM t3...;

SELECT /*+ BKA(t1) NO_BKA(t2) */ * FROM t1 INNER JOIN t2 WHERE ...;

SELECT /*+ NO_ICP(t1, t2) */ * FROM t1 INNER JOIN t2 WHERE ...;

SELECT /*+ SEMIJOIN(FIRSTMATCH, LOOSESCAN) */ * FROM t1 ...;

SELECT /*+ NO_ICP(t1) */ * FROM t1 WHERE ...;

案例 排序过多

```
SELECT sl.sku,sl.positionCode,ps.positionLev,
  SUM(CASE WHEN sk.skipQty > 0 THEN 0 ELSE CASE WHEN ps.positionLev = 3
           THEN (IFNULL(sl.quantity,0)-IFNULL(ll.lockedQty,0)+IFNULL(fl.frozenQty,0)) ELSE 0 END END) jhqty,
  SUM(CASE WHEN sk.skipQty > 0 THEN 0 ELSE CASE WHEN ps.positionLev = 4
           THEN (IFNULL(sl.quantity,0)-IFNULL(ll.lockedQty,0)+IFNULL(fl.frozenQty,0)) ELSE 0 END END) bhqty,
  SUM(CASE WHEN sk.skipQty > 0 THEN 0 ELSE CASE WHEN ps.positionLev = 5
           THEN (IFNULL(sl.quantity,0)-IFNULL(ll.lockedQty,0)+IFNULL(fl.frozenQty,0)) ELSE 0 END END) byqty,
  SUM(CASE WHEN sk.skipQty > 0 THEN 0 ELSE CASE WHEN ps.positionLev = 6
           THEN (IFNULL(sl.quantity,0)-IFNULL(ll.lockedQty,0)+IFNULL(fl.frozenQty,0)) ELSE 0 END END) ryqty
  FROM wms_si_stock sl
  JOIN wms_bd_product_information p ON p.sku=sl.sku
  JOIN wms_bd_positions ps ON ps.warehouseIdentity=sl.warehouseIdentity AND ps.positionCode=sl.positionCode
  LEFT JOIN (SELECT l.warehouseIdentity,l.sku,l.positionCode,l.isbox,l.boxNo,IFNULL(SUM(quantity),0) lockedQty
  FROM wms_si_locked l GROUP BY l.warehouseIdentity,l.sku,l.positionCode,l.isbox,l.boxNo) ll
  ON sl.warehouseIdentity=ll.warehouseIdentity AND sl.sku=ll.sku AND sl.positionCode=ll.positionCode AND
  sl.isbox=ll.isbox AND sl.boxNo=ll.boxNo
  LEFT JOIN (SELECT warehouseIdentity,sku,positionCode ,SUM(quantity) frozenQty FROM wms_so_frozen WHERE STATUS=1
  AND 0>quantity AND checkPlanId !='' AND checkPlanId IS NOT NULL GROUP BY warehouseIdentity,sku,positionCode) fl
  ON sl.warehouseIdentity=fl.warehouseIdentity AND sl.sku=fl.sku AND sl.positionCode=fl.positionCode
  LEFT JOIN (SELECT warehouseIdentity,sku,positionCode,boxNo,SUM(quantity) skipQty FROM wms_si_skiped WHERE STATUS=0
  and warehouseIdentity = '1'
  GROUP BY warehouseIdentity,sku,positionCode,boxNo) sk
        ON sk.warehouseIdentity=sl.warehouseIdentity
        AND sk.sku=sl.sku
        AND sk.positionCode=sl.positionCode AND sk.boxNo=sl.boxNo
  where 1=1
  and sl.warehouseIdentity = '1'
  AND ps.positionLev != 7
  GROUP BY sl.sku,sl.positionCode,ps.positionLev,sl.warehouseIdentity
  \G;
```

引自： 知数堂 郑松华

案例 排序过多

```
*************************** 1. row ***************************
           id: 1
  select_type: PRIMARY
        table: ps
         type: ALL
possible_keys: idx_pos_pos,idx_pos_lev,idx_pos_posLev
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 108416
        Extra: Using where; Using temporary; Using filesort
*************************** 2. row ***************************
           id: 1
  select_type: PRIMARY
        table: sl
         type: ref
possible_keys: idx_stock_all,idx_st_skuP,idx_st_posCd
          key: idx_st_posCd
      key_len: 152
          ref: wms.ps.positionCode
         rows: 1
        Extra: Using where
*************************** 3. row ***************************
           id: 1
  select_type: PRIMARY
        table: p
         type: ref
possible_keys: idx_proif_sku
          key: idx_proif_sku
      key_len: 153
          ref: wms.sl.sku
         rows: 1
        Extra: Using index
*************************** 4. row ***************************
           id: 1
  select_type: PRIMARY
        table: <derived2>
         type: ref
possible_keys: <auto_key0>
          key: <auto_key0>
      key_len: 368
          ref: wms.ps.warehouseIdentity,wms.sl.sku,wms.ps.positionCode,w
         rows: 10
        Extra: NULL
      key_len: 305
          ref: wms.ps.warehouseIdentity,wms.sl.sku,wms.ps.positionCode
         rows: 10
        Extra: NULL
*************************** 6. row ***************************
           id: 1
  select_type: PRIMARY
        table: <derived4>
         type: ref
possible_keys: <auto_key0>
          key: <auto_key0>
      key_len: 367
          ref: wms.ps.warehouseIdentity,wms.sl.sku,wms.ps.positionCode,wms.sl.boxNo
         rows: 2
        Extra: NULL
*************************** 7. row ***************************
           id: 4
  select_type: DERIVED
        table: wms_si_skiped
         type: ref
possible_keys: idx_skip_sku,idx_skip_stat
          key: idx_skip_stat
      key_len: 2
          ref: const
         rows: 22
        Extra: Using where; Using temporary; Using filesort
*************************** 8. row ***************************
           id: 3
  select_type: DERIVED
        table: wms_so_frozen
         type: ALL
possible_keys: idx_fz_skuP
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 37532
        Extra: Using where; Using temporary; Using filesort
*************************** 9. row ***************************
           id: 2
  select_type: DERIVED
        table: l
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 15957
```

案例 排序过多

```
SELECT s.sku,s.positionCode,s.positionLev,
SUM(CASE WHEN s.skipQty > 0 THEN 0 ELSE CASE WHEN s.positionLev = 3
        THEN (IFNULL(s.quantity,0)-IFNULL(s.lockedQty,0)+IFNULL(s.frozenQty,0)) ELSE 0 END END) jhqty,
SUM(CASE WHEN s.skipQty > 0 THEN 0 ELSE CASE WHEN s.positionLev = 4
        THEN (IFNULL(s.quantity,0)-IFNULL(s.lockedQty,0)+IFNULL(s.frozenQty,0)) ELSE 0 END END) bhqty,
SUM(CASE WHEN s.skipQty > 0 THEN 0 ELSE CASE WHEN s.positionLev = 5
        THEN (IFNULL(s.quantity,0)-IFNULL(s.lockedQty,0)+IFNULL(s.frozenQty,0)) ELSE 0 END END) byqty,
SUM(CASE WHEN s.skipQty > 0 THEN 0 ELSE CASE WHEN s.positionLev = 6
        THEN (IFNULL(s.quantity,0)-IFNULL(s.lockedQty,0)+IFNULL(s.frozenQty,0)) ELSE 0 END END) ryqty
from (
SELECT sl.sku,sl.positionCode,ps.positionLev,sl.warehouseIdentity,sl.quantity
(SELECT IFNULL(SUM(quantity),0)
FROM wms_si_locked ll
ON sl.sku=ll.sku AND sl.positionCode=ll.positionCode and  sl.warehouseIdentity=ll.warehouseIdentity   AND
sl.isbox=ll.isbox AND sl.boxNo=ll.boxNo) lockedQty,
(SELECT  SUM(f1.quantity) frozenQty
FROM wms_so_frozen fl WHERE fl.STATUS=1
AND 0>fl.quantity AND fl.checkPlanId !='' AND fl.checkPlanId IS NOT NULL
and sl.sku=fl.sku AND sl.positionCode=fl.positionCode AND sl.warehouseIdentity=fl.warehouseIdentity )frozenQty,
(SELECT SUM(sk.quantity)  FROM wms_si_skiped sk
 WHERE sk.STATUS=0
and sk.warehouseIdentity = '1'      and sk.warehouseIdentity=sl.warehouseIdentity AND sk.sku=sl.sku AND
sk.positionCode=sl.positionCode AND sk.boxNo=sl.boxNo) skipQty
FROM wms_si_stock sl
JOIN wms_bd_product_information p ON p.sku=sl.sku
JOIN wms_bd_positions ps ON ps.warehouseIdentity=sl.warehouseIdentity AND ps.positionCode=sl.positionCode
where 1=1
and sl.warehouseIdentity = '1'
AND ps.positionLev != 7
) s
GROUP BY s.sku,s.positionCode,s.positionLev,s.warehouseIdentity
```

引自： 知数堂 郑松华

案例 排序过多

修改后的执行计划：

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	<derived2>	ALL	NULL	NULL	NULL	NULL	54224	Using temporary; Using filesort
2	DERIVED	ps	ALL	idx_pos_pos,idx_pos_lev,idx_pos_posLev	NULL	NULL	NULL	108416	Using where
2	DERIVED	sl	ref	idx_stock_all,idx_st_skuP,idx_st_posCd	idx_st_posCd	152	wms.ps.positionCode	1	Using where
2	DERIVED	p	ref	idx_proif_sku	idx_proif_sku	153	wms.sl.sku	1	Using index
5	DEPENDENT SUBQUERY	sk	ref	idx_skip_box,idx_skip_sku,idx_skip_stat	idx_skip_box	214	wms.sl.boxNo,wms.sl.positionCode	1	Using where
4	DEPENDENT SUBQUERY	fl	ref	idx_fz_skuP,idx_fz_posCd	idx_fz_skuP	304	wms.sl.sku,wms.sl.positionCode	5	Using where
3	DEPENDENT SUBQUERY	ll	ref	idx_lk_skuP,idx_lk_posCd,idx_lock_box	idx_lk_skuP	304	wms.sl.sku,wms.sl.positionCode	1	Using where

之前的SQL 运行了 9000多秒 修改之后运行了7s

SQL优化三板斧

- 确认join字段类型一致和字符一致
- 去除不必要的排序及临时表
- 前置运算移到Where条件后增加过滤

总结

1.需要有全局观，知道自已的业务使用了MySQL的功能

2.需要要去了解一下MySQL有那些特性及功能能被所用

3.对于新特性

- 如果是一个重大的更新，如果慎重
- 对于Tips的功能，可以轻松使用

总结

- SQL优化的几个层面
  - 第一层面
    - 通过基础的书集学习SQL，正确编写SQL，通过索引优化（看书可以搞定，理解索引对优化的帮助）
  - 第二层面
    - SQL改写能力，有集合的意识，借助于优化器输出改写SQL，能搞明白Join的顺序， 搞明白 Access和filter的区别，通过改写得到正确的结果（通过练习就可达到）
  - 第三层面
    - 有一定的全局意识，并且针对看到的SQL，不使用优化器输出，也可以进行优化（已经对group by , join 及不同版本的优化器特性熟记于心）， SQL优化已经成为工作中。（需要有系统的方法论）

谢谢，希望有所帮助