生产ecief top1 SQL优化建议

1.

```
select
    list_id, RISK_CATEGORY, PROPOSAL_ID, CUSTOMER_ID, RISK_AMOUNT, PARTITION_INDI, INSERT_TIME,
    UPDATE_TIME, INSERT_BY, UPDATE_BY, STATUS, ITEM_CATEGORY, EXPIRY_DATE
    from t_customer_risk_sum_old where 1=1
      AND  PROPOSAL_ID = '8017081702000408'\G

mysql> select count(*) from t_customer_risk_sum_old;
+----------+
| count(*) |
+----------+
|  1865297 |
+----------+
1 row in set (0.53 sec)

mysql>
```

该表共有186万多条数据。

```
mysql> explain select
    ->     list_id, RISK_CATEGORY, PROPOSAL_ID, CUSTOMER_ID, RISK_AMOUNT, PARTITION_INDI, INSERT_TIME,
    ->     UPDATE_TIME, INSERT_BY, UPDATE_BY, STATUS, ITEM_CATEGORY, EXPIRY_DATE
    ->     from t_customer_risk_sum_old where 1=1
    ->       AND  PROPOSAL_ID = '8017081702000408'\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: t_customer_risk_sum_old
   partitions: NULL
         type: ALL
possible_keys: NULL
          key: NULL
      key_len: NULL
          ref: NULL
         rows: 1745027
     filtered: 10.00
        Extra: Using where
1 row in set, 1 warning (0.00 sec)

mysql>
```

查询走了全表扫，公扫描了174万多条记录，结果集只有一条，却要扫描这么多条数据，明显查询效率及其低下。

```
mysql> show create table t_customer_risk_sum_old\G;
*************************** 1. row ***************************
       Table: t_customer_risk_sum_old
Create Table: CREATE TABLE `t_customer_risk_sum_old` (
  `list_id` varchar(40) NOT NULL,
  `RISK_CATEGORY` varchar(20) NOT NULL,
  `PROPOSAL_ID` varchar(40) NOT NULL COMMENT '投保单ID',
  `CUSTOMER_ID` bigint(20) NOT NULL,
  `RISK_AMOUNT` decimal(18,2) DEFAULT NULL,
  `PARTITION_INDI` char(2) DEFAULT NULL COMMENT 'PARTITION_INDI',
  `INSERT_TIME` datetime DEFAULT NULL COMMENT '插入时间',
  `UPDATE_TIME` datetime DEFAULT NULL COMMENT '更新时间',
  `INSERT_BY` bigint(20) DEFAULT NULL COMMENT '插入人员',
  `UPDATE_BY` bigint(20) DEFAULT NULL COMMENT '更新人员',
```

```
  `STATUS` varchar(2) DEFAULT NULL COMMENT '状态(0在途 1有效 2  无效)',
  `ITEM_CATEGORY` varchar(20) DEFAULT NULL COMMENT '风险类别明细',
  `EXPIRY_DATE` date DEFAULT NULL COMMENT '失效时间',
  PRIMARY KEY (`list_id`),
  KEY `INX_CUST_RISK_SUM_OLD__CUSTOMER_ID` (`CUSTOMER_ID`),
  KEY `INX_CUST_RISK_SUM_OLD_RISKTYPE` (`RISK_CATEGORY`,`ITEM_CATEGORY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.00 sec)

ERROR:
No query specified

mysql>
```

where条件后的PROPOSAL_ID列没有索引，故查询走的是全表扫描。
PROPOSAL_ID列类型为varchar(40)，考虑到该列为投保单ID，纯数字，不会有字母及特殊字符出现。
故觉得varchar(40)类型并不适合。浪费存储空间，还影响查询新能

优化建议 ： 如果确定要经常根据PROPOSAL列查询结果，建议在PROPOSAL_ID列创建索引
（但该列类型为varchar(40)，若创建索引需慎重，可以考虑将PROPOSAL_ID列类型改为bigint(支持19位整形数字)）

1 避免使用字符串索引

字符串索引与数字索引有一些方面如果没做好会非常的慢。

事情的起因是线上日志发现的mysql慢查询。100万数据量的标准，联合查询全部走索引的情况下，尽然要600多毫秒。很不解，但是将索引列由varchar(50)型改为bigint型后，数据提升了30倍。究其原因就索引树上搜索时要进行大量的比较操作，而字符串的比较比整数的比较耗时的多。

所以建议一般情况下不要在字符串列建立索引，如果非要使用字符串索引，可以采用以下两种方法：

1.只是用字符串的最左边n个字符建立索引，推荐n<=10;比如index left(address,8),但是需要知道前缀索引不能在order by中使用，也不能用在索引覆盖上。

2.对字符串使用hash方法将字符串转化为整数，address_key=hashToInt(address)，对address_key建立索引，查询时可以用如下查询where address_key = hashToInt('beijing,china') and address = 'beijing,china';

参考 ： http://blog.csdn.net/hanyingzhong/article/details/48653177