



# 性能优化利器—— 数据库审核平台(Themis)实践

韩锋  
宜信技术研发中心数据库架构师

面临的挑战

审核平台选型

审核平台实践

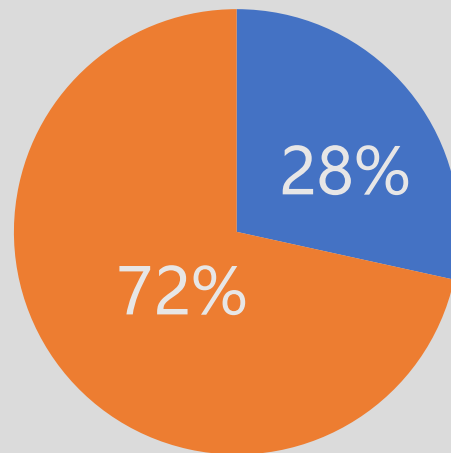
不足及发展

# 面临的挑战





某核心系统核心表  
XXX使用SQL占比  
1061 / 3730



```
SELECT /*+ INDEX (A1 xxxxx) */ SUM(A2.CRKSL), UM(A2.CRKSL*A2.DJ) ...  
FROM xxxx A2, xxxx A1  
WHERE A2.CRKFLAG=xxx AND A2.CDATE >=xxx AND A2.CDATE <xxx;
```

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	Pstart	Pstop
0	SELECT STATEMENT				9890G(100)			
1	SORT AGGREGATE		1	41				
2	MERGE JOIN CARTESIAN		3505T	127P	9890G (1)	999:59:59		
3	PARTITION RANGE ITERATOR		25M	1010M	170K (1)	00:34:12	153	243
4	TABLE ACCESS FULL	<del>XXXXX.XXXXXX</del>	25M	1010M	170K (1)	00:34:12	153	243
5	BUFFER SORT		135M		9890G (1)	999:59:59		
6	INDEX FULL SCAN	<del>INDEX_XXXXX</del>	135M		382K (1)	01:16:34		





重运维，轻架构、优化

重商业产品，轻开源产品

重手工，轻平台、工具

重初级，轻中高级



- Oracle结构设计规范
- Oracle开发规范
- MySQL结构设计规范
- MySQL开发规范





## 平台的选型

代表做法	特点	优点	缺点
智能分析引擎	自研SQL分析引擎，分析语句成本，并自动实现审核、分流、限流等操作。	<ul style="list-style-type: none"><li>可自动审核</li><li>扩展后可线上使用，实现分流等</li></ul>	<ul style="list-style-type: none"><li>难度大</li><li>效率不高</li></ul>
工具+人工审核	自研工具加后期人工审核，事后过滤、人工标记，跟踪全流程	<ul style="list-style-type: none"><li>对SQL精细粒度控制，灵活度大。</li><li>完成对SQL整个生命周期的管理。</li><li>技术难度较小</li></ul>	<ul style="list-style-type: none"><li>人工投入</li></ul>
商业产品	直接抽取SQL，自主分析，仍需人工介入	<ul style="list-style-type: none"><li>功能强大，有技术支持</li><li>周期短，见效快</li></ul>	<ul style="list-style-type: none"><li>费用较高</li><li>扩展性差</li></ul>

- 转变思想，全民动员，人人开发
- 知识沉淀，做好标准，方便落地
- 小步快跑，落地实施，不断修正
- 结合自身，定制目标，学做减法
- 他山之石可以攻玉，大胆引进

# 审核平台实践

### 提供能力

- 快速发现问题

### 操作方式

- WEB界面

### 审核力度

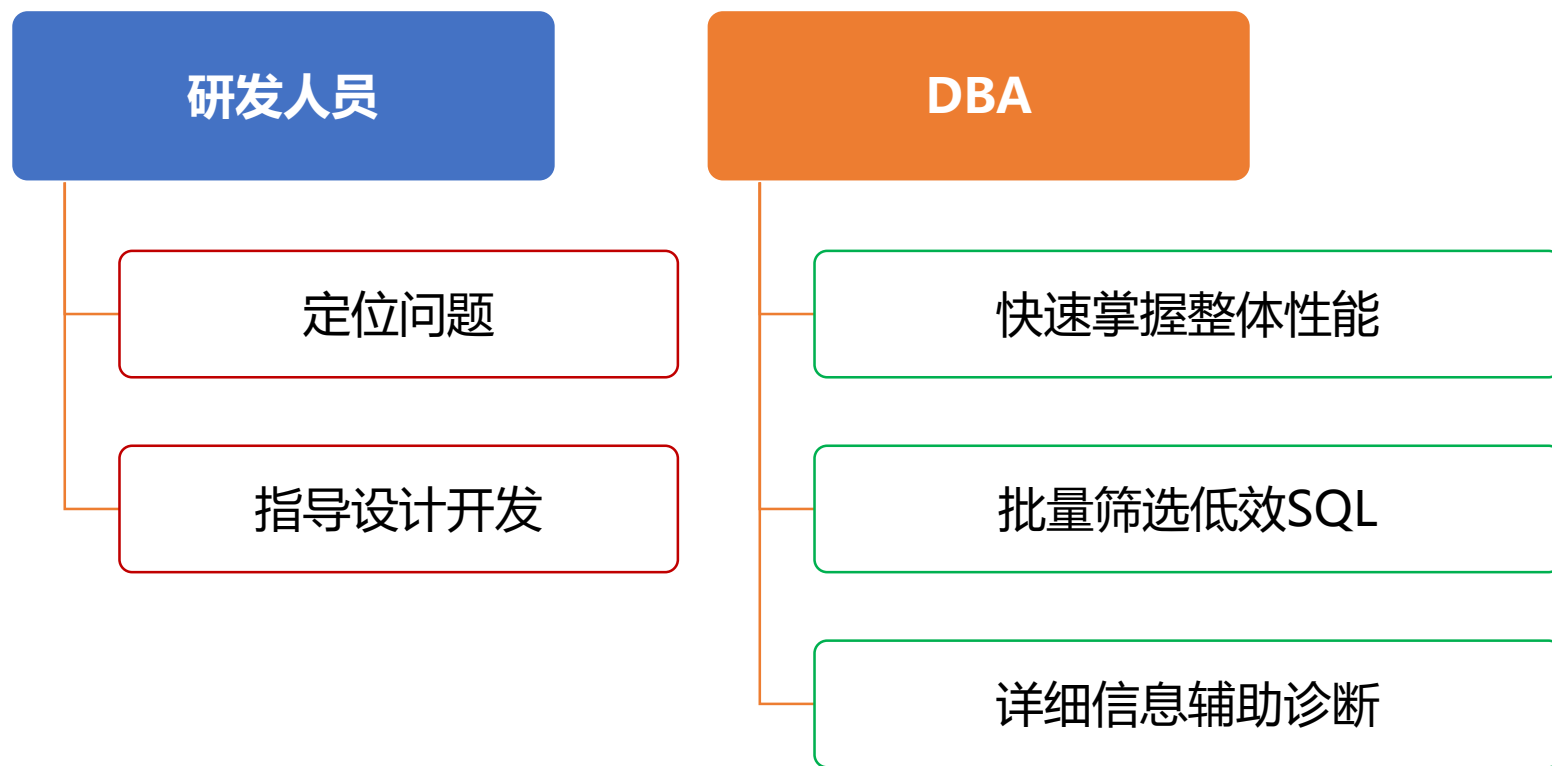
- Schema

### 审核维度

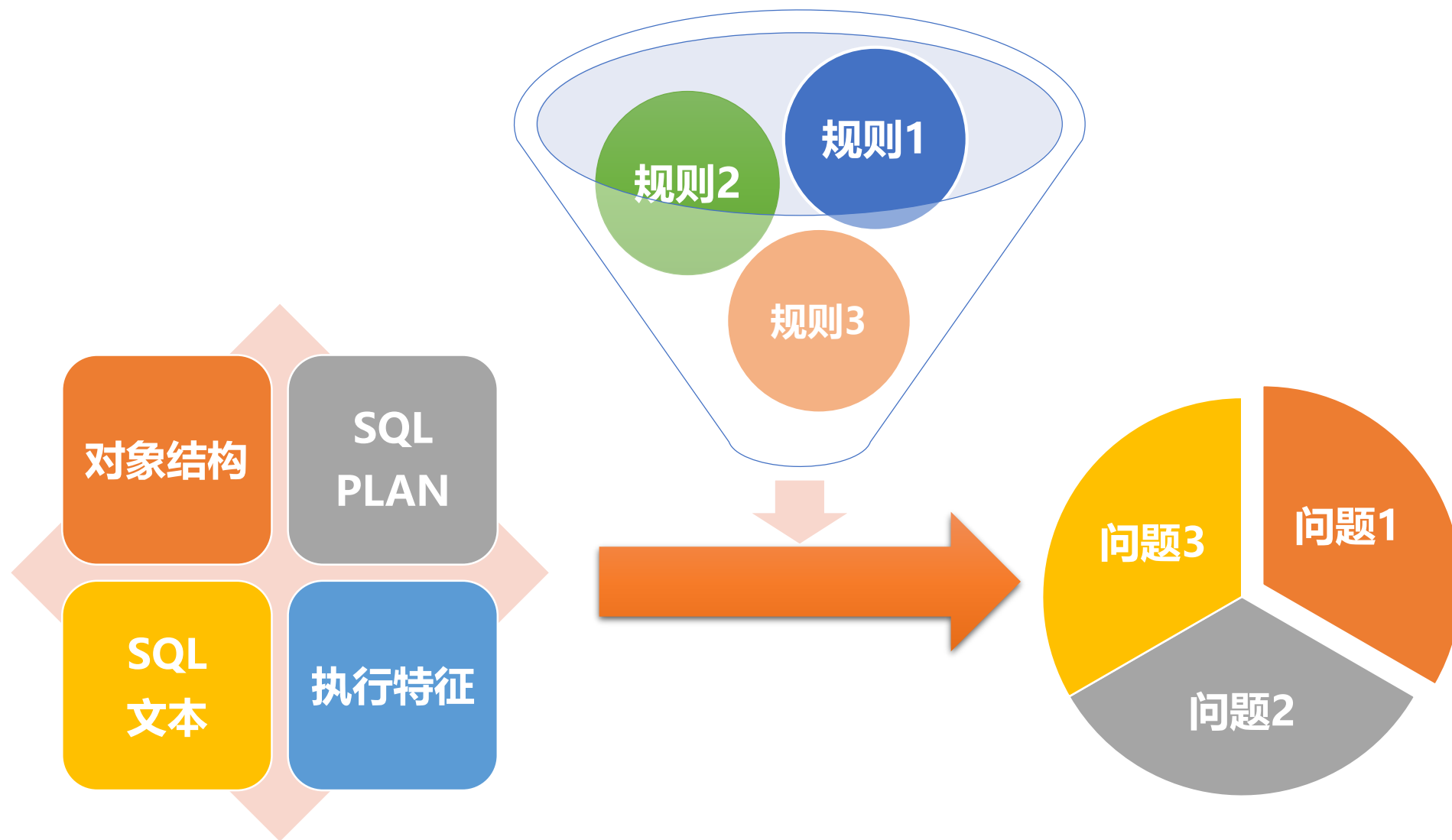
- 结构、语句、文本、执行计划、执行特征

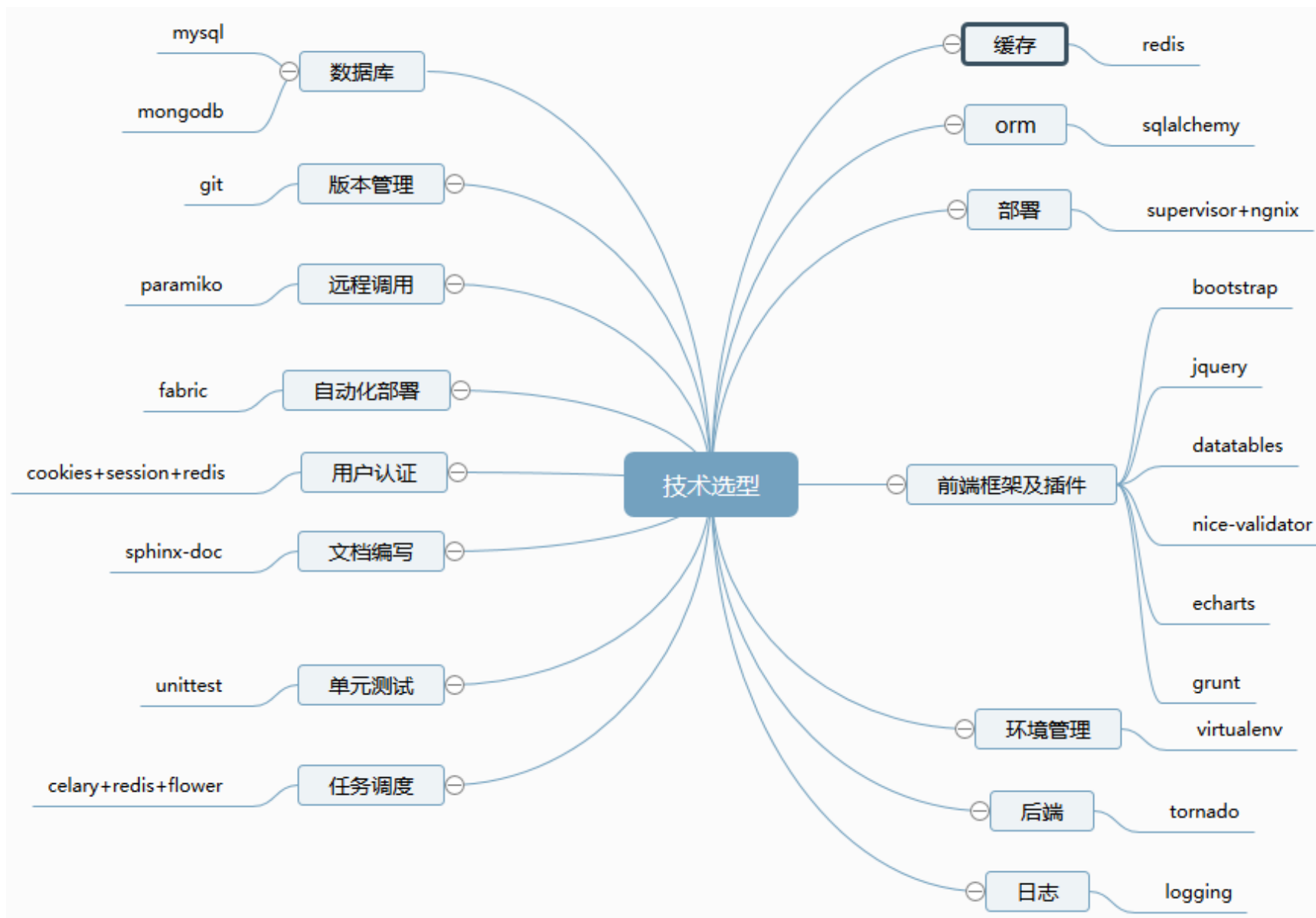
### 审核结果

- 报告(WEB , EXCEL)

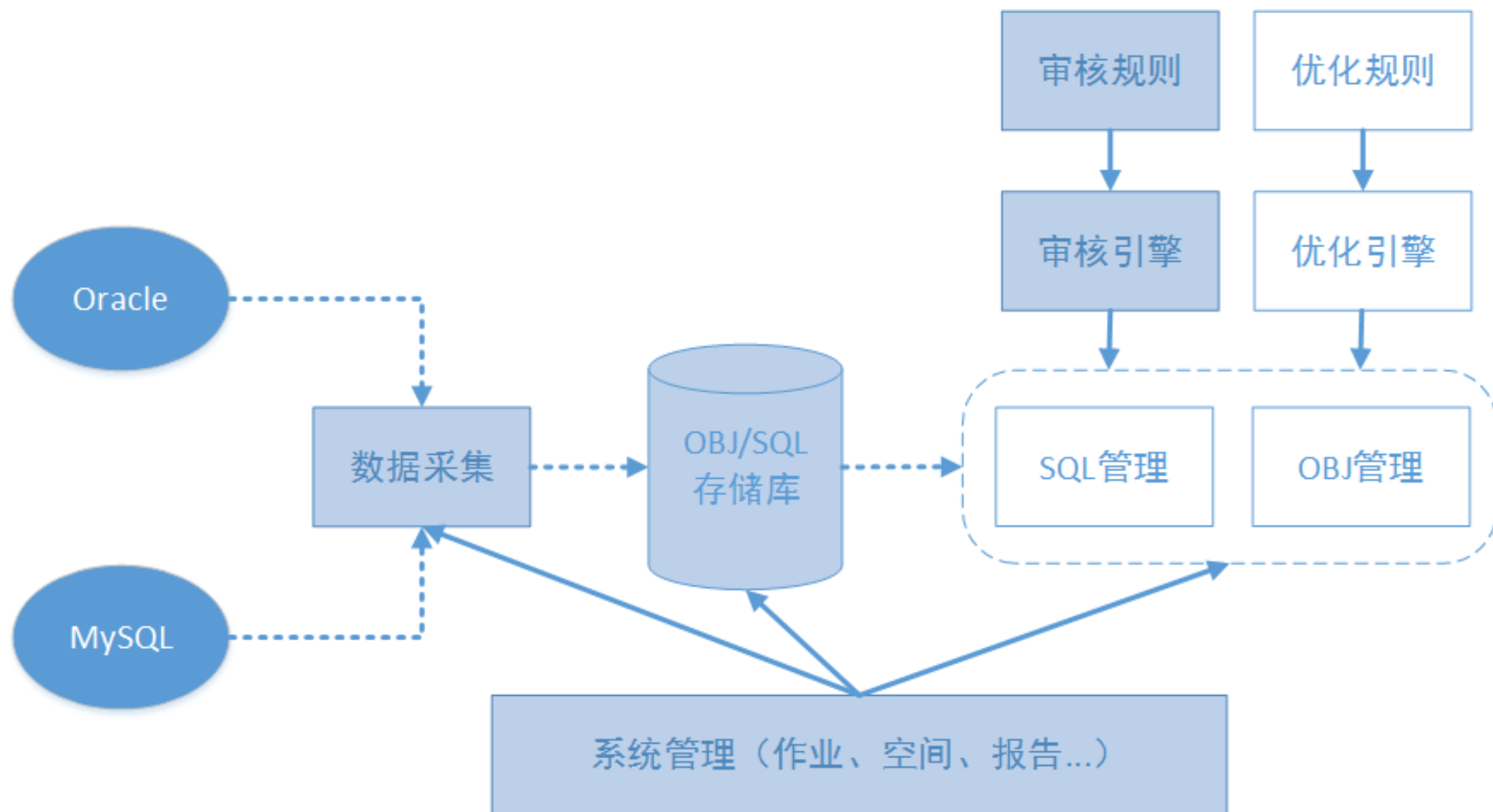


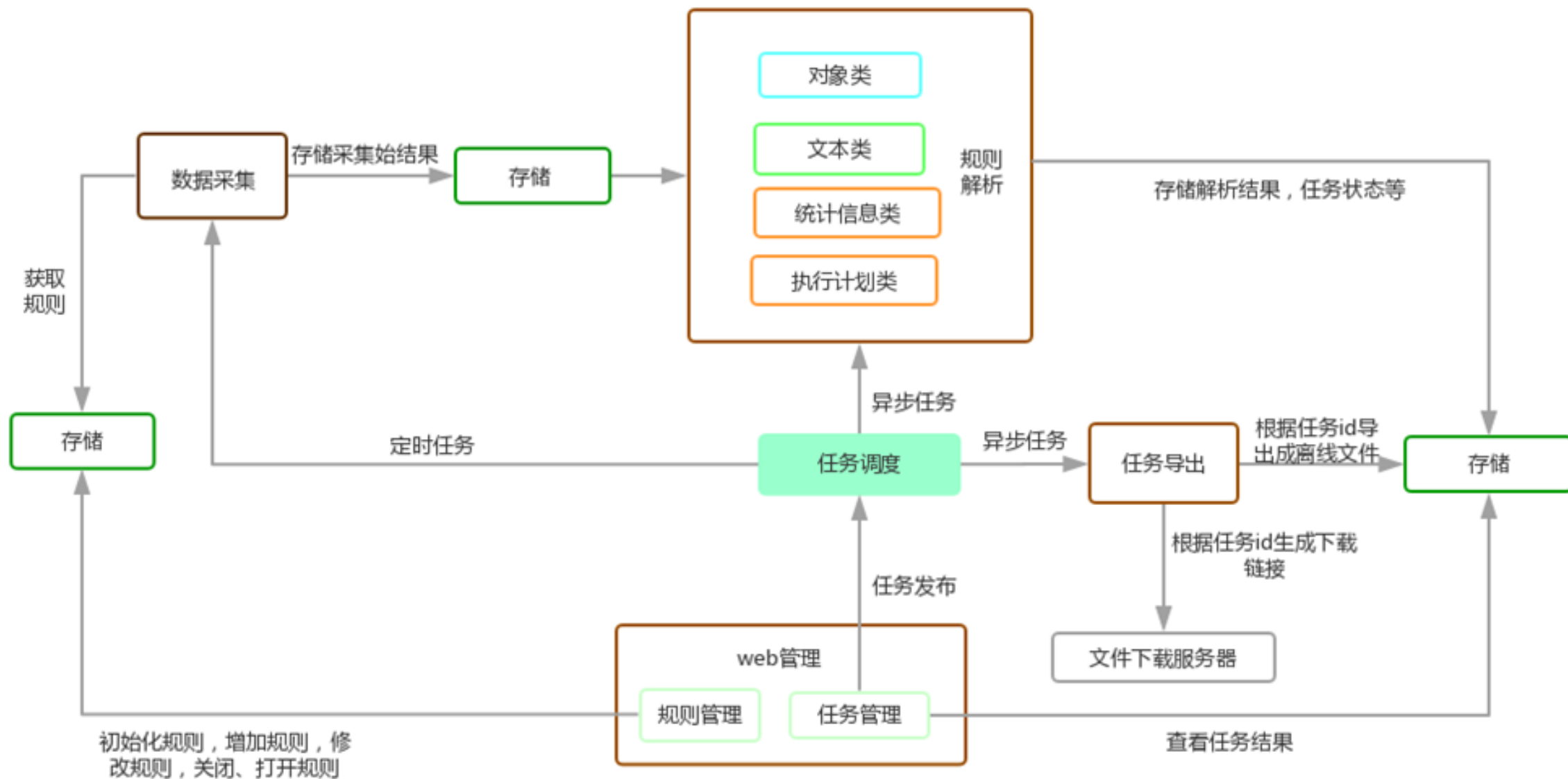






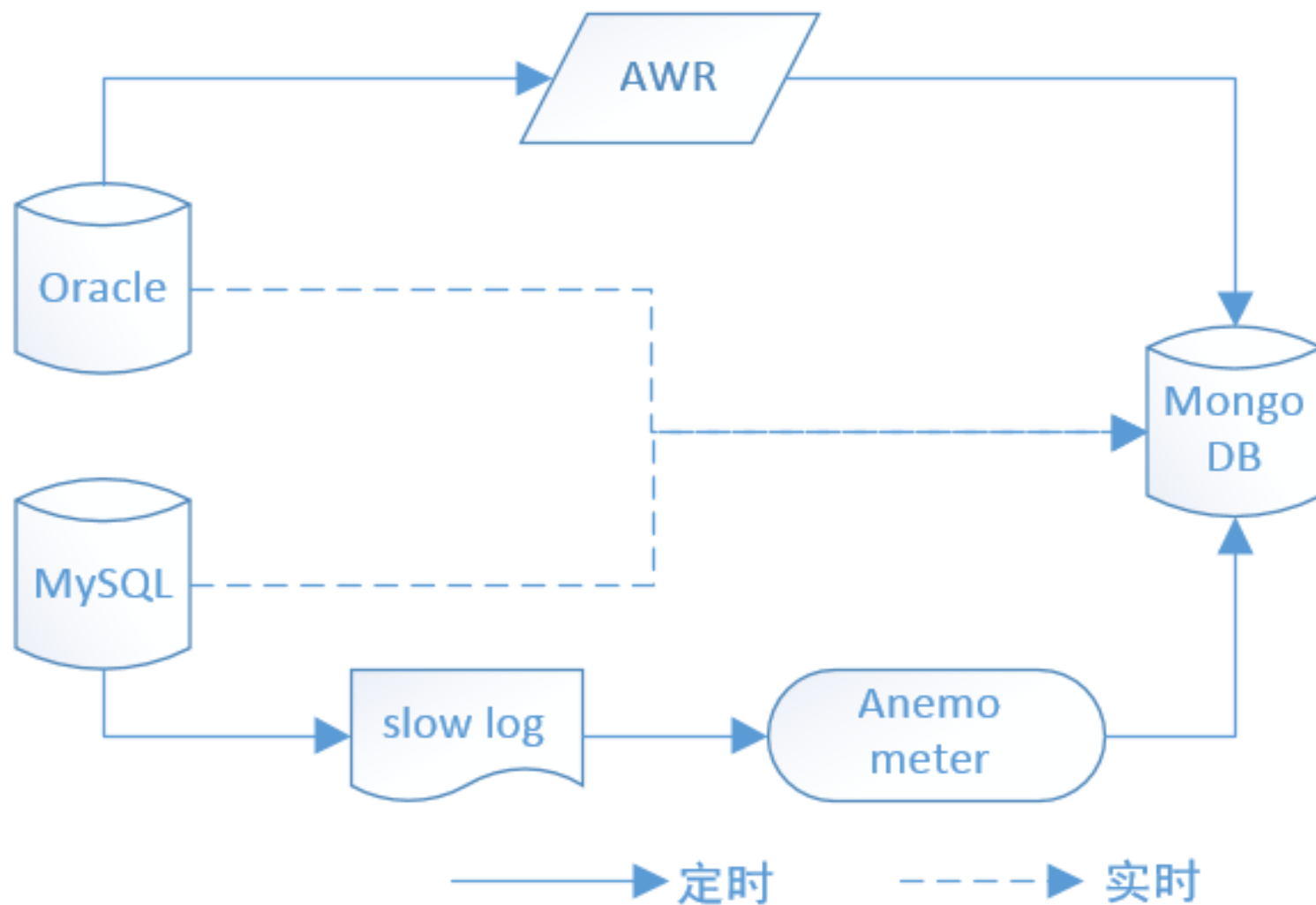
审核类别	示例规则
对象级	大表未分区
	未创建主键
语句级	多表关联
	标量子查询
执行计划级	大表全表扫描
	笛卡尔积
执行特征级	扫描块数与返回记录比例过低
	子游标数过多







	采集内容	Oracle	MySQL
对象	统计信息	√	√ *
	存储特征	√	√ *
	结构信息	√	√ *
	访问特征	√	
SQL	SQL文本	√	√
	执行计划	√	√ *
	游标	√	
	绑定变量	√	
	执行特征	√	√





规则解析是系统的核心部分，主要包括规则判断、计分等。

规则部分可简单划分如下：

- **数据库类型**

Oracle、MySQL

- **规则复杂度**

简单、复杂规则

- **审核对象**

对象类、文本类、执行计划类、执行特征类

```
{
  "db_type" : "mysql",
  "input_parms" : [],
  "max_score" : 20,
  "output_parms" : [],
  "rule_complexity": "simple",
  "rule_cmd": "like .\\%",
  "rule_desc" : "谓词条件使用like %xxx,无法使用索引",
  "rule_name" : "LIKE_UNINDEX",
  "rule_status" : "ON",
  "rule_summary" : "谓词条件使用like %xxx,无法使用索引",
  "rule_type" : "TEXT",
  "solution" : [
    "从业务角度出发, 分析是否可使用精确运算符或类似like 'xx%'",
  ],
  "weight" : 2
}
```

规则类别	规则说明
表、分区	大表过多
	超过指定规模没有分区
	单表或单分区数据量过多
	存在并行属性
	分区数量过多
索引	外键没有索引
	字段重复索引
	聚簇因子多大索引
	字段重复索引
字段	字段数量过多
	记录长度多长
	字段类型不匹配

```
# sql变量, 返回传入用户的组合索引数量和所有索引数量
sql = ""
SELECT 'COMBINEINDEX',
       COUNT(DISTINCT IC.INDEX_NAME) AS COMBINEINDEXNUMBER
FROM DBA_IND_COLUMNS IC
WHERE IC.INDEX_OWNER = '@username@'
      AND IC.COLUMN_POSITION > 1
UNION ALL
SELECT 'ALLINDEX',
       COUNT(1)
FROM DBA_INDEXES I WHERE I.OWNER = '@username@'
""
```

对象类规则解析比较简单，原理是从目标数据里查询数据字典信息，再根据结果进行分数的计算。

规则类别	规则说明
访问路径	大表扫描
	大索引扫描
	大索引快速全扫描
	索引跳跃扫描
	分区全扫描
	非连续分区扫描
	跨分区扫描
表间关联	笛卡尔积
	多表关联
	嵌套循环层次过多
类型转换	存在隐式类型转换
绑定变量	未使用绑定变量

```
mysql> explain format=json select * from film where film_id in
***** 1. row *****
EXPLAIN: {
  "query_block": {
    "select_id": 1,
    "nested_loop": [
      {
        "table": {
          "table_name": "film_actor",
          "access_type": "ref",
          "possible_keys": [
            "PRIMARY",
            "idx_fk_film_id"
          ],
          "key": "PRIMARY",
          "used_key_parts": [
            "actor_id"
          ],
          "key_length": "2",
          "ref": [
            "const"
          ],
          "rows": 19,
          "filtered": 100,
          "using_index": true
        }
      },
      {
        "table": {
          "table_name": "film",
          "access_type": "const",
          "possible_keys": [
            "PRIMARY"
          ],
          "key": "PRIMARY",
          "used_key_parts": [
            "film_id"
          ],
          "key_length": "4",
          "ref": [
            "const"
          ],
          "rows": 1,
          "filtered": 100,
          "using_index": false
        }
      }
    ]
  }
}
```

*schemaless*



```
"USERNAME" : "CLIC111123",
"SQL_ID" : "9ckavqbv8dap8",
"DB_SID" : "CEDB",
"BYTES" : 12,
"OBJECT_TYPE" : "TABLE",
"ETL_DATE" : "2016-08-04",
"PARTITION_ID" : null,
"PARTITION_STOP" : null,
"DEPTH" : 2,
"COST" : 3,
"OTHER_TAG" : null,
"OBJECT_NODE" : null,
"OPERATION_DISPLAY" : "TABLE ACCESS",
"IO_COST" : 3,
"PARTITION_START" : null,
"OPTIONS" : "BY INDEX ROWID",
"OPTIMIZER" : null,
"OBJECT_OWNER" : "CLIC111123",
"CPU_COST" : 30481,
"IPADDR" : "10.100.33.77",
"DISTRIBUTION" : null,
"ID" : 2,
"PARENT_ID" : 1,
"OBJECT_NAME" : "TC_BS_TRANSPORT",
"OTHER" : null,
"PLAN_HASH_VALUE" : NumberLong(3152128743),
"POSITION" : 1,
"OPERATION" : "TABLE ACCESS",
"CARDINALITY" : 1,
```

```
db.@collection_name@.find({
  "OPERATION":"PARTITION RANGE",
  "OPTIONS":"ALL",
  "USERNAME":"@username@",
  "ETL_DATE":"@etl_date@"
}).forEach(function(x){
  db.@sql@.find({
    "SQL_ID":x.SQL_ID,
    "PLAN_HASH_VALUE":x.PLAN_HASH_VALUE,
    "ID":{"$eq:x.ID+1}
  }).forEach(function(y){
    db.@tmp@.save({
      "SQL_ID":y.SQL_ID,
      "PLAN_HASH_VALUE":y.PLAN_HASH_VALUE,
      "OBJECT_NAME":y.OBJECT_NAME,
      "ID":y.ID,
      "COST":x.COST,
      "COUNT":""})
  });
});
```

左侧，采集执行计划数据样本。

右侧，规则实现样例（mongo语句）。

## 大表全表扫描规则 →

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		1	8	410 (1)	00:00:05
* 1	TABLE ACCESS FULL	T_TEST	1	8	410 (1)	00:00:05

```
db.sqlplan.find({SQL_ID:'fzzxntg1g9u6r'},  
  {OBJECT_TYPE:1,OBJECT_NAME:1,OPERATION_DISPLAY:1,OPTIONS:1,_id:0})
```

```
{ "O" : "SELECT STATEMENT", "OPTIONS" : null, "OBJECT_NAME" : null }
```

```
{ "O" : "    TABLE ACCESS", "OPTIONS" : "FULL", "OBJECT_NAME" : "T_TEST" }
```



```
db.@sql@.find({"OPERATION":"TABLE ACCESS",
              "OPTIONS":"FULL",
              "USERNAME":"@username@",
              "ETL_DATE":"@etl_date@"}
).forEach(function(x)
{
  if(db.obj_tab_info.findOne({
    "TABLE_NAME":x.OBJECT_NAME,
    $or:
    [{"NUM_ROWS":{"$gt:@params1@"}},
     {"PHY_SIZE(MB)":{"$gt:@params0@"}}
  ]})
)db.@tmp@.save({
  "SQL_ID":x.SQL_ID,
  "PLAN_HASH_VALUE":x.PLAN_HASH_VALUE,
  "OBJECT_NAME":x.OBJECT_NAME});})"
```



- 过滤执行计划
- 按统计信息筛选
- 按存储特征筛选
- 获得执行计划
- 后台存储并计分
- 汇总展示报告

```
mysql> explain format=json select * from rbac_city;
```

```
+-----+
| EXPL |
+-----+
| {
  "que
  "s
  "t
    if isinstance(arg, dict):
      level = level + 1
      for {
        "_id":ObjectId("57359bf4199f9c8ed0d84c53"),
        "checksum": "XXXXXX",
        "item_type": "query_block",
        "item_level": "1",
        "select_id": 1,
      elif isi
      for
        "citem_type": "table",
        "citem" : [ObjectId("57359bf4199f9c8ed0d84c54")]
      }
    {
      "_id":ObjectId("57359bf4199f9c8ed0d84c54"),
      "checksum": "XXXXXX",
      "item_type": "table",
      "item_level": "1_1",
      "table_name": "r_bac_city",
      "rows": 3295,
      "filtered": 100,
      "access_type" : "ALL",
      "citem" : [ObjectId("57359bf4199f9c8ed0d84c55")]
    }
  }
}
+-----+
1 row
mysql>
```

规则说明
select *
嵌套select子句
谓词中出现反向操作符
多个过滤条件通过or连接
存在子查询
存在三个以上的多表关联
存在全连接或外连接
delete中必须出现where
update中出现order by子句
inlist元素过多
重复查询子句
出现union集合操作

实现就是基于正则表达式，做模式匹配。

- **示例 — BAD\_JOIN**

“rule\_cmd” : “(cross join)|(outer join)”

- **示例 — SUB QUERY**

```
left_bracket = []
sql_content = []
for k in sql_length:
    if sql[k] == "(":
        left_bracket.append(k)
    if sql[k] == ")":
        start = left_bracket.pop() + 1
        stop = k - 1
        sql_content.append(sql[start:stop])
```

规则说明
扫描块数与返回记录数比例过低
子游标过多
elapsed_time
cpu_time
buffer_gets
disk_reads
direct_writes
Executions

待执行sql或mongo语句

查询语句

数据库类型

mysql

规则名称

规则名称

最大扣分

最大扣分

规则类型

OBJ

规则概要

规则概要

扩展字段

对象扩展字段

规则权重

规则权重

规则描述

规则描述

解决方案

解决方案，每行一条

增加输入参数

减少输入参数

增加输出参数

减少输出参数

提交

规则是平台核心，其丰富程度代表整体能力。为了满足不同需求，平台允许动态增加自有规则，只要遵守统一格式即可。

COUNT_RECORD_TAB	单表或单分区记录数量过大	ON	0.5	3	OBJ	Oracle	10000000	0	0	0	0	TABLE
COUNT_SUBPART_TAB	复合分区数量过多	ON	0.5	3	OBJ	Oracle	200	0	0	0	0	PART_TABLE
COUNT_SUMPART_FULL_TAB	分区表数量过多	ON	1	5	OBJ	Oracle	10	0	0	0	0	PART_TABLE
COUNT_SUMPART_SINGLE_TAB	分区数量过多	ON	0.5	3	OBJ	Oracle	200	0	0	0	0	PART_TABLE
DBLINKS_NUM	存在DBLINK	ON	10	10	OBJ	Oracle	0	0	0	0	0	DBLINK
DUPLICATE_INDEX	字段重复索引	ON	0.3	3	OBJ	Oracle	0	0	0	0	0	COLUMN

平台允许对规则动态关闭，打开，修改规则参数等。

ip地址：

10.120.12.51

端口号：

schema：

规则类型：

plan

text

对象

执行特征

日期选择：

选择日期

执行任务

上面为任务发布界面

下面为任务结果查看界面

导出报告

查看报告

操作用户	用户名	创建时间	状态	类型	开始日期	结束日期	选择
system	NEWDX	2016-12-29 15:16:42	成功	TEXT	2016-12-20	2016-12-21	<input checked="" type="checkbox"/>
system	NEWDX	2016-12-21 15:06:43	成功	SQLPLAN	2016-12-20	2016-12-20	<input type="checkbox"/>



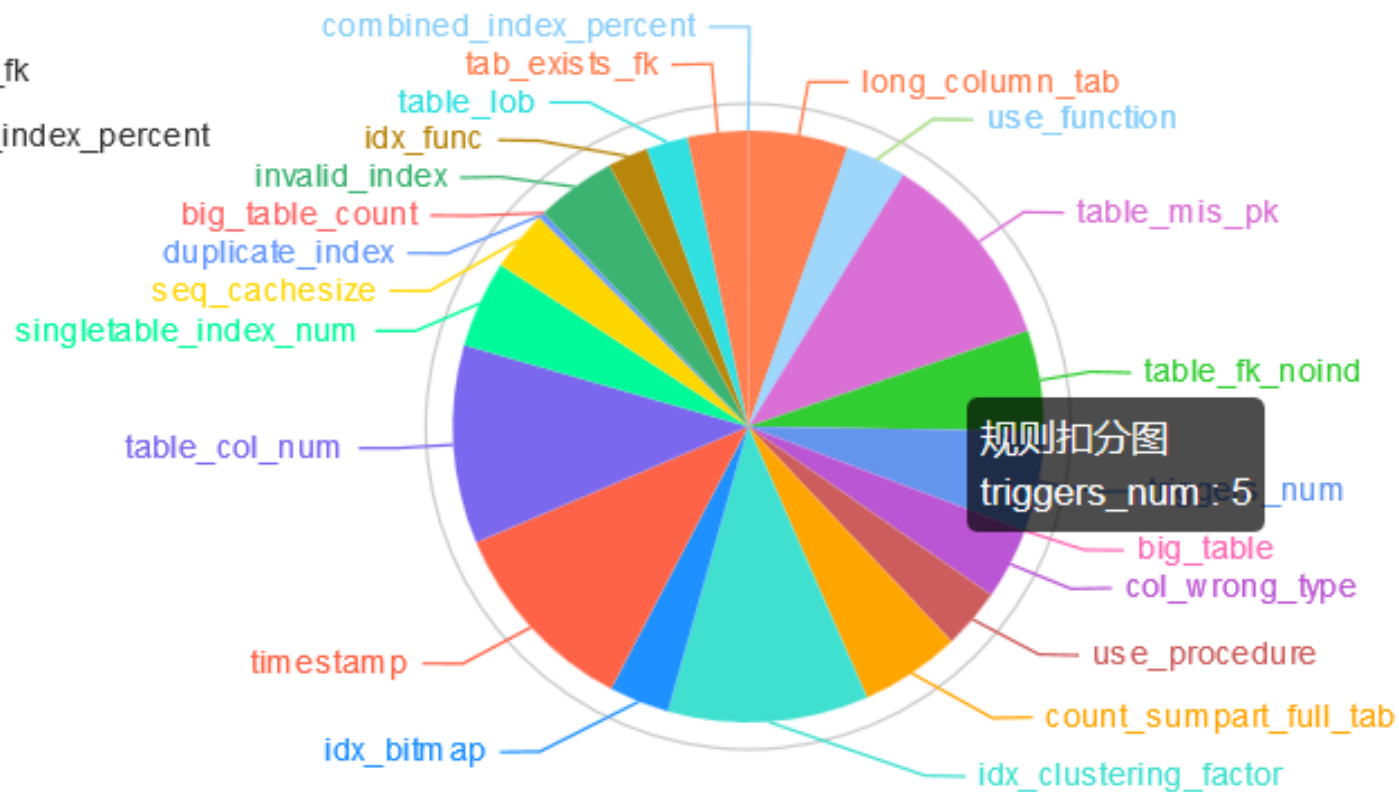
## 规则扣分

- long\_column\_tab
- use\_function
- table\_mis\_pk
- table\_fk\_noind
- triggers\_num
- big\_table
- col\_wrong\_type
- use\_procedure
- count\_sumpart\_full\_tab
- idx\_clustering\_factor
- idx\_bitmap
- timestamp
- table\_col\_num
- singletable\_index\_num
- seq\_cachesize
- duplicate\_index

- big\_table\_count
- invalid\_index
- idx\_func
- table\_lob
- tab\_exists\_fk
- combined\_index\_percent

规则总分: 55.854

规则扣分详情

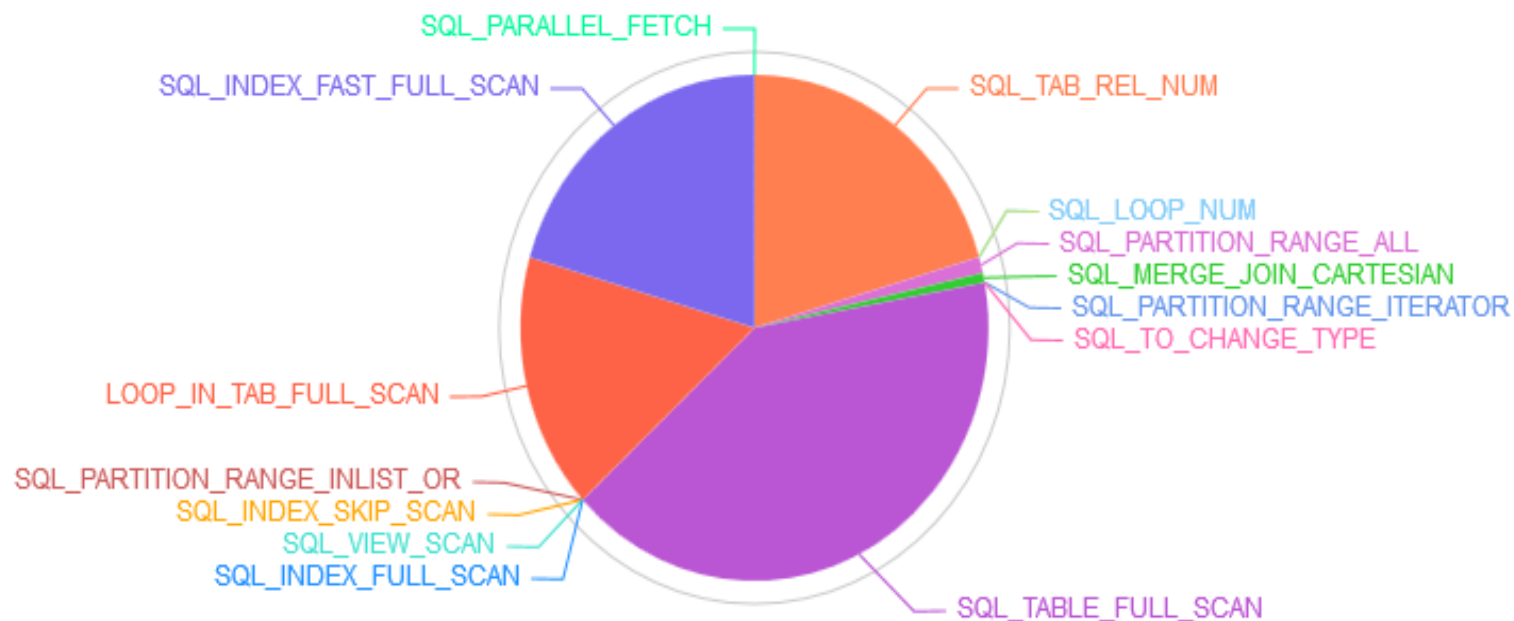


规则名称	规则描述	参数个数	违反次数	扣分
big_table	超过指定规模且没有分区的表	1	0	0
big_table_count	大表数量	1	1	0
col_wrong_type	表字段类型不匹配	3	153	3.558139534883721
combined_index_percent	组合索引数量过多或没有索引	1	13	0
count_sumpart_full_tab	分区表数量过多	1	11	5
duplicate_index	字段重复索引	0	1	0.3
idx_bitmap	是否使用位图索引	0	6	3
idx_clustering_factor	索引的聚簇因子	1	238	10

- SQL\_TAB\_REL\_NUM
- SQL\_LOOP\_NUM
- SQL\_PARTITION\_RANGE\_ALL
- SQL\_MERGE\_JOIN\_CARTESIAN
- SQL\_PARTITION\_RANGE\_ITERATOR
- SQL\_TO\_CHANGE\_TYPE
- SQL\_TABLE\_FULL\_SCAN
- SQL\_PARTITION\_RANGE\_INLIST\_OR
- SQL\_INDEX\_SKIP\_SCAN
- SQL\_VIEW\_SCAN
- SQL\_INDEX\_FULL\_SCAN
- LOOP\_IN\_TAB\_FULL\_SCAN
- SQL\_INDEX\_FAST\_FULL\_SCAN
- SQL\_PARALLEL\_FETCH

规则总分: 68.516

规则扣分详情



规则名称	规则描述	违反次数	扣分
SQL_TABLE_FULL_SCAN	大表全表扫描	1213	20
SQL_TAB_REL_NUM	过多的表关联，影响性能	184	10
SQL_INDEX_FAST_FULL_SCAN	大索引快速全扫描	148	10
LOOP_IN_TAB_FULL_SCAN	嵌套循环内层表访问方式为全表扫描	12	8
SQL_PARTITION_RANGE_ALL	分区全扫描	1	0.5
SQL_MERGE_JOIN_CARTESIAN	笛卡尔积	3	0.3
SQL_LOOP_NUM	嵌套层次过深	0	0
SQL_PARTITION_RANGE_ITERATOR	跨分区扫描	0	0
SQL_TO_CHANGE_TYPE	隐式转换	0	0
SQL_PARTITION_RANGE_INLIST_OR	非连续分区扫描	0	0

SQL\_TABLE\_FULL\_SCAN

## 解决方案

- 1.缺索引评估创建索引
- 2.取max、min值评估创建索引
- 3.索引失效重建索引，分区表维护记得维护索引
- 4.对条件字段使用函数或表达式a.函数、表达式放到等于号的右边b.创建函数索引(下策)
- 5.出现隐式转换a.不同类型的谓词匹配先显式转换b.表定义根据数据选择正确的数据类型
- 6.使用isNULL做查询条件a.不建议使用null值b.null值较少的情况可创建组合索引或者伪列索引(createindexidx\_1ontab1(col1,0)c.将null定义一个普通变量
- 7.使用不等运算符<>!=做查询条件a.尽量少用不等判断；b.如果列值是连续，可把否定操作更改为两个区间；c.如果列值不多，可用inlist枚举其他所有值
- 8.模糊匹配'%a%'%'a'建议精确匹配
- 9.sql逻辑，比如最大值，改用窗口函数
- 10.弱选择sql，返回结果集较大建议a.添加更多的谓词减少数据的访问，比如时间b.改造分区表c.使用覆盖索引
- 11.hintfull禁用hint1
- 2.统计信息不准确数据批量加载程序触发收集统计信息

搜索:

sqlid	sqltext	plan_hashvalue	pos	object_name	COST	COUNT
00akdkgu8tmys	SELECT COUNT(*) FROM (SELECT data_record	3398998570	17	ICP_BORROW_CONTRACT	12681	空
00akdkgu8tmys	SELECT COUNT(*) FROM (SELECT data_record	3398998570	13	ICP_CONTRACT_REMARK	1779	空

SQL文本[sql\_id:04qwkz1cjsrc]

```
select l.lending_id lendingId,
       l.lender_id lenderId,
       l.type type,
       l.invest_amt investAmt,
       pti.match_irr_low irrYearMin,
       pti.match_irr_upper irrYearMax,
       i.reinvest reinvest,
       i.recommend_deadline recommendDeadline,
       l.recommend_deadline_end lastMatchingDate,
```

执行计划

sql	OPTIONS	OBJECT_OWNER	OBJECT_NAME
SELECT STATEMENT	null	null	null
NESTED LOOPS	SEMI	null	null
NESTED LOOPS	null	null	null
TABLE ACCESS	BY INDEX ROWID	PHENIX	TP_DIGITAL_CER
INDEX	RANGE SCAN	PHENIX	IDX_TP_DIGITAL_CER_LENDER_ID

执行特征

搜索

PER_ELAPSED_TIME	DISK_READS_DELTA	PER_DISK_READS	PER_BUFFER_GETS	CPU_TIME_DELTA	BUFFER_GETS_DELTA	ELAPSED_TIME
32.423500000000004	649848	649848	1268174	11.4503	1268174	32.423500000000004

显示第 1 至 1 项结果, 共 1 项

上页

1

对象信息

搜索

LAST_ANALYZED	BLOCKS	COL_NUM	TABLE_TYPE	LAST_DDL_TIME	OBJECT_TYPE	OBJECT_NAME	TABLE_NAME	NUM_ROWS
2016-11-16 22:29:21	634035	24	NORMAL	2016-11-11 04:39:36	TABLE	TP_LENDING	TP_LENDING	32192022

显示第 1 至 1 项结果, 共 1 项

上页

1

mysql在解析json格式执行计划中暴露出的问题...

【会话进入sleep状态，假死】

解决方法执行会话之前设置wait\_timeout=3,这个时间根据实际情况进行调整。

【数据量过大，长时间没有结果】

会话处于query状态，但是数据量很大或因为数据库对format=json支持不是很好，长时间解析不出来，会影响其他会话。解决方法使用pt-kill工具杀掉会话。为了防止误杀，可打个标识“eXplAin format=json”，然后使用pt-kill识别eXplAin关键字。







# THANK YOU !

<https://github.com/bjbean/Themis>