陆金所AI SQL审核系统介绍

王英杰

陆金所数据架构团队负责人

1. 项目背景
2. 研发过程
3. 未来展望

1. 项目背景
2. 研发过程
3. 未来展望

1. 项目背景

- SQL的性能对核心系统可用率至关重要。
- 人工SQL审核系统会遭遇开发和DBA的人力资源瓶颈。

1.1 人工SQL审核流程

01 代码提交

02 差异比较

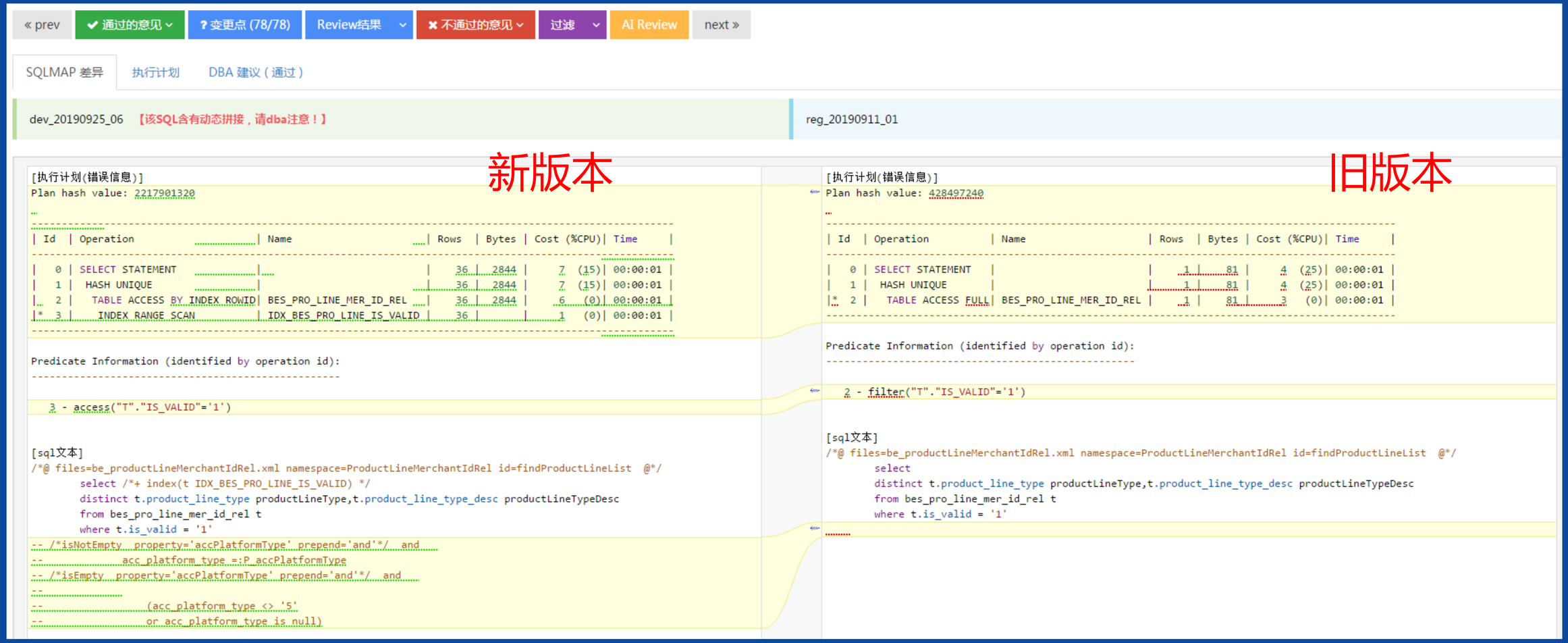
03 人工审核

04 上线批准

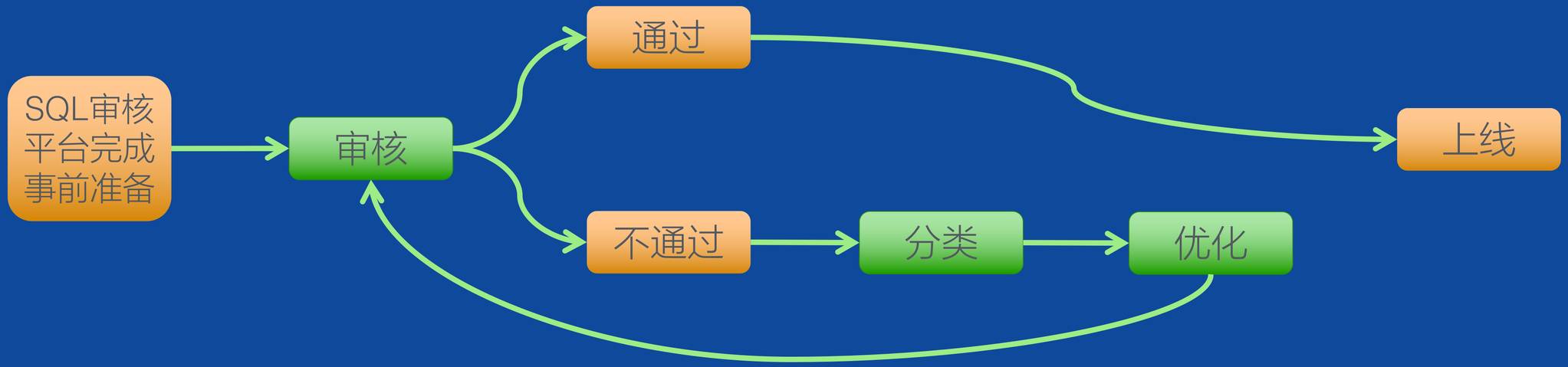
人工SQL审核系统特点：

- 人工SQL审核系统对所有应用、每个版本下的每一笔SQL进行管理
- 审核系统会自动生成每个版本的SQL文本差异以及导致的执行计划差异
- 开发和DBA会审核每个版本里发生变更的SQL、导致的执行计划差异以及对性能的影响
- 发版平台对于人工SQL审核系统建立强依赖

1.2 人工SQL审核平台



1.3 审核流程的展开



- 审核和分类过程就是一个打标签数据的过程
- 打标签过程需要大量的开发和DBA人力资源

1. 项目背景
2. 研发过程
3. 未来展望

2. 研发条件

- 人工SQL审核系统运行了3年，我们有了下面这些数据：
  - 每个应用版本的SQL代码改动数据和当时在生产环境的执行计划变动数据
  - 每个应用版本上线前后的数据字典和统计信息备份
  - 每个应用版本上线前DBA的审核意见数据
  - 每个应用版本上线后在每笔SQL在生产环境的运行时效数据
- 那么，是否可以基于历史数据 + AI算法来模拟DBA对SQL的审核?

2.1 研发思路

- 找到SQL代码、执行计划、统计信息、绑定变量和执行效率之间的相关性。
- 通过相关性基于SQL代码、执行计划、统计信息、绑定变量来预测执行效率。

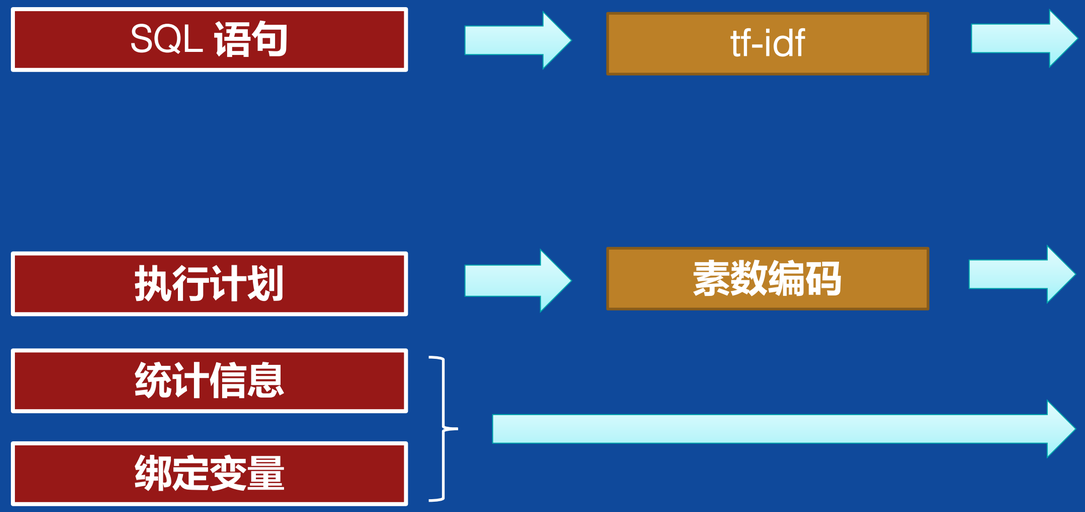
2.2 第一个版本——实现对SQL执行效率的预测

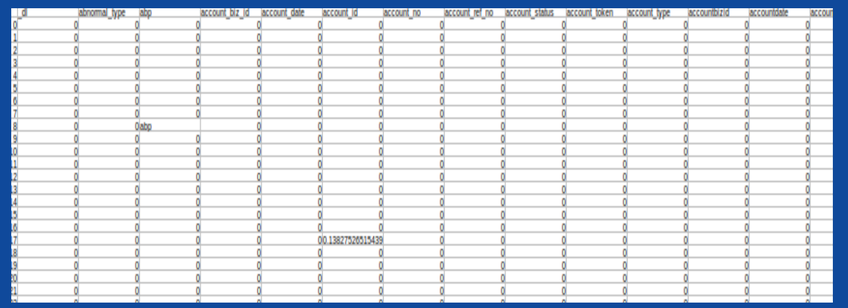
- 找SQL各个维度的信息和生产执行效率的相关性

2.2 特征嵌入

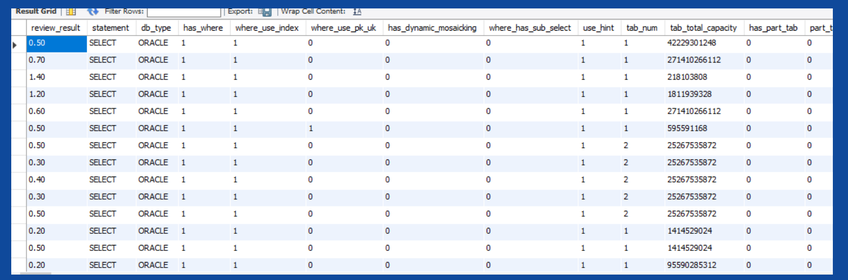
目标：将模型**不可读的TEXT**格式转换为**可读的数字**格式

- SQL语句使用tf-idf（词频逆文档频）转化
- 执行计划使用素数编码转化
- 统计信息和绑定变量直接并入矩阵





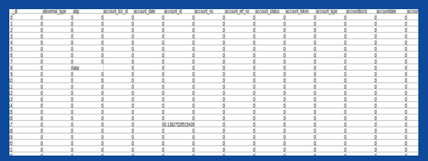
id	abnormal_type	abp	account_biz_id	account_date	account_id	account_no	account_ref_no	account_status	account_token	account_type	accountbizid	accountdate	accou
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	abp	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	00.13827526515439	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0

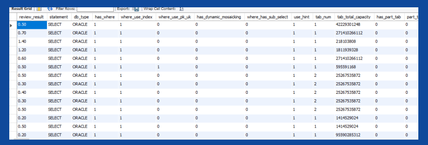


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

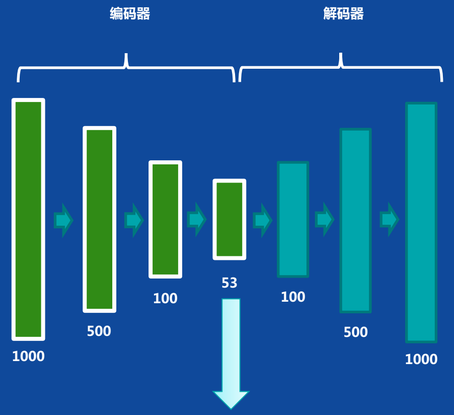
review_result	statement	db_type	has_where	where_use_index	where_use_pk_uk	has_dynamic_mosaicking	where_has_sub_select	use_hint	tab_num	tab_total_capacity	has_part_tab	part_t
0.50	SELECT	ORACLE	1	1	0	0	0	1	1	42229301248	0	0
0.70	SELECT	ORACLE	1	1	0	0	0	1	1	271410266112	0	0
1.40	SELECT	ORACLE	1	1	0	0	0	1	1	218103808	0	0
1.20	SELECT	ORACLE	1	1	0	0	0	1	1	1811939328	0	0
0.60	SELECT	ORACLE	1	1	0	0	0	1	1	271410266112	0	0
0.50	SELECT	ORACLE	1	1	1	0	0	1	1	595591168	0	0
0.50	SELECT	ORACLE	1	1	0	0	0	1	2	25267535872	0	0
0.30	SELECT	ORACLE	1	1	0	0	0	1	2	25267535872	0	0
0.40	SELECT	ORACLE	1	1	0	0	0	1	2	25267535872	0	0
0.30	SELECT	ORACLE	1	1	0	0	0	1	2	25267535872	0	0
0.50	SELECT	ORACLE	1	1	0	0	0	1	2	25267535872	0	0
0.20	SELECT	ORACLE	1	1	0	0	0	1	1	1414529024	0	0
0.50	SELECT	ORACLE	1	1	0	0	0	1	1	1414529024	0	0
0.20	SELECT	ORACLE	1	1	0	0	0	1	1	9590285312	0	0

2.2 信息压缩＋特征提取

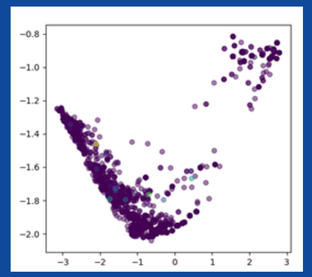




SQL的特征是个特别宽的**稀疏矩阵**， 这也是需要引用信息压缩的主要原因

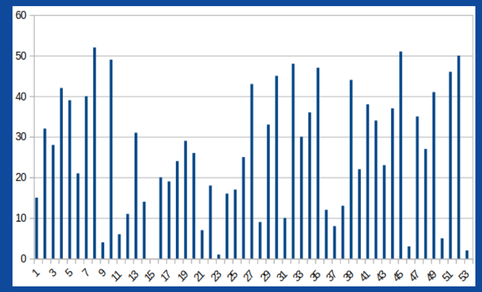


数据分布有明显的两极化



VAEs 信息压缩

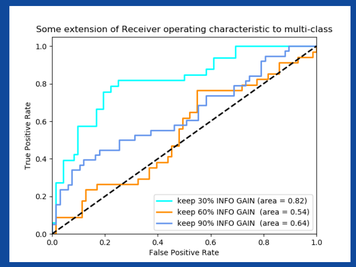
m×53



随机深林特征裁剪， 每个特征对应的权重（information gain）

粗粒度学习

m×38

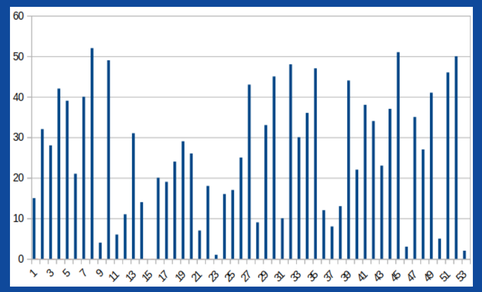


宏观roc

2.2 对SQL的执行时长进行预测

使用线性回归对SQL的执行时长进行预测：

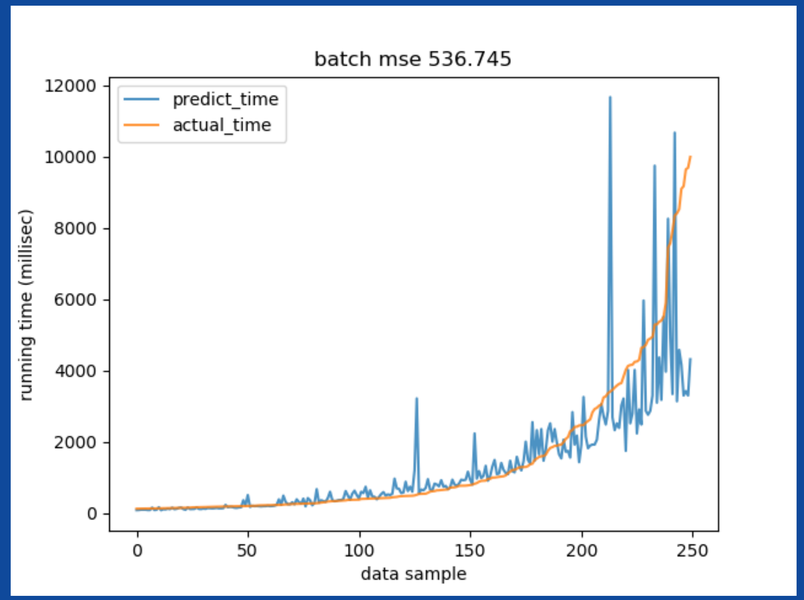
- 在生产环境选取300条SQL按照执行时长从低到高排序，生成实际曲线（橙色）
- 使用线性回归生成预测曲线（蓝色）



m×38

线性回归
Y=WX

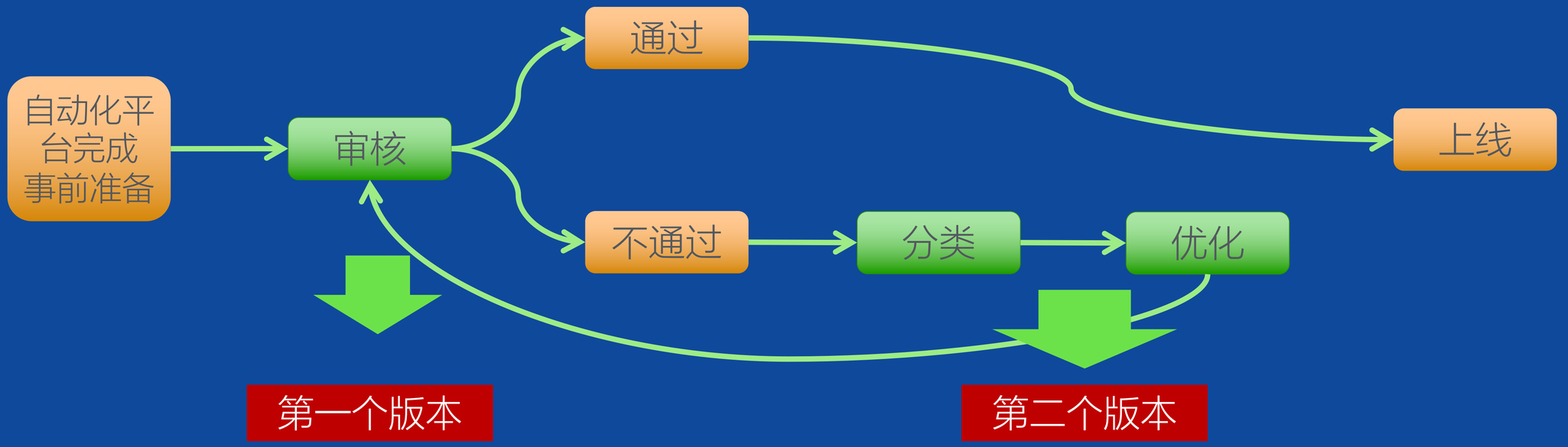
细粒度学习



2.2 在陆金所上线后的效果

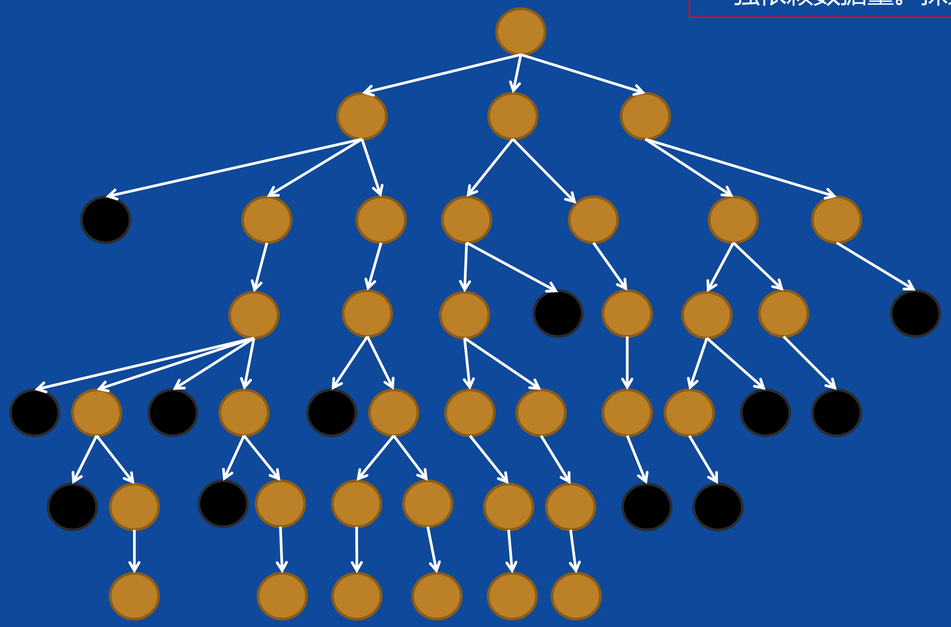
- 基于拟合度下，把AI作为SQL审核平台的第一道环节，AI审核通过即直接上线，人工不再审核。
- SQL审核总量里80%的SQL都是符合性能预期的，这些SQL由AI直接审核后上线，也节省了80%的人工审核工作量。

2.3 第二个版本——实现对SQL优化方案的搜索

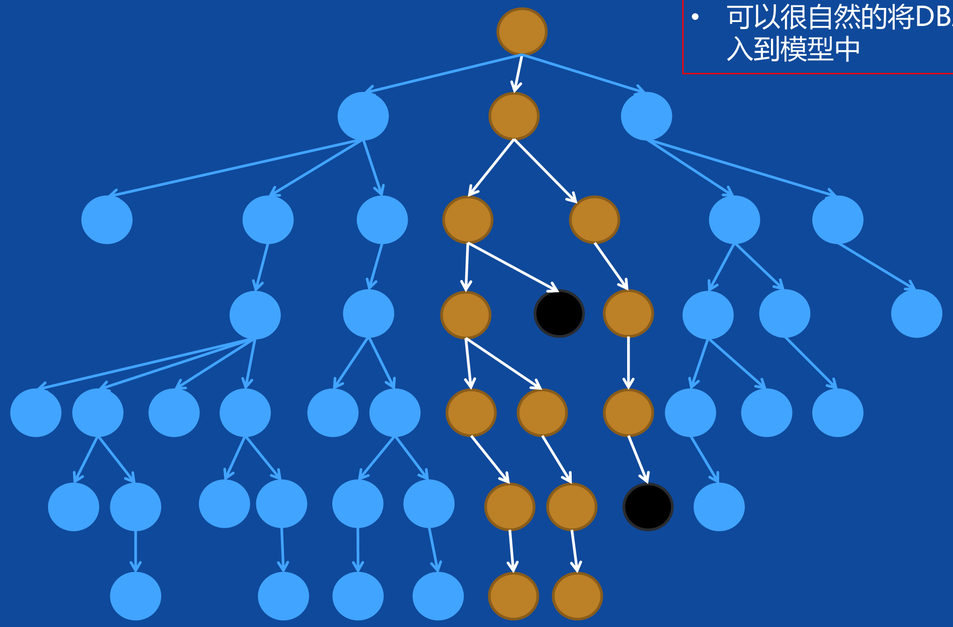


2.3 方案的选择

频率派:
- 基于数据分布得到目标与输入的联合分布
- 强依赖数据量。探索空间大



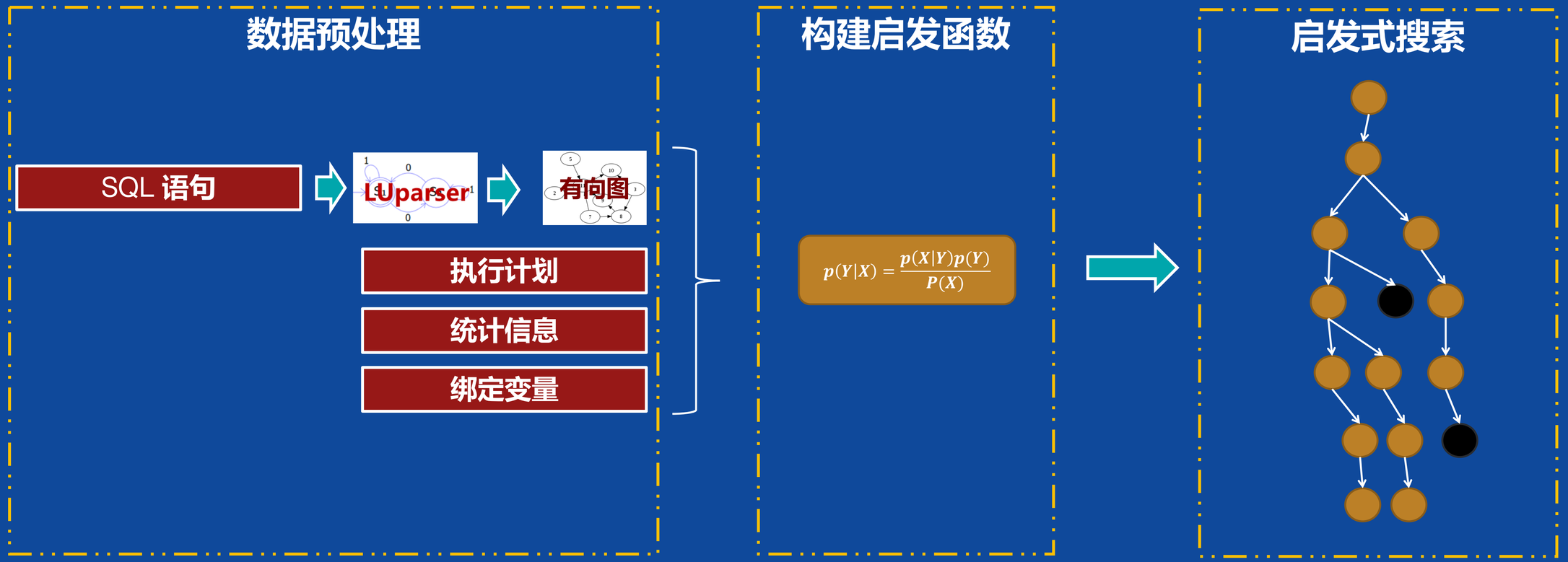
贝叶斯派:
- 基于经验知识得到目标与输入的条件概率
- 先验的加入缩小探索空间
- 可以很自然的将DBA的知识融入到模型中



频率派 P(Y,X)

贝叶斯派 P(Y|X) ~ P(X|Y)P(Y)

2.3 基于贝叶斯的启发式搜索

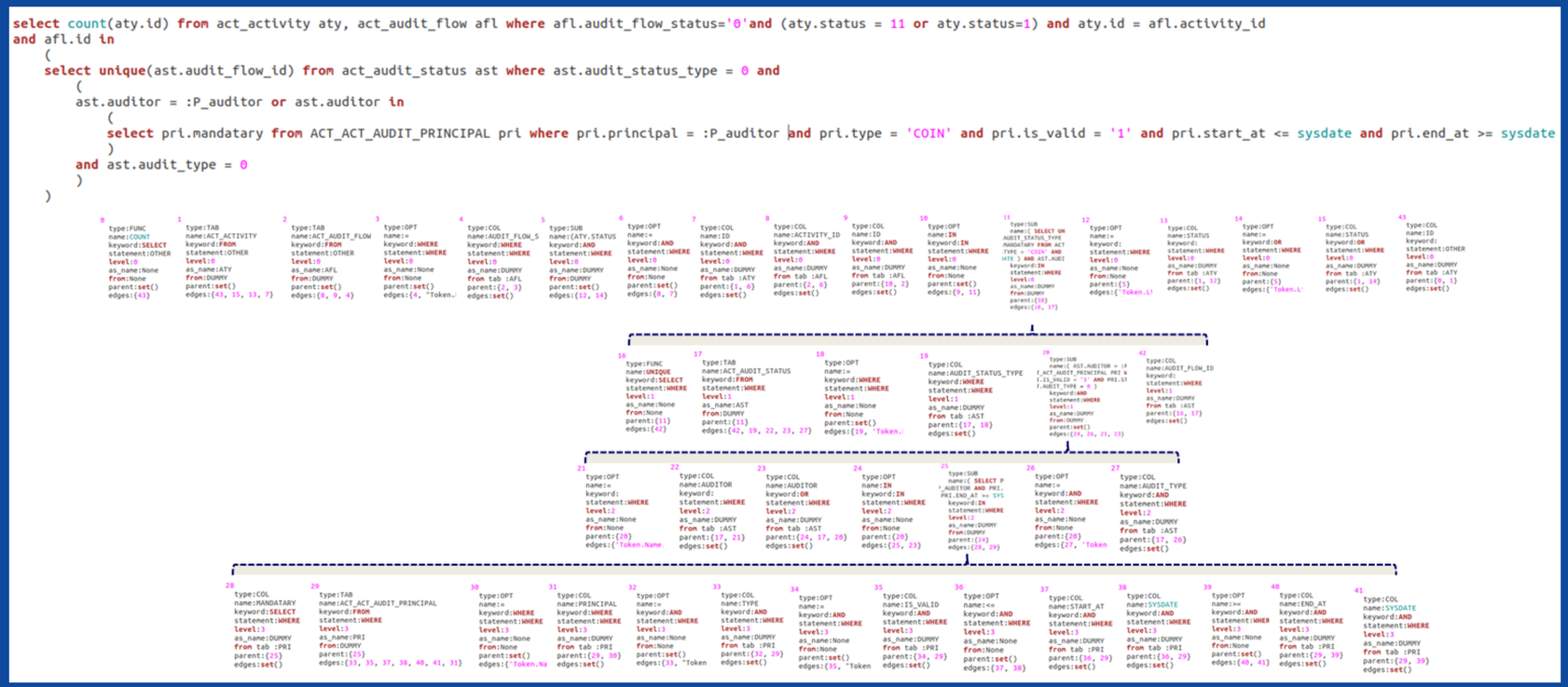


2.4 特征预处理：LUparser——陆金所SQL解析器

目标：

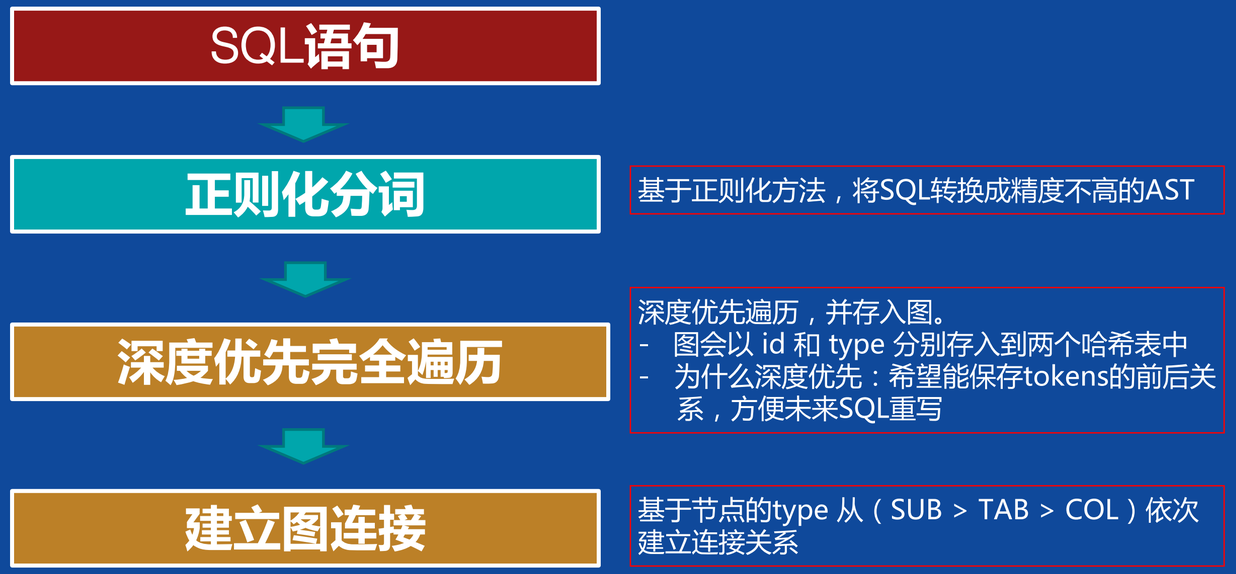
- 解析出SQL中列，表, 查询条件，子查询或视图的连接关系，把信息保存在有向图（direct graph）。
- 为了让机器能看懂SQL代码的结构。

```
select count(aty.id) from act_activity aty, act_audit_flow afl where afl.audit_flow_status='0'and (aty.status = 11 or aty.status=1) and aty.id = afl.activity_id
and afl.id in
    (
    select unique(ast.audit_flow_id) from act_audit_status ast where ast.audit_status_type = 0 and
        (
        ast.auditor = :P_auditor or ast.auditor in
            (
            select pri.mandatary from ACT_ACT_AUDIT_PRINCIPAL pri where pri.principal = :P_auditor and pri.type = 'COIN' and pri.is_valid = '1' and pri.start_at <= sysdate and pri.end_at >= sysdate
            )
        and ast.audit_type = 0
        )
    )
```



2.4 LUparser——陆金所SQL解析器

- Luparser的SQL解析过程，为了让机器能读懂SQL代码的结构。



2.5 基于贝叶斯的启发式搜索

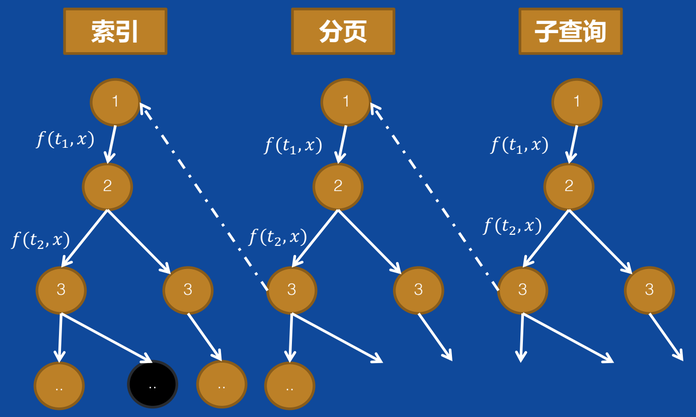
结合 历史评审结+
DBA review 经验

构建建议/问题状态，种类

搜索地图：状态库

索引	表链接	分页	子查询	…
有无索引	链接关系	排序	…	…
其他索引	有无视图	…	…	…
构建索引	联合查询	…	…	…
绑定变量	…	…	…	…

状态与状态之间基于启发函数做关联



t：当前搜索状态

执行计划

统计信息和数据字典

$$f(S,X)$$

基于贝叶斯的启发函数

- 状态（S）： python方法， 输出必须包含数值（计算启发值）
- 状态到状态：通过启发函数来选择。
- 搜索终止条件：当发现没有可选的状态，或候选状态为空

2.5 训练启发函数

$$f(S, X)$$

1. 对DBA建议做分词
2. 同义词转换，抓取关键字，应用成搜索地图
3. 在给定X下，计算联合概率和条件概率

1.基于历史评审意见

- ✖ 其他问题，请找 DBA 沟通
- ✖ 执行计划不稳定，需要修改SQL，添加 HINT
- ✖ update 语句未主动更新 updated_at
- ✖ update 语句 where 条件不固定，存在误更新的风险
- ✖ insert 语句未主动更新 created_at 或 updated_at
- ✖ select 语句动态拼接条件不明确，存在性能风险
- ✖ date 类型比较没有使用显式类型转换，存在性能风险
- ✖ 关联表太多，执行计划不稳定，存在性能风险
- ✖ 关联表不明确，执行计划不稳定，存在性能风险
- ✖ 标量自查询出现在 select 子句的列表中，存在性能风险
- ✖ 重要查询 条件使用 OR 关联，建议使用 UNION 进行查询拼接
- ✖ HINT 使用不恰当
- ✖ SELECT 列表不明确(select *)
- ✖ 动态拼接可能存在语法错误
- ✖ 分区表对应的SQL未带有 partition key
- ✖ iterate中不能使用 or 进行语句拼接
- ✖ 核心SQL需要添加HINT
- ✖ 后台使用的SQL没有带上be标识
- ✖ 此SQL为全动态拼接,不允许上线
- ✖ 索引选择性不高

真实dba建议

分词

映射成状态

索引	表链接	分页	子查询	...
有无索引	链接关系	排序	...	...
其他索引	有无视图	...	...	...
构建索引	联合查询	...	...	...
绑定变量	...	...	...	...

统计状态与状态之间的条件和联合概率

$$P(S_{t+1}, S_t, X)$$
$$P(S|X)$$

2.基于DBA经验对函数做调整

- 状态修正

e.g. 索引问题更倾向于查找区分度和直方图

2.6 陆金所上线后的效果

- 上线前对接代码库和发版平台，对审核不通过的SQL，先给出一轮优化建议供开发和DBA参考。
- 上线后对接监控平台，监控平台捕获慢查询后，调用AI SQL审核服务接口，自动给出优化建议。
- 同时AI明确优化建议优于CBO的执行计划才会给出建议反馈。

2.7 真实案例介绍

- 基于启发式搜索查找最优索引策略。

```
1              select
2                 id id,
3                 created_at createdAt,
4                 created_by createdBy,
5                 updated_at updatedAt,
6                 updated_by updatedBy,
7                 user_id userId,
8                 activity_id activityId,
9                 task_id taskId,
10                status status,
11                crm_user_id crmUserId,
12                SYNC_CHECK syncCheck,
13                REF_ID refId,
14                BATCH_ID batchId,
15                partition_key partitionKey
16             from act_task_result
17             where
18             task_id = :P_taskId
19             and id >= :P_startId
20             and id <= :P_endId
21             and partition_key in
22 /*iterate  property='partitionKeyList' close=')' conjunction=',' open='('*/
23  (
24             :P_partitionKeyList
25          ,
26             :P_partitionKeyList
27          ,
28             :P_partitionKeyList
29          )
30 -- /*isNotEmpty  property='status' prepend='and'*/  and
31 --             status = :P_status
```

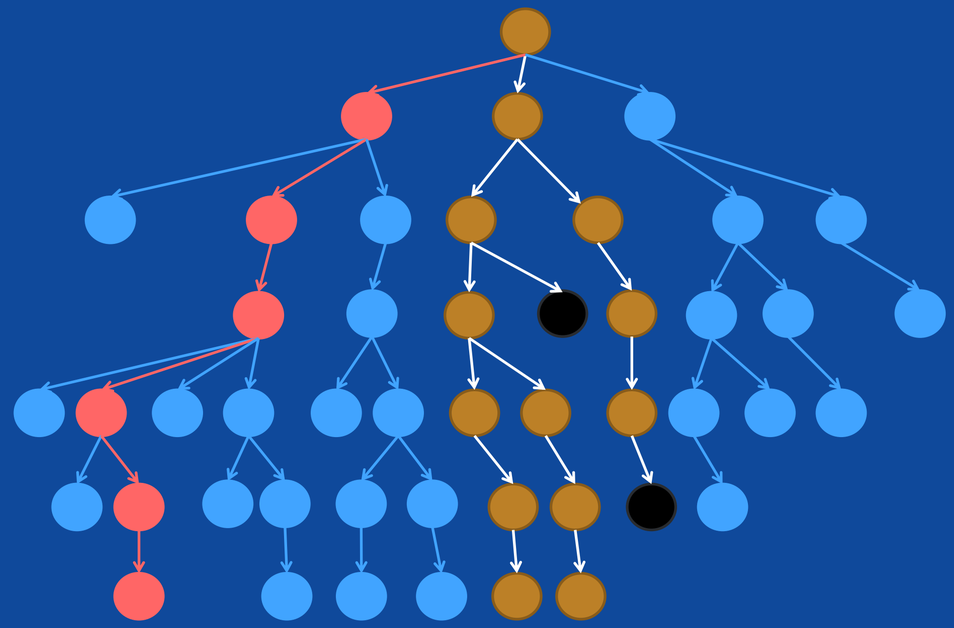
一. 项目背景

二. 研发过程

三. 未来展望

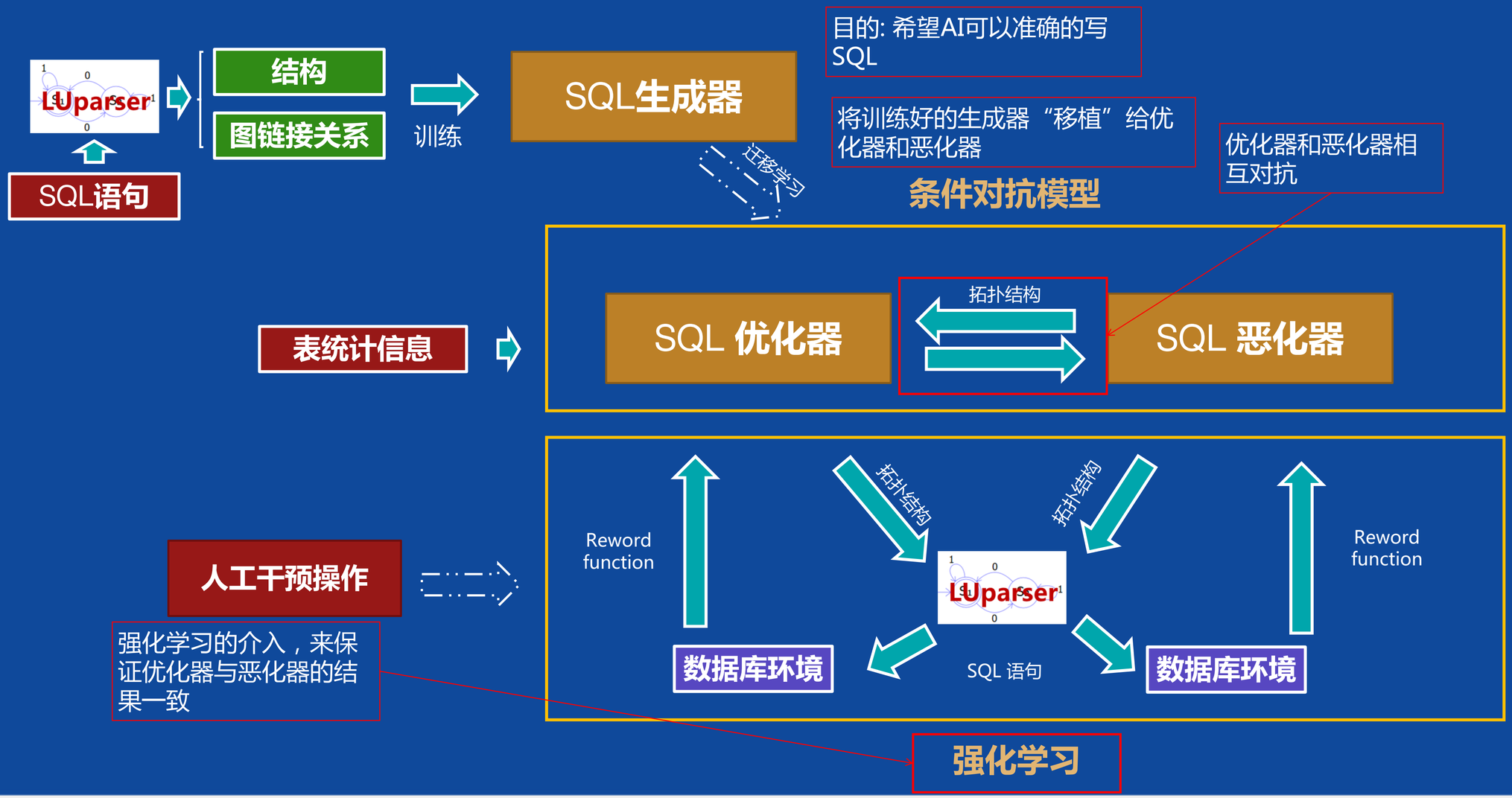
3.1 背景

- 贝叶斯派基于DBA经验，也受限于DBA经验。
- 能否通过强化学习让AI训练出突破现有DBA经验的优化方案。



如果全局最优解在DBA认知范围外，则贝叶斯派将永远无法找到该解

3.2 第三个版本——基于对抗模型的算法



Geekbang 极客邦科技 | InfoQ 极客传媒

InfoQ官网 全新改版上线

促进软件开发领域知识与创新的传播

关注InfoQ网站
第一时间浏览原创IT新闻资讯

免费下载迷你书
阅读一线开发者的技术干货

CHINA · BEIJING

THANKS

Global
Architect Summit