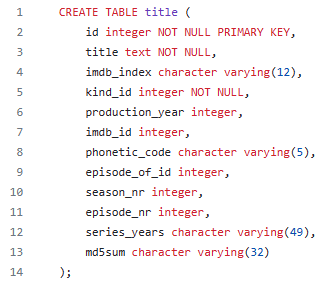
# 基于机器学习的数据库代价模型任务

## 任务描述

代价模型是数据库中一个重要的模块，主要用于为查询优化器生成的执行计划预测执行时间，进而选择执行时间最短的执行计划作为最终的执行计划。一般查询优化器会基于规则构建代价模型。但由于数据库的复杂性，代价模型有可能会偏离实际的执行情况。机器学习有很强的建模能力，考虑将其应用于代价估算，研究者尝试用一些基于学习的方法来实现更好的代价估算。其中一种基于学习的方法是采用机器学习方式来构建代价模型。本任务希望你利用机器学习模型实现准确的代价模型。

下面通过具体例子介绍本任务。如下是IMDB互联网电影资料库中一张名为title的数据表的模式（Schema）：



例如该表包含 production\_year (发行的年份)，kind\_id (电影种类类型)，imdb\_id (这部电影在IMDB中的id号)等描述电影的字段，这些字段在数据表内以列的形式展现。

查询语句如下:

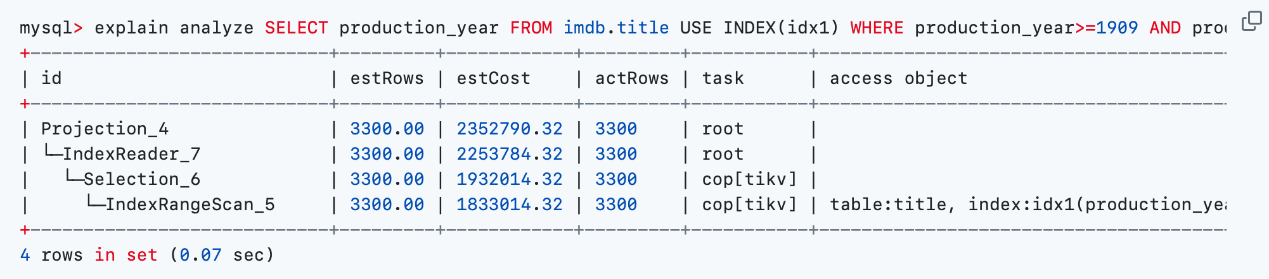
**SELECT production\_year**

**FROM imdb\_title USE INDEX(idx1)**

**WHERE production\_year >= 1909 AND production\_year <= 1909 AND kind\_id >= 0 AND kind\_id <= 6 AND season\_nr >= 0 AND season\_nr <= 57;**

注意到这个查询语句使用了USE Index(idx1), 代表指定使用的索引是 idx1。使用不同的索引也会导致不同的执行代价。

在数据库中可以使用 explain analyze 关键字对查询语句进行分析得到执行计划:



执行计划根据树状组织，每一行对应一个算子，主要的信息有:

id: 对应的算子名称，为了简单，在本任务的实验数据中只会出现 HashAgg, HashJoin, Sort, Selection, Projection, TableReader, TableScan (包含 TableRangeScan, TableFullScan 和 TableRowIDScan), IndexReader, IndexScan(包含 IndexRangeScan 和 IndexFullScan), IndexLookup 这些算子，你不需要掌握他们执行的逻辑，仅需当作黑盒进行建模即可，如若进一步想了解，可查看参考文献[1];

estRows: 优化器对该算子返回行数的估算值。为了简化问题，本实验中actRows等于estRows;

estCost: 优化器对该计划的代价估计值，也可称为估计代价;

execution: 实际的执行情况，包含执行时间，代价与执行时间呈正相关比关系;

不过请不要使用训练数据中的 execution info, memory, disk 和 actRows 字段的信息作为特征。如 execution\_info 列

图片包含 文本

描述已自动生成

文本, 表格

描述已自动生成Memory列与Disk列

actRows列

表格

描述已自动生成

因为

这些信息需要在实际执行计划后才能获得，不应该在估算计划代价时被使用。除此之外的特征，你可以任意使用作为你的特征。(这些形式的数据会在测试数据以及验证数据中出现，并且我们会提供解析的方法。为Plan.py代码)

## 补充知识

你需要从执行计划提取特征来训练机器学习模型，从执行计划提取特征的方式有如下两种：

1. 最简单的方式是从各个节点抽取特征然后求和，以这种方式我们可以得到如下特征：

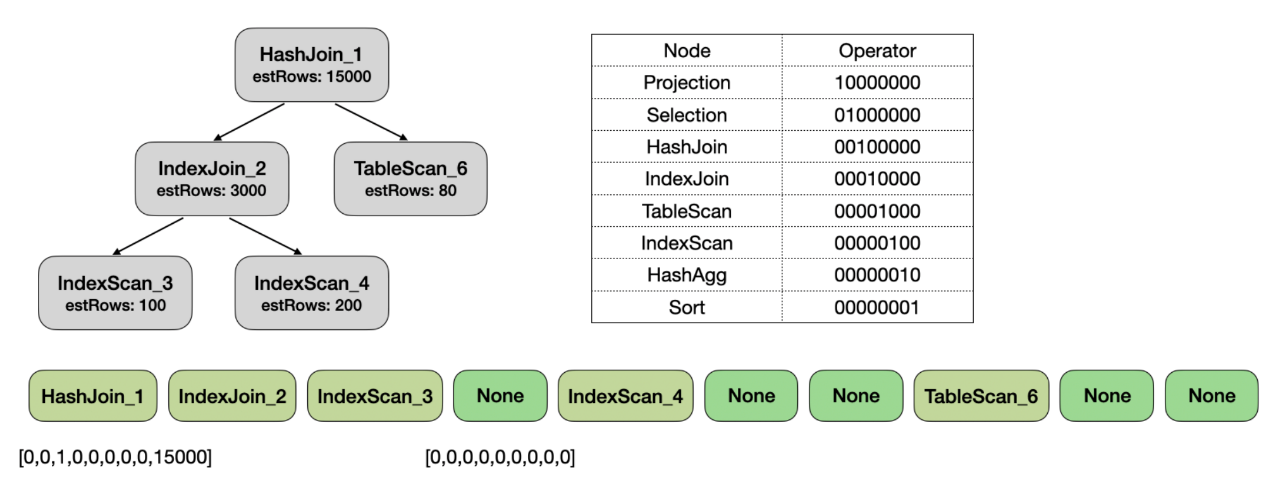
a. 节点总数；

b. 每种算子出现的次数；

c. 每种算子对应的 estRows 之和。

然而，这种抽取特征的方式将执行计划的树形结构丢弃了。

2. 第二种方式是从各个节点抽取特征以后，按照 DFS 前序遍历执行计划的顺序将各个节点的特征拼接起来。



NOTICE1: 在 DFS 前序遍历执行计划时，当我们遍历完该节点的所有孩子之后，我们会插入一个空节点表示所有孩子已经遍历完。这样执行计划和 DFS 前序遍历得到的序列构成了一对一的映射。

NOTICE2: 不同执行计划有不同数量的节点，因此得到的特征向量长度也不同。我们需要对特征向量做 padding，让它们的长度一致。

提取了特征以后，你便可以使用特征训练机器学习模型。准确地学习执行计划与实际基数之间的关系，便可使用该准确的机器学习模型作为代价估计模块。

## 数据集说明

**tests\_plans.json**

由如下该格式的若干数据组成，提取数据解析该plan的代码由我们提供:

{

"query": "SELECT production\_year FROM imdb.title USE INDEX(idx1) WHERE production\_year>=1909 AND production\_year<=1909 AND kind\_id>=0 AND kind\_id<=6 AND season\_nr>=0 AND season\_nr<=57",

"plan": [

"id\testRows\testCost\tactRows\ttask\taccess object\texecution info\toperator info\tmemory\tdisk",

"Projection\_4\t3300.00\t2352790.32\t3300\troot\t\ttime:9.72ms, loops:5, Concurrency:1\timdb.title.production\_year, row\_size: 8\t28.8 KB\tN/A",

"└─IndexReader\_7\t3300.00\t2253784.32\t3300\troot\t\ttime:9.55ms, loops:5, cop\_task: {num: 1, max: 9.28ms, proc\_keys: 3300, tot\_proc: 6ms, tot\_wait: 1ms, rpc\_num: 1, rpc\_time: 9.26ms, copr\_cache: disabled}\tindex:Selection\_6, row\_size: 48\t77.8 KB\tN/A",

" └─Selection\_6\t3300.00\t1932014.32\t3300\tcop[tikv]\t\ttikv\_task:{time:5ms, loops:8}, scan\_detail: {total\_process\_keys: 3300, total\_process\_keys\_size: 211200, total\_keys: 3301, rocksdb: {delete\_skipped\_count: 0, key\_skipped\_count: 3300, block: {cache\_hit\_count: 10, read\_count: 0, read\_byte: 0 Bytes}}}\tge(imdb.title.season\_nr, 0), le(imdb.title.season\_nr, 57), row\_size: 48\tN/A\tN/A",

" └─IndexRangeScan\_5\t3300.00\t1833014.32\t3300\tcop[tikv]\ttable:title, index:idx1(production\_year, kind\_id, season\_nr)\ttikv\_task:{time:5ms, loops:8}\trange:[1909 0,1909 6], keep order:false, row\_size: 48\tN/A\tN/A"

]

}

可以观察plan.py代码可知上述数据可被抽取为如下特征，id, est\_rows, act\_rows, estCost, exec\_info的含义 请阅读本文档第一节任务描述，请不要使用mem 与 disk, act\_rows 与 exec\_info，你的模型作为cost\_model时，可以认为还拿不到这些数据，但这些数据可供你阅读以对问题有更深的认识。

文本

描述已自动生成

**train\_plans.json**

由如下该格式的若干数据组成，作为模型评测使用:

{

"query": "SELECT production\_year FROM imdb.title USE INDEX(idx1) WHERE production\_year>=1909 AND production\_year<=1909 AND kind\_id>=0 AND kind\_id<=6 AND season\_nr>=0 AND season\_nr<=57",

"plan": [

"id\testRows\testCost\tactRows\ttask\taccess object\texecution info\toperator info\tmemory\tdisk",

"Projection\_4\t3300.00\t2352790.32\t3300\troot\t\ttime:10.1ms, loops:5, Concurrency:1\timdb.title.production\_year, row\_size: 8\t28.8 KB\tN/A",

"└─IndexReader\_7\t3300.00\t2253784.32\t3300\troot\t\ttime:9.97ms, loops:5, cop\_task: {num: 1, max: 9.79ms, proc\_keys: 3300, tot\_proc: 4ms, tot\_wait: 2ms, rpc\_num: 1, rpc\_time: 9.73ms, copr\_cache: disabled}\tindex:Selection\_6, row\_size: 48\t77.8 KB\tN/A",

" └─Selection\_6\t3300.00\t1932014.32\t3300\tcop[tikv]\t\ttikv\_task:{time:4ms, loops:8}, scan\_detail: {total\_process\_keys: 3300, total\_process\_keys\_size: 211200, total\_keys: 3301, rocksdb: {delete\_skipped\_count: 0, key\_skipped\_count: 3300, block: {cache\_hit\_count: 7, read\_count: 3, read\_byte: 191.8 KB}}}\tge(imdb.title.season\_nr, 0), le(imdb.title.season\_nr, 57), row\_size: 48\tN/A\tN/A",

" └─IndexRangeScan\_5\t3300.00\t1833014.32\t3300\tcop[tikv]\ttable:title, index:idx1(production\_year, kind\_id, season\_nr)\ttikv\_task:{time:4ms, loops:8}\trange:[1909 0,1909 6], keep order:false, row\_size: 48\tN/A\tN/A"

]

}

## 评测指标

由于执行数据内没有真实执行基数，仅有估计基数 est\_cost，因此将loss设计为预测基数语实际执行时间进行对齐的损失：

当然我们也提供evaluation.py对你的模型进行评估，评估使用预测基数与实际执行时间的相关性作为指标，较好的机器学习模型会让这两个指标呈现比较好的正相关性。

## 参考文献

[1] <https://docs.pingcap.com/zh/tidb/dev/explain-overview>