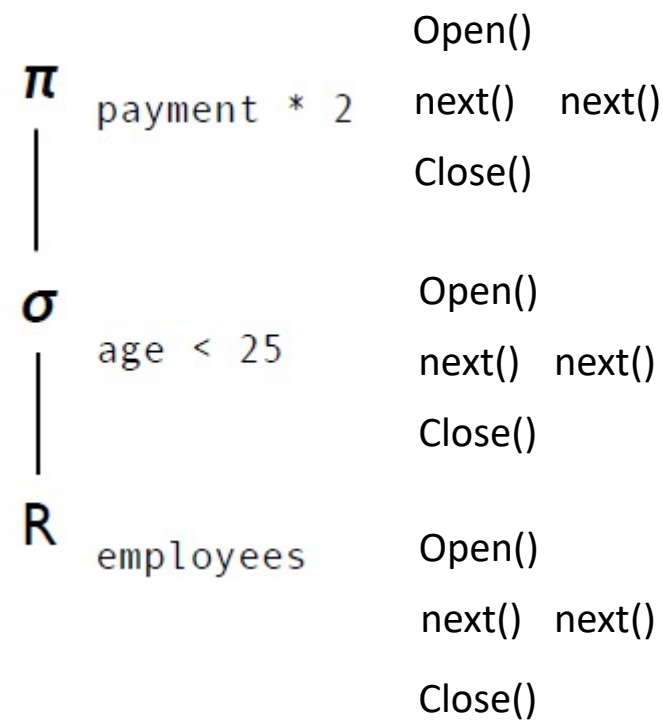
查询执行
Query Execution

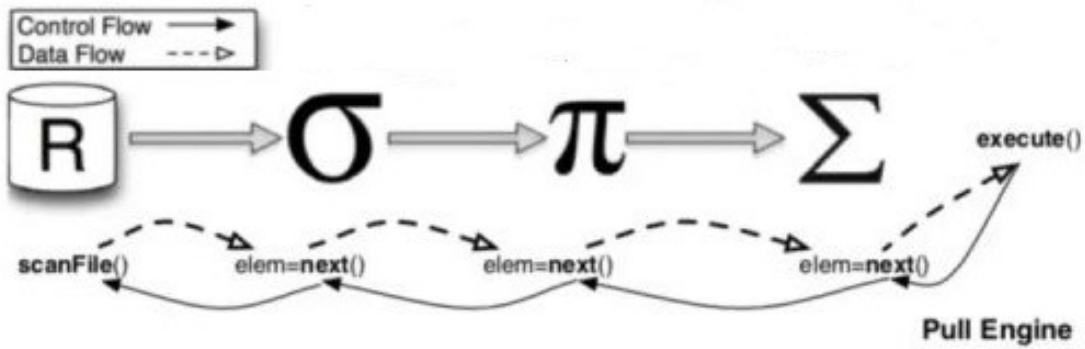
查询执行

- 火山模型
  - 物理计划是一颗树形结构
  - 操作流
    - 从树顶依次往孩子节点要数据，直到底层算子提供数据
  - 数据流
    - 从叶子依次往上层返回数据

Volcano Model

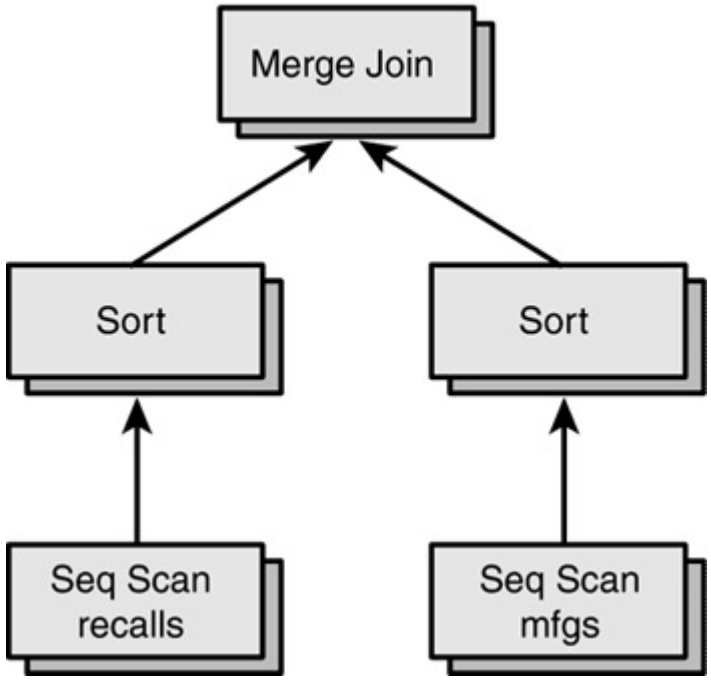
- 每个数据库操作都使用共同的接口
  - Open()：准备取第一条数据。
  - Next()：取下一条数据。
  - Close()：完成操作。
- 自上而下执行整个流水线






```
SELECT SUM(R.B)
FROM R
WHERE R.A < 10
```

Volcano Model on Join



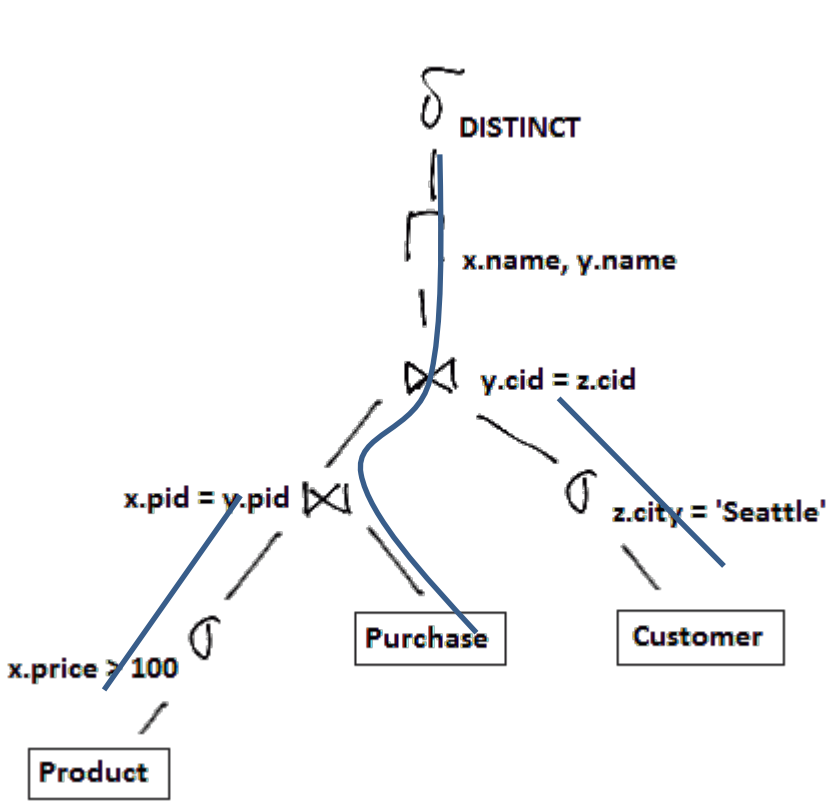
Merge Join- Next() : 不停取孩子中Sort的数据，如果左右相等，返回一个join结果
Sort-Open(): 将所有数据排序
Sort-Next(): 从排好序的数据中返回一行

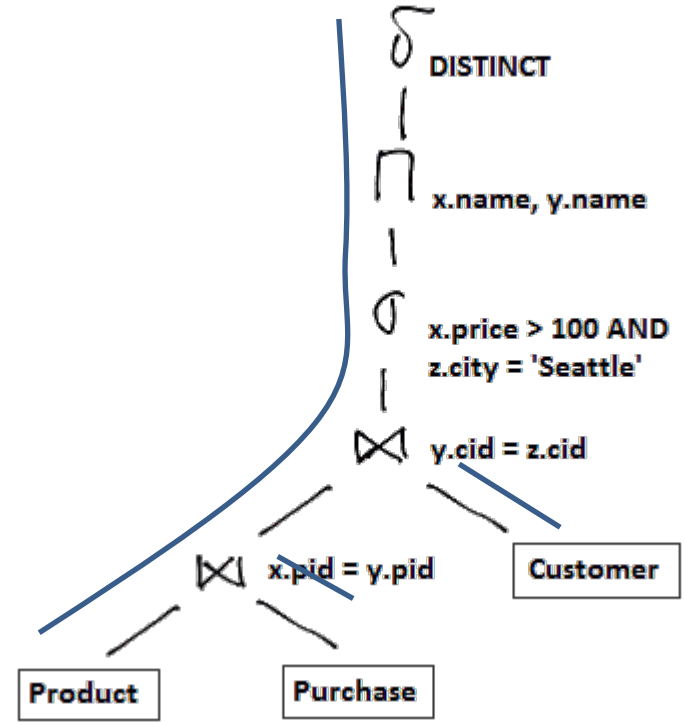
阻塞算子

- 典型
  - 构建哈希表(哈希连接)
  - 排序(合并连接)

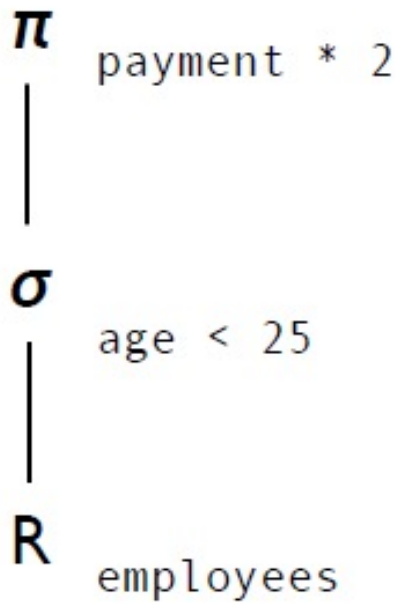
流水线的概念

Scan算子及阻塞算子之间的数据路径





批处理执行 vs 流水线执行

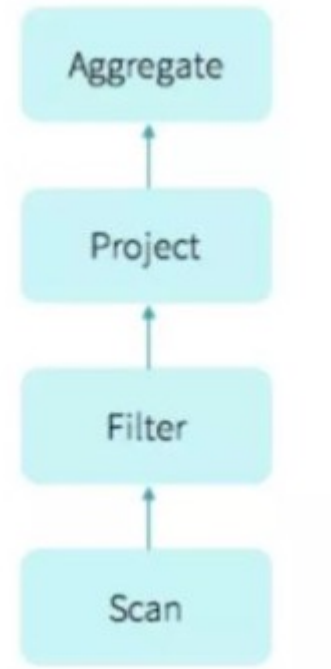


- 依次执行每个操作：将一个操作在所有数据上执行完之后再执行下一个操作。
- 在逐个数据上执行所有的操作。

火山模型的优缺点

◆优点：
- 实现简单易扩展
- 节省内存资源

◆缺点：
- 冗余的流控指令
- 效率低：虚函数嵌套，CPU的分支预测不友好



火山模型的优化

◆背景
- 介质：磁盘->内存
- 问题：IO墙->内存墙

◆优化
- 操作符融合
- RowSet迭代
- 推送模型/局部批处理?
- 编译执行

影响性能的因素：数据访问

Numbers Everyone Should Know

L1 cache reference	0.5 ns
Branch mispredict	5 ns
L2 cache reference	7 ns
Mutex lock/unlock	100 ns
Main memory reference	100 ns
Compress 1K bytes with Zippy	10,000 ns
Send 2K bytes over 1 Gbps network	20,000 ns
Read 1 MB sequentially from memory	250,000 ns
Round trip within same datacenter	500,000 ns
Disk seek	10,000,000 ns
Read 1 MB sequentially from network	10,000,000 ns
Read 1 MB sequentially from disk	30,000,000 ns
Send packet CA->Netherlands->CA	150,000,000 ns

Google

流水线在内存数据库中的优化

vectorwise

"MonetDB/X100: Hyper-Pipelining Query Execution " Boncz, Zukowski, Nes, CIDR'05

Memory Hierarchy