QUERY OPTIMIZATION

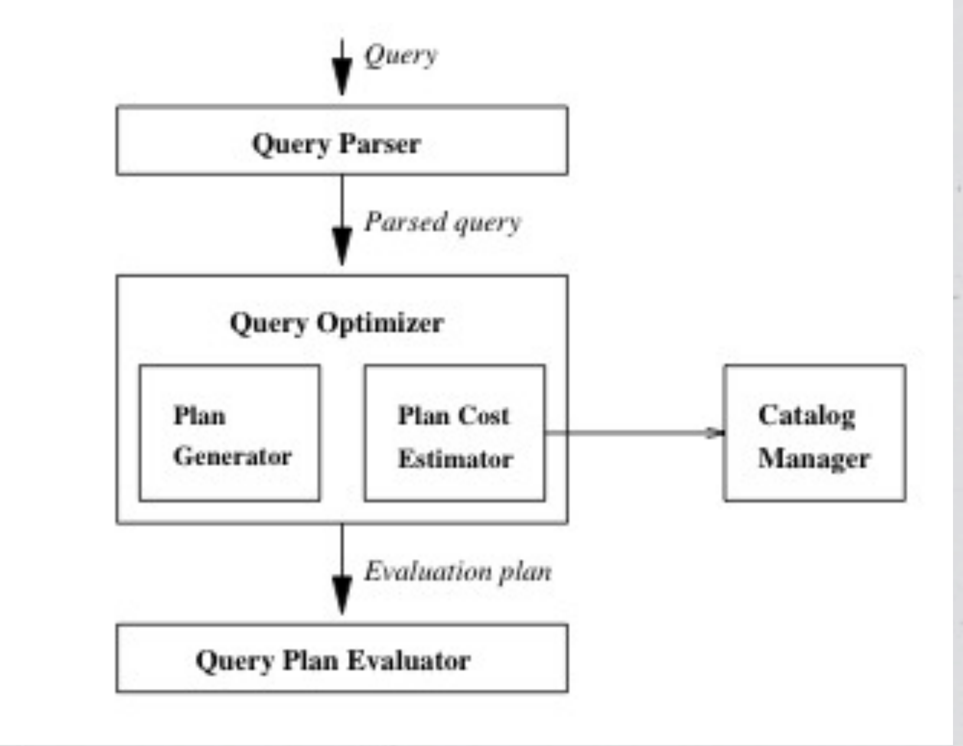
Vaibhav Bajpai

Contents

- Overview
- Heuristic-based Optimization
- Cost-based Optimization
- Example

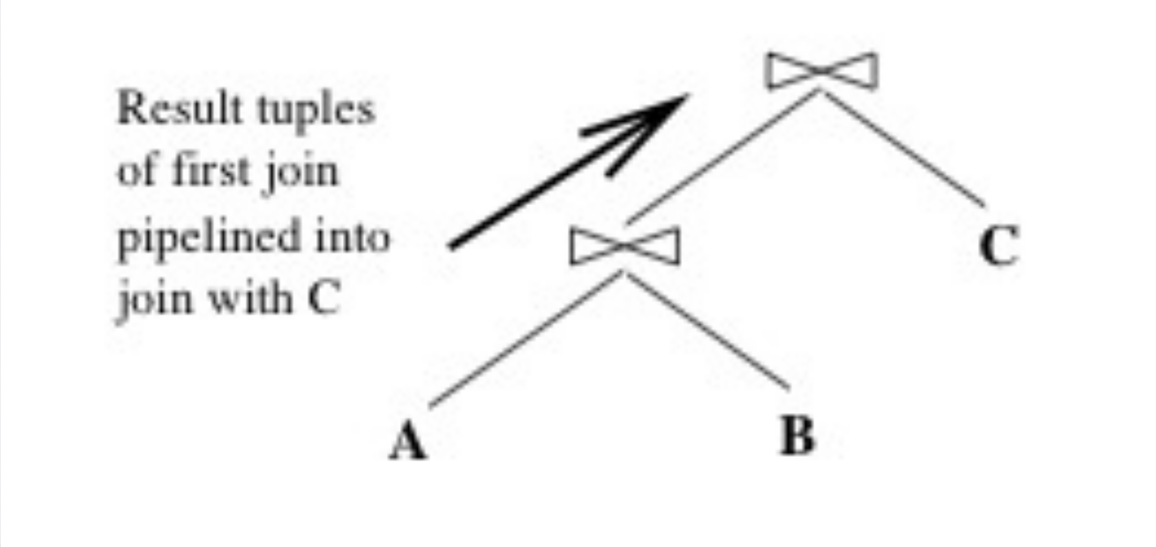
Overview

- Optimization
  - find a better equivalent plan
  - ideally want to find a best plan
  - practically want to avoid worst plans



Overview

- Blocking Operator
  - Materialization
- Non-Blocking Operator
  - Pipelining



Heuristic-based Approach

(Logical-Level Optimization)

- applying algebraic-equivalences …
- though cannot determine the **actual** costs …
- allows to **push** selection and projection ahead of joins …

Heuristic-based Approach

Equivalences

- Selection

$$\pi_a(\sigma_c(R)) \;\equiv\; \sigma_c(\pi_a(R))$$

$$R \bowtie_c S \;\equiv\; \sigma_c(R \times S)$$

$$\sigma_c(R \times S) \;\equiv\; \sigma_c(R) \times S$$
$$\sigma_c(R \bowtie S) \;\equiv\; \sigma_c(R) \bowtie S$$

$$\sigma_c(R \times S) \;\equiv\; \sigma_{c_1 \wedge c_2 \wedge c_3}(R \times S)$$

$$\sigma_{c_1}(\sigma_{c_2}(\sigma_{c_3}(R \times S)))$$

$$\sigma_{c_1}(\sigma_{c_2}(R) \times \sigma_{c_3}(S)).$$

- Projection

$$\pi_a(R \times S) \;\equiv\; \pi_{a_1}(R) \times \pi_{a_2}(S)$$

$$\pi_a(R \bowtie_c S) \;\equiv\; \pi_{a_1}(R) \bowtie_c \pi_{a_2}(S)$$

Cost-based Approach

(Physical-Level Optimization)

- input: logical-plan + physical situation + histograms
- output: efficient physical plan
- approach: **enumerate** all physical plans and pick the best one

Cost-based Approach

System R Optimizer

- works well with less than **10 joins**
- considers both combination of **CPU and IO** costs
- **cartesian products** are avoided
- **only left deep plans** are considered

Cost-based Approach

Query Blocks: Unit of Optimization

- A query is parsed into collection of query blocks
- A query evaluation plan is chosen for each block!
- A query block contains **no nesting** -
  - exactly **one** SELECT and **one** FROM
  - atmost **one** WHERE, GROUP BY, HAVING

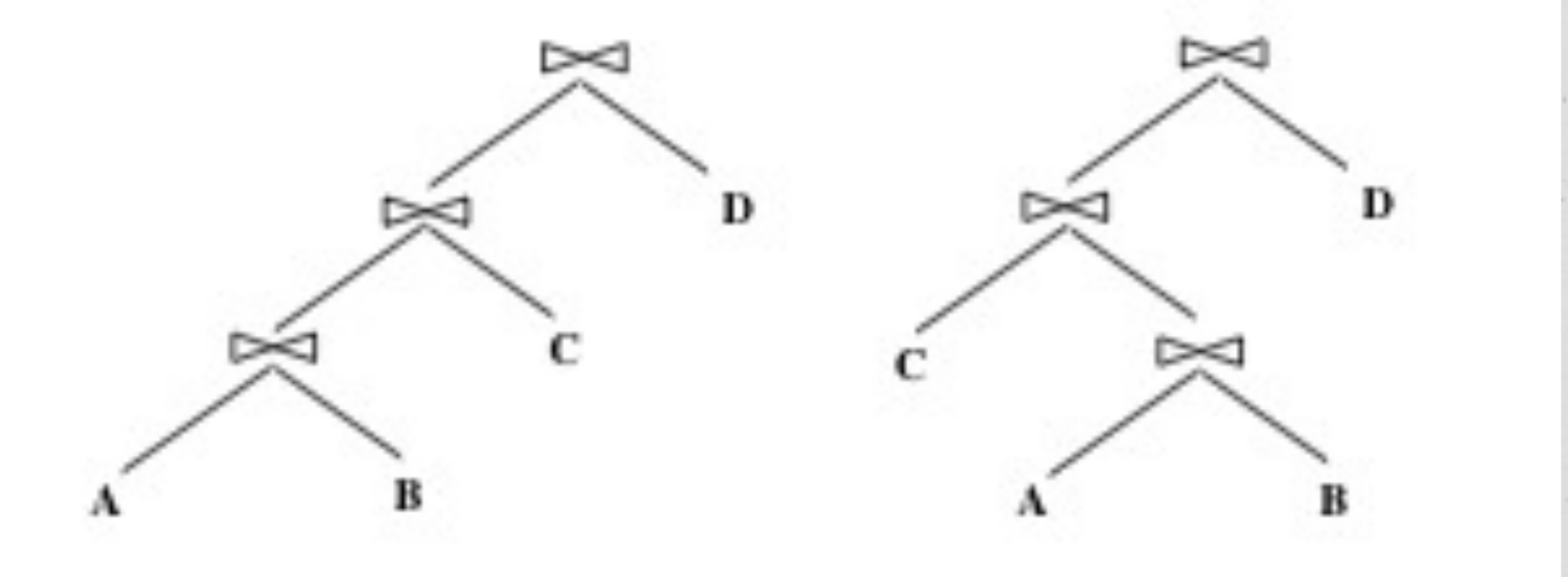
Cost-based Approach

Enumeration of Alternative Plans

- Single Relation Plans
  - Clustered Index (NPages(X)+NPages(Y)) * RF
  - Unclustered Index (NPages(X)+NTuples(Y)) * RF
- Multiple Relation Plans
  - No. of plans increase exponentially with increase in joins.
  - Need to **restrict search space! (ONLY left-deep join!)**

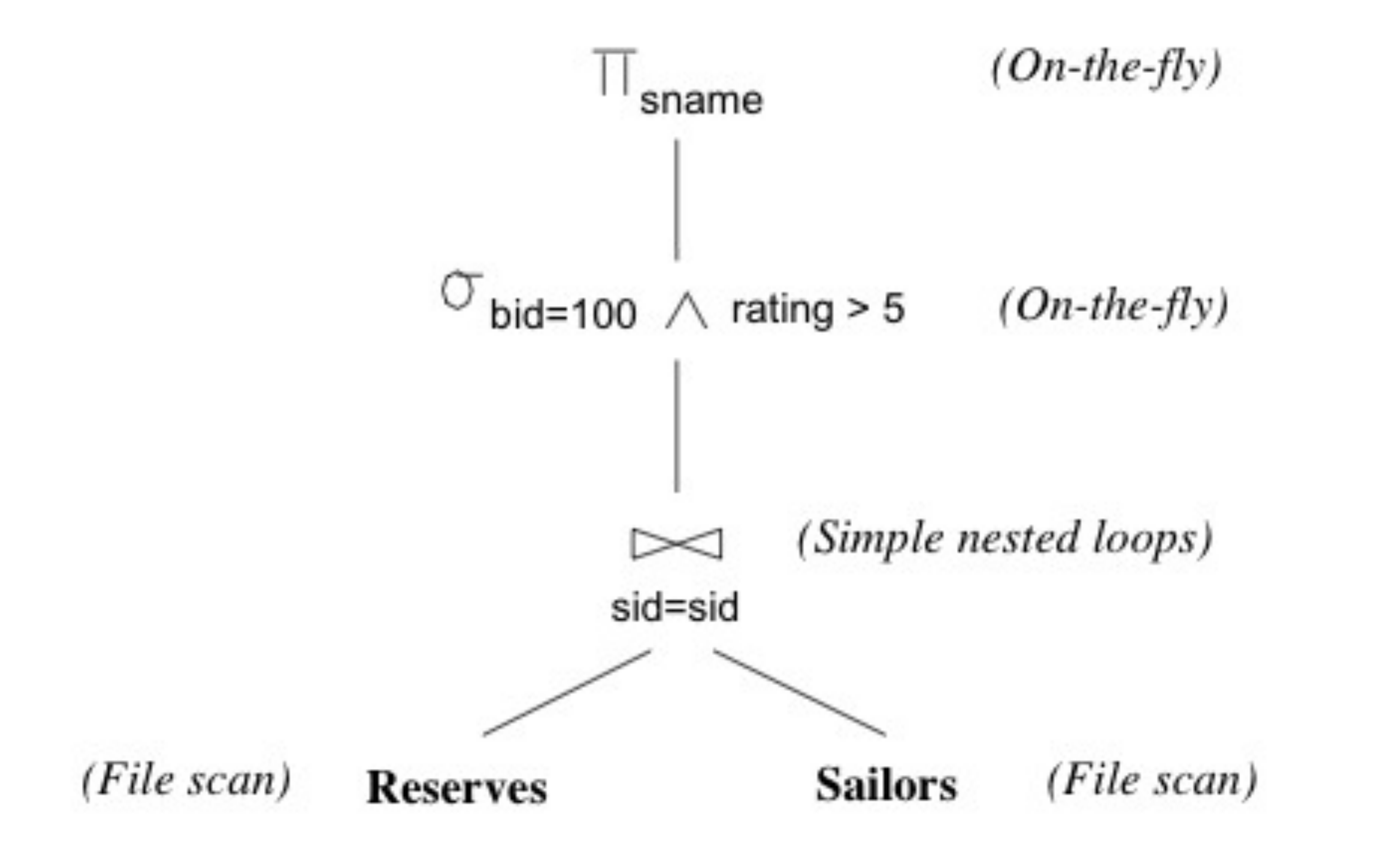
Cost-based Approach
Enumeration of Alternative Plans

Left Deep Joins



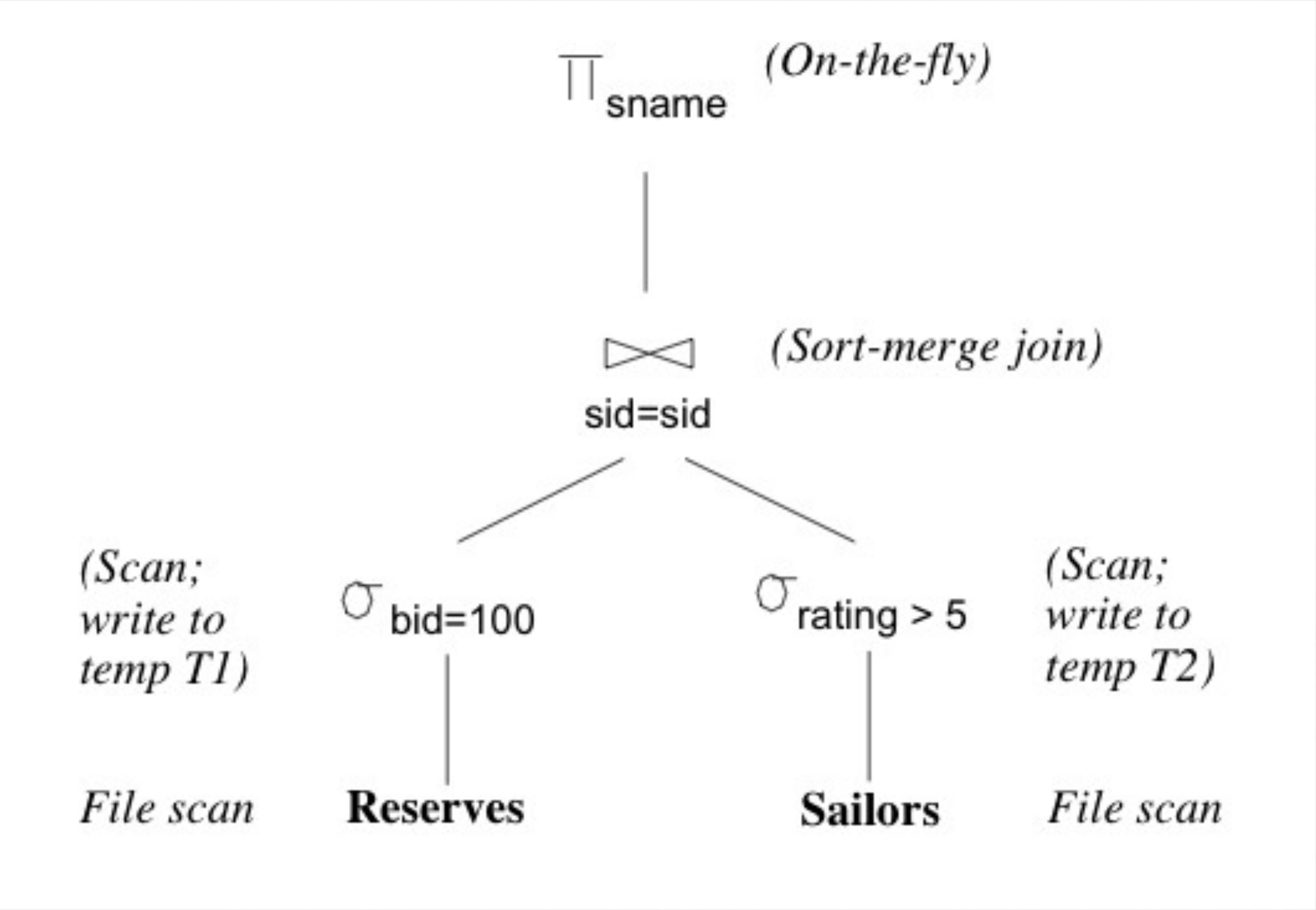
- Why?
  - generates **fully pipelined plans!**
  - many join algorithms scan the right relation multiple times

Example!

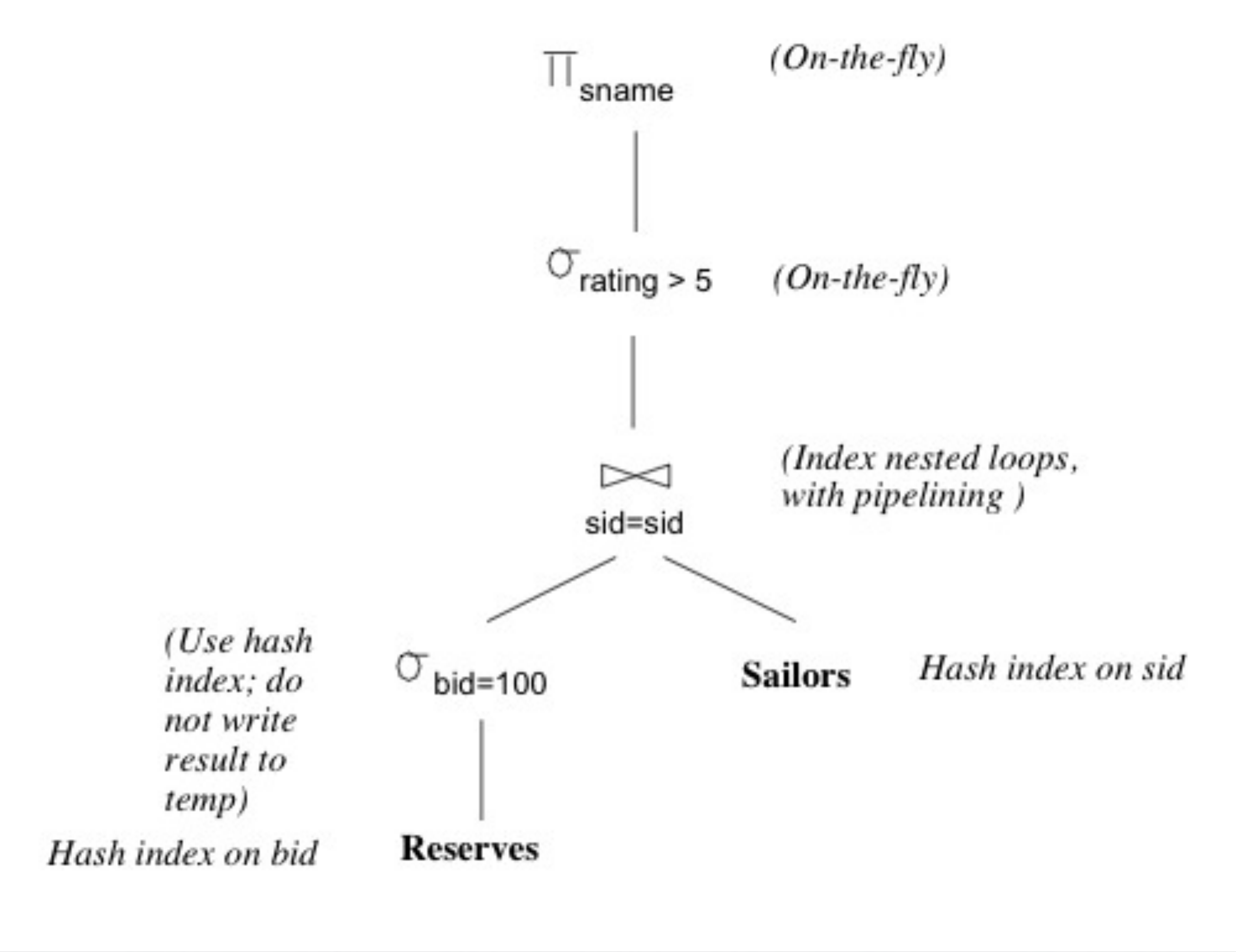


π_{sname}
(On-the-fly)
$\sigma_{bid=100 \wedge rating > 5}$
(On-the-fly)
$\bowtie_{sid=sid}$
(Simple nested loops)
(File scan)
Reserves
Sailors
(File scan)

Example!


Π sname
(On-the-fly)
⋈ sid=sid
(Sort-merge join)
(Scan; write to temp T1)
σ bid=100
σ rating > 5
(Scan; write to temp T2)
File scan
Reserves
Sailors
File scan

Example!


π_{sname}
(On-the-fly)
$\sigma_{rating > 5}$
(On-the-fly)
$\bowtie_{sid=sid}$
(Index nested loops, with pipelining)
(Use hash index; do not write result to temp)
$\sigma_{bid=100}$
Sailors
Hash index on sid
Hash index on bid
Reserves