Joins (1)

Physical Join Operators
Block Nested Loops
Merge Join(s)

Objectives:

- Introduce three primary join families
  - Block nested loops
  - Merge Join(s)
  - Hash Join(s)
- Examples, first two

Three Primary Classes of Join Algorithm

1. Block-Nested Loops (vs. Simple Nested Loops.)
2. Merge-Join
3. Hash-Joins

Simple Nested Loops

- R join S

For each r ∈ R
  For each s ∈ S
    if pred(r,s) then output result

// ignores that data is on disk, read in blocks

Block-Nest Loops - Assumptions

- Dedicate Most Memory to Outer-Loop
- Consider that there are many tuples per page.

Block Nested-Loops Detailed

```
Join(relation R, relation S, buffer_space M){ // M in units of disk pages
    Until done( table_scan(S,M-1)){          // Read M-1 pages of S
        if equijoin then build index on portion of S in memory
        Open(R);
        Until done(table_scan(R, 1){
            for each tuple r of R in memory{
                "find joining tuples of S"  // if equijoin use index
                "output them"               // if thetajoin another loop
                    }}}}
```


What to Notice:

1. Outer-loop iterates over "inner" relation
2. If equi-join, loop structure nested 3 deep
3. If theta-join, loop structure nested 4 deep
4. Call structure
   - not parameterized by theta/equi
   - theta predicate not passed in

In Your Text, Notice:

- Implementation of operators
  →Use an abstract iterator method
  - Open()
  - Next()
  - Close()
- Hide I/O by abstracting iterators.
  – IT WORKS
- They too ignore a lot of the arguments in the call structure.

Cost Model - Notation

- B(R) , number of blocks in R
- T(R) , number of tuples in R
- T(R)/B(R), an estimate of the number of tuples per block.

Number of I/O's (cost) of Block-Nested Loops

Run through the complete inner relation, S, for each chunk of R

If S fits in M-1 buffers → B(R) + B(S)

If S does not fit

→ B(S)/M-1 iterations

I/O Cost = (B(S)/M-1) * (M-1 + B(R))

What About Writing the Result

- Book convention –
  - no cost in the operator for the result
    - in some cases, results are immediate input to another operator.
      - no I/O cost for output
    - introduce an explicit new operator
      - "Save intermediate result to disk"