IX. Query Processing

Lecture Topics

- Query Interpretation
- Basic Operations
- Costs of Basic Operations

Steps in Query Interpretation

1. Translation
   - check SQL syntax
   - check existence of relations and attributes
   - replace views by their definitions
   - transform query into an internal form (similar to relational algebra)
2. Optimization
   - generate alternative **access plans**, i.e., procedure, for processing the query
   - select an efficient access plan
3. Processing
   - execute the access plan
4. Data Delivery

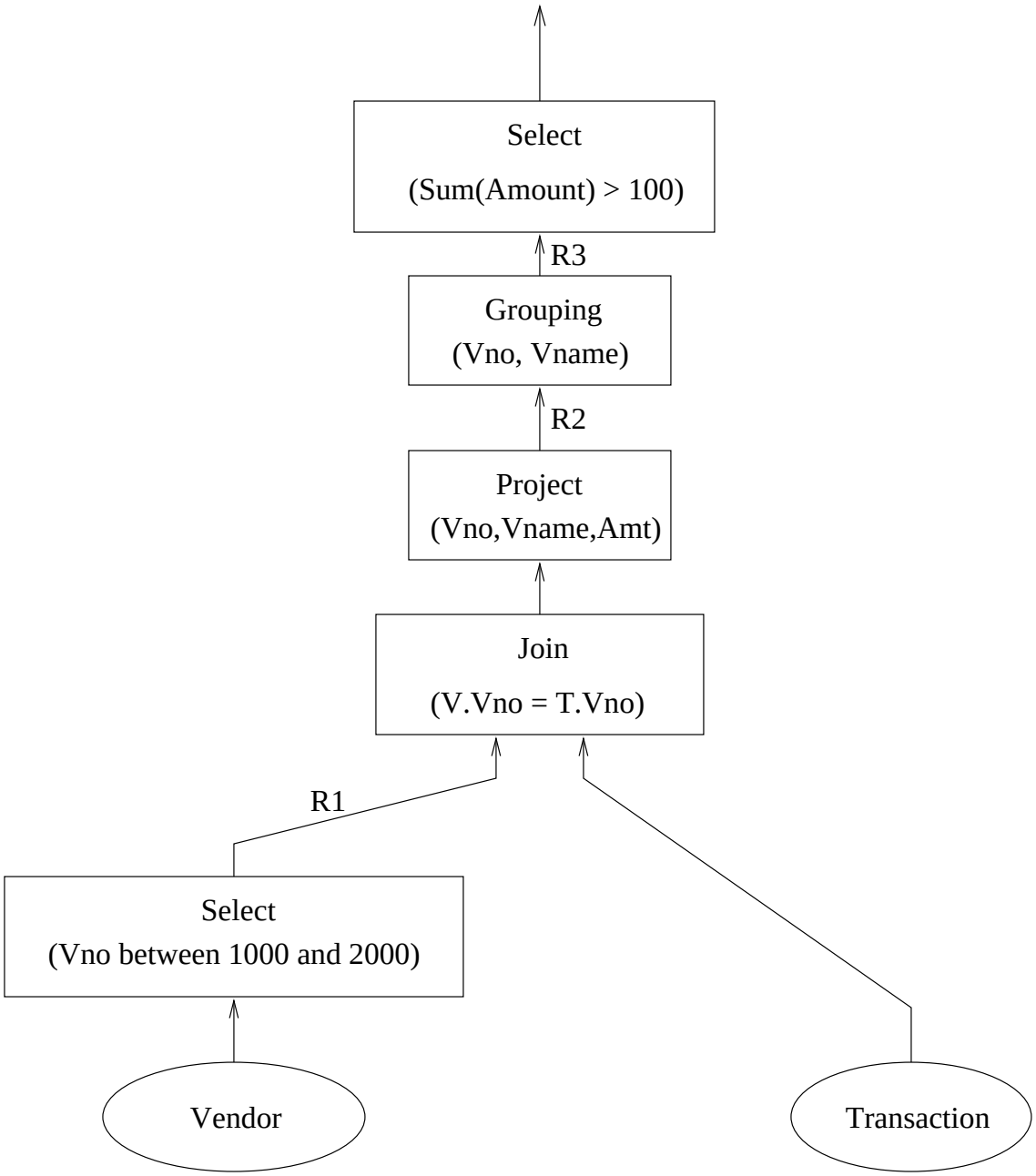
Example

select V.Vno, Vname, **count**(*), **sum**(Amount)
from Vendor V, Transaction T
where V.Vno = T.Vno **and** V.Vno **between** 1000 and 2000
group by V.Vno, Vname
having sum(Amount) $>$ 100

The following is a possible access plan for this query:

1. Scan the Vendor table, select all tuples where Vno is between 1000 and 2000 and place the result in a temporary relation R_1
2. Join the tables R_1 and Transaction, eliminate attributes other than Vno, Vname, and Amount, and place the result in a temporary relation R_2. This may involve:
   - sorting R_1 on Vno
   - sorting Transaction on Vno
   - merging the two sorted relations to produce R_2
3. Perform grouping on R_2, and place the result in a temporary relation R_3. This may involve:
   - sorting R_2 on Vno and Vname
   - group tuples with identical values of Vno and Vname
   - count the number of tuples in each group, and add their Amounts
4. Scan R_3, select all tuples with **sum**(Amount) $>$ 100 to produce the result.

Pictorial Access Plan



Some Basic Query Processing Operations

- Selection
  - scan without an index
  - scan with an index
- Projection
- Joining
  - nested loop join
  - hash join
  - sort-merge join
  - and others . . .
- Grouping and Duplicate Elimination
  - by sorting
  - by hashing
- Sorting

Tuple Selection

Sequential Scanning

When no appropriate index is available, we must check all file blocks holding tuples for the relation.

Even if an index is available, scanning the entire relation may be faster in certain circumstances:

- the relation is very small
- the relation is large, but we expect most of the tuples in the relation to satisfy the selection criteria

For example:

select Anum
from Accounts
where Balance $>$ 1.00

Joining Relations

select C.Cnum, Balance
from Customer C, Accounts A
where C.Cnum = A.Cnum

Two relations can be joined using the "nested loop" join method. Conceptually, that method works like this:

```
foreach tuple c in Customer do
        foreach tuple a in Accounts do
                if c.Cnum = a.Cnum then
                        output c.Cnum, a.Balance
        end
end
```

Other Techniques for Join

There are many other ways to perform a join. For example, if there is an index on the Cnum attribute of the Accounts relation, an **index join** could be used. Conceptually, that would work as follows:

foreach tuple c in Customer do
    use the index to find Accounts tuples a where a.Cnum $=$ c.Cnum
    for each such tuple a
        output c.Cnum, a.Balance
end

Examples of other join techniques:

- **Sort-Merge Join**: sort the tuples of Accounts and of Customer on their Cnum values, then merge the sorted relations.
- **Hash Join**: assign each tuple of Accounts and of Customer to a "bucket" by applying a hash function its Cnum value. Within each bucket, look for Account and Customer tuples that have matching Cnum values.

Costs of Basic Operations

Alternative access plans may be compared according to cost.

The cost of an access plan is the sum of the costs of its component operations.

There are many possible cost metrics. However, most metrics reflect the amounts of system resources consumed by the access plan. System resources may include:

- disk block I/O's
- processing time
- network bandwidth

A Simple Cost Model

We will use a simple cost model for examples.

The cost of an operation is will be an estimate of the number of file block retrievals required to perform that operation.

The number of file block retrievals depends on the number of input tuples and on the tuple blocking factor.

The blocking factor is the number of tuples that are stored in each file block.

For a relation R:

- $|R|$ will represent the number of tuples in R
- $b(R)$ will represent the blocking factor for R
- $|R|/b(R)$ is the number of blocks used to store R

We will assume that an index search has a cost of 2 plus the cost of retrieving the data blocks themselves.

Cost of Selection

Mark(Studnum, Course, Assignnum, Mark)

select Studnum, Mark
from Mark
where Course = PHYS **and** Studnum = 100 **and** Mark $>$ 90

Indices:

- clustering index CourseInd on Course
- non-clustering index StudnumInd on Studnum

Assume:

- $|\mathsf{Mark}| = 10000$
- $b(\mathsf{Mark}) = 50$
- 500 different students
- 100 different courses
- 100 different marks

Selection Strategy: Use CourseInd

Assuming *uniform distribution* of tuples over the courses, there will be about $|\mathsf{Mark}|/100 = 100$ tuples with Course = PHYS.

Searching the CourseInd index has a cost of 2. Retrieval of the 100 matching tuples adds a cost of $100/b(\mathsf{Mark})$ data blocks, for a total cost of 4.

Selection of N tuples from relation R using a clustered index has a cost of $2 + N/b(R)$

Selection Strategy: Use StudnumInd

Assuming *uniform distribution* of tuples over student numbers, there will be about $|\mathsf{Mark}|/500 = 20$ tuples for each student.

Searching the StudnumInd has a cost of 2. Since this is not a clustered index, we will make the pessimistic assumption that each matching record is on a separate data block, i.e., 20 blocks will need to be read. The total cost is 22.

Selection of N tuples from relation R using a clustered index has a cost of $2 + N$

Selection Strategy: Scan the Relation

The relation occupies 10,000/50 = 200 blocks, so 200 block I/O operations will be required.

Selection of N tuples from relation R by scanning the entire relation has a cost of $\|R\|/b(R)$

Cost of Joining

Mark(Studnum, Course, Assignnum, Mark)
Student(Studnum, Surname, Initials, Address, Birthdate, Sex)

select Surname, Course, Assignnum, Mark
from Mark natural join Student

Assume:

- $|\mathsf{Mark}| = 10,000$
- $b(\mathsf{Mark}) = 50$
- $|\mathsf{Student}| = 1,000$
- $b(\mathsf{Student}) = 20$

Cost of Nested-Loop Join

for each Mark block M **do**
  for each Student block S **do**
    for each tuple m in M **do**
      for each tuple s in S **do**
        if m.Studnum = s.Studnum **then**
          join m and s and add to the result

Each of the 200 blocks of Mark must be retrieved once.

Each of the 50 blocks of Student must be retrieved once for each block of Mark, for a total of 200x50 = 10,000 blocks retrieved.

The total cost is 10,200 block retrievals.

The cost of a nested-loop join of R_1 and R_2 is either:

$$\frac{|R_1|}{b(R_1)} + \frac{|R_1|}{b(R_1)}\frac{|R_2|}{b(R_2)}$$

or

$$\frac{|R_2|}{b(R_2)} + \frac{|R_1|}{b(R_1)}\frac{|R_2|}{b(R_2)}$$

depending on which relation is the **outer** relation.

Cost of Index Join

Suppose there is a unique, non-clustered index on Student.Studnum.

for each Mark block M **do**
    for each tuple m in M **do**
        use the index to locate Student tuples s with s.Studnum $=$ m.Studnum
        join m and s and add to the result

Each of Mark's 200 blocks must be retrieved once. For each tuple of Mark, a single tuple of Student will be retrieved. This will require an index lookup and a data block retrieval, for a cost of 3.

The total cost is 10000*3 + 200 = 30200.

The cost of a unique index join, where R_1 is the outer relation, and R_2 is the inner, indexed relation, is:

$$\frac{|R_1|}{b(R_1)} + 3R_1$$

Summary

An **access plan** is a detailed plan for the execution of a query. It describes:

- the sequence of basic operations (select, project, join, sort, etc.) used to process the query
- how each operation will be implemented, e.g., which join method will be used, which indices will be used to perform a selection.

Each operation may be assigned a cost, and an access plan may be assigned a cost by summing the costs of its component operations.

The cost assigned to an operation depends on assumptions about the data in the relations. These assumptions will be discussed further in the next section of the notes.