Query Processing

Dr. Jyotismita Chaki

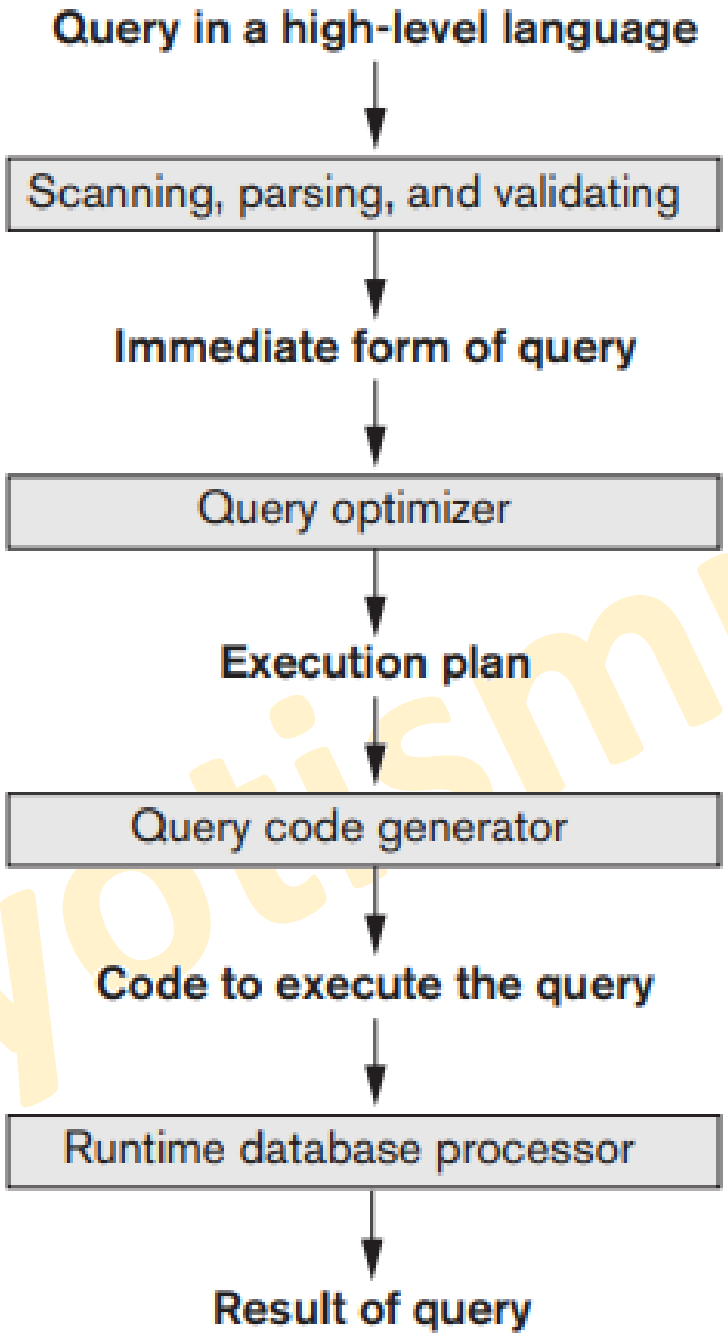
Query processing

- A query expressed in a high-level query language such as SQL must first be **scanned, parsed, and validated**.
- The **scanner** identifies the query tokens—such as SQL keywords, attribute names, and relation names—that appear in the text of the query, whereas the **parser** checks the query syntax to determine whether it is formulated according to the syntax rules (rules of grammar) of the query language.
- The query must also be validated by checking that all attribute and relation names are valid and semantically meaningful names in the schema of the particular database being queried.

Query processing

- An internal representation of the query is then created, usually as a tree data structure called a query tree.
- It is also possible to represent the query using a graph data structure called a query graph, which is generally a directed acyclic graph (DAG).
- A query has many possible execution strategies, and the process of choosing a suitable one for processing a query is known as query optimization.

Query processing



Code can be:

Executed directly (interpreted mode)

Stored and executed later whenever needed (compiled mode)

Relational Algebra

- The basic set of operations for the formal relational model.
- These operations enable a user to specify basic retrieval requests as relational algebra expressions.
- The result of a retrieval query is a new relation.
- A sequence of relational algebra operations forms a **relational algebra expression**, whose result will also be a relation that represents the result of a database query.
- Importance:
  - Provides a formal foundation for relational model operations.
  - Used as a basis for implementing and optimizing queries in the query processing and optimization modules that are integral parts of RDBMS.
  - Some of its concepts are incorporated into the SQL standard query language for RDBMS

Relational Algebra: Unary Relational Operations: SELECT

- The SELECT operator is unary; that is, it is applied to a single relation.
- Used to choose a subset of the tuples from a relation that satisfies a selection condition: selects some rows
- A filter that keeps only those tuples that satisfy a qualifying condition.
- Can also be visualized as a horizontal partition of the relation into two sets of tuples
- Denoted by: $\boldsymbol{\sigma_{<\text{select condition}>}(R)}$ [σ (sigma): SELECT operator, selection condition: Boolean expression (condition) specified on the attributes of relation R, R: relational algebra expression whose result is a relation]
- For example, to select the EMPLOYEE tuples whose department is 4: $\boldsymbol{\sigma_{Dno = 4}(EMPLOYEE)}$
- $|\sigma_c (R)| \leq |R|$ for any condition C

Relational Algebra: Unary Relational Operations: SELECT

- The Boolean expression specified in is made up of a number of clauses of the form:
  - <attribute name> < comparison op> <constant value>
  - <attribute name> < comparison op> < attribute name>
  - where is the name of an attribute of R, is normally one of the operators {=, <, ≤, >, ≥, ≠}, and is a constant value from the attribute domain.
- For example, to select the tuples for all employees who either work in department 4 and make over $25,000 per year, or work in department 5 and make over $30,000, we can specify the following SELECT operation: $\sigma_{(Dno=4\ AND\ Salary>25000)\ OR\ (Dno=5\ AND\ Salary>30000)}(EMPLOYEE)$

Relational Algebra: Unary Relational Operations: SELECT

- {=, <, ≤, >, ≥, ≠} can apply to attributes whose domains are ordered values: numeric or date domains
- If the domain of an attribute is a set of unordered values, then only the comparison operators in the set {=, ≠} can be used.
- An example of an unordered domain is the domain Color = { ‘red’, ‘blue’, ‘green’, ‘white’, ‘yellow’, …}, where no order is specified among the various colors.
- The Boolean conditions AND, OR, and NOT have their normal interpretation, as follows:
  - (cond1 AND cond2) is TRUE if both (cond1) and (cond2) are TRUE; otherwise, it is FALSE.
  - (cond1 OR cond2) is TRUE if either (cond1) or (cond2) or both are TRUE; otherwise, it is FALSE.
  - (NOT cond) is TRUE if cond is FALSE; otherwise, it is FALSE.

Relational Algebra: Unary Relational Operations: SELECT

- The fraction of tuples selected by a selection condition is referred to as the **selectivity** of the condition.
- SELECT operation is commutative
  - $\sigma_{<cond1>}(\sigma_{<cond2>}(R)) = \sigma_{<cond2>}(\sigma_{<cond1>}(R))$
- We can always combine a cascade (or sequence) of SELECT operations into a single SELECT operation with a conjunctive (AND) condition
  - $\sigma_{<cond1>}(\sigma_{<cond2>}(\ldots(\sigma_{<condn>}(R))\ldots)) = \sigma_{<cond1>}$ AND $\sigma_{<cond2>}$ AND...AND $\sigma_{<condn>}(R)$
- $\sigma_{(Dno=4 \text{ AND } Salary>25000) \text{ OR } (Dno=5 \text{ AND } Salary>30000)}$ (EMPLOYEE)

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5

Relational Algebra: Unary Relational Operations: PROJECT

- Selects certain columns (or attributes) from the table and discards the other columns.
- The result can be visualized as a vertical partition of the relation into two relations:
  - the needed columns (attributes)
  - the discarded columns
- The general form of the PROJECT operation: $\boldsymbol{\pi_{<attribute\_list>}(R)}$, where π (pi) is the symbol used to represent the PROJECT operation, and is the desired sublist of attributes from the attributes of relation R.
- Its degree is equal to the number of attributes in <attribute list>
- For example, to list each employee's first and last name and salary, we can use the PROJECT operation as follows: $\pi_{Lname, Fname, Salary}(EMPLOYEE)$
- **Commutativity** does not hold on PROJECT.

Relational Algebra: Unary Relational Operations: PROJECT

- PROJECT operation can do **duplicate elimination.**
- For example, consider the following PROJECT operation: $\pi_{Sex, Salary}(EMPLOYEE)$

Sex	Salary
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Relational Algebra: In-line expression

- To retrieve the first name, last name, and salary of all employees who work in department number 5, we must apply a SELECT and a PROJECT operation.
  - $\pi_{Fname, Lname, Salary}(\sigma_{Dno=5}(EMPLOYEE))$

Fname	Lname	Salary
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Relational Algebra: Unary Relational Operations: Rename

- We can explicitly show the sequence of operations, giving a name to each intermediate relation, and using the assignment operation, denoted by ← (left arrow), as follows:
  - DEP5_EMPS ← $\sigma_{Dno=5}$(EMPLOYEE)
  - RESULT ← $\pi_{Fname, Lname, Salary}$(DEP5_EMPS)
- To rename the attributes in a relation, we simply list the new attribute names in parentheses, as in the following example:
  - TEMP ← $\sigma_{Dno=5}$(EMPLOYEE)
  - R(First_name, Last_name, Salary) ← $\pi_{Fname, Lname, Salary}$(TEMP)

TEMP

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston,TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston,TX	M	40000	888665555	5
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble,TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

R

First_name	Last_name	Salary
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

Relational Algebra: Unary Relational Operations: Rename

- We can rename either the relation name or the attribute names, or both—as a unary operator.
- The general RENAME operation when applied to a relation R of degree n is denoted by any of the following three forms:
  - $\rho_{S(B1, B2, \dots, Bn)}(R)$: renames both the relation and its attributes
  - $\rho_{S}(R)$: renames the relation only
  - $\rho_{(B1, B2, \dots, Bn)}(R)$: renames the attributes only
  - where the symbol ρ (rho) is used to denote the RENAME operator, S is the new relation name, and B1, B2, … , Bn are the new attribute names.

Relational Algebra: UNION

- The result of this operation, denoted by R ∪ S, is a relation that includes all tuples that are either in R or in S or in both R and S.
- Duplicate tuples are eliminated.
- For example, to retrieve the Social Security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5:
  - DEP5_EMPS ← $\sigma_{Dno=5}$(EMPLOYEE)
  - RESULT1 ← π_{Ssn}(DEP5_EMPS)
  - RESULT2(Ssn) ← π_{Super_ssn}(DEP5_EMPS)
  - RESULT ← RESULT1 ∪ RESULT2

RESULT1

Ssn
123456789
333445555
666884444
453453453

RESULT2

Ssn
333445555
888665555

RESULT

Ssn
123456789
333445555
666884444
453453453
888665555

Relational Algebra: INTERSECTION and MINUS

- INTERSECTION: The result of this operation, denoted by R ∩ S, is a relation that includes all tuples that are in both R and S.
- SET DIFFERENCE (or MINUS): The result of this operation, denoted by R – S, is a relation that includes all tuples that are in R but not in S.
- UNION and INTERSECTION are commutative operations:
  - R ∪ S = S ∪ R and
  - R ∩ S = S ∩ R
- UNION and INTERSECTION are associative operations:
  - R ∪ (S ∪ T) = (R ∪ S) ∪ T and
  - (R ∩ S) ∩ T = R ∩ (S ∩ T)
- MINUS operation is not commutative:
  - R − S ≠ S − R

Relational Algebra: Example: Union, Intersection, Minus

(a) **STUDENT**

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

(b)

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

(c)

Fn	Ln
Susan	Yao
Ramesh	Shah

(d)

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

(e)

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

(a) Two union-compatible relations. (b) STUDENT ∪ INSTRUCTOR. (c) STUDENT ∩ INSTRUCTOR. (d) STUDENT – INSTRUCTOR. (e) INSTRUCTOR – STUDENT.

Relational Algebra: Cartesian Product / Cartesian Join

- Denoted by ×
- Produces a new element by combining every member (tuple) from one relation (set) with every member (tuple) from the other relation (set).
- Result of R(A1, A2, … , An) × S(B1, B2, … , Bm) is Q(A1, A2, … , An, B1, B2, … , Bm): degree n + m attributes
- If $|R| = n_R$ and $|R| = n_S$, then $|R \times S| = n_R * n_S$

Relational Algebra: Cartesian Product / Cartesian Join: Example

- We want to retrieve a list of names of each female employee's dependents
  - FEMALE_EMPS $\leftarrow \sigma_{Sex='F'}(EMPLOYEE)$
  - EMPNAMES $\leftarrow \pi_{Fname,\ Lname,\ Ssn}(FEMALE\_EMPS)$
  - EMP_DEPENDENTS $\leftarrow$ EMPNAMES $\times$ DEPENDENT
  - ACTUAL_DEPENDENTS $\leftarrow \sigma_{Ssn=Essn}(EMP\_DEPENDENTS)$
  - RESULT $\leftarrow \pi_{Fname,\ Lname,\ Dependent\_name}(ACTUAL\_DEPENDENTS)$

Relational Algebra: Binary relational operations: JOIN / INNER JOIN

- Denoted by ⋈
- Used to combine related tuples from two relations into single “longer” tuples.
- Suppose that we want to retrieve the name of the manager of each department.
- The general form of a JOIN operation on two relations5 R(A1, A2, … , An) and S(B1, B2, … , Bm) is $R \bowtie_{<\text{join condition}>} S$
- The result of the JOIN is a relation Q with n + m attributes Q(A1, A2, … , An, B1, B2, … , Bm) in that order
- To get the manager’s name, we need to combine each department tuple with the employee tuple whose Ssn value matches the Mgr_ssn value in the department tuple.
- We do this by using the JOIN operation and then projecting the result over the necessary attributes, as follows:
  - DEPT_MGR ← DEPARTMENT $\bowtie_{\text{Mgr\_ssn=Ssn}}$ EMPLOYEE
  - RESULT ← $\pi_{\text{Dname, Lname, Fname}}$(DEPT_MGR)

Relational Algebra: Binary relational operations: OUTER JOIN

- A set of operations, called **outer joins**, were developed for the case where the user wants to keep all the tuples in R, or all those in S, or all those in both relations in the result of the JOIN, regardless of whether or not they have matching tuples in the other relation.
- This satisfies the need of queries in which tuples from two tables are to be combined by matching corresponding rows, but without losing any tuples for lack of matching values.
- For example, suppose that we want a list of all employee names as well as the name of the departments they manage *if they happen to manage a department*; if they do not manage one, we can indicate it with a NULL value.

RESULT

Fname	Minit	Lname	Dname
John	B	Smith	NULL
Franklin	T	Wong	Research
Alicia	J	Zelaya	NULL
Jennifer	S	Wallace	Administration
Ramesh	K	Narayan	NULL
Joyce	A	English	NULL
Ahmad	V	Jabbar	NULL
James	E	Borg	Headquarters

Relational Algebra: Variations of OUTER JOIN

- Left outer join:
  - The LEFT OUTER JOIN operation keeps every tuple in the first, or left, relation R in R⟕S; if no matching tuple is found in S, then the attributes of S in the join result are filled or padded with NULL values.

$$\text{TEMP} \leftarrow (\text{EMPLOYEE} \bowtie_{\text{Ssn=Mgr\_ssn}} \text{DEPARTMENT})$$
$$\text{RESULT} \leftarrow \pi_{\text{Fname, Minit, Lname, Dname}}(\text{TEMP})$$

- Right outer join:
  - Keeps every tuple in the second, or right, relation S in the result of $R \bowtie S$.
- Full outer join:
  - Keeps all tuples in both the left and the right relations when no matching tuples are found, padding them with NULL values as needed. denoted by ⟗