Query Processing

Key Points

- understanding the relational algebra notation
- building a query tree for a simple SQL query

The Relational Model: Queries

The relational model provides a means for specifying data and operations on that data.

- provides the theory underlying SQL, though SQL is not an exact implementation
- data is structured as tuples grouped into relations
- query languages
  - *relational algebra* is a procedural language consisting of operators and operands which are combined to build expressions
    - the basic operands are relations
    - expressions specify how to compute the desired results
      - e.g. join BOOK and BOOK_AUTHORS using the book title, then restrict the result to tuples with the author ‘C.J. Cherryh’ and finally pick out just the title from each tuple
  - *relational calculus* is a declarative language based on predicate calculus
    - the desired result is expressed without specifying how to compute it
      - “get the title of books for which there exists an author named ‘C.J. Cherryh’”

The Relational Model: Queries

- SQL is based on both relational algebra and relational calculus
- relational algebra is used in query processing

Relational Algebra Operators

category	operator	notation	semantics	notes
set operations	union	R ⋃ S	all tuples in R or S	R, S must have the same schema (same attributes, same domain for each attribute)
	intersection	R ⋂ S	all tuples in both R and S	
	difference	R−S	all tuples in R but not in S	
extracting some of the data in a relation	project	$\pi_{a1,a2,\ldots,an}(R)$	all tuples in R, with just the specified attributes	duplicates are removed corresponds to SQL SELECT DISTINCT
	select	$\sigma_C(R)$	only the tuples in R satisfying condition C	corresponds to SQL WHERE
combining relations	Cartesian product	R×S	all possible pairings of a tuple in R with a tuple in S	duplicate columns identified by originating relation e.g. R.A corresponds to SQL ,
	natural join	R ⋆ S or R ⋈ S	theta-join where the join condition is equality on all attributes with the same name in R and S	duplicate columns are eliminated corresponds to SQL NATURAL JOIN
	theta-join	$R \bowtie_C S$	equivalent to $\sigma_C(R \times S)$	equijoin when *c* contains only equality comparisons duplicate columns identified by originating relation e.g. R.A corresponds to SQL JOIN … ON
renaming relations and attributes	rename	$\rho_S(R)$ $\rho_{S(a1,a2,\ldots,an)}(R)$	(essentially) change name of R to S also change R's attributes to a1,a2,...,an	actually creates a new relation with the new names

Questions

How do you represent SELECT DISTINCT?

- this is just the π operation

extracting some of the data in a relation	project	$\pi_{a1,a2,\ldots,an}(R)$	all tuples in R, with just the specified attributes	duplicates are removed corresponds to SQL SELECT DISTINCT
	select	$\sigma_C(R)$	only the tuples in R satisfying condition C	corresponds to SQL WHERE

Query Processing

Steps:

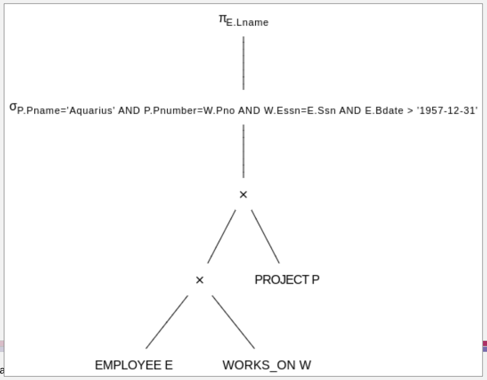
- *scanner* identifies language tokens
  - SQL keywords, column names, table names, …
- *parser* checks query syntax
- query is *validated* to make sure column and table names are valid
- internal representation of query (*query tree* or *query graph*) is built
  - each internal node of the tree is a relational algebra operation

done by the “prepare” step of a prepared statement

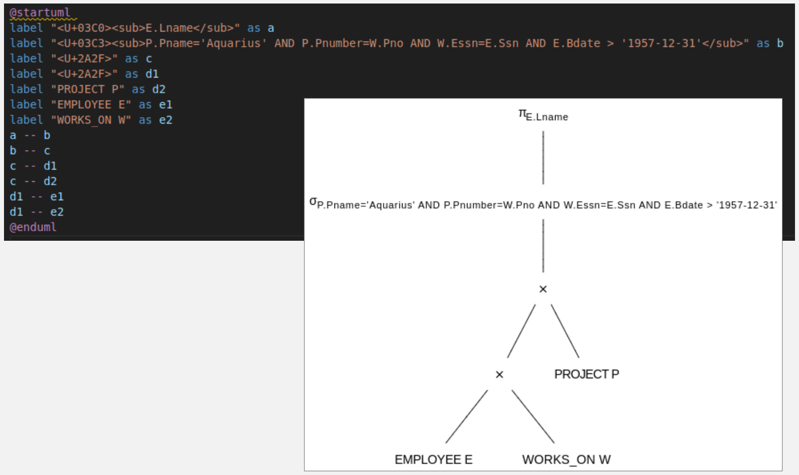
- devise *execution plan* for retrieving the result of the query from the data files
  - execution plan = query tree + algorithm for carrying out each operation

From Queries to Query Trees

```
SELECT E.Lname
FROM EMPLOYEE E, WORKS_ON W, PROJECT P
WHERE P.Pname='Aquarius' AND P.Pnumber=W.Pno
  AND W.Essn=E.Ssn AND E.Bdate > '1957-12-31'
```


$\pi_{E.Lname}$
$\sigma_{P.Pname='Aquarius' AND P.Pnumber=W.Pno AND W.Essn=E.Ssn AND E.Bdate > '1957-12-31'}$
×
×
PROJECT P
EMPLOYEE E
WORKS_ON W

From Queries to Query Trees

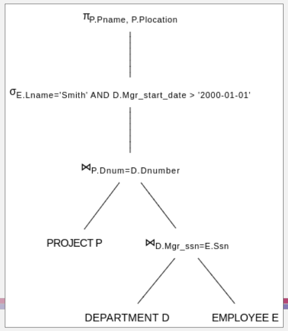


```
@startuml
label "<U+03C0><sub>E.Lname</sub>" as a
label "<U+03C3><sub>P.Pname='Aquarius' AND P.Pnumber=W.Pno AND W.Essn=E.Ssn AND E.Bdate > '1957-12-31'</sub>" as b
label "<U+2A2F>" as c
label "<U+2A2F>" as d1
label "PROJECT P" as d2
label "EMPLOYEE E" as e1
label "WORKS_ON W" as e2
a -- b
b -- c
c -- d1
c -- d2
d1 -- e1
d1 -- e2
@enduml
```

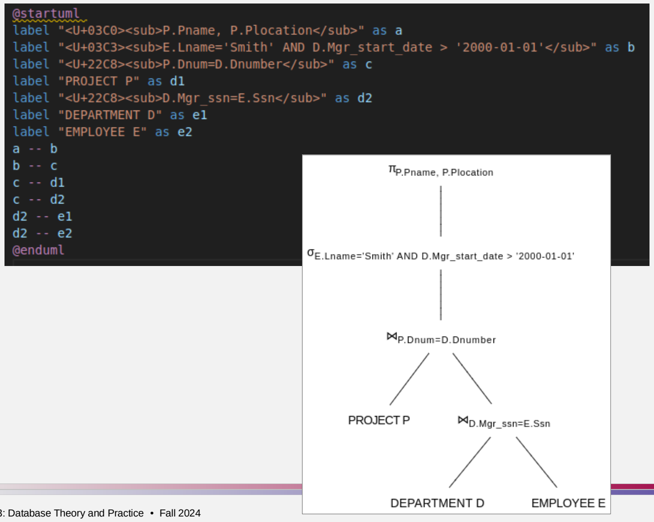
$\pi_{E.Lname}$
$\sigma_{P.Pname='Aquarius' AND P.Pnumber=W.Pno AND W.Essn=E.Ssn AND E.Bdate > '1957-12-31'}$
×
×
PROJECT P
EMPLOYEE E
WORKS_ON W

From Queries to Query Trees

```
SELECT P.Pname, P.Plocation
FROM PROJECT P JOIN ( DEPARTMENT D JOIN EMPLOYEE E
                      ON D.Mgr_ssn=E.Ssn )
  ON P.Dnum=D.Dnumber
WHERE E.Lname='Smith'
  AND D.Mgr_start_date > '2000-01-01'
```


$\pi_{P.Pname, P.Plocation}$
$\sigma_{E.Lname='Smith' AND D.Mgr\_start\_date > '2000-01-01'}$
$\bowtie_{P.Dnum=D.Dnumber}$
PROJECT P
$\bowtie_{D.Mgr\_ssn=E.Ssn}$
DEPARTMENT D
EMPLOYEE E

From Queries to Query Trees



```
@startuml
label "<U+03C0><sub>P.Pname, P.Plocation</sub>" as a
label "<U+03C3><sub>E.Lname='Smith' AND D.Mgr_start_date > '2000-01-01'</sub>" as b
label "<U+22C8><sub>P.Dnum=D.Dnumber</sub>" as c
label "PROJECT P" as d1
label "<U+22C8><sub>D.Mgr_ssn=E.Ssn</sub>" as d2
label "DEPARTMENT D" as e1
label "EMPLOYEE E" as e2
a -- b
b -- c
c -- d1
c -- d2
d2 -- e1
d2 -- e2
@enduml
```

$\pi_{P.Pname, P.Plocation}$
$\sigma_{E.Lname='Smith' AND D.Mgr\_start\_date > '2000-01-01'}$
$\bowtie_{P.Dnum=D.Dnumber}$
PROJECT P
$\bowtie_{D.Mgr\_ssn=E.Ssn}$
DEPARTMENT D
EMPLOYEE E

From Queries to Query Trees

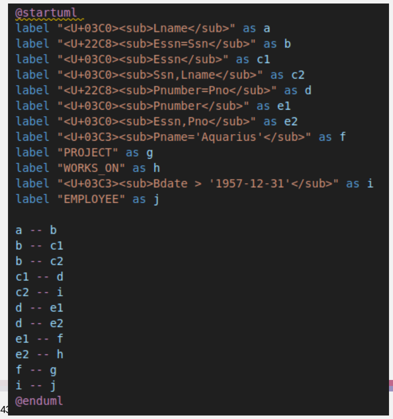
Complex queries are decomposed into *query blocks* with a single SELECT-FROM-WHERE (plus GROUP BY, HAVING if present).

- nested queries involve multiple blocks

```
SELECT Lname, Fname
FROM EMPLOYEE
WHERE Salary > ( SELECT MAX(Salary)
                 FROM EMPLOYEE
                 WHERE Dno=5 )
```

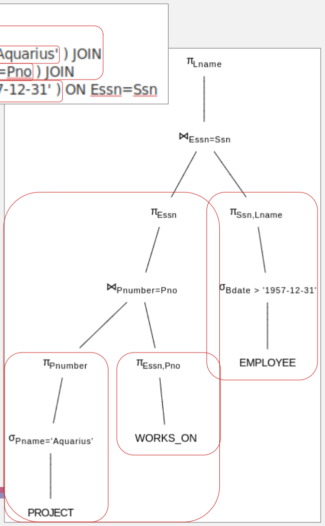
From Queries to Query Trees

```
SELECT Lname
FROM ( SELECT Essn
       FROM ( SELECT Pnumber FROM PROJECT WHERE Pname='Aquarius' ) JOIN
            ( SELECT Essn,Pno FROM WORKS_ON ) ON Pnumber=Pno ) JOIN
     ( SELECT Ssn,Lname FROM EMPLOYEE WHERE Bdate > '1957-12-31' ) ON Essn=Ssn
```



```
@startuml
label "<U+03C0><sub>Lname</sub>" as a
label "<U+22C8><sub>Essn=Ssn</sub>" as b
label "<U+03C0><sub>Essn</sub>" as c1
label "<U+03C0><sub>Ssn,Lname</sub>" as c2
label "<U+22C8><sub>Pnumber=Pno</sub>" as d
label "<U+03C0><sub>Pnumber</sub>" as e1
label "<U+03C0><sub>Essn,Pno</sub>" as e2
label "<U+03C3><sub>Pname='Aquarius'</sub>" as f
label "PROJECT" as g
label "WORKS_ON" as h
label "<U+03C3><sub>Bdate > '1957-12-31'</sub>" as i
label "EMPLOYEE" as j

a -- b
b -- c1
b -- c2
c1 -- d
c2 -- i
d -- e1
d -- e2
e1 -- f
e2 -- h
f -- g
i -- j
@enduml
```


π_{Lname}
$\bowtie_{Essn=Ssn}$
π_{Essn}
$\pi_{Ssn,Lname}$
$\bowtie_{Pnumber=Pno}$
$\sigma_{Bdate > '1957-12-31'}$
EMPLOYEE
$\pi_{Pnumber}$
$\pi_{Essn,Pno}$
$\sigma_{Pname='Aquarius'}$
WORKS_ON
PROJECT