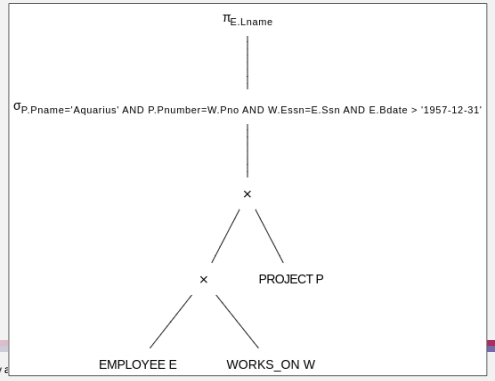
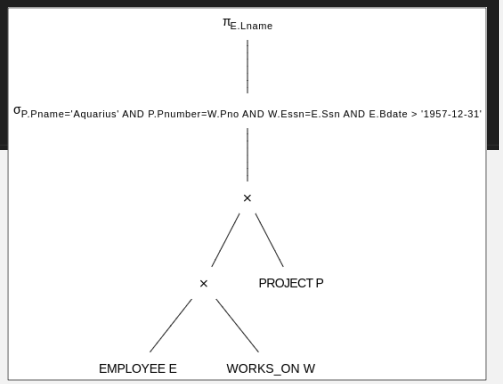
From Queries to Query Trees

```
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'
```



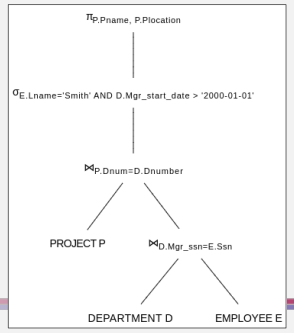
From Queries to Query Trees

```
@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
```



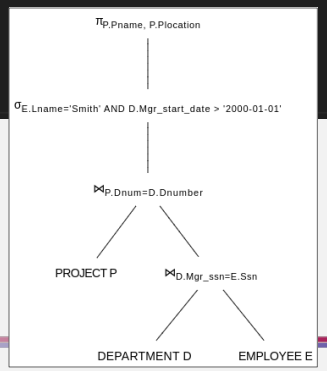
From Queries to Query Trees

```
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'
```



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
```



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

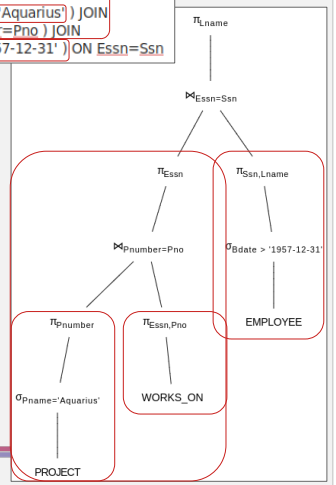
```
SELECT Lname, Fname
FROM EMPLOYEE
WHERE Salary > ( SELECT MAX(Salary)
                 FROM EMPLOYEE
                 WHERE Dno=5 )
```


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
```



Relational Algebra Operators

category	operator	notation	semantics	notes
set operations	union	$R \cup S$	all tuples in R or S	R, S must have the same schema (same attributes, same domain for each attribute)
	intersection	$R \cap S$	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	$\pi_{a1,a2,...,an}(R)$	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	$R \times S$	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 $R \bowtie S$	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	$\rho_S(R)$ $\rho_{S(a1,a2,...,an)}(R)$	(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

majors with at least one student

name, ID of CPSC majors

name, ID of students with no major

class years of non-CPSC majors

majors with no students

majors with more than one student

majors with exactly one student

names of departments with at least one student major

names of departments with no student majors

```
STUDENT(name,id,year,major)
DEPARTMENT(abbrev,deptname)
```

For each of the SQL queries, match the equivalent relational algebra expression.

a. $\pi_{major}(STUDENT)$

b. $\pi_{name,id}(\sigma_{major=CPSC}(STUDENT))$

c. $\pi_{name,id}(\sigma_{major=null}(STUDENT))$

d. $\pi_{year}(\sigma_{major \neq CPSC}(STUDENT))$

e. $\pi_{abbrev}(DEPARTMENT) - \pi_{major}(STUDENT)$

f. $\pi_{major}(\sigma_{S1.major=S2.major \text{ and } S1.id \neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$

g. $\pi_{major}(STUDENT) - \pi_{major}(\sigma_{S1.major=S2.major \text{ and } S1.id \neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$

h. $\pi_{deptname}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$

i. $\pi_{deptname}(\pi_{deptname,abbrev}(DEPARTMENT) - \pi_{deptname,abbrev}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$

SELECT abbrev FROM DEPARTMENT D WHERE NOT EXISTS (SELECT * FROM STUDENT S WHERE D.abbrev=S.major)

SELECT name,id FROM STUDENT WHERE major IS NULL

SELECT major FROM STUDENT

SELECT DISTINCT major FROM STUDENT WHERE major NOT IN (SELECT major FROM STUDENT S1, STUDENT S2 WHERE S1.major=S2.major AND S1.id<>S2.id)

SELECT DISTINCT major FROM STUDENT S1, STUDENT S2 WHERE S1.major=S2.major AND S1.id<>S2.id

```
STUDENT(name,id,year,major)
DEPARTMENT(abbrev,deptname)
```

For each of the SQL queries, match the equivalent relational algebra expression.

a. $\pi_{major}(STUDENT)$
b. $\pi_{name,id}(\sigma_{major=CPSC}(STUDENT))$
c. $\pi_{name,id}(\sigma_{major=null}(STUDENT))$
d. $\pi_{year}(\sigma_{major\neq CPSC}(STUDENT))$
e. $\pi_{abbrev}(DEPARTMENT) - \pi_{major}(STUDENT)$
f. $\pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
g. $\pi_{major}(STUDENT) - \pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
h. $\pi_{deptname}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$
i. $\pi_{deptname}(\pi_{deptname,abbrev}(DEPARTMENT) - \pi_{deptname,abbrev}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$

SELECT DISTINCT deptname FROM (SELECT deptname,abbrev FROM DEPARTMENT WHERE (deptname,abbrev) NOT IN (SELECT deptname,abbrev FROM DEPARTMENT JOIN STUDENT ON abbrev=major))

SELECT DISTINCT year FROM STUDENT WHERE major<>'CPSC'

SELECT DISTINCT deptname FROM DEPARTMENT JOIN STUDENT ON abbrev=major

```
STUDENT(name,id,year,major)
DEPARTMENT(abbrev,deptname)
```

For each of the SQL queries, match the equivalent relational algebra expression.

a. $\pi_{major}(STUDENT)$
b. $\pi_{name,id}(\sigma_{major=CPSC}(STUDENT))$
c. $\pi_{name,id}(\sigma_{major=null}(STUDENT))$
d. $\pi_{year}(\sigma_{major\neq CPSC}(STUDENT))$
e. $\pi_{abbrev}(DEPARTMENT) - \pi_{major}(STUDENT)$
f. $\pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
g. $\pi_{major}(STUDENT) - \pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
h. $\pi_{deptname}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$
i. $\pi_{deptname}(\pi_{deptname,abbrev}(DEPARTMENT) - \pi_{deptname,abbrev}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$

SELECT name,id FROM STUDENT WHERE major='CPSC'

SELECT DISTINCT deptname FROM DEPARTMENT D WHERE NOT EXISTS (SELECT * FROM STUDENT S WHERE D.abbrev=S.major)

SELECT abbrev FROM DEPARTMENT WHERE abbrev NOT IN (SELECT major FROM STUDENT)

SELECT DISTINCT major FROM STUDENT

```
STUDENT(name,id,year,major)
DEPARTMENT(abbrev,deptname)
```

For each of the SQL queries, match the equivalent relational algebra expression.

a. $\pi_{major}(STUDENT)$
b. $\pi_{name,id}(\sigma_{major=CPSC}(STUDENT))$
c. $\pi_{name,id}(\sigma_{major=null}(STUDENT))$
d. $\pi_{year}(\sigma_{major\neq CPSC}(STUDENT))$
e. $\pi_{abbrev}(DEPARTMENT) - \pi_{major}(STUDENT)$
f. $\pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
g. $\pi_{major}(STUDENT) - \pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
h. $\pi_{deptname}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$
i. $\pi_{deptname}(\pi_{deptname,abbrev}(DEPARTMENT) - \pi_{deptname,abbrev}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$

SELECT DISTINCT deptname FROM DEPARTMENT D WHERE EXISTS (SELECT * FROM STUDENT S WHERE S.major=D.abbrev)

```
STUDENT(name,id,year,major)
DEPARTMENT(abbrev,deptname)
```

For each of the SQL queries, match the equivalent relational algebra expression.

a. $\pi_{major}(STUDENT)$
b. $\pi_{name,id}(\sigma_{major=CPSC}(STUDENT))$
c. $\pi_{name,id}(\sigma_{major=null}(STUDENT))$
d. $\pi_{year}(\sigma_{major\neq CPSC}(STUDENT))$
e. $\pi_{abbrev}(DEPARTMENT) - \pi_{major}(STUDENT)$
f. $\pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
g. $\pi_{major}(STUDENT) - \pi_{major}(\sigma_{S1.major=S2.major\ and\ S1.id\neq S2.id}(\rho_{S1}(STUDENT) \times \rho_{S2}(STUDENT)))$
h. $\pi_{deptname}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$
i. $\pi_{deptname}(\pi_{deptname,abbrev}(DEPARTMENT) - \pi_{deptname,abbrev}(DEPARTMENT \bowtie_{abbrev=major} STUDENT)$