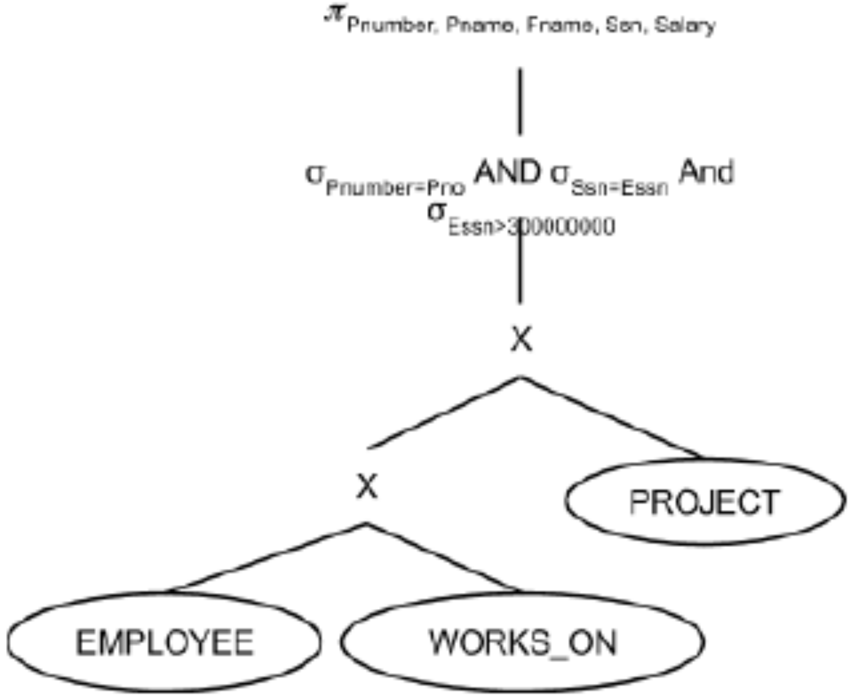
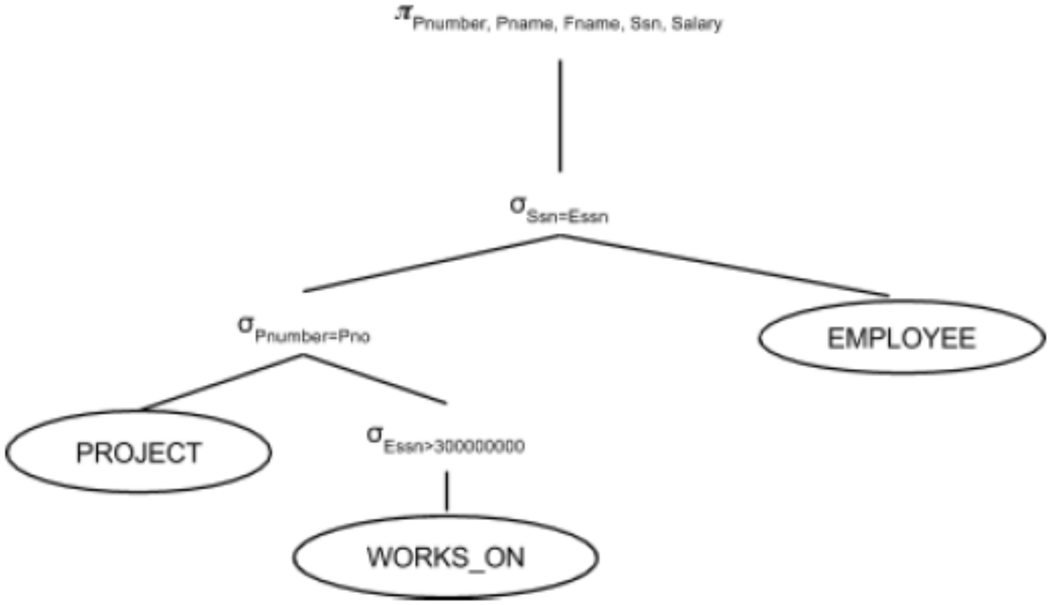
Homework_2 solutions:

1.

a)

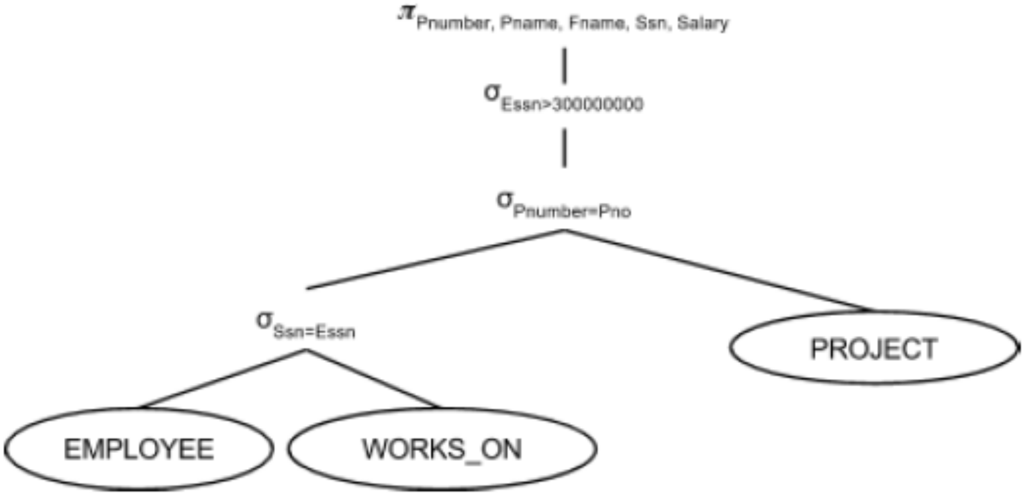

π Pnumber, Pname, Fname, Ssn, Salary
σ Pnumber=Pno AND σ Ssn=Essn And σ Essn>300000000
X
X
PROJECT
EMPLOYEE
WORKS_ON

b) Representation 1:



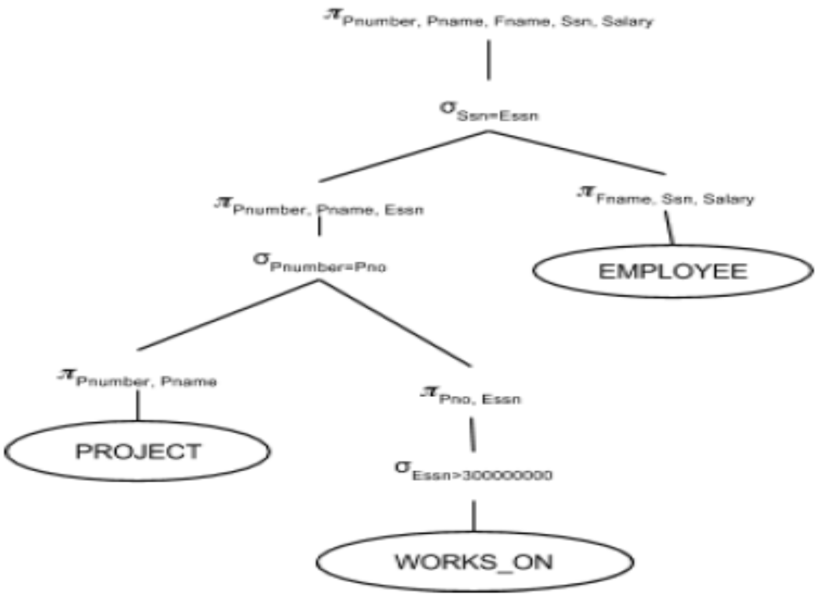
This representation is used when Selectivity of "Essn>300000000" is very low.

Representation 2:


$\pi_{Pnumber, Pname, Fname, Ssn, Salary}$
$\sigma_{Essn>300000000}$
$\sigma_{Pnumber=Pno}$
$\sigma_{Ssn=Essn}$
PROJECT
EMPLOYEE
WORKS_ON

This representation is used when Selectivity of "Essn>300000000" is very high.

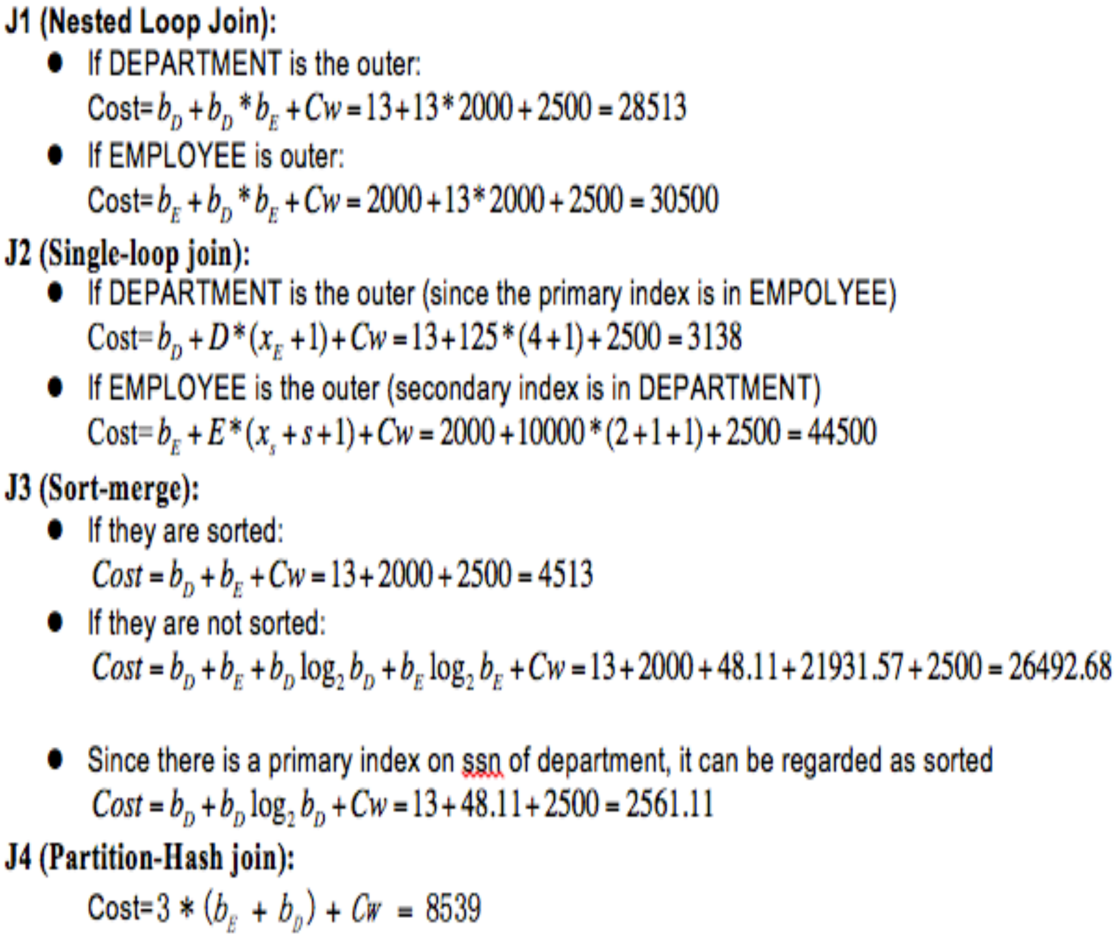
c)


$\pi_{Pnumber, Pname, Fname, Ssn, Salary}$
$\sigma_{Ssn=Essn}$
$\pi_{Pnumber, Pname, Essn}$
$\pi_{Fname, Ssn, Salary}$
$\sigma_{Pnumber=Pno}$
EMPLOYEE
$\pi_{Pnumber, Pname}$
$\pi_{Pno, Essn}$
PROJECT
$\sigma_{Essn>300000000}$
WORKS_ON

Above is the final optimized tree. Go through the algorithm on pg. 704 in textbook.

d) Go through the textbook.

2.



J1 (Nested Loop Join):

- If DEPARTMENT is the outer:
  Cost=$b_D + b_D * b_E + Cw = 13 + 13 * 2000 + 2500 = 28513$
- If EMPLOYEE is outer:
  Cost=$b_E + b_D * b_E + Cw = 2000 + 13 * 2000 + 2500 = 30500$

J2 (Single-loop join):

- If DEPARTMENT is the outer (since the primary index is in EMPOLYEE)
  Cost=$b_D + D * (x_E + 1) + Cw = 13 + 125 * (4 + 1) + 2500 = 3138$
- If EMPLOYEE is the outer (secondary index is in DEPARTMENT)
  Cost=$b_E + E * (x_s + s + 1) + Cw = 2000 + 10000 * (2 + 1 + 1) + 2500 = 44500$

J3 (Sort-merge):

- If they are sorted:
  $Cost = b_D + b_E + Cw = 13 + 2000 + 2500 = 4513$
- If they are not sorted:
  $Cost = b_D + b_E + b_D \log_2 b_D + b_E \log_2 b_E + Cw = 13 + 2000 + 48.11 + 21931.57 + 2500 = 26492.68$

- Since there is a primary index on ssn of department, it can be regarded as sorted
  $Cost = b_D + b_D \log_2 b_D + Cw = 13 + 48.11 + 2500 = 2561.11$

J4 (Partition-Hash join):

Cost=$3 * (b_E + b_D) + Cw = 8539$

3.

Refer to the textbook.

4.

Semantic query optimization uses constraints specified on the database schema to convert a query into another, which is more efficient.

5.

R (left) and S (right) are the tables and R.A and S.B are the attributes to be joined.
|R| = n and |S| = m.
T is the output.

Algorithm:

```
Sort the tuples in R on A;
Sort the tuples in S on B;
Set i=1, j=1;
While(i<=n and j<=m)
Do{
        If R.Ai > S. Bj
        Then {
                (*Right outer join, NULL tuple output to the result for left table*)
                Output combined tuple <NULL, Sj> to T;
                Set j = j+1;
}
Elseif R.Ai < S.Bj
Then set i = i+1;
Else{
(*Matched value found*)

        Output combined tuple <Ri,  Sj> to T;

        (*Output other matched tuples in S, if any*)
        Set jj = j+1;
        While(jj<=m and R.Ai == S.Bjj)
        Do{
                Output combined tuple <Ri,  Sjj> to T;
                Set jj = jj + 1;
        }

        (*Output other matched tuples in R, if any*)
        Set ii = i+1;
        While(jj<=m and R.Aii == S.Bj)
```

```
    Do{
        Output combined tuple <Rii,  Sj> to T;
        Set ii = ii + 1;
    }

    Set i=ii, j=jj;

}

}
```

(*In case R (left table) reaches the end first, the following code will create tuples for all remaining tuples in S with the R-part left NULL*)

```
If(j<m or (j==m and R.Ai<S.Bj))
Then{
    While(j<=m){
        Output combined tuple <NULL, Sj> to T;
        Set j = j+1;
    }
}
```