DJ Wiley

A01099945

Homework #2

**Exercise 17.24**

Solution:

The EMPLOYEE file has the following fields. Name, Ssn, Department_code, Phone, Address, Birth_date, Sex, Job_code, and Salary. The primary index is the Ssn field. The secondary indexes are the following. Department-code, Job_code, and Salary. Those fields are also non-key fields.

First find the record pointers that satisfy the complex selection condition Department_code = 5 using the secondary index on Department_code.

Second, using those results, retrieve the record pointers that satisfy the complex selection condition Job_code = 12 using the secondary index on Job_code.

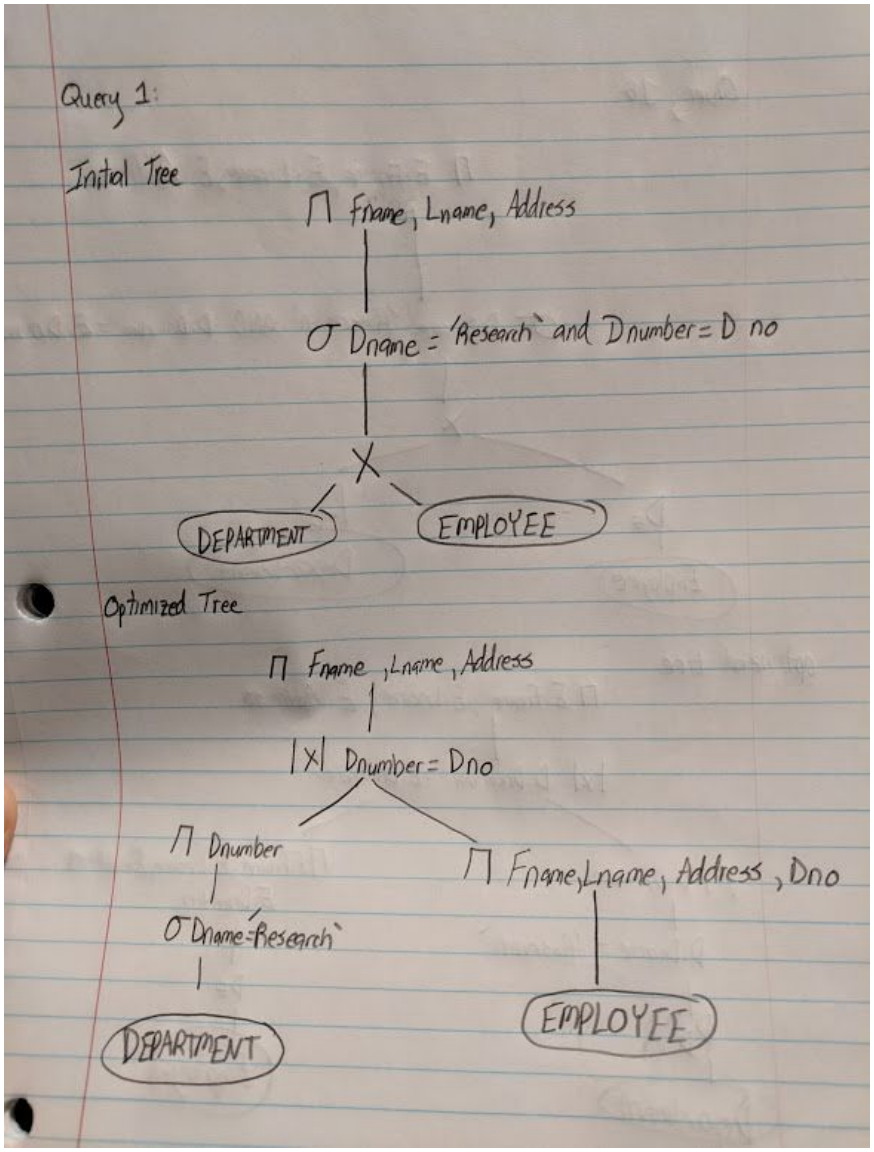
Lastly, using the previous results, retrieve the record pointers that satisfy the complex selection condition Salary = 50000 using the secondary index on Salary. Voila

**Exercise 18.13 (only show the final tree for part (b) for queries Q1 and Q1B, do not do other queries or part (a) or (c))**

Solution:

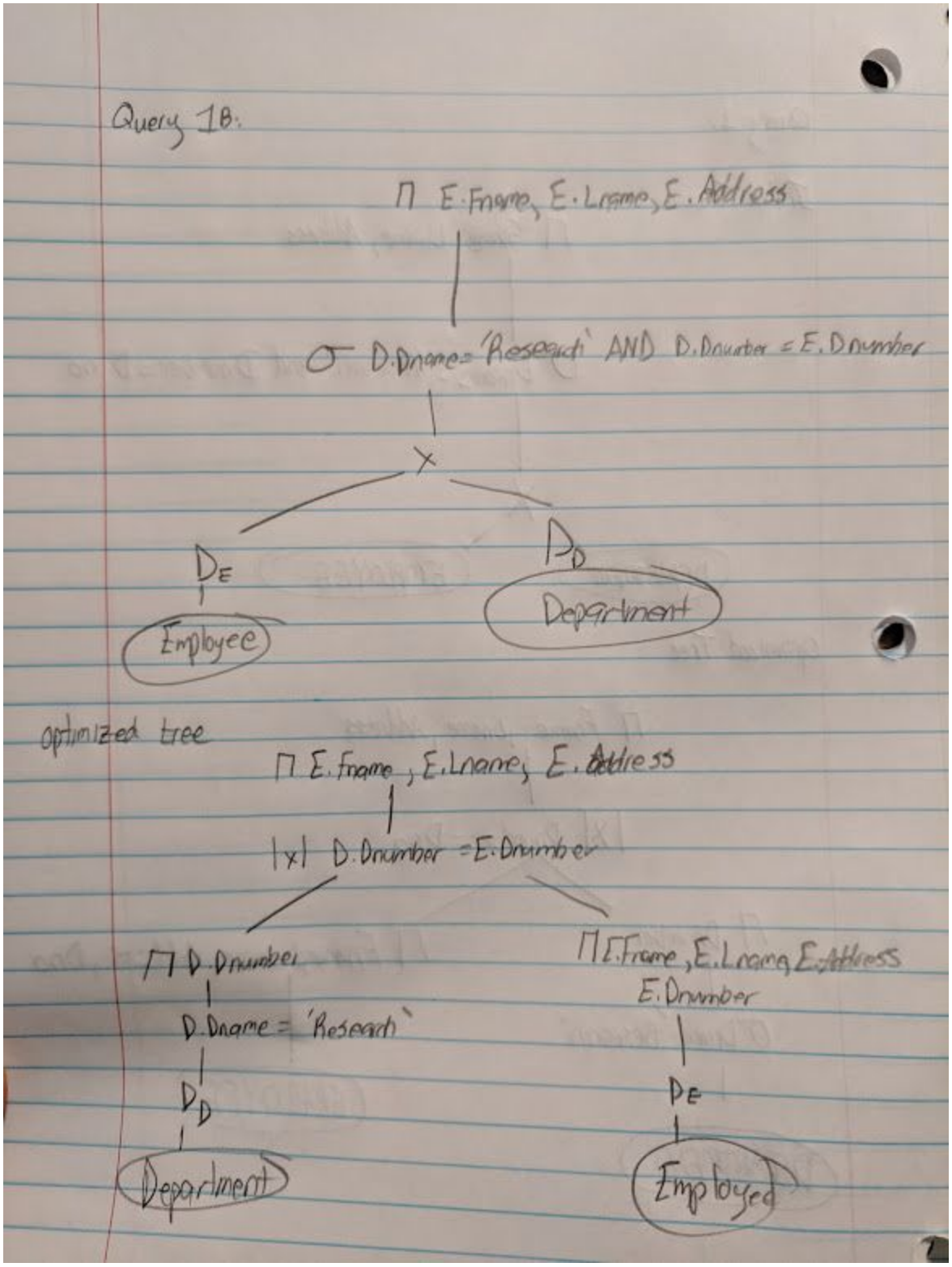
Query 1:

```
SELECT Fname, Lname, Address
FROM EMPLOYEE, DEPARTMENT
WHERE Dname='Research' AND Dnumber=Dno;
```




Query 1b:

SELECT E.Fname, E.LName, E.Address
FROM EMPLOYEE E, DEPARTMENT D
WHERE D.DName='Research' AND D.Dnumber=E.Dno;




**Exercise 18.14**

Solution:

First compute the number of runs, Nr, for the merge phase. This formula was provided by the text.

Nr = [b\Nb]

b = 4096 Blocks in the file

Nb = 64 Buffer space

Nr = [4096/ 64] = 64

The total number of passes is dependent on the degree from which the merging occurs. This will be labeled as DM

DM = min (Nr, Nb – 1)

= min (64, 64 – 1) = 63

The total number of passes is calculated by the following: $[\log_{DM}(NR)] = [\log_{63}(64)] = 2$

**Exercise 19.21**

Solution:

Query Plan 1:

Dno = Dnumber

/ \

Salary >40000 Department

/

EMPLOYEE

In the EMPLOYEE table there are 500 unique values in the Salary column with a minimum value of 1 and a maximum value of 500. Each of these values are expressed in units where 1 unit = 100 dollars.

$\sigma_{Salary>40000}$ is estimated to be (500-400)/500 = 1/5

Because of this, the salaries can be considered to be spread evenly. The cost of accessing the index is equal to Blvl + (1/5) x (LEAF BLOCKS)

= 1 + (1/5) x 50 = 11

Number of dta blocks to be accessed = (1/5)(NUM_ROWS)

= (1/5)(10000) = 2000

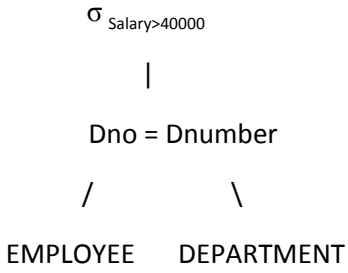
This means that 10000 rows are stored in 2000 blocks. It also means that 500 rows can be stored in each block. As a result, the TEMPORARY table will need 400 blocks to store the resultant. So the cost of writing the TEMPORARY table to the disk is the cost of writing 400 blocks. The cost for nested loop joins on the temp table and the DEPARTMENT table will be b+(b*b).

So... 5 + (5*400) = 2005 block accesses.

Total cost = Cost of index access + Data blocks accessed + blocks for TEMP table + Nested loop performance

= 11 + 2000 + 400 + 2005 = 4416 block accesses

Query plan 2:




We could again use a nested loop to join. Instead of creating a TEMP table we could use a pipeline approach to pass joining rows to the select operator.

Using the nested loop join algorithm query would give us:

Num data blocks = 2000

Num rows in DEPARTMENT = 50

Num leaf blocks = 50

50 + (50x2000) = 100050 block accesses

OPTIMIZATION:

If there were 50 main memory buffer pages for the join then the whole DEPARTMENT table could be plugged there. The new cost of the join and pipelined select query would be 50 + 2000 = 2050

**Exercise 14.26**

Solution:

a)

i. A -> B Cannot hold. Tuple 1 has 10 ->b1, and Tuple 2 has 10 -> b2. There is a violation
ii. B -> C Holds true. b1 -> c1 and b3 -> c4. No violation
iii. C -> B Does not hold true. c1->b1 and c1->b4. There is a violation

iv. B -> A Does not hold true. b1->10 and b1->13. There is a violation

v. C -> A Does not hold true. c1->10, c1->11, c1->13. There is a violation in the c1 tuple

b) The relation mentioned in this problem does have a primary key. The primary key is (B, C) because of the dependency and unique relation from B -> C.

**Exercise 14.27**

Solution:

AB is not the candidate key. The closure of AB does not give us all the attributes of the relation. ABD is the candidate key.

To check if AB is the candidate key of R we need to find the closure of AB

| Closure set of $\{AB\}^+$ | Functional Dependency used |
|---|---|
| {A, B} | Trivial |
| {A, B, C} | AB -> C |

Follow the same format as above to find the closures of ABD

| Closure set of $\{ABD\}^+$ | Functional Dependency used |
|---|---|
| {A, B, D} | Trivial |
| {A, B, C, D} | AB -> C |
| {A, B, C, D, E} | CD -> E |

As you can see, all the attributes from relation R can be identified using the key ABD, hence ABD is a candidate key for relation R.

**Exercise 14.31 (only consider BCNF, do not worry about the other normal forms)**

Solution:

a) The relation Book is not in BCNF.

Explanation:

The primary key is (Book_Title, Author_Name). For a relation to be 1NF the relation should contain only atomic values. The relation of Book contains only atomic values, therefore the relation Book is in 1NF. Because the relation is 1NF, it cannot be BCNF

b) Book_author is the only one that can be normalized to fit the BCNF. The relation contains the superkey of the table.