Q1. Relational Algebra [9 marks, 3 marks each]

Consider the following relational database, where the primary keys are underlined.

employee (*person_name*, *city*)
works (*person_name*, *company_name*, *salary*)
company (*company_name*, *business*)

Give an expression in relational algebra to express each of the following queries

a) Find the names of all employees who ***do not*** work for company "Microhard", if there exist some employees who are self-employed and do not work for any company.

$\Pi_{person\_name}(employee) - \Pi_{person\_name}(\sigma_{company\_name = \text{"Microhard"}}(works))$

b) Find the names and cities of all employees who work for IT companies (IT is the name of the business).

$\Pi_{person\_name,\ city}(employee \bowtie works \bowtie \sigma_{business = \text{"IT"}}(company)))$

c) For ***each*** company with number of employees greater than 100, list the company name and the business.

$t \leftarrow company\_name\ \gamma\ \textbf{count}(person\_name)\ \textbf{as}\ num\_employees(works)$

$\Pi_{company\_name,\ business}(company \bowtie (\sigma_{num\_employees > 100}\ t))$

Q2. Indexing [10 marks]

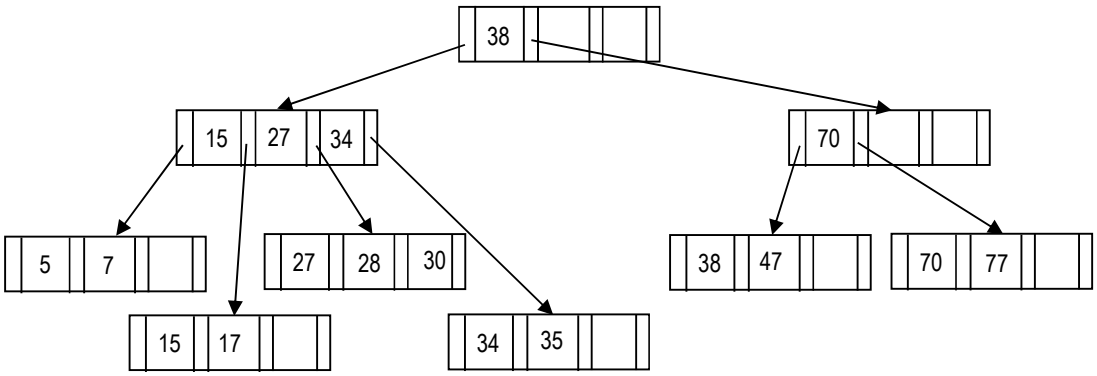
a) Consider a file with 30,000 records of size 100 bytes stored on a disk with block size 1,024 bytes and a record cannot span multiple blocks. Suppose that a ***secondary index*** is constructed on a candidate key of the file (i.e., the search key of the secondary index is a candidate key of the file). Suppose that the search key is 10 bytes long and a pointer is 6 bytes long. Find the number of block accesses required to perform a binary search for a record using the index. Show the steps clearly (no steps, no marks).

[4 marks]

Number of index entries per block = $\lfloor (1024 / (10+6)) \rfloor = 64$
Number of index entries = number of records in the data file = 30000
Number of index blocks = $\lceil (30000/64) \rceil = 469$
Number of block accesses required to perform a binary search for an index record = $\lceil \log_2 469 \rceil = 9$
Number of block accesses required to search for a record using the index = 9 + 1 = 10

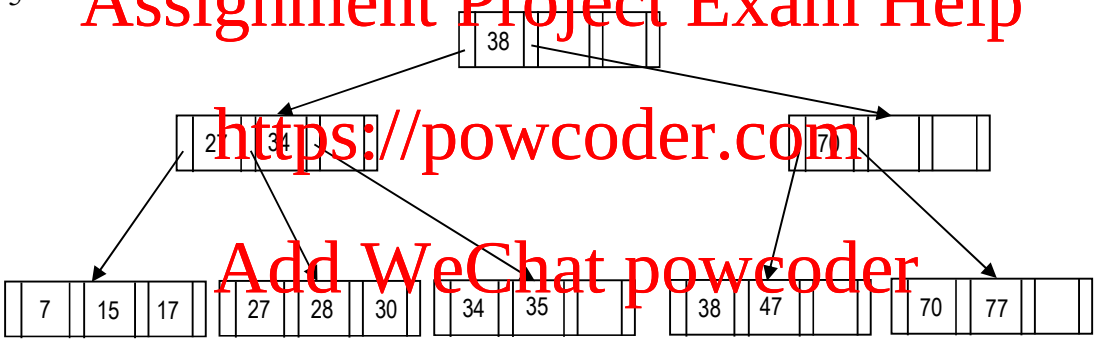
b) Consider the following B^+-tree with n=4. What is the ***minimum*** number of search-key values you must delete for the tree to shrink down ***by one level***? Show the sequence of deletions and draw a diagram for ***each*** deletion.

[6 marks]

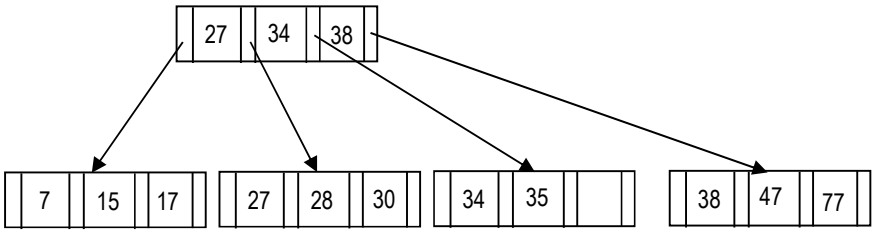


2 search-key values have to be deleted.

-5

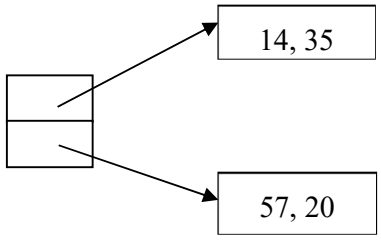


-70

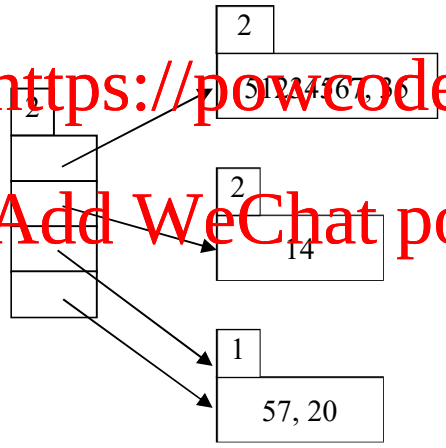


Q3. Hashing [6 marks]

Suppose that we are using ***extendable hashing*** on a file that contains records with integer search-key values. Suppose the hash function is $h(x) = x \bmod 32$ which generates 5-bit values and each bucket can hold **two** records. In the following figure, some records have been inserted. Draw the structure after a new record with search-key value equal to **your own** student ID is inserted. You need to show the ***i* value of the bucket address table** and the i_j **value of each bucket** in your diagram.



Suppose the student ID is 51234567.



Query processing and optimization [25 marks]

Consider the following relations, where the keys are underlined:

ENGINEER (ID, Name)
PROJECT (PID, ICEngID)

The ICEngID attribute in PROJECT is the ID of the engineer who is in charge of the project and PID is the ID of the project.

Consider the following query.

```
SELECT    *
FROM      ENGINEER E, PROJECT P
WHERE     E.ID=P.ICEngID
```

Given the following statistics and indices:

- number of tuples in ENGINEER: 1,600
- number of tuples in PROJECT: 3,200
- size of a tuple in ENGINEER: 50 bytes
- size of a tuple in PROJECT: 80 bytes
- disk block size: 512 bytes
- tuples do not span across blocks
- V(ICEngID, PROJECT) = V(ID, ENGINEER) = 1,600
- 3-level B^+-tree primary index on ID for ENGINEER
- 4-level B^+-tree primary index on PID for PROJECT
- 3-level B^+-tree secondary index on ICEngID for PROJECT

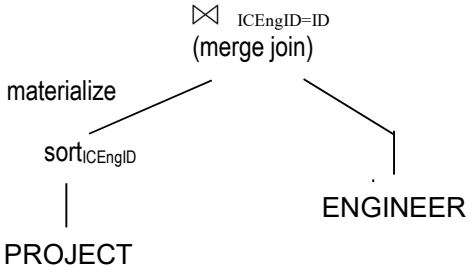
a) Estimate the number of output tuples for the query. Explain.

[3 marks]

Since ID is a key for ENGINNER, then a tuple of PROJECT will join with one tuple from ENGINNER. Therefore, the number of output tuples is the number of tuples in PROJECT, i.e. 3,200.

b) Draw a fully annotated ***evaluation plan*** if the query is computed with the ***merge-join*** algorithm for the ***worst-case estimate***.

[4 marks]



c) What is the **minimum** amount of memory in number of blocks for the ***worst-case estimate*** of the evaluation plan in part b)?

[1 mark]

3 memory blocks.

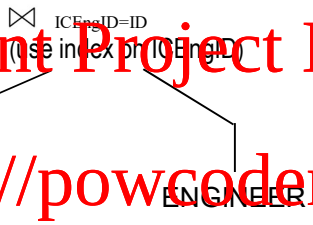
d) What is the ***worst-case cost*** in ***number of disk block transfers*** of the evaluation plan in part b)? Show the steps clearly (no steps, no marks).

[11 marks]

- Number of blocks in ENGINEER = $\lceil 1{,}600 / \lfloor 512/50 \rfloor \rceil = 160$ blocks
- Number of blocks in PROJECT = $\lceil 3{,}200 / \lfloor 512/80 \rfloor \rceil = 534$ blocks
- Sort PROJECT on ICEngID
  - initial number of runs = $\lceil 534 / 3 \rceil = 178$
  - number of merge passes = $\lceil \log_2 178 \rceil = 8$
  - cost = 534 * (2 * 8 + 1) = 9,078
- Cost of writing the sorting output = 534
- Cost of merge join = 160 + 534 = 694
- Total cost = 9,078 + 534 + 694 = 10,306

e) Is it possible to reduce the ***worst-case cost*** in ***number of disk block transfers*** if the query is computed with the ***indexed nested-loop join*** algorithm instead? Draw a revised evaluation plan and show the steps clearly to support your answer (no steps, no marks).

[6 marks]



- Number of projects per engineer
  = 3200 / V(ICEngID, PROJECT)
  = 3200 / 1600 = 2
- Cost of indexed nested-loop join = 160 + 1,600 * (3+2) = 8,160.
- So, the cost is reduced.

- END -