University of Glasgow | School of Computing Science

Assessed Coursework

Course Name	Database Systems (H)			
Coursework Number	1/1			
Deadline	Time:	16h30	Date:	11/03/2022
% Contribution to final course mark	20%			
Solo or Group ✓	Solo		Group	✓
Anticipated Hours	Average 20 hours			
Submission Instructions	Submission of only <u>one</u> PDF document; please read the description.			
Please Note: This Coursework cannot be Re-Assessed				

You must complete an "Own Work" form via https://studentltc.dcs.gla.ac.uk/ for all coursework

Data Management 2022

Task 1 [Marks: 20]

***Objective*: Experimentation with different structures of hash files over *unique and non-unique* attributes and assessing the impact of the attribute's uniqueness on the expected cost.**

The relation EMPLOYEE(SSN, DNO) has as primary key the SSN attribute. DNO is a non-unique attribute indicating the department number an employee is working on. There are 1,000 tuples (employees) and 100 departments. We assume that DNO is uniformly distributed across the employees such that there are 10 employees per department. The blocking factor of the file that accommodates the EMPLOYEE relation is 5 employees per block.

1.A: Consider a hash file that stores the EMPLOYEE relation generated by a hash function over the unique SSN. We assume that the hash function uniformly distributes the employees in M = 5 buckets. Assume the SQL1 query:

```
SQL1: SELECT * FROM EMPLOYEE WHERE SSN = x.
```

- Calculate the expected number of block accesses of SQL1 for any random SSN = x value in the *average case* scenario.
- Consider that the tuples of all the blocks of each bucket of the hash file are now sorted with respect to SSN. In this case, calculate the expected number of block accesses of SQL1 for any random SSN = x value in the *average case* scenario.

[10 Marks]

1.B: Consider a hash file that stores the EMPLOYEE relation generated by a hash function over the non-unique DNO. We assume that the hash function uniformly distributes the employees in M = 5 buckets. Assume the SQL2 query:

```
SQL2: SELECT * FROM EMPLOYEE WHERE DNO = y.
```

- Calculate the expected number of block accesses of SQL2 for any random DNO = y value in the *average case* scenario.
- Consider that the tuples of all the blocks of each bucket of the hash file are sorted with respect to DNO. Then, calculate the expected number of block accesses of SQL2 for any random DNO = y value in the *average case* scenario.

[10 Marks]

Task 2 [Total Marks: 20]

***Objective*: Experimentation with building a B+ Tree over a non-unique, non-ordering attribute and calculating the expected cost of *range selection* queries using B+ Tree.**

Consider the relation EMPLOYEE (SSN, TaxCode). TaxCode is a non-ordering and non-unique attribute. We have built a B+ Tree over the Tax code with the following context:

- There are r = 100,000,000 tuples of employees;
- There are n = 12,288 distinct values of the Tax code attribute;
- The size of a data block is B = 64 bytes and the size of an employee tuple is R = 32 bytes;
- The size of any pointer is P = 10 bytes and the size of the Tax code attribute is V = 5 bytes;
- The Tax code values are integers starting from 1 to 12288, i.e., Tax code ∈ {1, 2, 3, …, 12288};
- The Tax codes values are uniformly distributed over the tuples;
- All leaf nodes of the B+ Tree are 100% full of Tax code values.
- Any internal tree node and leaf node can fit in 1 block.

Assume the SQL3 query over the Tax code:

```
SQL3: SELECT * FROM EMPLOYEE WHERE TaxCode <= x
```

with random tax code value x ∈ {1, 2, 3, …, 12288}. Let us define the ratio w = x/n denoting the fraction of distinct tax code values from 1 to x out of all the distinct tax code values n. The ratio w is a real number ranging from w = 1/n when x = 1, to w = 1 when x = n. This ratio denotes the percentage of the distinct tax code values being considered in the WHERE clause of SQL3. For instance, w = x/n = 0.5 indicates that we focus on retrieving employees whose distinct tax code values refer to 50% of all the distinct available tax code values, i.e., starting from 1 to 6144. In this example, the range query could be: SELECT * FROM EMPLOYEE WHERE TaxCode <= 6144

Which should be the minimum ratio w = x/n such that searching by using the B+ Tree is *more* efficient than using a linear scan of the file in terms of block accesses?

[20 Marks]

Task 3 [Total Marks: 20]

Objective: **Experimentation with adopting B+ Trees for processing *aggregate* queries and calculating the corresponding expected cost.**

Consider the relation EMPLOYEE (SSN, Surname, Salary), which has r = 4,500 employees.

We are thinking of building two B+ Tree indexes:

- B+ Tree B1 will be built over the unique attribute SSN with order p = 30 tree pointers per internal node (i.e., 29 values per node), and pL = 5 pointers per leaf node (i.e., 5 values/pointers per leaf node + 1 sibling node pointer).
- B+ Tree B2 will be built over the non-unique SALARY with order p = 5 tree pointers per internal node (i.e., 4 values per node), and pL = 5 pointers per leaf node (i.e., 5 values/pointers per leaf node + 1 sibling node pointer). There are 625 distinct SALARY values. The salary values are *uniformly distributed* across all the employees.

All leaf nodes of each B+ Tree should have 100% full of values. Any internal tree node and leaf node can fit in 1 block. We can store 8 block-pointers per block.

Consider the aggregate query:

```
SQL4: SELECT MIN(SALARY), AVG(SALARY) FROM EMPLOYEE
```

3.A. Build the B1 B+ Tree and report on the levels and the structure of the leaf nodes. Then provide an algorithm/procedure where you use the B1 tree to process the SQL4 query and calculate the expected cost. (You could e.g., simply describe the steps of your procedure/algorithm or provide a pseudocode and explain the cost calculation).

[10 Marks]

3.B. Build the B2 B+ Tree and report on the levels and the structure of the leaf nodes. Then provide an algorithm/procedure where you use the B2 tree to process the SQL4 query and calculate the expected cost. (You could e.g., simply describe the steps of your procedure/algorithm or provide a pseudocode and explain the cost calculation).

[10 Marks]

Notes & Submission

Note 1: The Assessed Group Work is graded out of 60 Marks: Task 1 [20 marks], Task 2 [20 marks] and Task 3 [20 marks]. The group submits *one* product, and all group members receive the *same* grade.

Note 2: Answer ALL tasks.

Note 3: **Only one member** of your group will need to submit a document including your group details and answers in **one** PDF document file (use whatever you like as title of the PDF file).