University of Glasgow

DB H SUMMER 2021
Duration: 90 minutes

DEGREES of MSci, MEng, BEng, BSc, MA and MA (Social Sciences)

Database Systems H

Answer All Questions

This examination paper is worth a total of 60 marks

INSTRUCTIONS TO INVIGILATORS

Please collect all exam question papers and exam answer scripts and retain for school to collect. Candidates must not remove exam question papers.

Note: The Exam Duration is 90 Minutes

Part A: Relational Modelling and SQL [20 Marks]

Question 1.

(a). Consider the following relation:

EMPLOYEE(SSN, NumberPlate, PassportNumber, Name, Department)

storing information about an employee (identified by the unique social security number SSN), their vehicle's number plate (which is unique for each vehicle), their passport number (which is unique for each employee), their name, and the department they belong to. A database analyst has decided to use the attribute SSN as the Primary Key (which is underlined in the schema).

(i) Is: SSN → PassportNumber a *transitive* functional dependency? Explain briefly your answer. **[1]**

(ii) Is: SSN → Name a *transitive* functional dependency? Explain briefly your answer. **[1]**

(iii) Is: PassportNumber → Department a *transitive* functional dependency? Explain briefly your answer. **[2]**

(iv) Is the composite attribute {PassportNumber, SSN} a candidate key? Based on the definition of the super key and the candidate key, provide an explanation to your answer. **[2]**

(v) Could we claim the following equivalence?

"If in a relation **R**(X, Y), it holds true that X → Y and Y → X then both X and Y are candidate keys and vice versa".

Explain your answer by providing an example. **[2]**

(b). Which is the expected result of the following query? Explain briefly your answer. **[2]**

```
SELECT * FROM EMPLOYEE WHERE
NOT EXISTS (SELECT 0 FROM EMPLOYEE WHERE 1 IS NOT NULL OR NULL);
```

(c). Consider the following relational schema:

WORKS_ON(ESSN, PID, HOURS)

DEPENDENT(ESSN, DEPENDENT_NAME)

such that the relation WORKS_ON stores for a specific employee (with unique social security number ESSN), the corresponding project they work in (with unique project identifier PID) and the number of hours per week (HOURS) they work on this project. An employee can work in more than one project, while more than one employee can work in a project. The relation DEPENDENT associates an employee (identified through the ESSN) with their dependent (storing the name of the dependent). An employee can have any number of dependents or none. The Primary Keys are underlined in both relations.

For each project in which all employees work at least 20 hours per week, show how many of the employees have more than 2 dependents. **[10]**

Part B: File Organization [20 Marks]

Question 2. Assume the relation EMPLOYEE(SSN, Name, DNO) which is stored in a file on a disk. We need 10 bytes for the integer Primary Key attribute SSN (Social Security Number), 150 bytes for the attribute Name, and 10 bytes for the department number attribute DNO. Consider that the relation has r = 12 tuples and the size of the file block is 512 bytes.

(a). We adopt external hashing for storing the relation EMPLOYEE using M = 3 buckets and the hash function $y = h(\text{SSN}) = \text{SSN mod M}$, i.e., we hash the key SSN attribute. We assume that the size of each bucket is equal to the size of a block, i.e., 512 bytes. The hash file accommodates r = 12 tuples with SSN values: **0, 1, 2, 3, 4, 5, 7, 9, 10, 13, 15, 16, 27.** If a bucket is *full*, then we use overflow blocks connected through *chain pointers*.

(i). Assumption 1: We assume that *all* buckets are equi-probable to be selected given any random selection query over the SSN attribute.

Based on Assumption 1, calculate the *expected* number of block accesses (I/O blocks read/write) for a random SQL selection query using the hashing structure:

```
SELECT * FROM EMPLOYEE WHERE SSN = x,
```

for any random x value, in the *worst-case* scenario and *average-case* scenario.

Note: In the worst-case scenario the tuple is found in the *last* overflow block of the chain, if exists.

[4]

(ii). Assumption 2: We assume that, given a random selection query over the SSN attribute, a bucket is chosen based on a specific probability which is expressed by the percentage of blocks belonging to that bucket out of the total number of blocks of the hash file.

Based on Assumption 2, calculate the *expected* number of block accesses (I/O blocks read/write) for a random SQL selection query using the hashing structure:

```
SELECT * FROM EMPLOYEE WHERE SSN = x,
```

for any random x value, in the *worst-case* scenario and *average-case* scenario. **[4]**

Note: In the worst-case scenario the tuple is found in the *last* overflow block of the chain, if exists.

(b). Assume that we issue many selection queries over the relation EMPLOYEE such that p% of those queries involve the SSN attribute and (100-p)% do not include the SSN attribute. The p% value is between 0% and 100%.

(i). Based on Assumption 1 in Question 2(a).(i), provide a *decision making rule* based on the p% value such that we should store the relation in a sequential file sorted by SSN instead of hashing the file with respect to SSN. **[4]**

(ii). Based on Assumption 2 in Question 2(a).(ii), provide a *decision making rule* based on the p% value such that we should store the relation in a sequential file sorted by SSN instead of hashing the file with respect to SSN. **[4]**

Note: Base your reasoning on the *worst-case expected* cost for executing any random query over the relation EMPLOYEE (in terms of block accesses). Moreover, in the worst-case scenario, the expected cost in block accesses over a sequential file of b blocks is estimated $\log_2 b$.

(c). Based on your derived decision rules from Question 2.(b).(i) and Question 2.(b).(ii), briefly explain the impact of adopting Assumption 2 on the performance of external hashing methodology in terms of the expected number of block accesses. **[4]**

Part C: Query Processing and Optimization [20 Marks]

Question 3. Consider the relation EMPLOYEE(SSN, Salary, Age, DNO), where SSN is the social security number and DNO is the department number, and the selection query:

```
SELECT *
FROM  EMPLOYEE
WHERE (AGE = 60 OR AGE = 25) AND (DNO = 5)
       AND (Salary = 60000 OR Salary = 30000)
```

Consider also the following information of the database system:

- **Clustering Index** on the Age *non-key ordering* attribute with *t* levels.
- **B+ Tree Secondary Index** on the DNO *non-key non-ordering* attribute with $x_{DNO} = 3$ levels.
- **B+ Tree Secondary Index** on the Salary *non-key non-ordering* attribute with $x_{Salary} = 3$ levels.
- The number of the distinct values of DNO is 100.
- The number of the distinct values of Salary is 500.
- The number of the distinct values of Age is 200.
- The relation EMPLOYEE has $r = 10000$ tuples, the size of each record is R = 250 bytes (DNO = 50 bytes, Salary = 100 bytes, Age = 50 bytes, SSN = 50 bytes), the block size is B = 512 bytes, and each pointer has size V = 50 bytes.
- The available memory in the database system is 200 blocks.
- **Assumption**: all the values of each attribute are uniformly distributed across the tuples.

(a). Calculate the level *t* of the multi-level Clustering index over the non-key ordering Age. **[3]**

(b). Which is the *selection selectivity* of the query? **[2]**

(c). Propose **three** *query processing plans* for implementing the selection query using the provided access paths and select the *best* in terms of the number of block accesses. **[7]**

Question 4. Consider the relations EMPLOYEE(SSN, NAME, SALARY) and DEPENDENT(ESSN, DEPENDENT_NAME) with $r = 500$ employees and $d = 30$ dependents, respectively. Each attribute in the EMPLOYEE relation has size 10 bytes. Each attribute in the DEPENDENT relation has size 10 bytes with m = NDV(ESSN) = 10 distinct social security numbers (NDV stands for Number of Distinct Values). The ESSN values are uniformly distributed across the DEPENDENT tuples. The size of the block is B = 64 bytes and any pointer has size V = 10 bytes. Consider the join query:

```
SELECT E.NAME, D.DEPENDENT_NAME
FROM   EMPLOYEE AS E, DEPENDENT AS D
WHERE  E.SSN = D.ESSN
```

and consider the following available access paths:

- **Clustering Index** over the ESSN in the relation DEPENDENT.
- **B+ Tree Secondary Index** over the SSN with $x_{SSN} = 3$ levels in the relation EPMPLOYEE.

(a). Calculate the *join cardinality* and the blocking factor of the result block, i.e., the number of matching tuples per block. **[3]**

(b). Provide **<u>two</u>** *join processing methods* using the available access paths and select the *best* one in terms of block accesses. **[5]**