**CITY UNIVERSITY OF HONG KONG**

Course code & title : CS3402 Database Systems

Session : Semester A 2019/20

This paper has 13 pages (including this cover page).

1. This paper consists of Five questions.

2. Write down your answer in the space provided.

*This is an* ***open-notes*** *examination.*

*Candidates are allowed to use the following materials/aids:*

*Book,printed lecture notes, personal notes, and other course handout materials.*

|  |  |
| --- | --- |
| **Seat Number** |  |
| **Student ID** | 55216557 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Q1 (10%) | Q2 (24%) | Q3 (30%) | Q4 (16%) | Q5 (20%) | Total (100%) |
|  |  |  |  |  |  |

**Question 1 (10 marks): Basic RDBMS Concepts**

Determine whether each of the following statements is true or false and justify your answers. No mark will be given if your justification is wrong.

1. (2 marks): To perform a binary search on a relation, we first need to build a B+-tree.

|  |
| --- |
| ***False, B-tree can be used to do binary search.*** |

1. (2 marks): To process any SQL query with a WHERE clause, selection has to be performed before projection.

|  |
| --- |
| ***True, in order to process any SQL query with a where clause, we have to select a subset of the tuples from a relation based on a selection condition. After that, we use projection to keep the result in another file.*** |

1. (2 marks): Primary index is built on prime attribute and clustering index is built on non-prime attribute (An attribute is prime attribute if it belongs to one candidate key).

|  |
| --- |
| ***False, Primary index is built on all prime attributes, not only a proper subset of candidate key.*** |

1. (2 marks): In an extendible hashing system, each hash code is a bit string value that is associated with exactly one bucket.

|  |
| --- |
| ***False***  ***One bucket only contains a bit string hash code.*** |

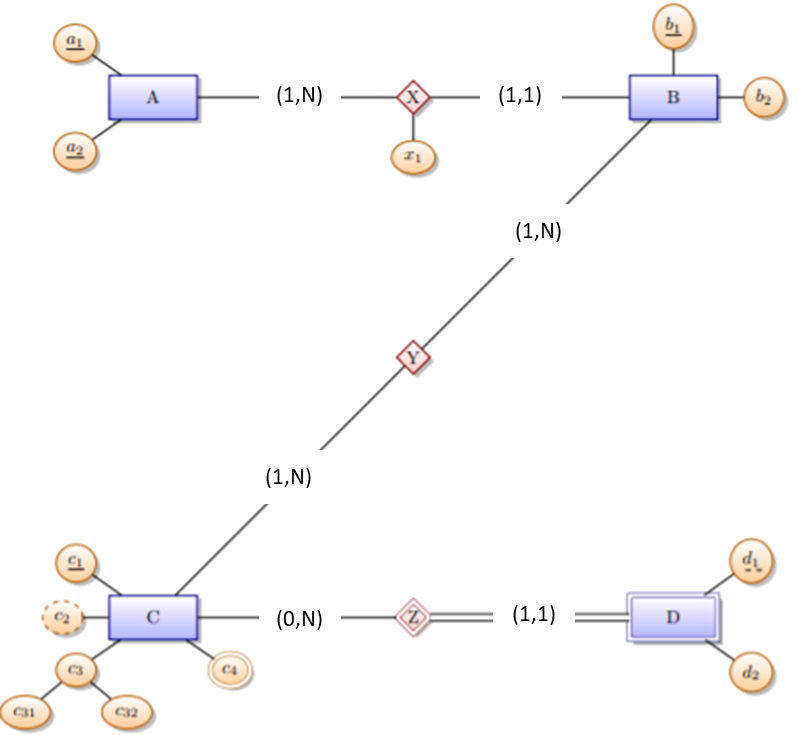
1. (2 marks): Given two relations R(a,b,c) and S (c,d,e), the results of R\*S and RS are not equivalent.

|  |
| --- |
| ***True***  R\*S = (a, b, c, d, e)  RS = (a, b, c, c, d, e) |

**Question 2 (24 marks): Database Design**

1. (12 marks) Take the following ER-model and translate it into a relational schema using the rules presented in class. Present the relational schema using the notation from the slides. For example, a relation R with attributes a1 and a2 where a2 is the primary key is written as R(a1, a2). You also need to specify foreign key constraints by arrows. You do

not have to show intermediate results.



(b) (12 marks) Consider the relation with schema R (A, B, C, D, E, F) and the following functional dependencies (FDs): A→BC, D→AF

一張含有 文字 的圖片

自動產生的描述

1. (4 marks) Please write down the candidate keys of the relation, and proof it with closure of attribute.

D→AF, A→BC (given)

D→ABCDF

DE→ABCDEF

D+ = {A, B, C, D, F}

E+ = {E}

So {D, E}+ is candidate key

1. (4 marks) Is relation R in BCNF? If it is, explain why it is. If it is not, explain why not and give a decomposition of R into a collection of relations that are in BCNF.

No, R is not 3NF. Transitivity dependency occur:

D→AF, A→BC (given)

D→BCF

R1(D, E)

R2(D, A, F)

R3(A, B, C)

1. (4 marks) About 3NF and BCNF, which of the following statements are **TRUE**? (Multiple choices)  
   (A) Every relation in 3NF is also in BCNF  
   (B) A relation R is in 3NF if every non-prime attribute of R is fully functionally dependent on every key of R  
   (C) Every relation in BCNF is also in 3NF  
   (D) No relation can be in both BCNF and 3NF

(E) If a 3NF table has only one candidate key, then this table is also in BCNF.

B, C

**Question 3 (30 marks): Querying**

1. (6 marks) Consider the following relations containing students and courses information:

STUDENT(SNAME, STUDENTID, BDATE, ADDRESS, DNUM)

COURSE(CNAME, COURSEID, LEVEL, LECTURER\_NAME)

COURSE\_TAKING(COURSEID, STUDENTID, GRADE)

Note the primary key of each relation is underlined, and the foreign key reference is indicated by arrows. Translate the relation definition into create table statements and include all necessary primary key and foreign key constraints.

CREATE TABLE STUDENT

(SNAME VARCHAR(15) NOT NULL,

STUDENTID CHAR(9) NOT NULL,

BDATE DATE

ADDRESS VARCHAR(30)

DNUM INT NOT NULL,

PRIMARY KEY (STUDENTID));

CREATE TABLE COURSE

(CNAME VARCHAR(15) NOT NULL,

COURSEID CHAR(9) NOT NULL,

LEVEL CHAR(2)

LETURER\_NAME CHAR(15) NOT NULL,

PRIMARY KEY (COURSEID),

UNIQUE (CNAME));

CREATE TABLE COURSE\_TAKING

(COURSEID CHAR(9) NOT NULL,

STUDENTID CHAR(9) NOT NULL,

GRADE CHAR(2) NOT NULL,

PRIMARY KEY (COURSEID, STUDENTID),

FOREIGN KEY (COURSEID) REFERENCES COURSE (COURSEID),

FOREIGN KEY (STUDENTID) REFERENCES STUDENT (STUDENTID));

1. (12 marks) Based on the schema in (a), compose SQL statements for the following queries.
2. (4 marks) Display the course id, name and the number of students taking each course.

SELECT S.COURSEID, S.SNAME, COUNT(T.STUDENTID)

FROM COURSE AS S, COURSE\_TAKING AS T

GROUP BY S.COURSEID;

1. (4 marks) Display the student id, student name and the GPA of each student who score lower than 60. Assume that all courses carry the same credit weight and the possible grades of each course are in the range of [0,100].

SELECT S.STUDENTID, S.SNAME, T.GPA

FROM STUDENT AS S, (

SELECT STUDENTID, AVERAGE(GRADE) AS GPA

FROM COURSE\_TAKING

GROUP BY STUDENT\_ID

) AS T

WHERE S.STUDENTID = T.STUDENTID AND T.GPA < 60;

.

.

1. (4 marks) Display the name of the student, if he/she has the same name as his/her lecturer.

SELECT SNAME

FROM STUDENT AS S, (

SELECT STUDENTID, LECTURER\_NAME

FROM COURSE\_TAKING AS T, COURSE AS C

WHERE T.COURSEID = C.COURSEID) AS G

WHERE S.SNAME = G.LECTURER\_NAME AND S.SID = G.SID;

1. (12 marks) Based on the schema in (a), write down the relational algebra expressions for the following queries.
2. (4 marks) Display id of the course with no enrolment.

Π COURSEID (COURSE - COURSE\*COURSE\_TAKING)

1. (4 marks) Display id of the student who takes all courses ‘Mary’ takes.

MARYID ← Π COURSEID, STUDENTID (COURSE\_TAKING \* (σSNAME = Mary STUDENT))

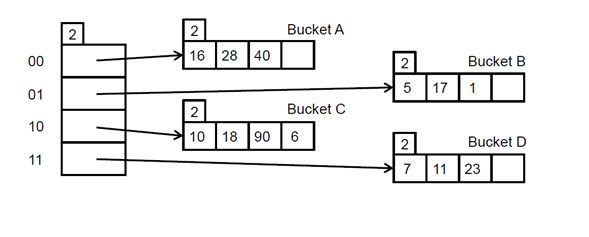
Π STUDENTID (COURSE ⋈ COURSEID=COURSEID MARYID) - Π STUDENTID MARYID

1. (4 marks) Display the name of the student, if the student gets the highest grade on some course he/she takes.



**Question 4 (16 marks):**

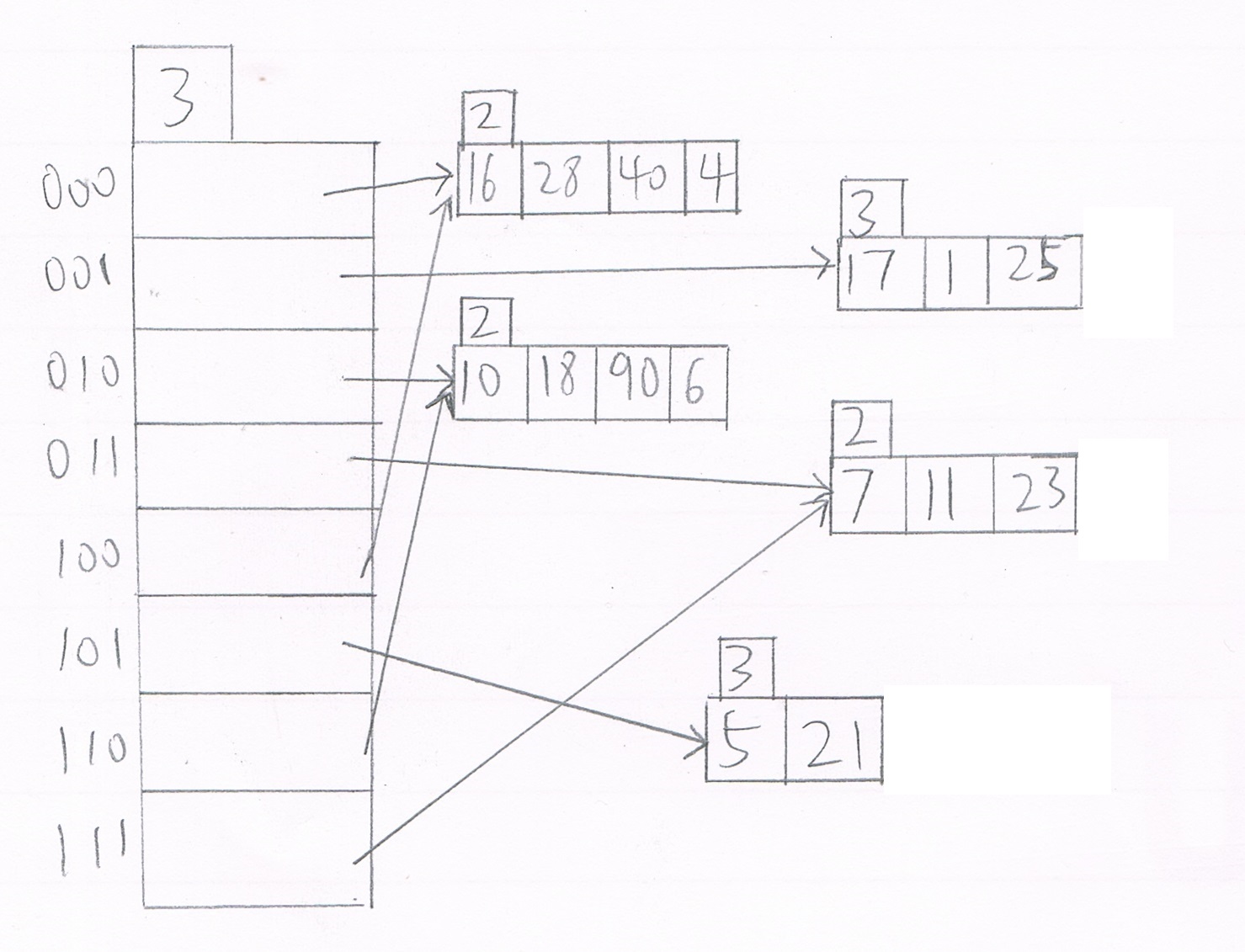
1. (8 marks)Consider the following extendible hash.



* 1. (4 marks) Draw the index after the following keys have been inserted: 4, 21, and 25.

Hints: decimal-binary conversion table

|  |  |
| --- | --- |
| Decimal | Binary |
| 1 | 1 |
| 4 | 100 |
| 5 | 101 |
| 7 | 111 |
| 10 | 1010 |
| 11 | 1011 |
| 16 | 10000 |
| 17 | 10001 |
| 18 | 10010 |
| 21 | 10101 |
| 23 | 10111 |
| 25 | 11001 |
| 28 | 11100 |
| 40 | 101000 |
| 90 | 1011010 |



* 1. (4 marks) Having inserted the above three keys in Question4(1), what is the minimum number of delete operations needed for the global depth to decrease? And which keys should be deleted to achieve that? (Hint: A bucket is merged with its split image if and only if it becomes empty.)

Minimum number of delete operations is 2. Key 5 and key 21 should be deleted. In this case, hash index 101 is deleted and only the following hash index left

000, 100 → 00

001 → 01

010, 110 → 10

011, 111 → 11

This reduce the hash index from the original 3 bits to 2 bits, and this decrease the global length from 3 to 2.

1. (8 marks)Consider a relational database with two table schemes:

Course (c-name, room, instructor)

Enrollment (student-name, c-name, grade)

(1) (4 marks) To facilitate as efficiently as possible such a query like “given a course, find out who are the students taking the course”, what file structure will you recommend to use, and why?

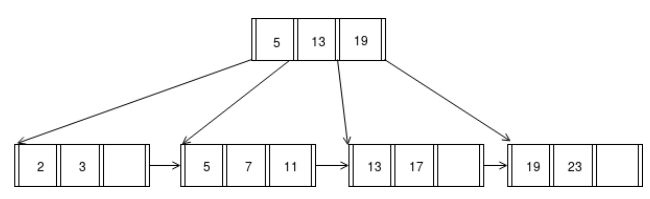
Hash file. Since block address is know by hash function, accessing any record is faster. Similarly updating or deleting a record is also very quick.

(2) (4marks) To support as efficiently as possible such a query like “given an instructor, find out who are the students taught by him/her”, what kind of indexing will you recommend to use and how to use it?

B+ tree index. It guarantees the number of disk operations is the order of logarithmic in the worst case. Also, the maximum height of the tree is the logarithm of the number of records. Hence, the size of index file is much smaller than using other indexing method.

**Question 5 (20 marks): Indexing**

1. (10 marks) Consider the following B+ tree with order d=4 and height h=2.

****

When answering the following questions, please follow the assumptions:

• A left pointer in an internal node guides towards keys < than its corresponding key, while a right pointer guides towards keys

• A leaf node underflows when the number of **keys** goes bellow .

• An internal node underflows when the number of **pointers** goes below .

1. (4 marks) Insert 10 into the B+tree. Draw the resulting tree.

一張含有 文字, 地圖 的圖片

自動產生的描述

1. (2 marks) After (1), how many pointers (parent-to-child and sibling-to-sibling) do you chase to find all keys between 5 and 15?

6

1. (4 marks) Then delete 23. Draw the resulting tree?

一張含有 物件 的圖片

自動產生的描述

1. (10 marks) Consider a disk with block size B=256 bytes. A block pointer is P=7 bytes long, and a record pointer is P R =7 bytes long. A file has r=30,000 EMPLOYEE records of fixed-length. Each record has the following fields: NAME (30 bytes), SSN (9 bytes), DEPARTMENTCODE (9 bytes), ADDRESS (40 bytes), PHONE (10 bytes), BIRTHDATE (8 bytes), SEX (1 byte), JOBCODE (4 bytes), SALARY (8 bytes, real number). An additional byte is used as a deletion marker. Suppose the file is stored with an unspanned organization and file is ordered by the key field SSN.
   1. (2 marks) Calculate the record size R in bytes and the blocking factor bfr.

Record size R = (30 + 9 + 9 + 40 +10 + 8 + 1 + 4 + 8) + 1 = 120 bytes

Blocking factor bfr = floor (256/120) = 2 records per block

* 1. (2 marks) If we want to construct a single-level primary index on SSN, calculate the index blocking factor bfr\_i;

Index record size Ri = (9 + 7) = 16 bytes

Index blocking factor bfr\_i = floor (256/16) = 16

* 1. (2 marks) Based on (1)&(2), calculate the number of index entries and the number of index blocks.

Number of index entries = 30000/2 = 15000 entries

Number of index blocks = 15000/16 = 938 blocks

* 1. (4 marks) If we want to construct a multi-level index on SSN, calculate the number of levels needed and the total number of blocks required.

Number of second-level index entries = 938 entries

Number of second-level index blocks = ceiling (938/16) = 59 blocks

Number of third-level index blocks = ceiling (59/16) = 4 blocks

Number of fourth-level index blocks = ceiling (4/16) = 1 blocks

The fourth level is the top index level since it has only one block.

Hence, the index has x = 4 levels.

Total number of blocks required = 938 + 59 + 4 + 1 = 1002 blocks.