SIM GLOBAL EDUCATION

UOW AUSTRALIA

Family Name .....

First Name .....

SCIT

School of Computing and

Information Technology

CSCI317
Database Performance Tuning

This paper is for students studying at the Singapore Institute of Management Pte Ltd.

S3-2021 FINAL EXAMINATION

Date: 9 September, 2021

Time: 10.00 am – 1.00 pm SGT + 30 minutes submission time

Exam value: 40% of the subject assessment
Marks available: 40 marks

DIRECTIONS TO CANDIDATES

(1) The answers to the questions included in the final examination must be hand-written with a BLACK or DARK BLUE PEN on the WHITE PIECES of paper in A4 format. No pencil and no other colour of paper is allowed.

(2) When finished, take the pictures of the hand-written solution, save the pictures in files (`pdf`, `jpeg`, `jpg`, `gif`, `bmp`, `png`, `tiff` formats are all acceptable), and submit the files through Moodle. Using mobile phone cameras is all right. It is possible to take more than one picture per answer to assure the good readability of an answer. The marks will be deducted for submissions in the different formats. No more than 20 files can be submitted and no more than 200Mbytes can be submitted. Please well plan your pictures.

(3) The files should have the names indicating a number of the respective question in the final examination paper like `q1`, `q2`, ... and `q1-1`, `q1-2`, ... when more than one picture is used for an answer of a question.

(4) All answers including the drawings must be hand-written. No printed material will be evaluated. No iPad or other tablet is allowed. The solutions must be hand-written on the pieces of paper. Submission of the typed and/or electronically processed text is a violation of the final/deferred/supplementary examination regulations and it will be considered as a medium level academic misconduct with all consequences coming from such fact.

(5) Marks will be deducted for the late submissions at a rate of 1 mark per 1 minute late.

Introduction

The questions 2, 4, 5, and 6 of the examination paper are related to the following simplified version of TPC-H benchmark database used in the laboratory classes.

```
CUSTOMER(
C_CUSTKEY       NUMBER(12)      NOT NULL,
C_NAME          VARCHAR(25)     NOT NULL,
C_ADDRESS       VARCHAR(40)     NOT NULL,
C_NATIONKEY     NUMBER(12)      NOT NULL,
C_ACCTBAL       NUMBER(6)       NOT NULL,
C_PHONE         NUMBER(12)      NOT NULL,
CONSTRAINT CUSTOMER_PKEY PRIMARY KEY(C_CUSTKEY) );

PART(
P_PARTKEY       NUMBER(12)      NOT NULL,
P_NAME          VARCHAR(55)     NOT NULL,
P_BRAND         CHAR(10)        NOT NULL,
P_SIZE          NUMBER(12)      NOT NULL,
P_RETAILPRICE   NUMBER(12,2)    NOT NULL,
CONSTRAINT PART_PKEY PRIMARY KEY (P_PARTKEY) );

PARTSUPP(
PS_PARTKEY      NUMBER(12)      NOT NULL,
PS_SUPPNAME     VARCHAR(55)     NOT NULL,
PS_AVAILQTY     NUMBER(12)      NOT NULL,
CONSTRAINT PARTSUPP_PKEY PRIMARY KEY (PS_PARTKEY,PS_SUPPNAME),
CONSTRAINT PARTSUPP_FKEY FOREIGN KEY(PS_PARTKEY)
         REFERENCES PART(P_PARTKEY) );

ORDERS(
O_ORDERKEY      NUMBER(12)      NOT NULL,
O_CUSTKEY       NUMBER(12)      NOT NULL,
O_TOTALPRICE    NUMBER(12,2)    NOT NULL,
O_ORDERDATE     DATE            NOT NULL,
CONSTRAINT ORDERS_PKEY PRIMARY KEY (O_ORDERKEY),
CONSTRAINT ORDERS_FKEY1 FOREIGN KEY (O_CUSTKEY)
                REFERENCES CUSTOMER(C_CUSTKEY) );

LINEITEM(
L_ORDERKEY      NUMBER(12)      NOT NULL,
L_PARTKEY       NUMBER(12)      NOT NULL,
L_LINENUMBER    NUMBER(12)      NOT NULL,
L_QUANTITY      NUMBER(12,2)    NOT NULL,
L_SHIPDATE      DATE            NOT NULL,
L_TAX           NUMBER(4,2)     NOT NULL,
     CONSTRAINT LINEITEM_PKEY PRIMARY KEY (L_ORDERKEY, L_LINENUMBER),
     CONSTRAINT LINEITEM_FKEY1 FOREIGN KEY (L_ORDERKEY)
                REFERENCES ORDERS(O_ORDERKEY),
     CONSTRAINT LINEITEM_FKEY2 FOREIGN KEY (L_PARTKEY)
                REFERENCES PART(P_PARTKEY) );
```

Assume that, the relational tables listed above occupy the following amounts of persistent storage:

```
CUSTOMER          100 Mbytes
PART              100 Mbytes
PARTSUPP          200 Mbytes
ORDERS            300 Mbytes
LINEITEM          500 Mbytes
```

Question 1 (7 marks)

Consider the following fragment of query processing plan.

```
PLAN_TABLE_OUTPUT
-------------------------------------------------------------------------------------------
Plan hash value: 4005457340

-------------------------------------------------------------------------------------------
| Id  | Operation            | Name     | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT     |          |   449K|    30M|       | 15866   (1)| 00:00:01 |
|*  1 |  HASH JOIN RIGHT ANTI|          |   449K|    30M|       | 15866   (1)| 00:00:01 |
|*  2 |   TABLE ACCESS FULL  | LINEITEM |     1 |     9 |       | 12152   (1)| 00:00:01 |
|*  3 |   HASH JOIN          |          |   450K|    26M|  2776K|  3712   (1)| 00:00:01 |
|   4 |    TABLE ACCESS FULL | CUSTOMER |  45000|  2241K|       |   389   (0)| 00:00:01 |
|*  5 |    TABLE ACCESS FULL | ORDERS   |   450K|  4833K|       |  2696   (1)| 00:00:01 |
-------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - access("L_ORDERKEY"="ORDERS"."O_ORDERKEY")
   2 - filter("L_TAX">20)
   3 - access("C_CUSTKEY"="O_CUSTKEY")
   5 - filter("O_CUSTKEY">=0)
```

(1) Find and draw a syntax tree of the query processing plan listed above. To draw a syntax tree, use the relational algebra operations (and NOT Oracle query processing plan operations !) explained during the lecture classes.

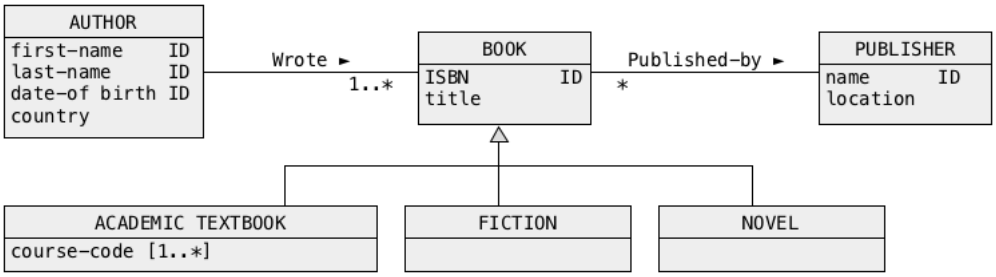
(3 marks)

(2) Discover and write `SELECT` statement that may have a query processing plan listed above.

(4 marks)

Question 2 (7 marks)

A conceptual schema given below represents a database domain where the authors write the books published by the publishers. For simplicity, we assume, that each book is written by one author. We consider three types of books: academic textbooks, fiction books, and novels. Textbooks are described by codes of courses the textbooks were used for. All other attributes are self-explanatory.



(1) Perform simplification of a conceptual schema given above and re-draw the simplified conceptual schema.

(2 marks)

(2) We would like to improve the performance of the following class of applications:

Find the names and location of publishers (`name, location` in a class `PUBLISHER`) who published the academic textbooks written by the authors living in a given country (`country` in a class `AUTHOR`) and used as academic textbooks by the courses with the give code(s) (attribute `course-code` in a class `ACADEMIC TEXTBOOK`).

The following application belongs to the class of applications given above.

Find the names and location of publishers who published the academic textbooks written by the authors living in a class `Australia` and used as academic textbooks in the courses `CSIT115, CSCI235, and CSCI317`.

To improve performance, consider the following transformations of a simplified conceptual schema: denormalization, decompositions, and appropriate implementation of generalization.

List the transformations applied and re-draw a conceptual schema after all transformations.

(5 marks)

Question 3 (6 marks)

A relational table `ORDERS` contains information about the orders submitted by the customers.

`ORDERS(order#, order-date#, product, quantity, price-per-unit)`

A relational table `ORDERS` has a primary key `(order#)`

Assume that:

(i) a relational table `ORDERS` occupies `1000` data blocks,
(ii) a blocking factor in a relational table `PARTSUP` is `50` rows per block,
(iii) a relational table `ORDERS` contains information about `200` products,
(iv) a relational table `ORDERS` contains information about `100` prices per unit,
(v) a primary key is automatically indexed,
(vi) an attribute `product` is indexed,
(vii) an attribute `price-per-unit` is indexed,
(viii) all indexes are implemented as B*-trees with a fanout equal to `10`,
(ix) a leaf level of an index on attribute `price-per-unit` consists of `30` data blocks,
(x) a leaf level of an index on primary key consists of `200` data blocks.

For each one of the following queries briefly describe how the database system processes each query and estimate the total number of read block operations needed to compute each query. There is no need to perform the final computations. A correctly constructed formula filled with the appropriate constants is completely sufficient.

Do not include the results from processing of `EXPLAIN PLAN` statement.

(1)
```
SELECT product
FROM ORDERS
WHERE product = 'bolt' OR quality = 'average';
```

(2)
```
SELECT count(*)
FROM ORDERS
WHERE product IN ('bolt', 'screw');
```

(3)
```
SELECT price-per-unit, COUNT(*)
FROM ORDERS
GROUP BY price-per-unit
HAVING count(*) > 5;
```

(4)
```
SELECT order#, product, quality
FROM ORDERS
ORDER BY order#, product;
```

(5)
```
SELECT *
FROM ORDERS
WHERE order# = 12345 OR product = 'bolt';
```

(6)
```
SELECT order#, product
FROM ORDERS;
```

Question 4 (6 marks)

Consider the following `SELECT` statements.

```
SELECT COUNT(DISTINCT O_TOTALPRICE)
FROM ORDERS;

SELECT *
FROM CUSTOMER JOIN ORDERS
              ON CUSTOMER.C_CUSTKEY = ORDERS.O_CUSTKEY
WHERE C_ACCTBAL = 1000 AND O_TOTALPRICE = 1000;

SELECT *
FROM ORDERS JOIN LINEITEM
            ON ORDERS.O_ORDERKEY = LINEITEM.L_ORDERKEY
WHERE O_ORDERDATE = '09-SEP-2021' OR L_QUANTITY = 500;

SELECT *
FROM LINEITEM JOIN PART
              ON LINEITEM.L_PARTKEY = PART.P_PARTKEY
WHERE L_QUANTITY = 100 AND P_NAME = 'bolt';

SELECT COUNT(PS_AVAILQTY)
FROM PARTSUPP;
```

(1) Find the smallest number of indexes that improve performance of all queries listed above. Please keep in mind, that an objective of this task is to get any possible improvement in performance with the smallest number of indexes used.

(3 marks)

(2) For each query briefly explain how the indexes will be used to process a query. Do not include the results from processing of `EXPLAIN PLAN` statement.

(3 marks)

Question 5 (6 marks)

Assume that you can invest 500 Mbytes of transient memory into implementation of `DEFAULT`, `KEEP` and `RECYCLE` data buffer caches.

Consider a sequence of `SELECT` statements given below.

```
SELECT * FROM CUSTOMER WHERE C_NATIONKEY = 'AU';
SELECT * FROM CUSTOMER WHERE C_NATIONKEY = 'CH';
SELECT * FROM PARTSUPP;
SELECT * FROM PART;
SELECT * FROM CUSTOMER WHERE C_NATIONKEY = 'PL';
SELECT * FROM CUSTOMER WHERE C_NATIONKEY = 'SG';
SELECT * FROM ORDERS;
SELECT * FROM LINEITEM;
SELECT * FROM PARTSUPP WHERE PS_AVAILQTY > 100;
SELECT * FROM CUSTOMER WHERE C_NATIONKEY = 'US';
SELECT * FROM CUSTOMER WHERE C_NATIONKEY = 'UK';
SELECT * FROM CUSTOMER WHERE C_NATIONKEY = 'AU';
SELECT * FROM ORDERS WHERE O_TOTALPRICE < 100;
```

(1) Decide what should be the size of `DEFAULT`, `KEEP`, and `RECYCLE` data buffer caches and assign the relational tables used in SQL statements given above to `DEFAULT`, `KEEP`, and `RECYCLE` data buffer caches. The objective of this task is to minimize the total number of physical read block operations needed to process `SELECT` statements listed above. To simplify the problem, assume that 1 data block has a size of 1 Mbyte. The sizes of the relevant relational tables are listed at the bottom of the page 2 of the final examination paper

You do not need to use SQL to implement this step. The simple declarations of the size of each data buffer cache and declaration which relational table is assigned to which data buffer cache is completely sufficient.

(3 marks)

(2) Consider the outcomes of step (1) and find the total number physical read block operations needed to process `SELECT` statements listed above. Assume that all data buffer caches are empty before processing of `SELECT` statements listed above. To simplify the calculations, assume that 1 data block has a size of 1 Mbyte. The sizes of relevant relational tables are listed at the bottom of the **Introduction** page of the final examination paper. Show your calculations.

(3 marks)

Question 6 (8 marks)

Consider a fragment of simple JDBC application listed below. It is a typical example of a pretty poor, from performance point of view, JDBC program.

(1) Rewrite JDBC code written below to improve the performance of the application it is included in. There is no need to write the entire JDBC application.

(4 marks)

(2) Explain all details why your version of JDBC code is more efficient than the original one.

(4 marks)

```
Statement stmt1 = conn.createStatement ();
       Statement stmt2 = conn.createStatement();
      ResultSet rset1 = stmt1.executeQuery( "SELECT * FROM PART " +
                                            "WHERE(P_PARTKEY >= 3000) " +
                                            "AND (P_PARTKEY <= 4000) " +
                                            "ORDER BY P_PARTKEY ASC" );
       long counter;
       long p_partkey = 0;
       while ( rset1.next() )
      { p_partkey = rset1.getInt(1);
        ResultSet rset2 = stmt2.executeQuery( "SELECT * FROM PARTSUPP "
                                              "WHERE (PS_AVAILQTY > 9500) " +
                                                "AND (PS_PARTKEY >= 3000) " +
                                                "AND (PS_PARTKEY <= 4000)" );
        long ps_partkey = 0;
         counter = 0;
         while ( rset2.next() )
        { ps_partkey = rset2.getInt(1);
          if (p_partkey == ps_partkey)
                  counter++;
         }
        if (counter >= 3)
          System.out.println( p_partkey);
       }
```

End of Examination