School of Computing and Information Technology

Student to complete:

Family name	
Other names	
Student number	
Table number	

CSCI317
Database Performance Tuning
Singapore Institute of Management

Final Examination Paper
Session 4 2019

Exam duration	3 hours
Weighting	60 % of the subject assessment
Marks available	60 marks
Items permitted by examiner	Non-programmable calculator
Directions to students	7 questions to be answered. Marks for each question are shown beside the question. All answers must be written in the answer booklet provided.

This exam paper must not be removed from the exam venue

Introduction
The questions 2, 4, 5, and 7 of the examination paper are related to the following simplified version of TPC-H benchmark database used in the laboratory classes.

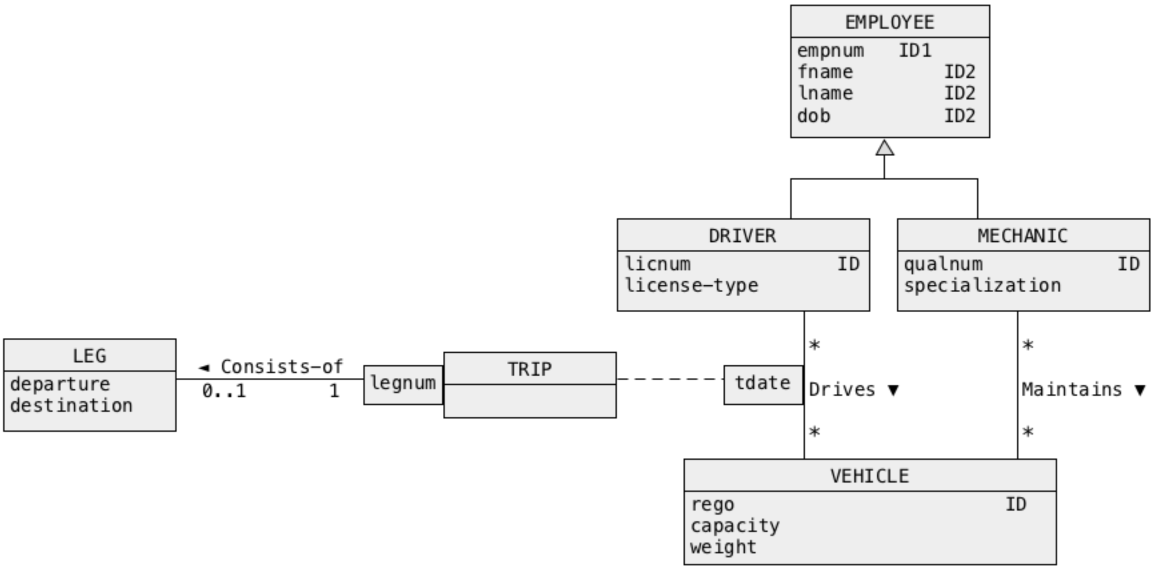
```
CUSTOMER(  C_CUSTKEY       NUMBER(12)      NOT NULL,
           C_NAME          VARCHAR(25)     NOT NULL,
           C_ADDRESS       VARCHAR(40)     NOT NULL,
           C_NATIONKEY     NUMBER(12)      NOT NULL,
     CONSTRAINT CUSTOMER_PKEY PRIMARY KEY(C_CUSTKEY) );

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,
     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 disk storage:

```
CUSTOMER        200 Mbytes
PART             50 Mbytes
PARTSUPP        500 Mbytes
ORDERS          300 Mbytes
LINEITEM        800 Mbytes
```

Question 1 **(10 marks)**

The following conceptual schema represents a database domain where drivers perform trips and mechanics maintain trucks. Trips consists of legs and each leg is described by a departure city and destination city. Both drivers and mechanics are the employees of a transportation company.



(1) Perform simplification of the conceptual schema above and re-draw the simplified conceptual schema. **(3 marks)**

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

Find the first and the last names of drivers (attributes `fname`, `lname` in a class `EMPLOYEE`) who travelled between two given cities (attributes `departure`, `destination` in a class `LEG`) and used a vehicle with a given weight (attribute `weight` in a class `VEHICLE`).

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

Find the first and the last names of drivers who travelled between `Sydney` and `Melbourne` and used a vehicle with a weight > `1000`.

Find the denormalizations of the simplified conceptual schema that improves the performance of the class of applications described above. When performing the denormalizations apply the following transformations of the simplified conceptual schema: migration of attributes, partitioning of classes of objects, and elimination of generalization. Re-draw the simplified conceptual schema after the denormalizations. **(7 marks)**

Question 2 **(8 marks)**

Consider the relational tables created by processing of `CREATE TABLE` statements listed on page 2 of the final examination paper.

Consider the following `SELECT` statement:

```
SELECT O_ORDERDATE, SUM(L_QUANTITY)
FROM LINEITEM JOIN ORDERS
              ON LINEITEM.L_ORDERKEY = ORDERS.O_ORDERKEY
GROUP BY O_ORDERDATE;
```

(1) Write SQL statement that denormalizes a relational table `LINEITEM`. The denormalization supposed to speedup the processing of a new `SELECT` statement that retrieves the same results as `SELECT` statement listed above and it uses only a denormalized table `LINEITEM`.

(2 marks)

(2) Write SQL statement that reloads data from the relational table `ORDERS` into the denormalized table `LINEITEM` created in a step (1).

(3 marks)

(3) Write a new `SELECT` statement that retrieves the same results as the original `SELECT` statement listed above and it uses only a denormalized table `LINEITEM`.

(1 mark)

(4) Explain what data redundancies are caused by the denormalization of a relational table `LINEITEM` and why a new `SELECT` statement created in a step (4) can be processed faster than the original `SELECT` statement.

(2 marks)

Question 3 **(10 marks)**

Assume that a relational table

`SHIPMENT(SUPPLIER#, PART#, QUANTITY, SDATE)`

contains 10^6 rows and that the attributes `(SUPPLIER#, PART#, SDATE)` form a composite primary key.

Assume that all shipments have been done by `20` suppliers, quantities of shipment vary from `1` to `100` with the same probability of each value of quantity, and average number of rows per disk block is equal to `10`.

Assume that all primary keys are automatically indexed by a database system and an index on primary key is implemented as non-clustered B*-Tree. The height of this index is equal to `3` and leaf level of an index occupies `50` data blocks.

A database administrator created a non-clustered B*-Tree index on attribute `QUANTITY` and found that height of this index is equal to `2` and leaf level of an index occupies `10` blocks.

Find the optimal query processing plans for each one of the queries listed above and estimate how many read block operations are needed to implement each one of the plans, i.e. estimate the total number read block operations needed to compute each one of the queries. Show your calculations. Express the query processing plans as the short stories about how the system plans to compute the queries given below.

There is no need to compute a value of `log` function. Each solution is worth 2 marks.

(1)
```
SELECT *
FROM SHIPMENT
WHERE SUPPLIER# = 123456;
```

(2)
```
SELECT SDATE
FROM SHIPMENT
WHERE SDATE > '1-JAN-2000';
```

(3)
```
SELECT QUANTITY
FROM SHIPMENT
WHERE SUPPLIER# = '123456' AND
      SDATE='1-JAN2000' AND
      PART#='777888';
```

(4)
```
SELECT COUNT(PART#)
FROM SHIPMENT;
```

(5)
```
SELECT QUANTITY, COUNT(*)
FROM SHIPMENT;
```

Question 4 **(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. Rewrite a code written below to improve the performance of the application it is included in. There is no need to write the entire JDBC application.

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

```
ResultSet rset1 = stmt1.executeQuery(
                  "SELECT P_PARTKEY FROM PART ORDER BY P_NAME" );
long p_partkey = 0;
while ( rset1.next() )
{
  p_partkey = rset1.getInt(1);
  ResultSet rset2 = stmt2.executeQuery(
                    "SELECT COUNT(*) FROM LINEITEM " +
                    "WHERE L_PARTKEY = " + p_partkey );
  long total;
  while ( rset2.next() )
  {
    total = rset2.getInt(1);
    if (total >= 30 )
       System.out.println( p_partkey + "   " + total);
  }
}
```

Question 5 **(8 marks)**

Consider the following `SELECT` statements and their query processing plans created by a query optimizer. The query processing plans are listed without the estimated costs Both `SELECT` statements given below retrieve exactly the same information from a database.

(1)
```
SELECT MAX(P_RETAILPRICE)
FROM PART;
```

```
------------------------------------ ...
| Id  | Operation          | Name |
------------------------------------ ...
|   0 | SELECT STATEMENT   |      |
|   1 |  SORT AGGREGATE    |      |
|   2 |   TABLE ACCESS FULL| PART |
------------------------------------ ...
```

(1)
```
SELECT DISTINCT P_RETAILPLRICE
FROM PART P1
WHERE NOT EXISTS (SELECT P_RETAILPRICE
                  FROM PART P2
                  WHERE P1.P_RETAILPRICE < P2.P_RETAILPRICE);
```

```
-------------------------------------- ...
| Id  | Operation            | Name |
-------------------------------------- ...
|   0 | SELECT STATEMENT     |      |
|   1 |  SORT UNIQUE NOSORT  |      |
|   2 |   MERGE JOIN ANTI    |      |
|   3 |    SORT JOIN         |      |
|   4 |     TABLE ACCESS FULL| PART |
|*  5 |    SORT UNIQUE       |      |
|*  6 |     TABLE ACCESS FULL| PART |
-------------------------------------- ...
```

(1) Draw the syntax trees of the query processing plans given above.

(2 marks)

(2) Decide which query processing plan is the most efficient one and provide a comprehensive justification of your decision.

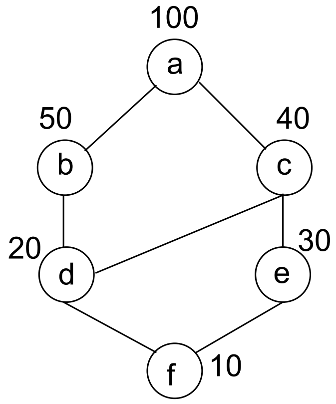
(4 marks)

(3) What hint would you provide to an unexperienced SQL programmer in relation to implementation of `SELECT` statements like the ones listed above ?

(2 marks)

Question 6 **(8 marks)**

Consider the following incomplete lattice of materialized views.



Assume that a view "a" has been already materialized and a cost of its materialization is 100. The costs of materialization of the other views "b", "c", "d", "e", and "f" are given in the diagram above.

Use an algorithm included in a presentation 19 Materialized views and practiced in Assignment 4, task 1 to find **no more than two other views that** can be materialized in order to reduce the costs of view processing in the best way.

Question 7 **(8 marks)**

Consider the following `SELECT` statements.

(1)
```
SELECT C_NAME, O_ORDERDATE
FROM CUSTOMER JOIN ORDERS
          ON C_CUSTKEY = O_CUSTKEY;
UNION
SELECT C_NAME, 0
FROM CUSTOMER
WHERE C_CUSTKEY NOT IN (SELECT C_CUSTKEY
                        FROM ORDERS);
```

(2)
```
SELECT SUM(L_QUANTITY)
FROM LINEITEM JOIN ORDERS
              ON L_ORDERKEY = O_ORDERKEY;
```

(3)
```
SELECT O_ORDERDATE, COUNT(*)
FROM ORDERS
GROUP BY O_ORDERDATE
HAVING COUNT(*) >= 1;
```

(4)
```
SELECT DISTINCT C_NAME
FROM CUSTOMER
WHERE (SELECT COUNT(*)
       FROM ORDERS
       WHERE O_CUSTKEY = CUSTOMER.C_CUSTKEY) > 3);
```

Your task is to find the more efficient implementations of each one of `SELECT` statements listed above. Remember that improved `SELECT` statements must retrieve from a database exactly the same information as the original ones.

Each one of the cases listed above is worth 2 marks.

End of Examination