计算机科学与工程系
Department of Computer Science and Engineering

CS307

Final Exam, Fall 2017
Duration 110 minutes

Stéphane Faroult

All printed and manuscript documents allowed, calculators (useless) allowed, other electronic devices forbidden.

There are several questions, of various difficulties, in this exam. They are organized in two completely independent parts. Questions are themselves independent.

Don't get stuck on a question – try to organize your time on the basis one point = one minute.

You must write your answers on the exam paper.

You are reminded of academic integrity requirements.
Papers strangely similar will get 0.

Points are indicative. They may be adjusted if one question is failed by far too many people.

Part 1: Quiz (25 points)
Part 2: Database Design (15 points)
Part 3: Indexing (20 points) (two independent questions)
Part 4: Sample from a database (20 points)
Part 5: Performance analysis (20 points) (two independent questions)

Part 1: Quiz questions (25 points)

To do

Part 2: Database Design (15 points)

I have found on the web a (MySQL) dump of a single table containing the text of all Shakespeare plays, with the following structure:

```
CREATE TABLE IF NOT EXISTS bill_play_text (
  line_id       bigint(20) NOT NULL AUTO_INCREMENT,
  play_name     varchar(100) NOT NULL,
  speech_number varchar(100) DEFAULT NULL,
  line_number   varchar(100) DEFAULT NULL,
  speaker       varchar(100) DEFAULT NULL,
  text_entry    varbinary(2000) NOT NULL,
  PRIMARY KEY (`line_id`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=111397 ;
```

This is an extract from the loading script (MySQL supports the insertion of multiple lines at once):

...

```
(34225, 'Hamlet', '17', '3.1.61', 'KING CLAUDIUS', 'Than is my deed to my most painted word:'),
(34226, 'Hamlet', '17', '3.1.62', 'KING CLAUDIUS', 'O heavy burthen!'),
(34227, 'Hamlet', '18', '3.1.63', 'LORD POLONIUS', 'I hear him coming: let''s withdraw, my lord.'),
(34228, 'Hamlet', '18', '', 'LORD POLONIUS', 'Exeunt KING CLAUDIUS and POLONIUS'),
(34229, 'Hamlet', '18', '', 'LORD POLONIUS', 'Enter HAMLET'),
(34230, 'Hamlet', '19', '3.1.64', 'HAMLET', 'To be, or not to be: that is the question:'),
(34231, 'Hamlet', '19', '3.1.65', 'HAMLET', 'Whether ''tis nobler in the mind to suffer'),
(34232, 'Hamlet', '19', '3.1.66', 'HAMLET', 'The slings and arrows of outrageous fortune,'),
(34233, 'Hamlet', '19', '3.1.67', 'HAMLET', 'Or to take arms against a sea of troubles,'),
...
```

What is called "line_number" (4th column) is composed of the Act number (main division of the play, usually between 1 and 5 - the curtain goes up and down between acts), the Scene number (usually there is a change of scene when another place is shown) and a verse number. "text_entry" is an indication about the movements of characters, as in rows 34228 and 34229 ("Exeunt" is Latin, it means (they) go out - "Exit" means (he) goes out).

This single-table structure is of course pretty bad. How would you reorganize data?

Part 3: Indexing (20 points)

Question 3.1 (10 points)

Consider the following two queries.

```
SELECT i.name, i.id, COUNT(c.id)
FROM cert_certificates c
JOIN cert_histories h ON h.cert_certificate_id = c.id
LEFT OUTER JOIN inspectors i ON h.inspector_id = i.id
LEFT OUTER JOIN cert_histories h2
  ON (h2.cert_certificate_id = c.id AND h.date_changed < h2.date_changed)
WHERE (h.cert_status_ref_id = ? OR h.cert_status_ref_id = ?)
  AND h2.id IS NULL
GROUP BY i.id, i.name
ORDER BY i.name

SELECT l.letter, c.number
FROM cert_certificates c
JOIN cert_type_letter_refs l ON c.cert_type_letter_ref_id = l.id
JOIN cert_histories h ON h.cert_certificate_id = c.id
LEFT OUTER JOIN cert_histories h2
  ON (h2.cert_certificate_id = c.id AND h.date_changed < h2.date_changed)
WHERE h.cert_status_ref_id = ?
  AND h2.id IS NULL
  AND h.inspector_id = ?
     ORDER BY l.letter, c.number
```

Assuming that:

- The `cert_certificates` table contains nearly 20000 records, the `cert_histories` table contains nearly 60000 records, and the other tables contain less than 10 records each;
- There are indexes on `id` for each table and on `cert_certificates.number`.

Specify what other index(es) can be created to improve the performance of the queries above.

Question 3.2 (10 points)

From a question on a forum:
Consider the following table

```
CREATE Table T1(
        ID  int primary key,
        Name VARCHAR(50) not null,
        Status CHAR(1) not null,
        GroupI INT not null,
        GroupII INT,
   constraint t1_U unique(GroupI, Name)
}
```

When we look at the indexes on the table we find this (each row is an index, what is listed is the names of the columns in the index, in indexing order)

```
     ID (Unique)
     GroupI, Name (Unique)
```

```
GroupI
GroupII, Name
GroupI, GroupII, Status, Name
GroupII, Name, GroupI
```

Which index(es) is (or are) probably useless?

Part 4: Sample from a database (20 points)

We want to extract a consistent subset of the film database, by picking films at random.
To do so, we create a work table `extracted_films` that only contains `movieid` values and which is populated as follows (we assume that we want a subset of 500 films):

```
insert into extracted_films(movieid)
select movieid
from movies
order by random()
limit 500;
```

You are reminded that the database contains tables `movies` (foreign key to `countries`), `countries`, `people` and `credits` (foreign key to `movies` and foreign key to `people`).

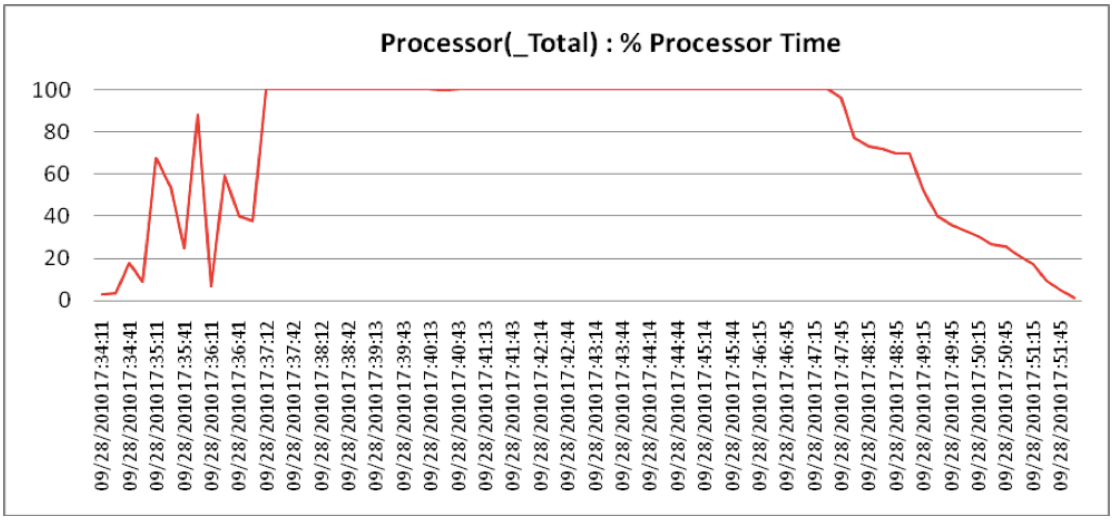
Write the select statements (you can use select *, in real life it would be turned into an insert statement into the correct table) that will extract a consistent subset of the database.
You will be careful to do it in a way that will not cause any trouble with foreign keys when reloading the data into already created empty tables.

Part 5: Performance analysis (20 points)

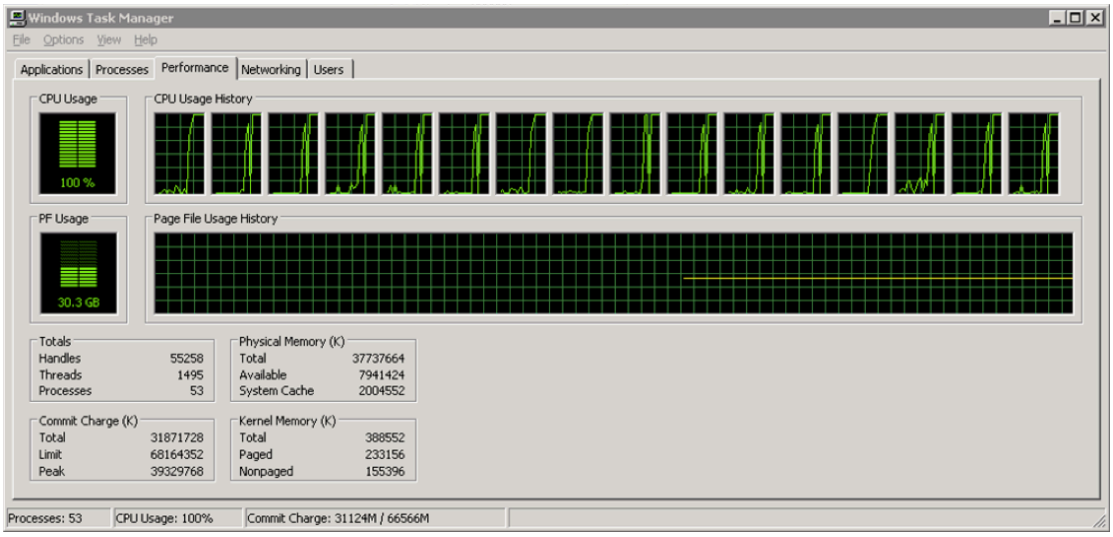
Question 5.1 (10 points)

A friend of mine was called a few years ago for auditing an application for which web access was randomly timing out. What probably was the reason for time-outs was that 100% usage of CPU on a powerful machine. Here is the magnified end of afternoon CPU peak.

CPU Peak



It was traced down to a stored procedure named `sp_VoucherUpdate` that, when it was manually started, exhibited the same behavior of using all of the CPU :



What follows is just the beginning of the procedure (between 25 and 30% of the full procedure). The audit report merely states that this procedure should be rewritten.
A few useful indications to help you understand the code better :

- `#name` indicates in SQL Server a temporary table (on which logging mechanisms that are applied are lighter)
- Function `left(col, n)` returns the n-most left characters of a text column.

I have also added comments about SQL Server peculiarities.
Could you suggest at least two specific modifications that are likely to improve performance ?

```
UPDATE  VoucherUpdate
SET WarningLevelId = 0,
```

```
    WarningMessage = NULL;
INSERT INTO #AllVoucher
SELECT  NULL AS ArchiveCode, T2.VoucherNo
FROM    VoucherUpdate T1
INNER   JOIN VoucherHeader T2
   ON LEFT(T1.VoucherNo,9) = LEFT(T2.VoucherNo,9)
UNION
SELECT  T2.ArchiveCode, T2.VoucherNo
FROM    VoucherUpdate
INNER   JOIN VoucherHeaderArchive T2
   ON LEFT(T1.VoucherNo,9) = LEFT(T2.VoucherNo,9);

WHILE (SELECT   COUNT(Sid)
       FROM    VoucherUpdate
       WHERE   IsDone  = 0) > 0
BEGIN
  INSERT INTO #TempVoucherUpdate
  -- SELECT TOP 2000 ... with SQL Server
  -- is the same as SELECT ... LIMIT 2000 with other
  -- database management systems
  SELECT  TOP 2000 *
  FROM    VoucherUpdate
  WHERE   IsDone  = 0;

  UPDATE  #TempVoucherUpdate
  SET WarningLevelId = 2,
      -- SQL Server uses + instead of || for concatenating strings
      WarningMessage = ISNULL(WarningMessage, '')
                     + 'Duplicate Voucher# in input file,'
  WHERE   LEFT(VoucherNo,9) IN (SELECT  LEFT(VoucherNo,9)
                                FROM    VoucherUpdate
                                GROUP BY VoucherNo
                                HAVING COUNT(*) > 1);
  UPDATE  #TempVoucherUpdate
  SET WarningLevelId = 2,
      WarningMessage = ISNULL(WarningMessage, '')
        + 'Duplicate Voucher# in Navision to process manually,'
    WHERE   VoucherNo IN (SELECT  LEFT(T1.VoucherNo,9)
                          FROM    VoucherUpdate T1
                          INNER   JOIN VoucherHeader T2
                             ON LEFT(T1.VoucherNo,9) = LEFT(T2.VoucherNo,9)
                          GROUP BY LEFT(T1.VoucherNo,9)
                          HAVING COUNT(T2.VoucherNo) > 1);

    UPDATE  T1
    SET VoucherNo = T2.VoucherNo
    FROM    #TempVoucherUpdate T1
            INNER   JOINVoucherHeader T2
               ON LEFT(T1.VoucherNo,9) = LEFT(T2.VoucherNo,9)
    WHERE   LEN(T1.VoucherNo) = 9
      AND     LEFT(T1.VoucherNo,9) NOT IN
                 (SELECT  LEFT(T1.VoucherNo,9)
                  FROM    VoucherUpdate T1
                          INNER   JOIN VoucherHeader T2
                             ON LEFT(T1.VoucherNo,9) = LEFT(T2.VoucherNo,9)
                  GROUP BY LEFT(T1.VoucherNo,9)
```

```
                HAVING COUNT(T2.VoucherNo) > 1)
 ...
END ;
```

Question 5.2 (10 points)

The following query comes straight from an audit report, in which it was mentioned as "costly" (I have slightly modified table and column names), without any suggestion about how to make things better. Table ORDPAC contains around 140000 rows, has a primary key (that doesn't appear in the query) *ordpac_id* and is indexed on (*dattr*, *etatr*). This query has been identified as unusually long . In this query, *getdate()* is a function that returns the current date and time (down to the second), *dateadd (datepart , number , date)* returns a date obtained by adding to *date* a *number* (signed integer) of ime units specified by *datepart*; in the query "ss" means "seconds" et *date(ss, 5, DATTR)* returns DATTR plus 5 seconds.
Here is the query:

```
select …..
       CODCOTSWP,
       DTCOTSWP,
       CODCOTTRE,
       DTCOTTRE
from
       ORDPAC
where
       ((STATE=0 and STATER=NULL) or
       (STATE=0 and STATER=1 and dateadd(ss,5,DATTR)<getdate()) or
       (STATE=0 and STATER=2 and dateadd(ss,5,DATTR)<getdate()))
```

a. Where is the problem? How to fix it??

b. The author of the report made no remark about the first line in the `where` clause. What do you think about it?