南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

2017-2018 Academic Year
Fall Semester
Answer Key

Course Name: Database Principles Course Code: CS307

Department: Computer Science Exam Paper Setter: Stéphane Faroult

General guidelines: only give 0 if nothing was written. Give at least 0.5 for effort, however wrong it is. You can give a little more if it makes some sense.

Part 1 (Quiz)

```
q1-q5    f,b,b,a,a   q1: 2 points
q6-q10   b,a,a,b,a
q11-q15  a,b,a,c,d   q11: 2 points
q16-q20  c,b,b,a,c
q21-q23  b,b,c
```

Part 2 (Database Design) 15 points

Table Plays

2 points for table + attributes, 3.5 points total with constraints

```
  playid + 0.5 for saying that it's the primary key
  title  + 0.5 for saying it's unique
  -- Accepted: number of acts
+ 0.5 for saying that all attributes are mandatory
```

Table Characters

2 points for table + attributes, 3.5 points total with constraints

```
  caractid + 0.5 for saying that it's the primary key
  name     + 0.5 for saying it's unique
  -- Accepted: playid - but -0.5 if not indicated as
  --                              a foreign key
  -- Having the play as attribute of the character
```

```
  -- is wrong because some characters (Falstaff
  -- for instance) appear in several plays
  -- but students aren't expected to be
  -- familiar with Shakespeare's plays
+ 0.5 for saying that all attributes are mandatory


-- Optional Table
--  Can be derived from lines but might
--  allow more referential integrity
-- table Character_play
--    caractid
--    playid
-- +1 (bonus) if added with the specification
--    primary key = (caractid, playid)
--    caractid foreign key to Characters
--    playid foreign key to Plays
--  No bonus otherwise
```

Table Speeches

2 points for table + attributes, 3.5 points with constraints

```
speech_id  + 0.5 for saying it's the primary key
caracterid + 0.5 for saying its a foreing key to characters
+ 0.5 for saying that all attributes are mandatory
```

Table Lines

No points for the table as it's based on the original one. Assign points as specified hereafter.

4.5 points with constraints

```
lineid  0.5 for saying that it's the primary key
playid  0.5 for saying that it's a foreign key to plays
act_num
scene_num
verse_num
speech_id  0.5 for saying it's a foreign key to
speeches
text_entry

2 points for storing separately act_num, scene_num and
verse_num
```

1 points for saying that playid + the combination of act_num, scene_num and verse_num is unique (grant the points even if act_num, scene_num and vers_num have not been separated)
+ 1 bonus points for saying that verse_num and speech_id can be null and other columns are mandatory

Part 3 (Indexing)

第一条 Question 3.1 (10 points)

Indexing should be driven by conditions:

- Correct answer - 5 points for
  `cert_histories(cert_certificate_id)`

  or better answer (7 points)
  `cert_histories(cert_certificate_id, date_changed)`
  Only 4 points if order or columns reversed

- Strong candidate for indexing (+ 2 points)
  `cert_histories(cert_status_ref_id)`
  + 1 point for reservations about selectivity

- Other possibilities (+ 1point)
  `cert_histories(inspector_id)`

第二条 Question 3.2 (10 points)

5 points for each
Index on GroupI useless
Index on GroupII, Name useless

Part 4 (Sample from a database) 20 points

Two important things:
1) Not forgetting a table
2) Extracting in the right order
   a. countries must be extracted before movies (reloaded first)

b. both people and movies must be extracted before credits

Valid orders:
countries,movies,people,credits
countries,people,movies,credits
people,countries,movies,credits

Grading: 4 points for correct order
Points for each extraction as indicated below.

Extraction of countries 4 points

```
select * from countries
where country_code in
         (select m.country_code
          from movies m
               join extracted_films e
                 on e.movieid=m.movieid)
Also OK: Join in the subquery replaced by
       select country_code from movies
       where movieid in
                (select movieid from extracted_films)
Also OK: Join on subquery, or use of WITH
```

Extraction of movies (join possible) - 3 points

```
select *
from movies
where where movieid in
         (select movieid from extracted_films)
```

Extraction of people - 5 points, only 4 if movies appears in the query (not needed)

```
select *
from people
where peopleid in
         (select c.peopleid
          from credits c
              join extracted_films e
              on e.movieid = c.movieid)
```

Extraction of credits - 4 points, -1 if movies appears in the query (not needed), -1 if people appears in the query (not needed). Join possible

```
select *
from credits
where movieid in
         (select movieid from extracted_films)
```

Part 5 (Performance analysis)

第三条 Question 5.1

Two suggestions needed, 5 points for each.

Can be any of:

1. replace `LEFT(T1.VoucherNo,9) = LEFT(T2.VoucherNo,9)`
   by `T1.VoucherNo like substr(T2.VoucherNo,1, 9) + '%'`
   (or anything vaguely similar - syntax not important, what is important is their saying that the function might kill the index, even if in this case it my not quite be true because the SQL Server optimizer normally know how to manage left())
2. Replace UNION with UNION ALL to avoid a sort
3. Don't run a COUNT in the while loop, count once, then maintain a count of how many rows were processed
4. replace

```
WHERE   LEFT(VoucherNo,9) IN (SELECT  LEFT(VoucherNo,9)
                              FROM    VoucherUpdate
                              GROUP BY VoucherNo
                              HAVING COUNT(*) > 1);
```

by a join with a like (give 4 points if problem seen here but join solution not suggested)

5. Combine the two updates of #TempVoucherUpdate into one (+1 bonus if suggesting a way to do it)

第四条 Question 5.2 (10 points)

a. Function date kills the index (3 points)
   Fix: DATTR < dateass(ss, -5,getdate()) at both places (2 points)
   +1 bonus if suggesting STATER in (1,2) and one condition on date

b. STATER = NULL is always wrong (2 points)
   First condition useless (1 point)
   Should be (STATE = 0 and STATER is NULL) (2 points)