UOW AUSTRALIA

School of Computing and Information Technology

Student to complete:

Family name	
Other names	
Student number	
Table number	

CSCI235
Database Systems
Wollongong Campus

Examination Paper
Autumn Session 2019

Exam duration	3 hours
Weighting	60 %
Items permitted by examiner	*None*
Aids supplied	None
Directions to students	6 questions to be answered. Answer each question in a separate booklet.

This exam paper must not be removed from the exam venue

Question 1 **(10 marks)**

Consider the relational schemas given below and the respective sets of functional dependencies valid in the schemas.

For each one of the relational schemas, determine the highest normal form, which is valid for a schema. **Justify your answer.** Justification must include the derivations of minimal keys from the functional dependencies and testing the validity of all normal forms (2NF, 3NF, BCNF) against the relational schemas, minimal keys, and functional dependencies.

If a schema is not in BCNF, then decompose it into a *minimum number of schemas* so that each one of them is in BCNF.

A correct guess without the comprehensive justifications scores no marks!

Do not change the attributes of relational schemas in Question 1.

(1) R = (A, B, C, D)
$A \rightarrow B$
$B \rightarrow A$
$B \rightarrow D$

(2.5 marks)

(2) R = (A, B, C, D)
$A, B \rightarrow C$
$A, B \rightarrow D$
$C \rightarrow D$

(2.5 marks)

(3) R = (A, B, C, D)
$B \rightarrow C, D$

(2.5 marks)

(4) R = (s#, c#, year, lecturer, dept)
s#, c#, year $\rightarrow$ lecturer
lecturer $\rightarrow$ dept

(2.5 marks)

THE QUESTIONS 2, 3 and 4 REFER TO THE RELATIONAL TABLES LISTED BELOW

The schemas of relational tables, specifications of primary, candidate, foreign keys and check constraints are given below.

```
CREATE TABLE SKILL(
    sname          VARCHAR(30)     NOT NULL, /* Skill name  */
    CONSTRAINT SKILL_pkey PRIMARY KEY (sname) );
CREATE TABLE APPLICANT(                    /* Applicants           */
    anumber        NUMBER(6)   NOT NULL,   /* Applicant number     */
    fname          VARCHAR(20) NOT NULL,   /* First name           */
    lname          VARCHAR(30) NOT NULL,   /* Last name            */
    dob            DATE        NOT NULL,   /* Date of birth        */
    city           VARCHAR(30) NOT NULL,   /* City                 */
    state          VARCHAR(20) NOT NULL,   /* State                */
    phone          NUMBER(10)  NOT NULL,   /* Phone number         */
    email          VARCHAR(50),            /* E-mail address       */
    CONSTRAINT APPLICANT_pkey PRIMARY KEY (anumber) );
CREATE TABLE EMPLOYER(                     /* Employers            */
    ename          VARCHAR(100)NOT NULL,   /* Employer name        */
    city           VARCHAR(30) NOT NULL,   /* City                 */
    state          VARCHAR(20) NOT NULL,   /* State                */
    phone          NUMBER(10)  NOT NULL,   /* Phone number         */
    fax            NUMBER(10),             /* Fax number           */
    email          VARCHAR(50)             , /* E-mail address     */
   CONSTRAINT EMPLOYER_pkey PRIMARY KEY (ename) );
CREATE TABLE POSITION(                     /* Advertised positions */
    pnumber        NUMBER(8)   NOT NULL,   /* Position number      */
    title          VARCHAR(30) NOT NULL,   /* Position title       */
    salary         NUMBER(9,2) NOT NULL,   /* Salary               */
    bonus          NUMBER(9,2),            /* End of year bonus    */
    ename          VARCHAR(100)NOT NULL,   /* Employer name        */
    CONSTRAINT POSITION_pkey PRIMARY KEY (pnumber),
    CONSTRAINT POSITION_fkey FOREIGN KEY (ename) REFERENCES EMPLOYER(ename) );
CREATE TABLE SPOSSESSED(
    anumber        NUMBER(6)   NOT NULL,   /* Applicant number      */
    sname          VARCHAR(30) NOT NULL,   /* Skill name            */
    slevel         NUMBER(2)   NOT NULL,   /* Skill level           */
    CONSTRAINT SPOSSESSED_pkey PRIMARY KEY (anumber, sname),
    CONSTRAINT SPOSSESSED_fkey1 FOREIGN KEY (anumber) REFERENCES APPLICANT
(anumber),
    CONSTRAINT SPOSSESSED_fkey2 FOREIGN KEY (sname) REFERENCES SKILL (sname),
    CONSTRAINT SPOSSESSED_check1 CHECK ( slevel between 1 and 10) );
CREATE TABLE SNEEDED(
    pnumber        NUMBER(8)   NOT NULL,   /* Position number        */
    sname          VARCHAR(30) NOT NULL,   /* Skill name             */
    slevel         NUMBER(2)   NOT NULL,   /* Skill level            */
    CONSTRAINT SNEEDED_pkey PRIMARY KEY (pnumber, sname),
    CONSTRAINT SNEEDED_fkey1 FOREIGN KEY (pnumber)REFERENCES POSITION (pnumber),
    CONSTRAINT SNEEDED_fkey2 FOREIGN KEY (sname) REFERENCES SKILL (sname),
    CONSTRAINT SNEEDED_check1 CHECK ( slevel between 1 and 10) );
CREATE TABLE APPLIES(
    anumber        NUMBER(6)   NOT NULL,   /* Applicant number     */
    pnumber        NUMBER(8)   NOT NULL,   /* Position number      */
    appdate        DATE        NOT NULL,   /* Application date     */
    CONSTRAINT APPLIES_pkey PRIMARY KEY (anumber, pnumber, appdate),
    CONSTRAINT APPLIES_fkey1 FOREIGN KEY (anumber) REFERENCES APPLICANT
(anumber),
    CONSTRAINT APPLIES_fkey2 FOREIGN KEY (pnumber) REFERENCES POSITION (pnumber)
);
```

Question 2 **(10 marks)**

This question is related to a sample database created through processing of CREATE TABLE statements listed on a page 3 of the examination paper.

Implement a stored PL/SQL procedure `APPLICATIONS` to list the applicants and their applications.

The names of applicants must be listed in the ascending order of last names of applicants. The position number and title of a position applied by an applicant must be listed in the ascending order of position number.

Execute the stored PL/SQL procedure `APPLICATIONS`. A fragment of expected sample printout is given below.

```
 7 James Bond:
   3 senior lecturer
   4 associate professor
   7 professor
 4 Michael Collins:
   2 lecturer
   7 professor
 5 Margaret Finch:
   2 lecturer
   3 senior lecturer
   7 professor
…
```

Question 3 **(10 marks)**

This question is related to a sample database created through processing of CREATE TABLE statements listed on a page 3 of the examination paper.

Implement a `statement trigger` that enforces the following consistency constraint.

```
A position cannot need more than 4 skills.
```

Write SQL statements that comprehensively test the trigger. A comprehensive test must consist of at least two cases for each operation: one accepted and one rejected. Assume that position 00000001 has 4 skills needed, and position 00000005 has 2 skills needed. The other values are up to you.

Question 4 **(10 marks)**

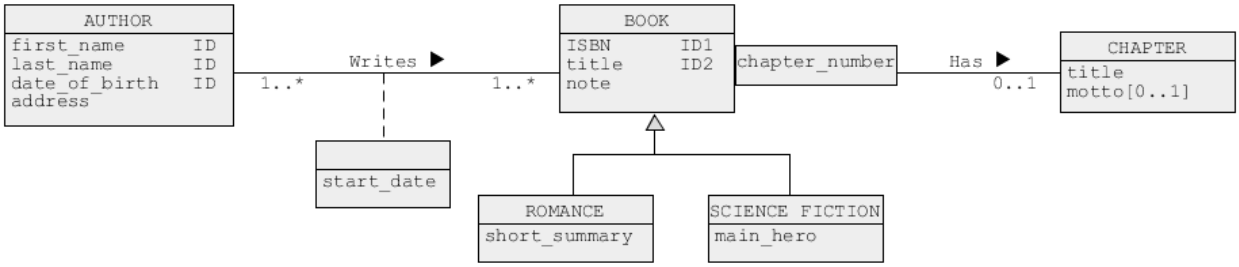
This question is related to a sample database created through processing of CREATE TABLE statements listed on a page 3 of the examination paper.

Show a sample concurrent execution of two database transactions such that the execution reveals `deadlocks`. Thoroughly explain why the execution reveals a deadlock. Remember to set an appropriate isolation level for the transactions. **Note, that a sample execution without the detailed explanations scores no marks**.

Use a technique of presentation of concurrent execution of two database transactions that explained to you during the lecture classes and such that the statements of the first transaction are listed on the left-hand side of a page and the statements of the second transaction are listed on the right-hand side of a page. Make sure that each statement starts in a different line to represent a different moment in time when its execution starts.

Question 5 **(10 marks)**

Consider the following conceptual schemas representing the sample database domains.



For the conceptual schema given above creates a sample BSON document whose contents are consistent the respective conceptual schema. Your documents must contain information about at least two instances for each author, book, chapter of the classes included in the schemas.

Question 6 **(10 marks)**

Consider a sample BSON document given below. Assume that all documents in a collection driver have the same structure as the document listed below.

```
db.driver.insert(
  { "first_name":"James",
    "last_name":"Bond",
    "licence":007,
    "address":{"street":"Northfields Ave",
               "bldg":3,
               "city":"Wollongong",
               "country":"Australia"},
   "trips":[ {"number":5,
              "truck_rego":"PKR856",
              "date":"12-DEC-2017",
              "legs": [ {"number":1,
                         "departure":"Sydney",
                         "destination":"Melbourne" },
                        {"number":2,
                         "departure":"Melbourne",
                         "destination":"Sydney" } ] },
             {"number":25,
              "truck_rego":"AL08UK",
              "date":"03-JUN-2018",
              "legs": [ {"number":1,
                         "departure":"Sydney",
                         "destination":"Melbourne" } ] }
           ]
    }
);
```

Use either a method find() or a method aggregate() available in MongoDB to write the implementations of the following queries. Implementation of each query is worth 2 marks.

(1) Find the `first name, last name` and `licence` number of all drivers who are located in `Wollongong, Australia`.

(2) Find the `first name, last name` and `licence` number of all drivers who haven't taken any trips at all.

(3) Find the `truck rego number`, `date` and `total length` (size of legs) of each trip performed by the truck.

Use either a method remove() or a method update() to write the implementations of the following data manipulation operations. Implementation of each data manipulation operation is worth 2 marks.

(4) Delete from a collection driver the documents that contain information about the drivers whose last name are `Thomas` and live in `Darwin, Australia`.

(5) Add a new leg in a trip number `25` from `Melbourne` to `Adelaide`.

End of examination paper