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
## Spring Session 2019

| | |
|---|---|
| Exam duration | 3 hours |
| Weighting | 60 % |
| Items permitted by examiner | *None* |
| Aids supplied | None |
| Directions to students | 6 questions to be answered.<br>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)
$A \rightarrow B$
$C \rightarrow D$

**(2.5 marks)**

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

**(2.5 marks)**

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

**(2.5 marks)**

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

**(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 Department (
    D#              NUMBER(5)          NOT NULL,   /* Department number          */
    DName           VARCHAR2(30)       NOT NULL,   /* Department name            */
    Manager#        CHAR(5)            NOT NULL,   /* Department manager number  */
    MSDate          DATE,                          /* Manager start date         */
    CONSTRAINT Department_PK PRIMARY KEY(D#),
    CONSTRAINT Department_CK UNIQUE(DName)
);

CREATE TABLE DeptLocation (
    D#              NUMBER(5)          NOT NULL,   /* Department number          */
    Address         VARCHAR2(50)       NOT NULL,   /* Department location        */
    CONSTRAINT DeptLocation_PK PRIMARY KEY(D#, Address),
    CONSTRAINT DeptLocation_FK FOREIGN KEY(D#) REFERENCES Department(D#)
);

CREATE TABLE Employee (
    E#              CHAR(5)            NOT NULL,   /* Employee number            */
    Name            VARCHAR2(30)       NOT NULL,   /* Employee name              */
    DOB             Date,                          /* Date of birth              */
    Address         VARCHAR2(50),                  /* Home address               */
    Sex             CHAR,                          /* M-Male, F-Female           */
    Salary          NUMBER(7,2),                   /* Salary                     */
    Super#          CHAR(5),                       /* Supervisor number          */
    D#              NUMBER(5),                     /* Department number          */
    CONSTRAINT Employee_PK PRIMARY KEY(E#),
    CONSTRAINT Employee_FK1 FOREIGN KEY (Super#) REFERENCES Employee(E#),
    CONSTRAINT Employee_FK2 FOREIGN KEY (D#) REFERENCES Department (D#)
);

CREATE TABLE Project (
    P#              NUMBER(10)         NOT NULL,   /* Project number             */
    PTitle          VARCHAR2(30)       NOT NULL,   /* Project title              */
    Sponsor         VARCHAR2(30),                  /* Project sponsor name       */
    D#              NUMBER(5)          NOT NULL,   /* Department number          */
    Budget          NUMBER(10,2)       NOT NULL,   /* Project budget             */
    CONSTRAINT Project_PK PRIMARY KEY(P#),
    CONSTRAINT Project_FK FOREIGN KEY (D#) REFERENCES Department(D#),
    CONSTRAINT Project_CK UNIQUE (PTitle)
);

CREATE TABLE WorksOn (
    E#              CHAR(5)            NOT NULL,   /* Employee number            */
    P#              NUMBER(10)         NOT NULL,   /* Project number             */
    Hours           NUMBER(3,1)        NOT NULL,   /* Working hours per week     */
    CONSTRAINT WorksOn_PK PRIMARY KEY(E#, P#),
    CONSTRAINT WorksOn_FK1 FOREIGN KEY(E#) REFERENCES Employee(E#),
    CONSTRAINT WorksOn_FK2 FOREIGN KEY(P#) REFERENCES Project(P#)
);
```

**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 function `DEPTPROJECT` that takes a `department number` as a parameter, and finds all numbers, titles and budgets of projects held by the department, and the numbers and names of employees who work on the project in that department.

The function must return a string of characters that contains the department number, name, project numbers, titles and budgets for the department, and employee numbers and name that work on each project in the department. If a department has more than 1 project, the projects must be listed in the descending order of the budgets. The employees that work on the project must be listed in the ascending order of their names.

Execute the stored PL/SQL function `DEPTPROJECT` for all departments. A fragment of sample printout is given below:

```
Department: 1 SALES
  Project: 1001 Computation 25000
       00110 Alvin
       00101 Peter

Department: 2 ACCOUNTING
Department: 3 GAMES
  Project: 1003 Racing car 225000
       00105 Robert
  Project: 1002 Study methods 15000
       00150 Bob
       00105 Robert
…
```

**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 and comprehensively test a **row trigger** that verifies the following consistency constraint.

**"An employee cannot work on more than 3 projects".**

You must consider both INSERT and UPDATE events.
You can add

```
PRAGMA AUTONOMOUS_TRANSACTION
```

to avoid mutating errors.

Comprehensive testing means that the trigger must reject SQL statements that violate the consistency constraint and accept SQL statements that do not violate the consistency constraint. It is a part of your task to find what SQL statements should be tested. Whenever SQL statement violates the consistency constraint a trigger must return `ORA-...` error message. If SQL statement does not violate the consistency constraint then a trigger must return no messages.

Assume that employee 00101 has 3 projects. There numbers are 1002, 1003, and 1005. Employee 00107 has 2 projects. There numbers are 1001 and 1003. The other values are up to you.

**Question 4** **(10 marks)**

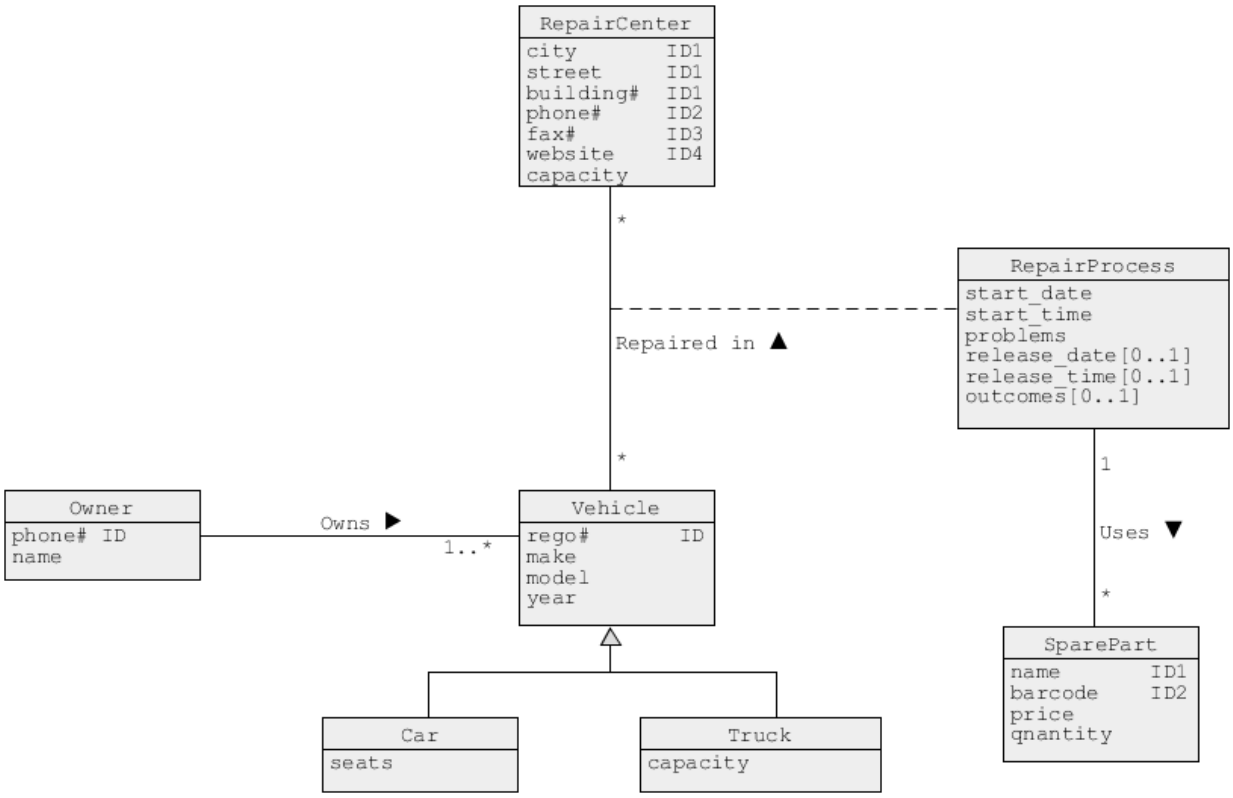
This question is related to a sample database created through processing of CREATE TABLE statements listed on a page 3 of the examination paper.

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 sample BSON documents whose contents are consistent the respective conceptual schema. Your documents must contain information about at least two instances for each repair center, vehicle (car and truck), owner, and Spare part 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 bookshop have the same structure as the document listed below.

```
db.bookshop.insert( {
    "_id":"185.3.16",
    "book": {
     "callnum":"185.3.16",
     "isbn":"1-292-06118-9",
     "title":"Database Systems",
     "authors":[
          {
               "fname":"Thomas",
               "lname":"Connolly"},
          {
               "fname":"Carolyn",
               "lname":"Begg"}
     ],
     "publisher":"Pearson Pty Ltd",
     "year":2015,
     "price":136.99,
     "topic":"Computer Science",
     "description":"This is the 6th edition. You can register
online to access the examples",
     "keywords":["Database", "XML", "Distributed"]
     }
});
```

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 `title, authors, publisher, year` and `price` of all books which are published `before 2015`.

(2) Find the `title, publisher, year` and `price` of all books which contain a word `C++` in the description.

(3) Find the `title, publisher, year` and `price` of all books which have `at least 2` keywords.

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 bookshop the documents that contain information about the author of books whose last name is `Bond` and the first name is `James`.

(5) Add keywords `Sorting` and `Searching` in the document which call number is `123.45.67`.

*End of examination paper*