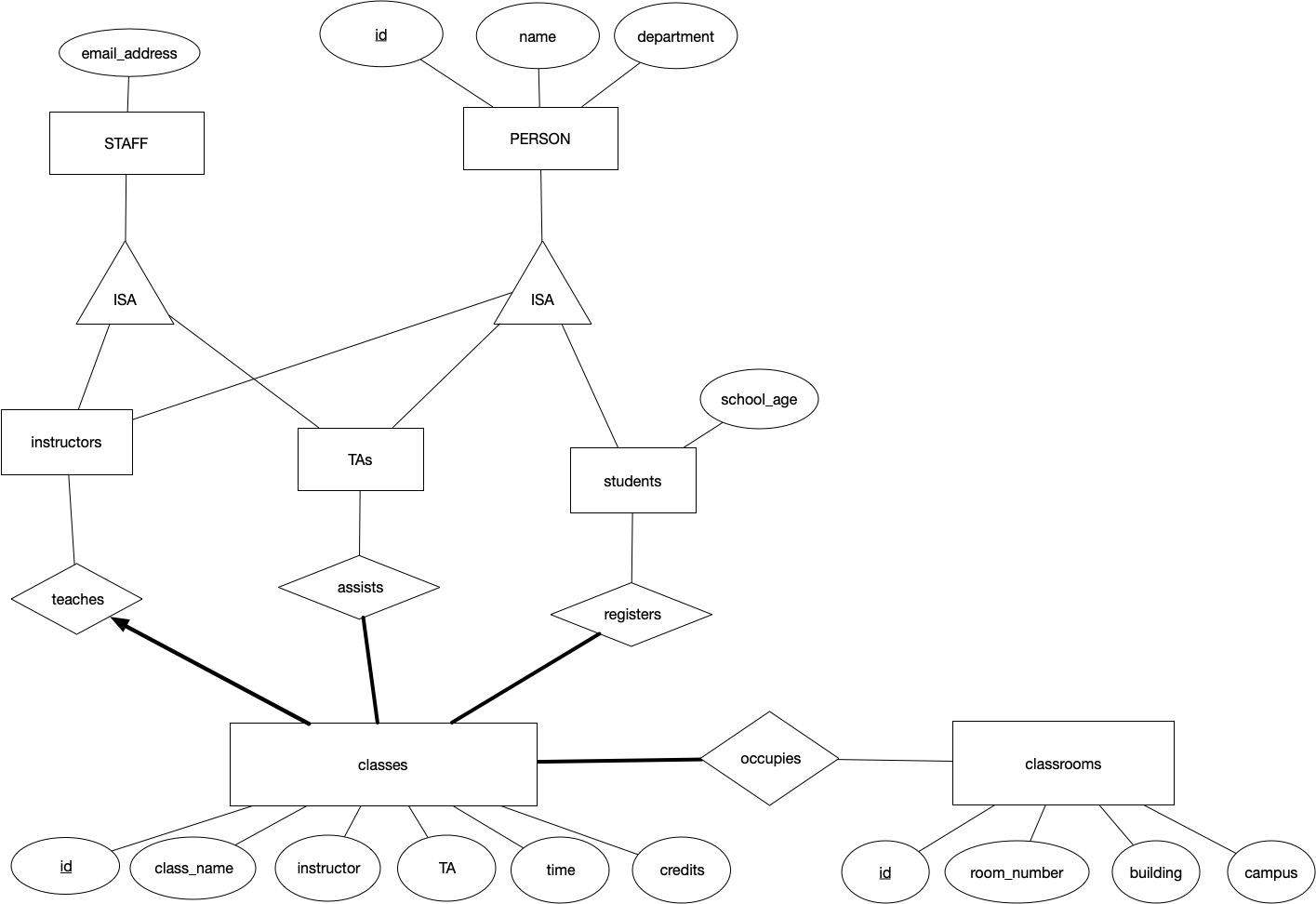
**INFSCI 2710 Database Management, Fall 2017**

**Homework 3: ER, Schema Refinement, Storage and Indexing (100 pts)**

**Due Date: Tue 11/27, hard copy at the beginning of the class.**

**Q1 [15 pts]** Draw an ER diagram for an online class registration system on the following description: The database must store information about instructors who teach the class, TAs who assist the class, students who register the class, classes that the system has, classrooms where the classes will be given. Instructors, TAs and students share some common attributes such as id, name and department. Instructors and TAs have a specific attribute, email address, while students have a specific attribute school age. Each class is described by id, class name, instructor, TA, time, credits. Each classroom is described by id, campus, building and room number. One instructor may give more than one classes, and so does TA. One class only has one instructor, but may have several TAs. You can draw ER diagram by hand, but make sure it is readable.

Answer:



**Q2 [15 pts]** Translate the ER diagram from Q1 into SQL DDL statements.

CREATE TABLE classes (

id INTEGER,

class\_name CHAR(20),

instructor INTEGER,

TA INTEGER,

time DATE,

credits INTEGER

);

CREATE TABLE instructors (

id INTEGER,

name CHAR(20),

department CHAR(20),

email\_address CHAR(20),

PRIMARY KEY(id)

);

CREATE TABLE teaches (

id INTEGER,

instructor INTEGER,

PRIMARY KEY(id),

FOREIGN KEY(instructor) REFERENCES classes

);

CREATE TABLE TAs (

id INTEGER,

name CHAR(20),

department CHAR(20),

email\_address CHAR(20),

PRIMARY KEY(id)

);

CREATE TABLE assists (

id INTEGER,

TA INTEGER,

PRIMARY KEY(id, TA),

FOREIGN KEY(id) REFERENCES TAs,

FOREIGN KEY(TA) REFERENCES classes

);

CREATE TABLE students (

id INTEGER,

name CHAR(20),

department CHAR(20),

school\_age INTEGER,

PRIMARY KEY(id)

);

CREATE TABLE registers (

id INTEGER,

Class\_id INTEGER,

PRIMARY KEY(id, Class\_id),

FOREIGN KEY(id) REFERENCES students,

FOREIGN KEY(Class\_id) REFERENCES classes(id)

);

CREATE TABLE classrooms (

id INTEGER,

campus CHAR(20),

building CHAR(20),

room\_number INTEGER,

PRIMARY KEY(id)

);

CREATE TABLE occupies (

id INTEGER,

Class\_id INTEGER,

PRIMARY KEY(id, Class\_id),

FOREIGN KEY(id) REFERENCES classrooms,

FOREIGN KEY(id) REFERENCES classes(id)

);

**Q3 [15 pts]** Consider a relation R in table (a). Complete the table (b) for given functional dependencies (FD). Please just answer yes, no or unknown.

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | D |
| A1 | B1 | C1 | D1 |
| A1 | B2 | C2 | D2 |
| A1 | B3 | C3 | D1 |
| A2 | B1 | C2 | D1 |
| A2 | B3 | C4 | D1 |
| A3 | B2 | C5 | D2 |
| A4 | B1 | C5 | D1 |

|  |  |  |  |
| --- | --- | --- | --- |
| FD | Satisfied by R  (Y/N/UNK) | Hold on R (Y/N/UNK) | Trivial (Y/N) |
| A→B | N | N | N |
| B→A | N | N | N |
| B→D | Y | UNK | N |
| BD→C | N | N | N |
| ABC→A | Y | Y | Y |
| BD→D | Y | Y | Y |
| AD→C | N | N | N |
| A→C | N | N | N |
| AB→C | Y | UNK | N |
| BC→D | Y | UNK | N |
| ABC→D | Y | UNK | N |
| D→ABC | N | N | N |
| BD→A | N | N | N |
| AB→D | Y | UNK | N |
| D→B | N | N | N |

**Table (a)**

**Table (b)**

**Q4 [10pts]** Consider a relation R(A,B,C,D,E,F,G) and a set of functional dependencies FD={A→B, AB→C, C→D, EG→F } which hold on R1. Using Armstrong's axioms verify if the following functional dependencies (FD) hold on R1 and if FD holds, provide your solution/proof that it holds:

|  |  |  |
| --- | --- | --- |
| FD | Yes/No | Proof if yes |
| A→D | Y | A🡪A(reflexivity)&A🡪B(union)=>A🡪AB&AB🡪C(transitivity)=>A🡪C&C🡪D(transitivity)=>A🡪D |
| B→C | N |  |
| BD→C | N |  |
| A→BCD | Y | A🡪A(reflexivity)&A🡪B(union)=>A🡪AB&AB🡪C(transitivity)=>A🡪C&A🡪B(union)=>A🡪BC  C🡪D&C🡪C(reflexivity)=>C🡪CD(augmentation)=>BC🡪BCD  (transitivity)=>A🡪BCD |
| AEG→BCDF | Y | A🡪BCD  (augmentation)=>AEG🡪BCDEG  EG🡪F(augmentation)=>BCDEG🡪BCDF  (transitivity)=>AEG🡪BCDF |

**Q5 [8 pts]** Consider a relation R(A,B,C,D,E,F,G) and a set of functional dependencies FD={A→B, AB→C, C→D, E→F } hold on R. Are decompositions in the table lossless and why?

|  |  |  |
| --- | --- | --- |
| Decomposition | Lossless? Y/N | Why |
| R1(ABCD) and R2(AEFG) | Y | R1 and R2 is lossless-join wrt F if and only if the closure of F contains R1∩R2🡪R1 or R1∩R2🡪R2. In this case, R1∩R2=A and it gives A🡪ABCD |
| R1(EBCD) and R2(AEFG) | N | R1 and R2 is lossless-join wrt F if and only if the closure of F contains R1∩R2🡪R1 or R1∩R2🡪R2. In this case, R1∩R2=E and it doesn’t give either E🡪ABCD or E🡪AEFG |
| R1(ABCDE) and R2(AEFG) | Y | R1 and R2 is lossless-join wrt F if and only if the closure of F contains R1∩R2🡪R1 or R1∩R2🡪R2. In this case, R1∩R2=AE and it gives either AE🡪ABCDE |
| R1(ABCD) and R2(BCDEFG) | N | R1 and R2 is lossless-join wrt F if and only if the closure of F contains R1∩R2🡪R1 or R1∩R2🡪R2. In this case, R1∩R2=BCD and it doesn’t give either A🡪ABCD or A🡪 BCDEFG |

**Q6 [8 pts]** Consider the following relations with the associated functional dependencies. Decide, whether those relations are in (a) BCNF, (b)3NF, (c) neither in BCNF nor 3NF normal form.

|  |  |  |
| --- | --- | --- |
| Relation, FD | Answer (a, b, or c) | Solution |
| R1(A,B,C)  {A→B, B→C} | c | A is the candidate key of R1, but both left side of FDs don’t contain key A for R1. |
| R2(A,B,C)  {A→B, A→C} | a, b | A is the candidate key of R2. Then left side of FDs all contain key A for R2, so R2 is BCNF, and also a 3NF. |
| R3(A,B,C,D)  {A→BCD, AB→BCD} | a, b | A is the candidate key of R3. Then left side of FDs all contain key A for R3, so R3 is BCNF, and also a 3NF. |
| R4(A,B,C,D)  {AB→CD, D→B} | b | AB is a candidate key of R4. The FD of AB🡪CD contains a key for R4, and the FD of D🡪B shows that B is part of key AB for R4 |

**NOTE**: For the next questions, Q7 – Q9, you are running a DBMS on a computer that has 4 kByte disk block size. Reminder: 1kByte=1024bytes, 1 MByte = 1024 kBytes.

**Q7 [4 pts]** Table T in your database D has size 6kBytes. How much space does Table T take on the drive? Show your calculations.

Your answer here (just one number): 8 kBytes

Your calculations here:

6 kBytes / 4 kBytes = 1.5 blocks, but disk block size is integer, so it will cost 2 blocks.

Therefore, the usage of space is 2\*4 kBytes = 8 kBytes = 8\*1024 bytes = 8192 bytes.

**Q8 [5 pts]** Table T in your database D has size 100MBytes. You execute a query: “select \* from T where num=1000”. How much data will be read from the drive? Show your calculations.

Your answer here (just one number): 100 MB = 102400 kBytes = 104,857,600 bytes

Your calculations here:

Because the whole data will be read from the drive in order to execute the query, so the amount of data read from the drive is 100\*1024\*1024 bytes = 104,857,600 bytes.

**Q9 [5 pts]** How your answer will be different from question 8, if clustered index for column num is used for table T. Assume that size of the required index structure is Y kBytes. And assume that n\*size\_of\_tuple = block\_size, where n is natural.

Your answer here (just one number): Y + 4 kBytes

Your calculations here:

Because block\_size is 4 kBytes, and size of required index structure is Y kBytes. The data will be read the index first, which is Y/4 blocks. After that, data will be read the match-up block. The total blocks to be read is Y/4+1 blocks. The data to be read is 4\*(Y/4+1) = Y+4 kBytes.

**Q10 [15 pts]** Draw a valid B+ tree below for the search keys (1, 2, 3, 4, …, 20). Assume the keys are inserted in their natural order. The order of the tree is 3.

(a) How many entries can the root have?

Answer: the root have a range of  entries.

(b) How many entries can other nodes contain?

Answer: each nodes contain a range of

(c) Draw this valid B+ tree