MSDS 7330 File Organization and Database Management Midterm Exam

**Name: Cory Nichols**

MidTerm Exam: Directions

This is a midterm exam for MSDS 7330, File Organization and Database Management. This document contains the questions for the exam. For your answers, create a Word document that clearly identifies every question number and your answer to that question. Name the Word file containing your answers ‘yourLastNameMSDS7330MidtermExam.doc’. For example, my Word file would have the name EngelsMSDS7330MidtermExam.doc.

Answer each question fully and completely. Show all of your work and state your assumptions where appropriate. Each answer is worth an equal number of points. For each multiple choice question, record the one letter of your chosen answer (1/2 the points) and write 2-5 sentences explaining why your chosen answer is correct (1/2 the points). Note that one-sentence explanations will receive 0 points even if they are correct.

The questions may have hints embedded within them regarding the answer. Follow these hints as appropriate for full points.

Collaboration is expected and encouraged; however, each student must hand in their own exam. To the greatest extent possible, answers should not be copied but, instead, should be written in your own words. Copying answers from anywhere is plagiarism, this includes copying text directly from the textbook. Do not copy answers. Always use your own words. Directly under each question list all persons with whom you collaborated and list all resources used in arriving at your answer. Resources include but are not limited to the textbook used for this course, papers read on the topic, and Google search results. Note that Google is not a reference. It is a tool to find references. Don’t forget to place your name in the Word document itself.

MidTerm Exam: Questions

1) Which of the following is a way in which a data definition language (DDL) differs from type declaration language such as Java?

a) Executing an action in the DDL results in the creation of an object in the database.

b) DDLs allow consistency constraints to be specified.

c) DDLs support authorization, giving different access rights to different users.

d) All of the above. [Hint: a description for this answer would identify the corresponding type definition language  characteristic/lack thereof for each of the three DDL characteristics.]

e) None of the above. [Hint: a description for this answer would identify at least two ways in which a DDL differs  from a type declaration language.]

**ANSWER: D – All of the above**

Type declaration languages often simply do not allow constraints OR authorization at a user level such as in SQL DDL. Further, DDL allows physical creation of objects in the database. In a type declaration language, objects are temporary or abstract. However, programming languages often have a much more complex data type structure.

2) Which of the following is NOT a disadvantage of using a file processing system to perform database functionality?

a) Data inconsistency.

b) Atomicity issues

c) Concurrent access anomalies

d) Application programs to access data.

**ANSWER: D – Application programs to access data**

File processing systems utilize application programs to access data. Database management systems are also application programs utilized to access data, however, they are more universal and provide for a platform where changes can be made easily, preserving data integrity and allowing multiple users to modify data. Write is atomic, no intermediate state.

3) In designing and setting up a database for a particular enterprise, what is the first step that should be taken? [Hint: Explain why your chosen answer should be the first step.]

a) Define the enterprise requirements

b) Define the integrity constraints on the data

c) Define the physical implementation of the database

d) Create/initialize the database.

**ANSWER: A – Define the enterprise requirements.**

The first step is always to define the enterprise requirements. These requirements will subsequently drive a good portion of the integrity constraints on the data. Once the requirements are defined, a data model can be created.

4) In designing and setting up a database for a particular enterprise, what is one of the last steps that should be taken? [Hint: Explain why your chosen answer should be one of the last steps.]

a) Define the enterprise requirements

b) Define the integrity constraints on the data

c) Define the physical implementation of the database

d) Create/initialize the database.

**ANSWER: D – Create/initialize the database.**

Defining the enterprise requirements is the first step. Further, creating a data model via an E:R model should be the second step. Integrity constraints and the physical level would come after the E:R diagram. Once all of these requirements are gathered, along with most likely triple checking your requirements are correct, THEN a database can be created.

5) What are three possible data abstraction levels in a database? [Hint: Explain what each abstraction level is and give an example for each level.]

a) (Physical, Logical, Table)

b) (Attribute, Type, Value)

c) (Physical, Logical, View)

d) (Tuple, Attribute, Table)

**ANSWER: C – Physical, Logical, View**

The physical level of abstraction is by far the most detailed level. It represents the hardware level procedures such as fetching data, data storage, indexing and other ‘physical heavy lifting’ the database is responsible for. The Logical level hides the detail of the physical level and presents all of the under-workings as relations, tuples and relationships. Most of the work from a programming and development perspective is done at the logical level. The view is the highest level of data abstraction. Data is in a relation/table form and allows for exact specification of the data at the logical level. It also provides for a decent security mechanism.

6) Which of the following data models is best suited for use with SQL? [Hint: Provide a definition of the data model and explain why SQL works well with it.]

a) Relational model

b) Entity-Relationship model

c) Object-Based Data model

d) Semi structured Data model

e) All of the above. [Hint: a description of this answer would explain each of the answers a-d above.]

f) None of the above. [Hint: a description for this answer would provide an example and description of a data model well suited for use with SQL.]

**ANSWER: A – Relational Model**

The relational model stores data in relations and tables. Each table has a set of attributes (columns) and tuples (rows). At the intersection of attributes and tuples are values. The relational model provides for integrity constraints in the form of domain, key and referential integrity constraints. These constraints help to preserve data integrity. SQL is able to interact extremely well with the relational model via DML and DDL. Relations can be joined together easily, tuples can be filtered, combined, calculated and modified quickly all under a standard (ANSI) language umbrella.

7) A Data Definition Language (DDL) is used to do which of the following?

a) Specify a database schema.

b) Specify the storage structure.

c) Specify consistency constraints.

d) Specify the access methods used in the database.

e) All of the above. [Hint: a description for this answer would explain each of the answers a-d above.]

f) None of the above. [Hint: a description for this answer would provide two examples, and explanations of what DDLs are used for.]

**ANSWER: E – All of the above.**

DDL can specify a database schema and storage structures via CREATE syntax and statements. Creating databases, tables, indexes, views, adding or subtracting columns all are executed via DDL. The DBA can grant or restrict access to users to SELECT, INSERT, UPDATE, DELETE and more operations. Keys and other constraints are also specified using DDL. This can happen with ALTER statements, CREATE statements and other methods.

8)  A Data Definition Language (DDL) is used to do which of the following?

a) Specify a database schema.

b) Specify the storage structure.

c) Specify consistency constraints.

d) Specify the access methods used in the database.

e) All of the above. [Hint: a description for this answer would explain each of the answers a-d above.]

f) None of the above. [Hint: a description for this answer would provide two examples, and explanations of what DDLs are used for.]

**ANSWER: E – All of the above.**

DDL can specify a database schema and storage structures via CREATE syntax and statements. Creating databases, tables, indexes, views, adding or subtracting columns all are executed via DDL. The DBA can grant or restrict access to users to SELECT, INSERT, UPDATE, DELETE and more operations. Keys and other constraints are also specified using DDL. This can happen with ALTER statements, CREATE statements and other methods.

9) Database systems are typically configured in a tiered structure that allows access to the database only through an authorized application. How many tiers are most appropriate for a large application accessed over the World Wide Web? [Hint: Explain why your chosen answer is appropriate (and better than the others) for the intended system.]

a) 2

b) 3

c) 4

d) 5

**ANSWER: B – 3.**

Three tier architectures are more preferable for the web. The client does not make direct database calls, it speaks to the application server instead. The application server communicates with the database for data access. This helps to centralize logic, control flow of data requests and uphold integrity of the underlying data.

10) In 1970, E.F. Codd published his paper, “A Relational Model of Data for Large Shared Data Banks,” in Communications of the ACM. This is recognized as the landmark paper on the relational model for databases. In this paper, which problem does Codd identify as likely to become a practical problem due to the inclusion of multiple types of data within a single database? [Hint: Discuss if this problem still exists or how it has been solved in modern database systems.]

a) Access control

 b) Usability of set operations

c) Redundancy/Normalization

d) Consistency

**ANSWER: D – Consistency.**

Codd mentions that consistency could become a ‘serious practical problem’ in his paper. Codd specifically mentions an issue with data inconsistency where a user introduces a new element that introduces inconsistency. The project, in Codd’s paper, does not have an integrity constraint, such as a foreign key, associated to the product as part of the department. Today, foreign keys would prevent such an entry from being made, keeping the database in a consistent state.

*Resource: “A Relational Model of Data for Large Shared Data Banks” Codd, E.F.*

11) Which of the following is the primary means used to uniquely identify a tuple in a relation during the operation of the database system? [Hint: Define the answer and explain how it works.]

a) Candidate key

b) Primary key

c) Superkey

d) Foreign key

**ANSWER: B – Primary key.**

The primary means to identify a tuple in a relation is via a primary key. The primary key is chosen from a population of candidate keys, which are minimal subsets of super keys in a relation. The primary key can be a unique id (a single attribute) or a combination of attributes.

12) Which of the following is the set of all possible keys that can be used to uniquely identify a tuple? [Hint: Define the answer and explain how it works.]

a) Candidate key

b) Primary key

c) Superkey

d) Foreign key

**ANSWER: C – Superkey.**

A superkey is the set of all possible attributes that can uniquely identify a tuple. No two tuples should have the same superkey. The superkey functions as a unique identifier for each tuple, however, it is not the most efficient key. Normally, superkeys are minimized into candidate keys and a primary key is chosen from the pool of candidate keys.

13) The key that is ultimately used to uniquely identify a tuple is chosen directly from the set of all of which of the following keys? [Hint: Define the answer and explain how it works.]

a) Candidate key

b) Primary key

c) Superkey

d) Foreign key

**ANSWER: A – Candidate Key.**

A primary key is chosen from the set of all candidate keys, which are minimal sets of the super key that uniquely identify each tuple in a relation. Tuples are also functionally dependent on candidate keys, however, the database developer normally chooses one candidate key as her primary key.

14) Which of the following keys creates a *referential integrity constraint*? [Hint: Define the answer and explain how it works.]

a) Candidate key

b) Primary key

c) Superkey

d) Foreign key

**ANSWER: D – Foreign Key.**

The foreign key creates a referential integrity constraint. Foreign keys function as integrity constraints in that they limit the domain of an attribute(s) in one relation based on the domain of an attribute in another relation. For example, in our university database example, the instructor relation may have department as part of the instructor schema. In order to establish consistency, we’d place a referential integrity constraint in the form of a foreign key on the department attribute in the instructor table that REFERENCES the department relation, specifically the department name, to ensure no erroneous values are inserted into the instructor table.

TABLE I STUDENT-TUTOR TABLE

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CourseID | StudentID | Date | TutorID | Topic | Room | Grade | Book | TutEmail |
| U1 | St1 | 23.02.03 | Tut1 | GMT | 629 | 4.7 | Deumlich | tut1@fhbb.ch |
| U2 | St1 | 18.11.02 | Tut3 | GIn | 631 | 5.1 | Zehnder | tut3@fhbb.ch |
| U1 | St4 | 23.02.03 | Tut1 | GMT | 629 | 4.3 | Deumlich | tut1@fhbb.ch |
| U5 | St2 | 05.05.03 | Tut3 | PhF | 632 | 4.9 | Dmmlers | tut3@fhbb.ch |
| U4 | St2 | 04.07.03 | Tut5 | AVQ | 621 | 5.0 | SwissTopo | tut5@fhbb.ch |

15) What is a likely primary key in Table I? [Hint: Explain why your chosen answer is correct.]

a) Date

b) StudentID, TutorID

c) CourseID, StudentID, TutorID

d) StudentID, TutEmail

**ANSWER: C – CourseID, StudentID, TutorID.**

This is a table in serious need of normalization. The likely primary key is CourseID, StudentID, TutorID. This compound key is necessary because studentID, TutorID does not allow us to uniquely identify the courses. A student could take multiple courses in this case, which would not allow us to determine exact grade or book attribute values. Further a student could have multiple tutors as evidenced by student 2. Date is repetitive and not unique, so it cannot be a primary key. StudentID, TutEmail suffers from the same disadvantages as studentID, TutorID.

16) What is a likely foreign key in Table I?

a) StudentID

b) TutorID

c) CourseID

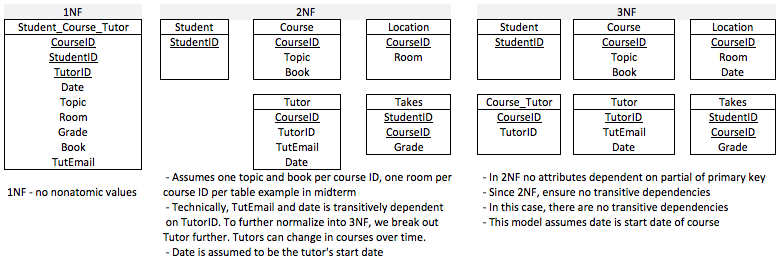
d) All of the above. [Hint: define the likely tables to which each of these three foreign keys point.]

e) None of the above. [Hint: describe one approach to create multiple tables resulting in the use of foreign keys in at  least one of the tables. Define the tables.]

**ANSWER: D – All of the Above.**

StudentID, TutorID and CourseID are all likely foreign keys. The StudentID attribute would likely reference ID in the students relation. The TutorID would likely reference ID in the tutors relation and the courseID attribute would likely reference ID in the course relation.

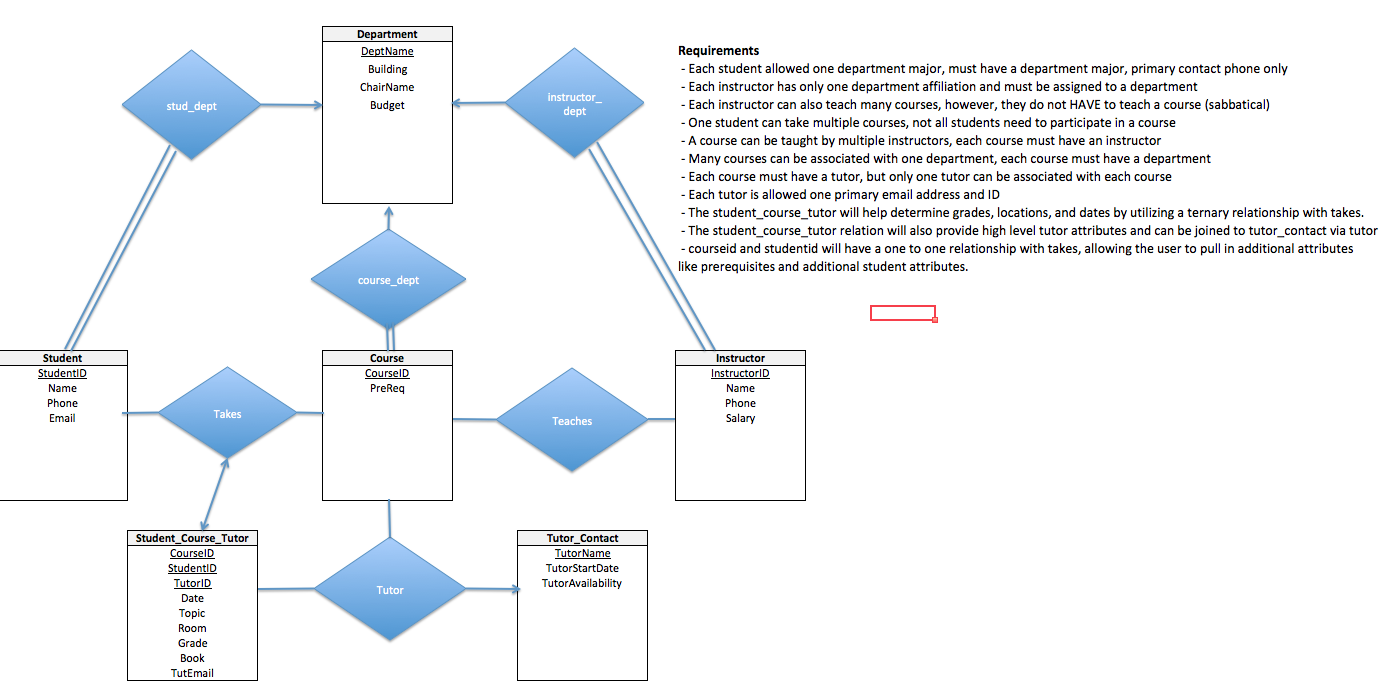
17) Define the set of relations that would result from normalizing this table into 3NF form. Draw the tables with their values.  [Hint: Show your work in arriving at this final set of relations.]



18) Table I is clearly just one table within a broader set of tables. Define and draw an E-R diagram that defines a complete  database in which this table exists and makes sense. You should have at least six entities in your final E-R diagram. [Hint: State your requirements.]

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CourseID | StudentID | Date | TutorID | Topic | Room | Grade | Book | TutEmail |
| U1 | St1 | 23.02.03 | Tut1 | GMT | 629 | 4.7 | Deumlich | tut1@fhbb.ch |
| U2 | St1 | 18.11.02 | Tut3 | GIn | 631 | 5.1 | Zehnder | tut3@fhbb.ch |
| U1 | St4 | 23.02.03 | Tut1 | GMT | 629 | 4.3 | Deumlich | tut1@fhbb.ch |
| U5 | St2 | 05.05.03 | Tut3 | PhF | 632 | 4.9 | Dmmlers | tut3@fhbb.ch |
| U4 | St2 | 04.07.03 | Tut5 | AVQ | 621 | 5.0 | SwissTopo | tut5@fhbb.ch |

**ANSWER:** Using the table in Q18 as an entity set, the graphic below, with requirements listed, could represent an actual database implementation:



*Resource Used: Database System Concepts, 6th Ed. Silberschatz, Sudarashan, Korth.*

19) Given the relation: *works( personName, companyName, salary, hireDate )* which of the attributes is an appropriate primary key? [Hint: Explain why your chosen answer is correct.]

a) personName

b) companyName

c) salary

d) hireDate

**ANSWER: A – personName.**

personName is the most appropriate (albeit flawed) primary key. It is likely the most unique attribute in the relation and will allow identification of a given tuple more effectively than companyName, salary and hiredate, which there would be many more similar values.

20) Given the relation: *works( personName, companyName, salary, hireDate )* which of the following attribute sets is an appropriate primary key? [Hint: Explain why your chosen answer is a better answer than your answer for question 19.]

a) personName, salary

b) salary, hireDate

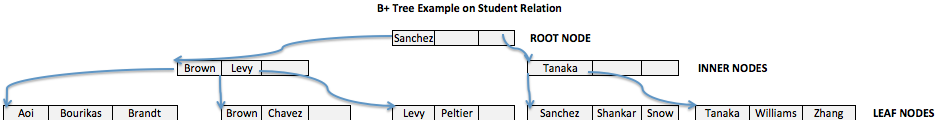
c) personName, hireDate

d) companyName, hireDate

**ANSWER: C – personName, hireDate.**

personName, hireDate is a more appropriate primary key than personName. This is because personName is open to duplicate values. This duplication of values would eliminate personName as a primary key. By concatenating hireDate to personName, our composite key would differentiate on employees with the same name by hire date. The most optimal primary key would be a unique ID in this case, however, we must work with what we have in the relation.

21) Create a B+-tree index for the *student* relation from the course textbook. This relation may be found on page 1279 of the textbook. Index on *name.*



Use the university relations from the textbook, defined in Appendix A, for the following questions. Simply write SQL queries to retrieve the stated information. [Hint: just write the SQL queries and not the resulting relations.]

22) Find the names of all students who have taken at least one Comp. Sci. course.

**ANSWER:**

SELECT distinct name

FROM student s

INNER JOIN takes t ON s.id = t.id

WHERE t.course\_id IN(

SELECT course\_id

FROM course

WHERE dept\_name = "Comp. Sci.");

23) Find the IDs and names of all students who have not taken any course offering before Spring 2009.

**ANSWER:**

SELECT distinct t.ID, s.name

FROM takes t

INNER JOIN student s

ON t.id = s.id

WHERE t.ID NOT IN(

SELECT id

FROM takes

WHERE year < 2009

);

The semester prior to Spring 2009 is Fall 2008. To simplify the subquery we use a less than operator on year = 2009. Because the data set starts at Spring 2009, we see all students who have taken a course will appear in our resulting relation.

24) For each department, find the maximum salary of instructors in that department. [Hint: You may assume that every  department has at least one instructor.]

**ANSWER:**

SELECT dept\_name, max(salary) max\_sal

FROM instructor

GROUP BY dept\_name

ORDER BY max\_sal DESC;

25) Find the lowest, across all departments, of the per-department maximum salary computed by the preceding query. [Hint:  store the preceding query as a relation, and then operate on that relation.]

**ANSWER:**

I executed this a few ways, with department name and without. As well as with only derived tables and with the preceding query stored as a relation.

**Using another relation:**

**DDL to Create Table:**

CREATE TABLE max\_salaries AS

(SELECT

dept\_name,

max\_sal

FROM (

SELECT dept\_name, max(salary) max\_sal

FROM instructor

GROUP BY dept\_name

ORDER BY max\_sal DESC

) r);

**DML to find minimum salary with department:**

SELECT

dept\_name,

max\_sal

FROM max\_salaries

WHERE max\_sal = (SELECT min(max\_sal) FROM max\_salaries);

**DML to find minimum salary only:**

SELECT MIN(max\_sal)

FROM max\_salaries

**Solution Using derived tables:**

SELECT

dept\_name,

max\_sal

FROM (

SELECT dept\_name, max(salary) max\_sal

FROM instructor

GROUP BY dept\_name

ORDER BY max\_sal DESC

) r

WHERE max\_sal = (SELECT MIN(max\_sal)

FROM (

SELECT

dept\_name,

max(salary) max\_sal

FROM instructor

GROUP BY dept\_name

ORDER BY max\_sal DESC

)t);