Remy Lagrois

Midterm

Section 402

1. **d) All of the above**. A declarative language could do these things but programs/functions must be specifically written to do each of them. A DDL is created specifically to perform these operations and so the functions needed to execute them are already present in the language, no new code needs to be written to perform them. (textbook ch 1, https://www.techopedia.com/definition/1175/data-definition-language-ddl)
2. **d) Application programs to access data**. In truth even this is more of a problem in a file processing system than in a DBMS since new programs must be written as the need arises. However, each of the others is a greater disadvantage. Concurrent-access anomalies and data inconsistencies are somewhat related in that two users may change the same information in two different places to two different values will result in disagreement between the two locations. Also it is more difficult to enforce atomicity (that is an operation either executes in full or not at all) on an FPS while that ability can be baked into a DBMS. (textbook ch 1)
3. **a) Define the enterprise requirements**. This step must be done first since it defines how you will go about doing the following steps. It’s difficult to define integrity constraints or the physical implementation if you don’t actually know what data, how much data, and how that data will be used. (textbook ch 1)
4. **d) Create/initialize the database.** Since you can’t create a database in nothingness you must at least first have the physical implementation planned out. The other two steps should also be done first since they may define what the physical implementation will be. For the reasons mentioned above, the enterprise requirements should be defined first. (textbook ch 1)
5. **c) (Physical, Logical, View).** The physical level is the actual hardware the data’s stored on and the configuration of said hardware. The logical level should be independent of the physical level and is how the data is conceptually organized, for example an E-R diagram will show the logical level. The view level is what a database user actually sees. It presents only the information the user needs or is interested in and different users may have different “views” while the logical and physical levels remain the same across the whole database. (textbook ch 1)
6. **a) Relational model** The relational model uses a series of tables (called relations) which have a fixed number of columns (attributes) and a mutable number of rows. These tables are all connected, that is they have relationships. For example a database for a hospital may have one table which contains information about the different departments within the hospital. This table could be related to another table containing information about individual doctors since a department is made up of a group of doctors. This doctors table in turn may be related to a table with patient information since a doctor has patients to care for.
7. **e) All of the above** All of these are done with a DDL. Most basically the storage structure can be done using the CREATE command (in SQL). This results in a new relation being created. Integrity constraints are added when specifying what attributes the table will contain (for example that it can’t be null or an attribute is numeric with so many digits/decimal places). Primary keys can be defined here as can foreign keys which define how tables are related to each other. (textbook ch 3, https://www.techopedia.com/definition/1175/data-definition-language-ddl)
8. **c) A relation containing the average salary of all instructors within a department where the average salary is less than $50,000**

a = SELECT AVG(salary)

FROM instructor

WHERE salary < 50000

b = SELECT dept\_name, AVG(salary)

FROM instructor

GROUP BY dept\_name

d = SELECT dept\_name, AVG(salary)

FROM instructor

GROUP BY dept\_name

HAVING salary < 50000

(textbook ch 3)

1. **b) 3** In three tier architecture none of the database calls are actually contained on the users machine. Instead the user interacts with an application (in our case it could be a web based application) which in turn interactions with the application server over a network. This application server is what actually interacts and execute calls on the database. This means you don’t need to widely distribute the software contained on the application server and a user can simply go to the web-based application.
3. **b) Primary key** Primary keys are used to identify individual tuples. They can consist of one or multiple attributes (though one is more ideal). The primary key must be unique to that tuple, if it’s one attribute no other tuple can have the same value. If it’s multiple attribute than that combination of values must belong to only one record. (textbook ch 2)
4. **c) Superkey** All the attributes or combination of attributes which are unique to a single record is called the superkey. The superkey therefore is a set of possible keys. (textbook ch 2)
5. **a) Candidate key** The candidate key is a subset of the superkey. The possible keys contained in the candidate key use the minimal number of attributes that can uniquely identify a record. One of the keys from this subset is selected to be the primary key (textbook ch2).
6. **d) Foreign key** A foreign key is an attribute in one table which references a record in a second table. If a record contains a foreign key then the table which that key references must contain a record with the same value as foreign key itself; this forms the integrity constraint. (textbook ch2).
7. **c) CourseID, StudentID, TutorID** This answer is the minimum number of attributes needed to ID a specific record. Date alone doesn’t work since a student may have more than one tutoring session on the same day. StudentID and TutorID together also isn’t enough as a student may have the same tutor for more than one course while StudentID and TutEmail suffers from the same problem. (textbook ch2).


11. **a) personName** Company name is not good since basically every company has more than one person working for it. Hire date also doesn’t work since it’s quite likely the same or multiple companies will hire more than one person on the same date. Salary also doesn’t work since many people are likely to make the same amount of money (this does depend on the company and jobs being listed but there’s not enough information given to be useful). A person’s name (first and last) is the most likely to be unique. (textbook ch2).
12. **c) personName, hireDate** This works better than the previous answer since it has two attributes which reduces the likelihood of duplicates. Salary/hire date and Company Name/hire date both don’t work since many companies will hire people in batches and often the batches are for the same type of position which are more likely to have the same salary. Answer C also works better than the next best option of personName/Salary since the hireDate is immutable. That is if a record using option C as the primary key is unique when it is created it will be unique forever since anyone hired with the same name in the future will have a different date. If salary were used it could be changed later possibly making tuples to no longer be unique. (textbook ch2).