James Zafiri
12/10/2023
CSC6302 Module 07

Quiz Three

1.) Superkeys: 2^(n) – 1 = 2^4 – 1 = 16 – 1 = 15 superkeys, which are listed below
(GearName, Cost, ClassId, Weight) , (GearName, Cost, ClassId) , (GearName, Cost, Weight) , (GearName, ClassId, Weight) , (GearName, Cost) , (GearName, ClassId) , (GearName, Weight) , (Cost, ClassId, Weight) , (Cost, ClassId) , (Cost, Weight) , (ClassId, Weight) , (GearName) , (Cost) , (ClassId) , (Weight)

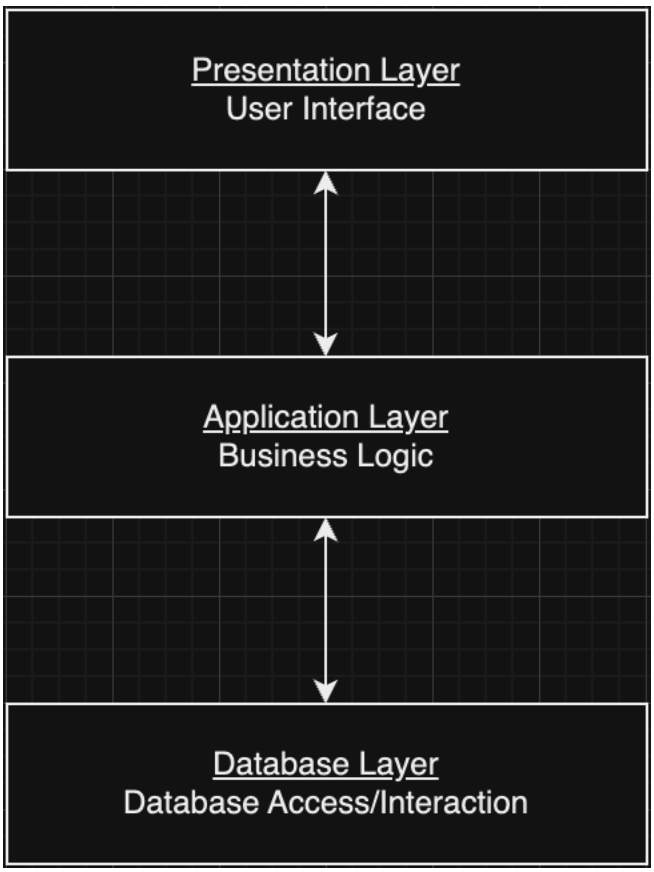
Candidate keys listed below:
(GearName, Cost, ClassId, Weight) , (GearName, Cost, ClassId) , (GearName, Cost, Weight) , (GearName, ClassId, Weight) , (GearName, Cost) , (GearName, Weight), (GearName, ClassId) , (GearName)

I made every candidate key one that included the GearName column. This is because it is the best way to make any row unique. I feel that it should be the primary key, or maybe (GearName, ClassId) since I am not exactly sure what kind of gears it is talking about here. It could be the latter incase two gears have the same name but feel as though they would all be distinct.

2.) Ensuring lossless decomposition means that you can still write a query and get the desired result sets, based on your known dependencies. It is breaking down a higher-level relation into multiple smaller relations while making sure that no information is lost in the process.

3.) In this table I am going to assume that the primary key is (IDSt, IDProf), or even (IDSt). This is because it contains the most unique way to receive the necessary information of the grade of that student based on the professor. While this table is in First Normal Form, I don't think the table is in Second Normal Form because there are partial dependencies. The Prof column does not depend on the IDSt, but the IDProf column. To fix this, I think there should be a separate table for students and one for professors. This way the column of the professor's name will not be depending on the student's ID, not causing a partial dependency. You can even have a connector table of foreign keys to help fix the issues, but as of right now it is not in Second Normal Form.

4.)



Presentation Layer: This layer is responsible for presenting information to the user and receiving their inputs. This can be a UI, Web Page, or Mobile App for example. This is for interacting with the user, but not handling business logic or directly interacting with the database.
Application Layer: This layer contains the logic of the application and business rules. It will process/manage the data based on certain requirements. This takes the user input, then applies the logic for the correct next action and acts like a middleman between the other two layers.
Database Layer: This layer is responsible for directly interacting with the database. It can perform operations on the database, as well as manage the access and storage of it. This layer translates the requests from the business logic into database queries, executes the queries, and then returns the results from the database.

5.) When writing a database application pros and cons:
   a.) Pros: This is more secure since you don't have open connections. Also, resource management since it's only used when you need it.
   Cons: This is slow because it takes time to establish a connection to the database. Frequent operations can definitely be time costly.
   b.) Pros: Reduces the time spent of repeatedly opening and closing connections for database operations. This in turn makes the operations faster as well.

Cons: This leads to less security from constantly having it open. It also consumes system resources from having the connection open so long, and there could be more limits to how many concurrent connections there are at once.

6.) There are multiple reasons why a returned ResultSet should be parsed into another data structure. The performance and flexibility will be improved because you can optimize your data for what you need to do with it, and different structures can give you different methods and operations to choose from. Abstraction is also another big reason because the code will be cleaner and easier to work with. When the ResultSet is parsed into another data structure it removes unneeded database-specific information, and it becomes a lot easier for the user to deal with.

7.) A partial dependency is when there is a column that only depends on some of the columns that are contained in the primary key instead of the entire key. Candidate keys are useful for identifying partial dependencies because any candidate key can be chosen as the primary key. You can go through all of them and test each key, so this way if any of the columns of the table do not depend on the entire key, it should not be chosen as the primary key.

8.) A transitive dependency is when a column in a table depends on something other than the primary of the key for uniqueness and context. Third Normal Form ensures that these are all eliminated, and the relation is decomposed into smaller relations.

9.) To guarantee a database schema is in First Normal Form:
- All columns must have a unique name.
- All columns must have a well-defined domain.
- All data in a column must be of the same type.
- All columns must contain one distinct value per row (Single Valued Attribute).
- The order in which you store data in a table does not matter.

10.) *attached separately