Normalization case studies examples

Business, Customers

The illustration of bellow uses particular, and course tables to explain why conversion of the database to first, second and third normal form is appropriate in a college environment.

Particular (Name, course_Code-> course_Number, department)

In terms of first normal form, the two relations above must meet the definition of a relation as explained above in order for them to be safely stored in the database. In the second normal form, the two relations must meet the definition of a relation where each relation has well defined unique keys with no partial key dependencies. As an example, course_Number has partial dependency on Course_Code which in turn has a partial dependency on Name within particular table. A gainLecturer_Room has a partial dependency on department which in turn has a partial dependency on Course_Number within Course table. These dependencies must be removed in order to attain second normal form for easy creation of rules in the database for storage of student details. Assuming that the second normal form relation for the first relation is Particular (Name, course Code-> department), a transitive dependency is created between the name and department which is again normalized to third normal form, resulting in to two other relations.

Denormalization table is accepted in situations where the process of updating a fact in the database is to be optimized by storing it in one place. It is also accepted where performance is of the database is to be maximized as well as storing the history. This can be justified in the situation where accompany stores information of customers in project table and customer table. In this case, the company management may want to require customer name from the time a project was started. In this case, the database administrator will just maintain the current name only in the customer table. However he will have to add the customer_Name attribute to the project table, in addition to the date itself, when the name was valid. As a result, , more fields are added to the project table which optimizes future update process as well as maintaining history of the customer in the customer table (Teorey et al., 2011, Bagui & Earp, 2011).

Business rules impact both database normalization and the decision to denormalize database tables in a significant manner. This is so that normalization helps to create rules and patterns that can be applied to any changes in the database. This is tremendously significant in a business where the same rules are required to manage business changes in trends and patterns of seasonal sales. Denormalization helps to, optimally, update the day to day business transactions stored as facts in one place within the database.

References

Teorey, T. J., Lightstone, S. S., Nadeau, T., & Jagadish, H. V. (2011). Database modeling and design: Logical design. Elsevier.

Singh, S. K. (2011). Database systems: Concepts, design and applications. Pearson Education India.

Bagui, S., & Earp, R. (2011). Database design using entity-relationship diagrams. CRC Press.