Database Management System – cs422 DE

Assignment 6 – Week 9

---

(1) 3NF essentially identifies

A. 1-* relationships
B. * - * relationships
C. 1-1 relationships
D. None of the above

ANS:

A

(2) While checking our tables for normalization, if we find that they are not even in 2NF then we must have missed some

A. 1-* relationships
B. * - * relationships
C. 1-1 relationships
D. None of the above

ANS:

B

(3) How to identify parent and child entities in a relationship?

ANS:

In a relationship, parent entity is identified as the one which has the primary key and child entity as the one which has the foreign key.

In the other way, the parent entity exists independently and has a unique identifier, and the child entity depends on the parent entity for its existence and referencing the parent entity via foreign key.

(4) Solve review question 17.2/ 16.2 (a,b,c,d,g,i) from 5th /4th edition of the course text book.

ANS:

The rules for deriving relations that represent

(a) strong entity types: Create an independent relation whose attributes are simple and constituent if it is composite.
e.g., Student -> strudentId, firstName, lastName, email

(b) weak entity types: Create a relation whose attributes are simple and whose primary key is partially or fully dependent on parent entity
e.g., Enrollment -> studentId, courseId, startDate

(c) one-to-many (1:*) binary relationship types: Put the primary key on the 1 side and foreign key on the * side.

e.g., One student has multiple enrollment

(d) one-to-one (1:1) binary relationship types: Choose parent and child entities and use one key as primary key on parent entity and as foreign key on child entity
e.g., Student and Passport relation

(e) one-to-one (1:1) recursive relationship types: Create a relation where the foreign key is referencing to that relation's primary key
e.g., Employee -> employeeId, employeeName, position, managerId

(f) superclass/subclass relationship types: Create a relation with a primary key and create another relation which foreign key is referencing to the primary key of other relation
e.g., Vehicle, Car, Motorbike relations where vehicle_id is the primary key of Vehicle relation and other two relations reference Vehicle table, using car_id and motorBike_id

(g) many-to-many (*:*) binary relationship types: Create a relation whose primary key is combined other two entities' primary keys.
e.g., Student, Course relations has many-to-many relationships
Enrollment -> studentId, courseId, startDate

(h) complex relationship types: Create a relation which contains the attributes of the relationship, as well as the primary keys of the related entities.
e.g., Employee, Project relationship is complex.
Assignment -> employeeId, projectId, startDate, endDate

(i) multi-valued attributes: Create a relation that contains the attribute and the primary key of the related entity.
e.g., Employee can have more than phoneNumber
EmployeePhoneNumber -> employeeId, phoneNumber

(5) Discuss how the technique of normalization can be used to validate the relations derived from the conceptual data model. (17.3/16.3)
ANS:
The technique of normalization can be used to validate the relations derived from the conceptual data model by ensuring that it conforms to the standard normal form, 1NF, 2NF and 3NF so that it can identify any issues with the relations and can refine them to ensure they are not causing data redundancy, anomalies, and others.

(6) Solve exercise 17.8/16.8 from the 5th /4th edition of the course text book. In the ERD, only those attributes are listed which are PK for that entity. You are required to add more attributes to the relations which will be logically applicable to that entity.
ANS:

PaymentMethod -> **pMethodNo**, pMethodName

Invoice -> **invoiceNo**, customerNo, invoiceDate, totalAmount, status

Order -> **orderNo**, orderDate, customerNo

OrderDetail -> orderId, productNo, price, quantity

Product -> **productNo**, productName, description

Customer -> **customerNo**, customerName, phoneNumber, address

Employee -> **employeeNo**, employeeName, phoneNumber, address, position, salary

Shipment -> **shipmentNo**, sMethodNo, invoideNo, shippedDate, estimatedDeliveryDate, trackingNumber, status

ShipmentMethod -> **sMethodNo**, shipmentName