KUTAISI INTERNATIONAL UNIVERSITY

Homework 2

Fundamentals of Databases 1
Dimitri Tabatadze
October 13, 2023

1. (a) The primary key is used for unique identification of an object and the primary index is used for arranging the data.

   (b) Without predefined order, we may generate $n!$ different private keys with n components - permutations (or reorderings).

   (c)
   - Primary key - Something that is used for uniquely identifying an object.
   - Candidate key - Something that *can* be used for uniquely identifying an object. Candidate keys have one or more alternatives.
   - Composite key - Multiple attributes combined in a predefined order used for uniquely identifying an object.
   - Foreign key - An attribute used for refering to an item in another table.

2. (a) When mapped, strong-weak entity relationship gets merged into the weak entity relation with the primary key being a composite of the weak entity's original primary key and the (now foreign) primary key of the strong entity. While on the other hand, simple `1:N` relations don't utilize the primary key of the entity on the side with cardinality `1` of the relationship as a part of composite primary key.

   (b) The foreign keys remain foreign, but in the case of the strong-weak relationship, they also become a part of composite primary key.

3. (a) Vertical

   (b) Horizontal

   (c) Horizontal

   (d) Vertical

   Enforcing uniqueness is more challenging to my mind - O(n) for every insertion. Duplicating the PKs from super types to the subtypes is needed during vertical mapping. Enforcing uniqueness over subtype relations is needed for horizontal mappnig.

4. (a)
```
student:   {[studID:int, name:string, semester:int]}
assistant: {[assistantID:int, name:string, researchArea:string, profID:int FK]}
course:    {[courseID:int, contactHours:int, title:string, profID:int FK]}
professor: {[profID:int, rank:int, name:string]}
enrolment:    {[studID:int FK, courseID:int FK]}
examination:  {[studID:int FK, courseID:int FK, grade:int, profID:int FK]}
requirement:  {[successor:int FK, predecessor:int FK]}
```
   (b) 7 final relations.

5. (a)
```
user:   {[ID:int, name:string]}
tweet:  {[ID:int, text:string, w_date:date, userID:int FK]}
follow: {[followerID:int FK, followingID:int FK]}
like:   {[userID:int FK, tweetID:int FK, l_date:date]}
```
   (b) 4 final relations.

6. :(

7. The following figures are for the standard cardinalities and min-max noation respectively: 1, 2

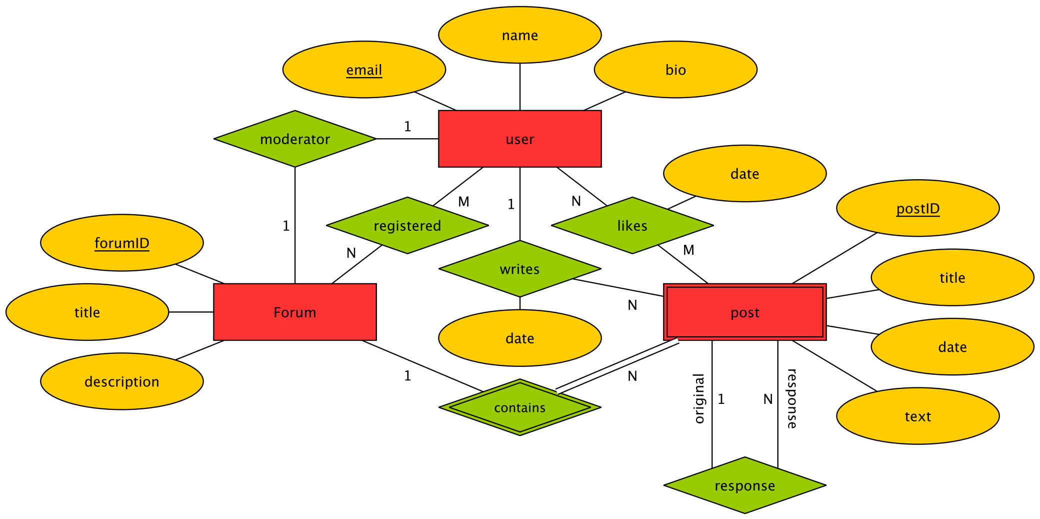

Figure 1: The ER diagram with associated cardinalities

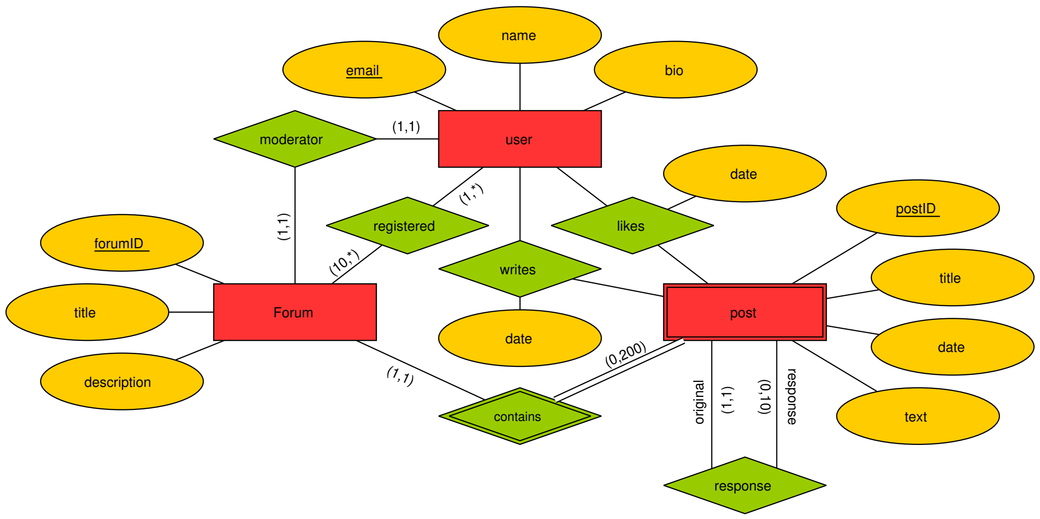

Figure 2: The ER diagram with associated min-max nnotation

8.
```
department:   {[depID:int, depName:string]}
employee:     {[emplID:int, name:string, function:string, works_in:int FK]}
parkingSpace: {[parkID:int]}
timeslot:     {[timeslotID:int, time:datetime]}
room:         {[roomID:int, buildingID:int FK, size:int, maxOcc:int]}
building:     {[buildingID:int, yearConstruction:date]}
reserve:      {[rommID:int FK, timeslotID:int FK, occasion:string, partyNo:int]}
```

9.
```
student:   {[studID:int, name:str]}
exam:      {[examPart:int, studID:int FK, finalGrade: int]}
professor: {[profID:int, name:string]}
examines:  {[examPart:int FK, studID:int FK, profID:int FK, grade:int]}
```