ST. XAVIER’S COLLEGE

**Maitighar, Kathmandu**

**(Affiliated to Tribhuvan University)**



**Database Management System**

**Theory Assignment #4**

**Submitted By**

Alok Shrestha

B.Sc. CSIT

Year II/IV Semester

013BSCIT005

**Submitted To**

Er. Sanjay Yadav

Lecturer

Department of Computer Science

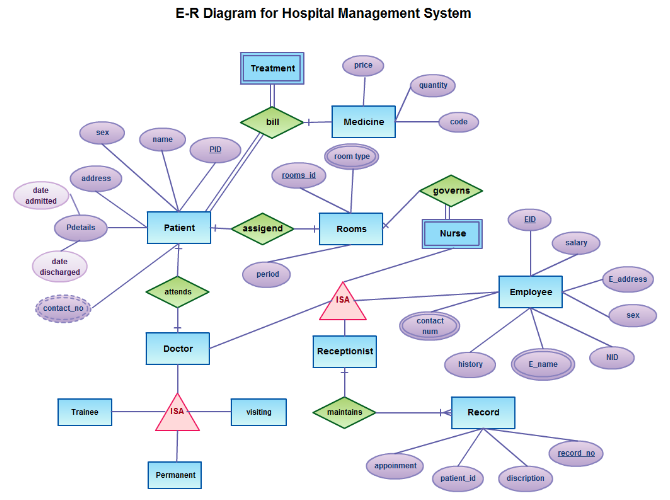
St. Xavier’s College

Maitighar, Kathmandu

**Submitted On**

August, 2015

**ER diagram with one case study**

****

**Design**

**Functional Design** is a paradigm used to simplify the **design** of hardware and software devices such as computer software and increasingly, 3D models. A **functional design** assures that each modular part of a device has only one responsibility and performs that responsibility with the minimum of side effects on other parts.

**Database design Process**

Design and implementation of an automated billing system for the University of North Carolina at Chapel Hill.

|  |
| --- |
| The database design process consists of the following steps[5]: |

|  |  |
| --- | --- |
| Step | Description |
| 1 | [Requirements collection and analysis](http://www.ibiblio.org/faint/finosaur/db/step1.html) |
| 2 | [Conceptual database design](http://www.ibiblio.org/faint/finosaur/db/step2.html) |
| 3 | [Choice of a DBMS](http://www.ibiblio.org/faint/finosaur/db/step3.html) |
| 4 | [Data model mapping](http://www.ibiblio.org/faint/finosaur/db/step4.html) (also called logical database design) |
| 5 | [Physical database design](http://www.ibiblio.org/faint/finosaur/db/step5.html) |
| 6 | [Database system implementation](http://www.ibiblio.org/faint/finosaur/db/step6.html) |

**Conceptual database design**

I have identified the following Access-specific entities (others are look-up tables derived from the SIS database structure):

Conventions used: underlined terms refer to primary keys (PK); "FK" refers to foreign keys. Data Architect is one possible tool to use to convert the following entities into appropriate Entity-Relationship Diagrams.

## Terms

1. code
2. description
3. start date
4. end date

### Examples: 992, Spring 1999, 1/6/99, 5/8/99

## Student Athletes

1. PID
2. ID
3. Last
4. First
5. Middle
6. Title
7. SSN
8. Sport 1
9. Sport 2
10. Male
11. First Year
12. E-mail
13. Line 1
14. Line 2
15. Line 3
16. City
17. State
18. Zip
19. Phone

## Award

1. TermID (FK to Terms)
2. PID (FK to Student Athletes)
3. Tuition
4. TuitionCap
5. Fees
6. FeesCap
7. Room
8. RoomCap
9. Board
10. BoardCap
11. Books
12. BooksCap
13. Room/Student?
14. Board/Student?
15. Granville?
16. Tuition/IS
17. Books/IS
18. Credit Hours
19. CountryValue (FK to Citizen)
20. ClassificationValue (FK to Classification)
21. CollegeValue (FK to College)
22. MajorValue (FK to Major)
23. Code (FK to Scholarship.Code)
24. SportValue (FK to Sport)
25. TaxStatusValue (FK to TaxStatus)
26. Special Program (FK to Special Program)
27. Notes
28. Effective Date
29. Cancellation Date
30. Withdrawal Date

## Vendors

1. VendorID
2. VendorName
3. Street
4. City
5. State
6. Zip
7. Contact Person Name
8. Phone
9. Fax
10. E-mail
11. Account#
12. Dept#

## Debits & Credits

1. Tx#
2. Term (FK to Award)
3. PID (FK to Award)
4. Order (FK to Transaction Types.Order, this IDs the transaction as tuition, fees, etc.)
5. Amount
6. Tax Code (FK to Tax Codes.Tax Code)
7. Reason (FK to Reasons.Reason)
8. Ticket#
9. X or K#

**Data model mapping or logical design** consists of converting the entity relationship diagram into tables in Fourth Normal Form (and documenting those instances that do not conform to this standard). In this database, the Student Athletes table has fields labeled "Sport 1" and "Sport 2". In 4NF, these fields would be separated into another table, comprised of PID and Sport as a joint primary key. I made the design decision to implement two identical text fields of 3 characters to simplify merges for queries and reports.

**Physical database design**, consists of executing the tables in an actual DBMS software file.

**Characteristics of relation**

* No Duplicate Tuples - A relation cannot contain two or more tuples which have the same values for all the attributes. i.e., in any relation, every row is unique.
* Tuples are unordered - The order of rows in a relation is immaterial.
* Attributes are unordered - The order of columns in a relation is immaterial.
* Attribute Values are Atomic - Each tuple contains exactly one value for each attribute.  
  It may be noted that many of the properties of relations follow the fact that the body of a relation is a mathematical set.

**E-R to Relational Mapping Algorithm**

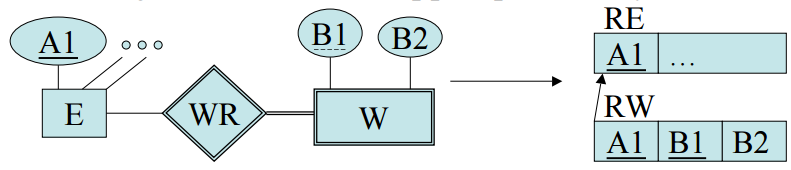
STEP 1: Mapping of regular entity types

* For each entity type E in the ER schema, create a relation R that includes all the simple attributes of E.
* Add only simple components from any composite attributes in E.
* Choose one of the key attributes of E to be a primary key of R.



STEP 2: Mapping of weak entity type

* For each weak entity type W with owner type E create a new relation RW that includes all the simple attributes of W as attributes of RW.
* In addition include a foreign key reference to the key of the translation RE of E.
* The key of RW will be the key of foreign key together with the mapped partial key from W.

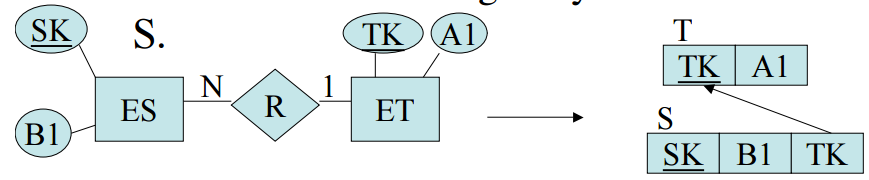


STEP 3: Mapping of binary 1:1 relationship type

* For each binary 1:1 relationship R in the ER Schema, identify the relations S and T that correspond to the entity types participating in R.
* There are three possible approaches:
  + The foreign key approach: Choose one of the relations, say S, and include in S a foreign key reference to the primary key of T. (Favor S over T if its corresponding entity participated totally in the relationship.)
  + Merged relation approach: When both relations correspond to entities that participated totally in the relationship, one can just merge the two relations into one.
  + Relationship relation approach: Set up a new relation with for the purpose of cross referencing the primary keys of tables S and T.

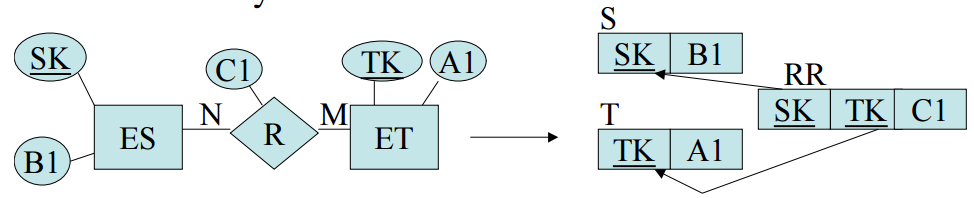
STEP 4: Mapping of binary 1:N relationship type

* For each 1:N binary relationship type R, identify the relations S and T corresponding to the entity types in this relationship.
* Further, suppose S is the N-side of the relationship.
* Then include a foreign key reference to R in S.



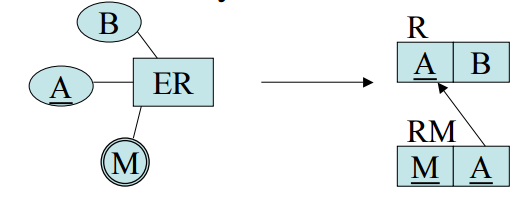
STEP 5: Mapping of binary M:N relationship type

* For each binary M:N relationship type R, create a new relation RR to represent R.
* Include as foreign keys of keys the primary keys of the relations corresponding to the two participating entities.
* Also add as attributes simple attributes of R as in the entity case.



STEP 6: Mapping of multivalued attributes

* For each multivalued attribute M, create a new relation RM. This relation will include an attribute corresponding to M as well as a foreign key reference to the relation corresponding to the entity that M was part of.
* The key will be both attributes.



STEP 7: Mapping of n-ary relation types

* For each n-ary relationship R with n >2, create a new relation RR to represent R.
* Include as foreign key attributes in RR the primary keys of the relations that correspond to the participating entities with cardinality constraints other than 1.
* The primary key of RR is the combination of these foreign keys.

**REFERENCES**

[1] http://creately.com/diagram/example/h7ithf1s1/E-R+Diagram+for+Hospital+Managment+System

[2] http://www.answers.com/Q/Characteristics\_of\_relations\_in\_relational\_database\_model

[3] K.L. University, http://www.kluniversity.in/elearn/materials/tdbqucsbsb77152TDBQUCSBSB.pdf

[4] http://www.ibiblio.org/faint/finosaur/db/index.html

[5] Elmasri, Ramez, & Navathe, Shamkant B., Fundamentals of Database Systems, Second Edition, Addison-Wesley Publishing Company, 1994, ISBN 0-8053-1748-1, pages 450-452.