**ST. XAVIER’S COLLEGE**

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



**Database Management System**

**Lab Assignment #4**

**Submitted by:**

PRANESH DHUNJU SHRESTHA  
013BSCCSIT026

**Submitted to:**

|  |  |
| --- | --- |
| **Er. Sanjay Kumar Yadav**  Lecturer  St. Xavier’s College |  |

**Date of Submission:** August 10, 2015

# ****3. Relational Database Characteristics****

1. Data in the relational database must be represented in tables, with values in columns within rows.
2. Data within a column must be accessible by specifying the table name, the column name, and the value of the primary key of the row.
3. The DBMS must support missing and inapplicable information in a systematic way, distinct from regular values and independent of data type.
4. The DBMS must support an active on-line catalogue.
5. The DBMS must support at least one language that can be used independently and from within programs, and supports data definition operations, data manipulation, constraints, and transaction management.
6. Views must be updatable by the system.
7. The DBMS must support insert, update, and delete operations on sets.
8. The DBMS must support logical data independence.
9. The DBMS must support physical data independence.
10. Integrity constraints must be stored within the catalogue, separate from the application.
11. The DBMS must support distribution independence.  The existing application should run when the existing data is redistributed or when the DBMS is redistributed.
12. If the DBMS provides a low level interface (row at a time), that interface cannot bypass the integrity constraints.

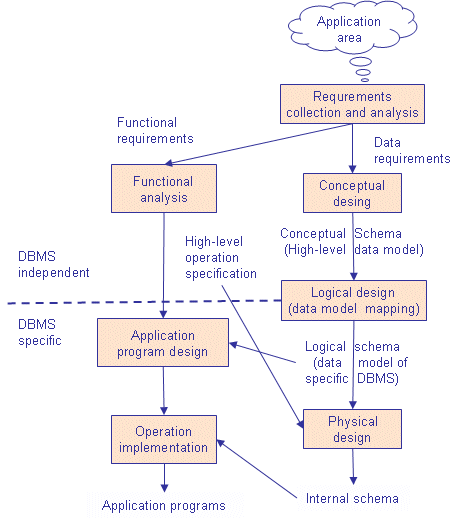
# 2.b Database Design

In the picture below there are the main phases of database design. Database design is connected with application design.

The requirements and the collection analysis phase produce both data requirements and functional requirements. The data requirements are used as a source of database design. The data requirements should be specified in as detailed and complete form as possible.

In parallel with specifying the data requirements, it is useful to specify the known functional requirements of the application. These consist of user-defined operations that will be applied to the database (retrievals and updates). The functional requirements are used as a source of application software design. Of course some functions may produce also needs for database design.

Note that some phases are database management system independent and some are dependent. The idea is to design first the database without thinking about the actual database system - just to concentrate on the data.



**Conceptual Design**

Once all the requirements have been collected and analyzed, the next step is to create a conceptual shema for the database, using a high level conceptual data model. This phase is called conceptual design.

The result of this phase is an Entity-Relationship (ER) diagram or UML class diagram. It is a high-level data model of the specific application area. It describes how different entities (objects, items) are related to each other. It also describes what attributes (features) each entity has. It includes the definitions of all the concepts (entities, attributes) of the application area.

During or after the conceptual shema design, the basic data model operations can be used to specify the high-level user operations identified during the functional analysis. This also serves to confirm that the conceptual schema meets all the indenfied functional requirements.

There are several notations to draw the ER diagram.

**Logical Design**

The result of the logical design phase (or data model mapping phase) is a set of relation shcemas. The ER diagram or class diagram is the basis for these relation schemas.To create the relation shemas is quite a mechanical operation. There are rules how the ER model or class diagram is transferred to relation shemas.The relation schemas are the basis for table definitions. In this phase (if not done in previous phase) the primary keys and foreign keys are defined.

### Physical Design

The goal of the last phase of database design, physical design, is to implement the database. At this phase one must know which database management system (DBMS) is used. For example, different DBMS's have different names for datatypes and have different datatypes.The SQL clauses to create the database are written. The idexes, the integrity constraints (rules) and the users' access rights are defined.Finally the data to test the database is added in.In parallel with these activities, application programs are designed.

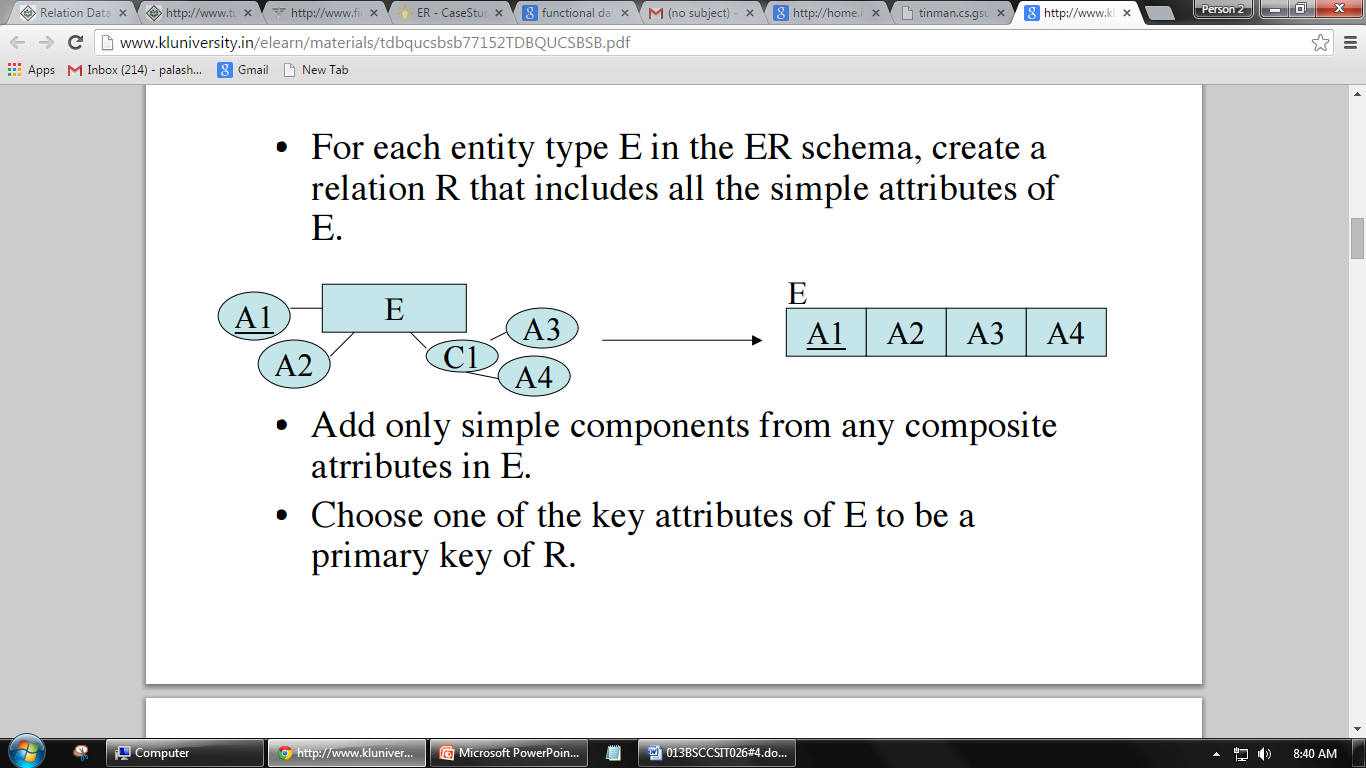
**4. ER to relational mapping algorithm**

**ER -> Relational Map Step 1**

• 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 atrributes in E.

• Choose one of the key attributes of E to be a primary key of R.

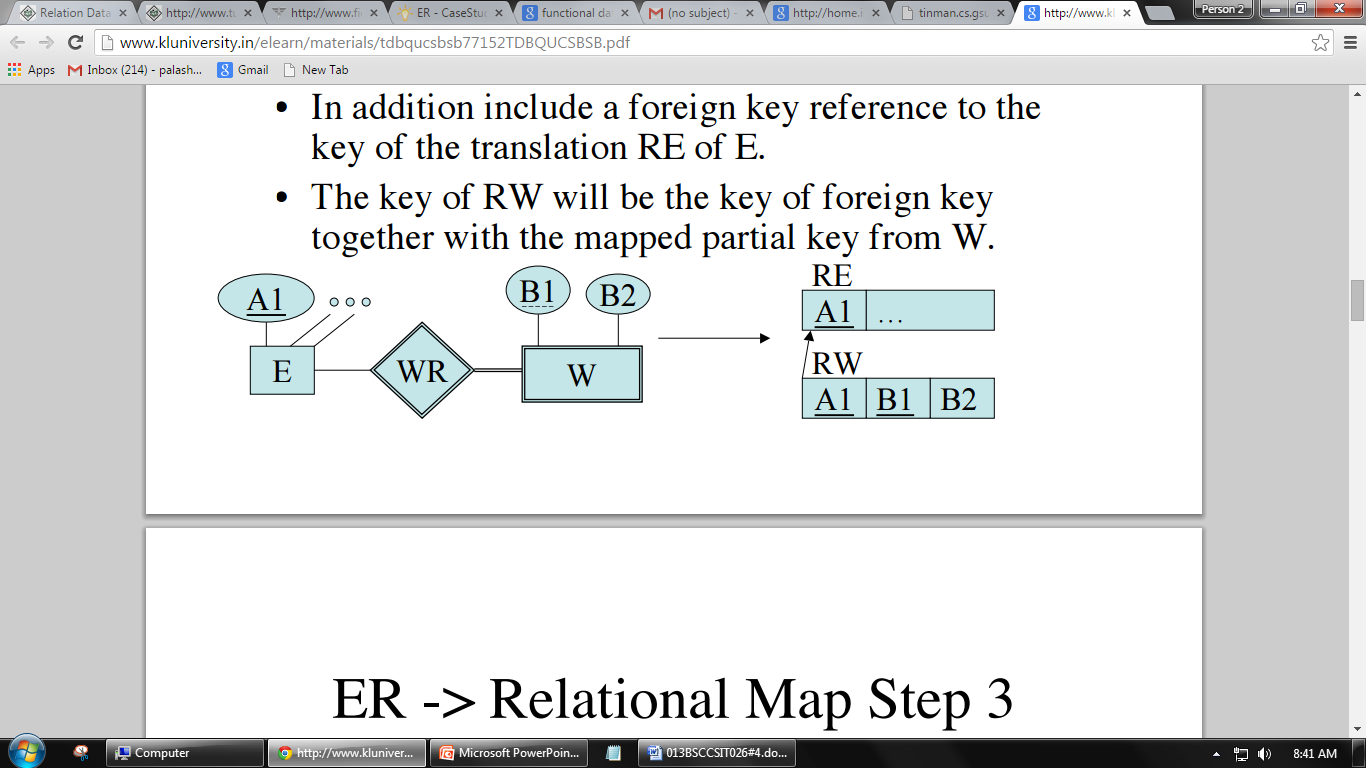


**ER -> Relational Map Step 2**

• 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.



**ER -> Relational Map Step 3**

• 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.

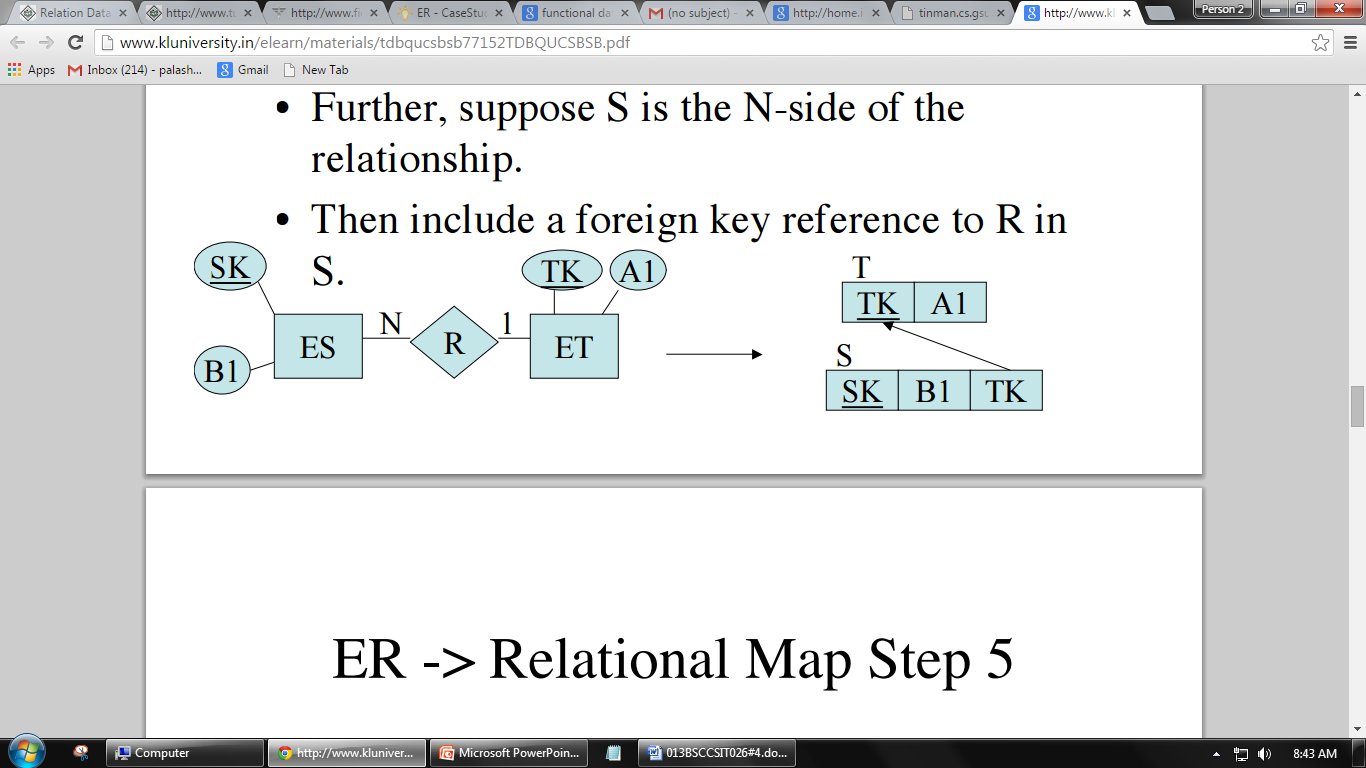
– Relationship relation approach: Set up a new relation with for the purpose of cross referencing the primary keys of tables S and T.

**ER -> Relational Map Step 4**

• 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.

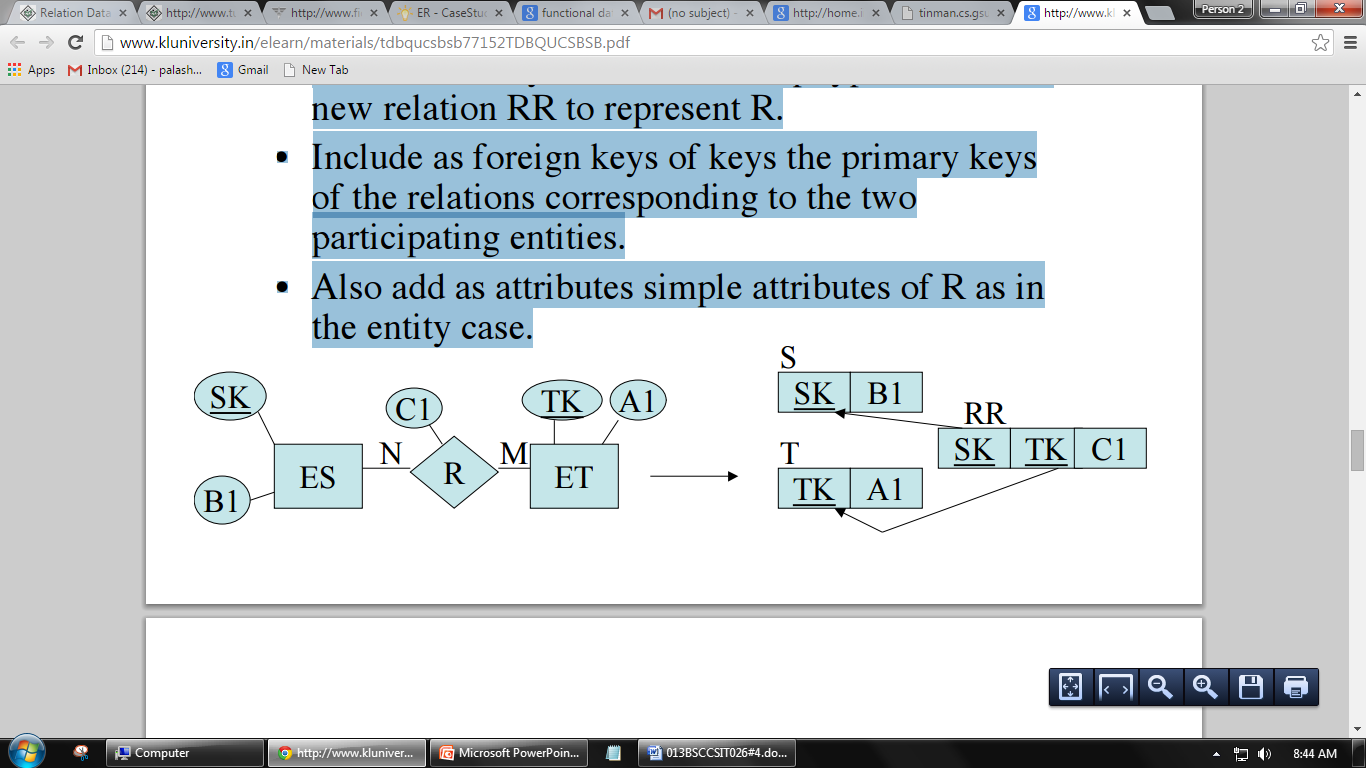


**ER -> Relational Map Step 5**

• 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.

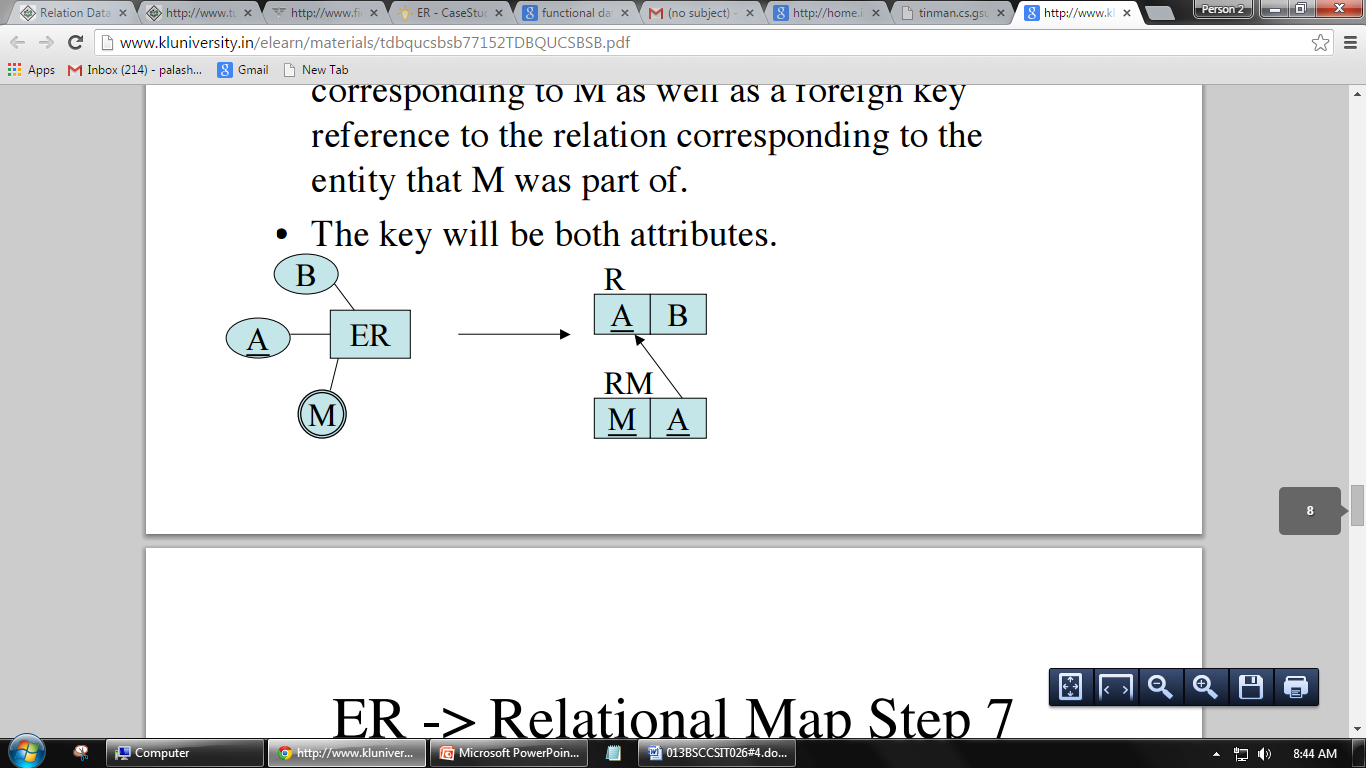
• Also add as attributes simple attributes of R as in the entity case.



**ER -> Relational Map Step 6**

• 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.

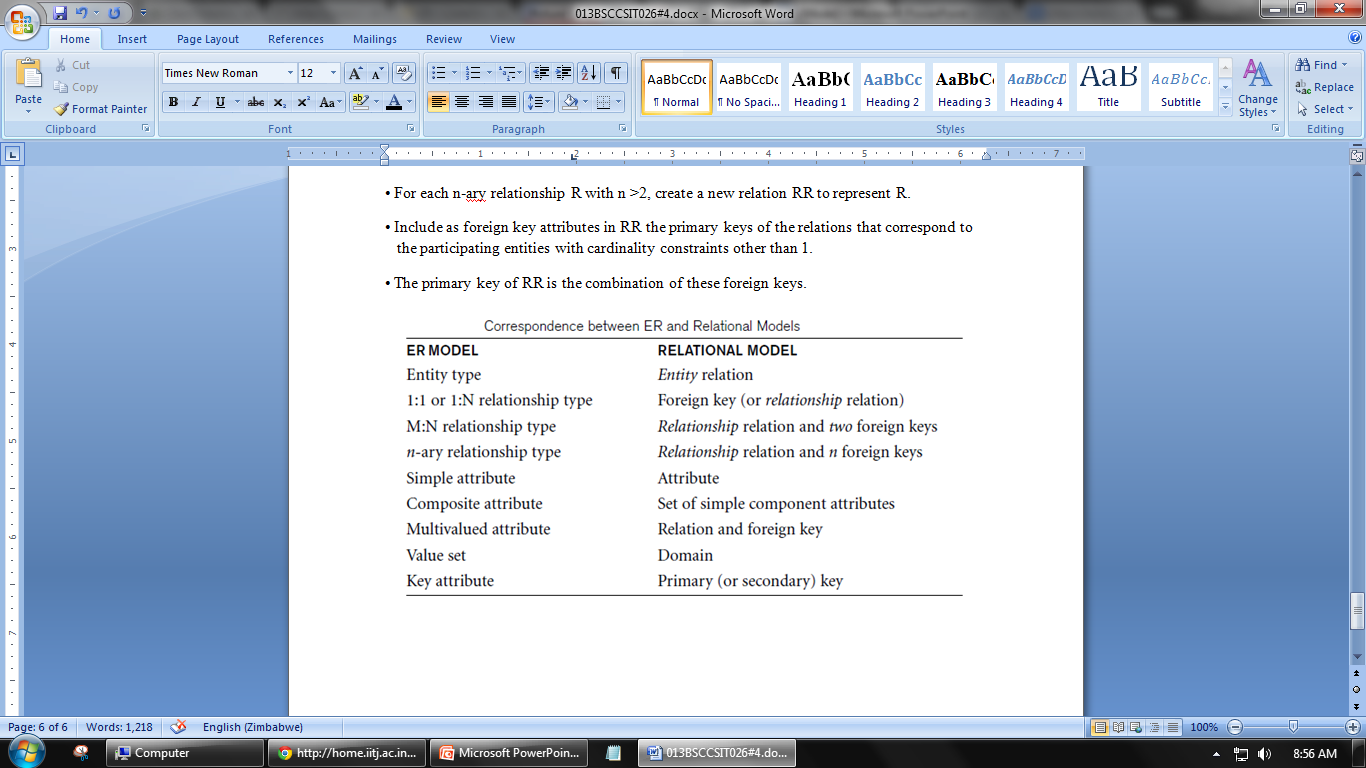


**ER -> Relational Map Step 7**

• 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.



1. ER diagram with one case study.
2. Design
   1. Functional design
   2. Database design
      1. Conceptual database design
      2. Logical database design
      3. Physical database design
3. Characteristics of relations
4. ER to relational mapping algorithm
   1. Mapping of regular entity type
   2. Mapping of weak entity type
   3. Mapping of binary 1:1 relationship type
   4. Mapping of binary 1:M relationship type
   5. Mapping of binary M:N relationship type
   6. Mapping of multivalued attributes
   7. Mapping of N- ary relationship type