

LOGICAL DATABASE DESIGN

Learning Units

- 8.1 Entity-relationship(E-R) modelling of data elements of an application.
- 8.2 Organization of data as relations
- 8.3 Normalization of relations
- 8.4 Creation of logical relational database
- 8.5 Objectives of database management system(DBMS)
- 8.6 Overview of DBMS.

LEARNING GOALS

In this module we will learn:

1. The Entity-Relationship(ER) modelling to develop a conceptual model of data.
2. How to organize data required in an application as relations
3. The need for normalizing relations
4. The various normal forms and their relevance
5. How to normalize relations to successive higher normal forms to form a relational database
6. The need for an integrated database in organizations
7. The goals of Data Base Management systems (DBMS)
8. The structure and organization of DBMS.

MOTIVATION

- When a DFD is developed we have a knowledge of all data elements required by an application
- Data dictionary lists all data elements but does not say anything about relationships between data elements
- Relationships are needed to logically group data elements into related sets or tables

MOTIVATION

- Such an organization
 - Reduces data duplication
 - Simplifies adding, deleting and updating data
 - Simplifies retrieval of desired data

MOTIVATION

- Logical databases give conceptual model.
- Logical databases need to be stored in physical media such as a hard disk for use by applications
- A system is needed to map the logical database to a physical medium which is transparent to an application program.
- Database management systems achieve this purpose

LOGICAL DATABASE DESIGN-INTRODUCTION

- Purpose to develop conceptual model of data
- This model specifies relationships among data items
- Using this, raw data are organized into tables of related data
- These tables are organized in such a way that:
 - a) duplication of data is reduced
 - b) operations of adding, deleting, changing data (together known as updating data) is simplified and systematized
 - c) systematization reduces accidental errors
 - d) Retrieval of data is facilitated
- Collection of these tables are called the database for the application

LOGICAL DATABASE DESIGN-INTRODUCTION

- Loosely one may call organization of related data put in a table as a “RELATION”
- Systematization by which related data are put in a table is called “NORMALIZATION”
- A method called entity-relationship analysis facilitates creation of relations

ENTITY-RELATIONSHIP MODEL

ENTITY: SPECIFIES DISTINCT REAL WORLD ITEMS IN AN APPLICATION

For example: vendor,item,student,course,teachers

RELATIONSHIP: MEANINGFUL DEPENDENCIES BETWEEN ENTITIES

For example: vendor supplies items
teacher teaches courses

Relationships are underlined above

ENTITY SETS

An entity set is collection of similar entities

Examples : * Set of all vendors of an organization is a vendor set

* Set of all items in a store is an item set

Every member of an entity set is described by its attributes

ATTRIBUTES

Attributes specify properties of members of entity set

Attributes also specify properties of relationships

Examples:

Entity : Vendor

Attributes : **vendor code**, vendor name, address

Relationship : supplies

Attributes : **vendor code**, **item code**, order no., qty. supplied, date of supply, price/unit

ENTITIES AND ATTRIBUTES

Example

Entity : Teacher

Attributes : **Teacher code**, teacher name, department, building, room no, phone no.

Relationship : Teaches

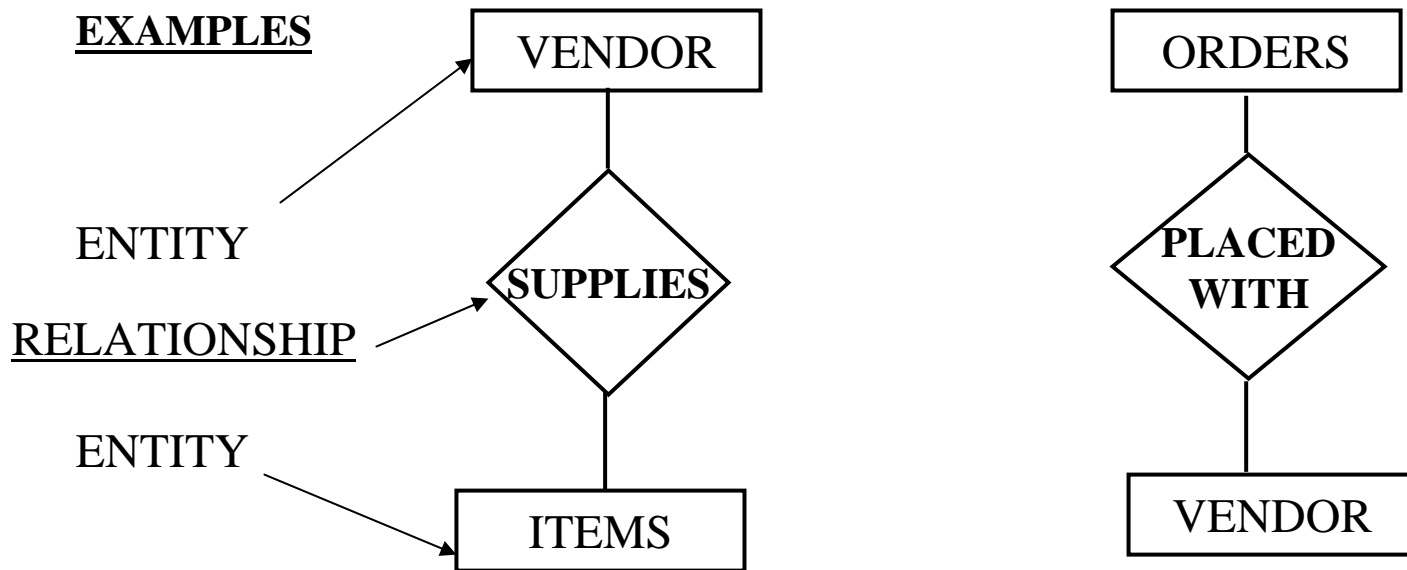
Attributes : **Teacher code**, **Course no**, course name, semester offered, credits, prerequisites

ENTITY-RELATIONSHIP DIAGRAM

- Some entities depend on one another

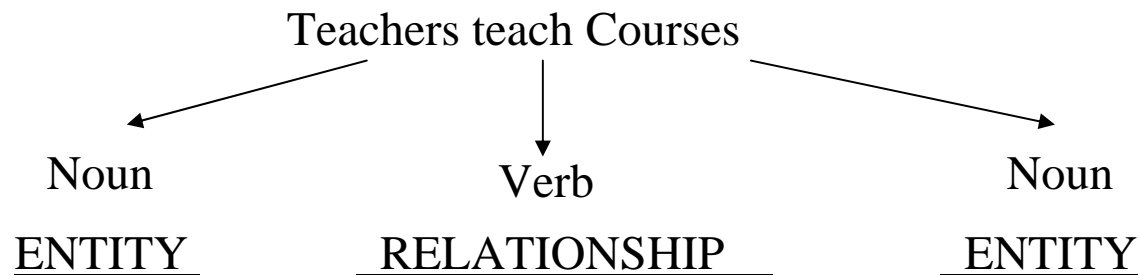
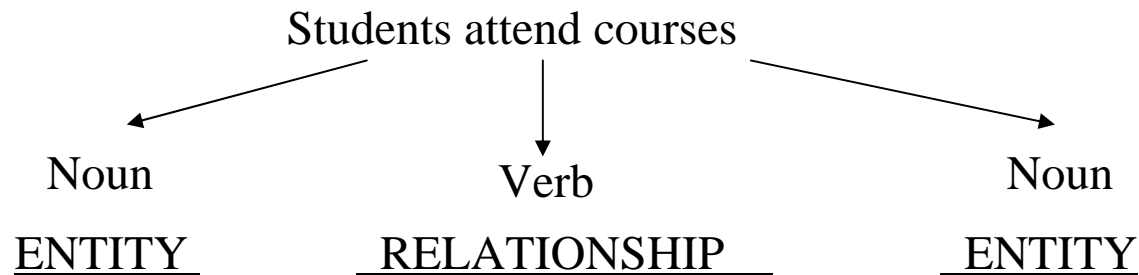
For example: entity vendor and entity items are related as vendors supply items

- These relationships are described by entity-relationship diagrams (or ER diagrams)
- In an ER diagram entities are represented by rectangles and relationships by diamond shaped boxes



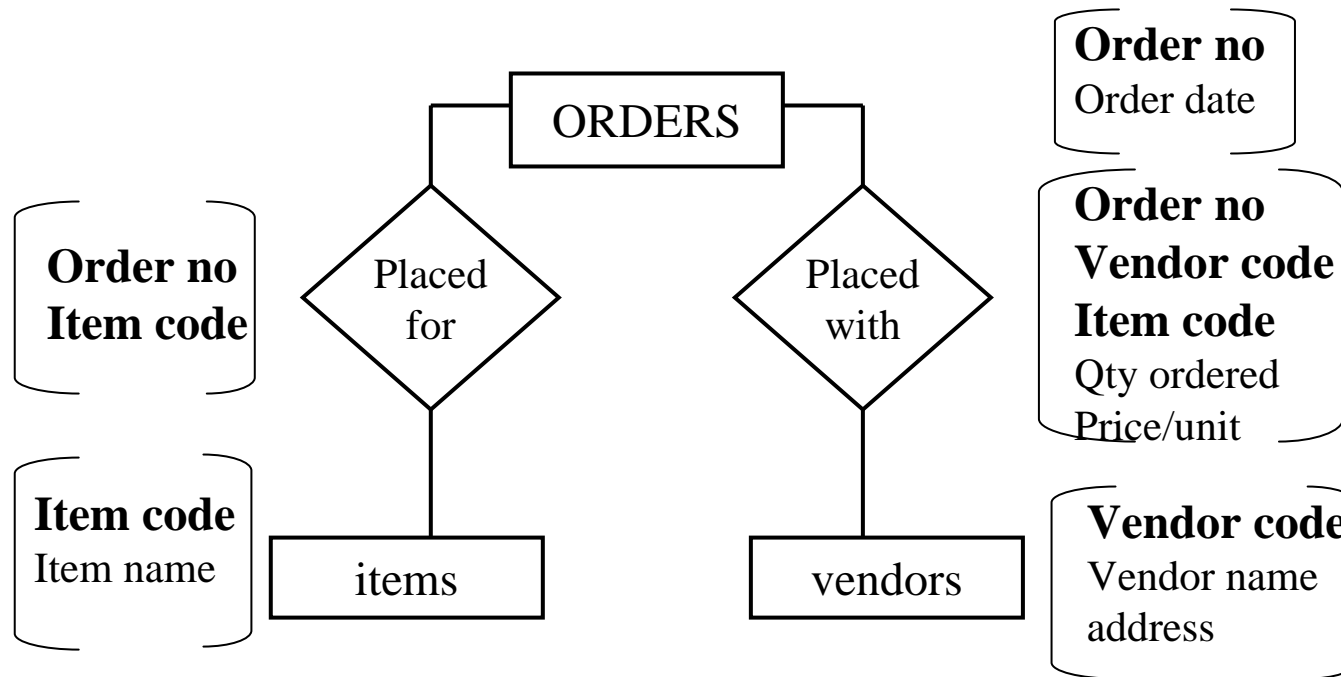
HOW TO IDENTIFY ENTITIES AND RELATIONSHIPS

- No algorithms to identify entities and relationship
- When a word statement is used to describe an applications nouns normally are entities and verbs relationships



ENTITY-RELATIONSHIP DIAGRAMS

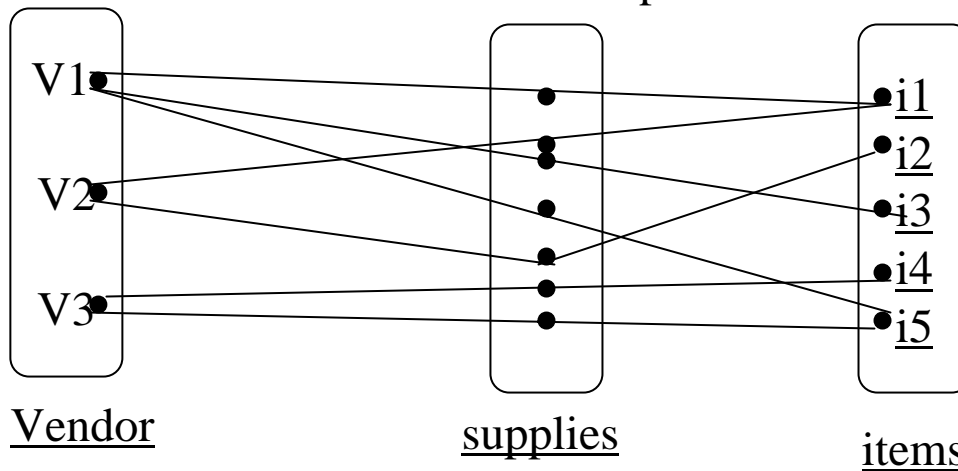
- ONE ENTITY MAY BE RELATED TO MANY OTHER ENTITIES BY MULTIPLE RELATIONSHIPS



- Underlined attributes are unique identifiers

RELATION CARDINALITY

- Relation cardinality - number of relationships in which an entity can appear
- An entity may appear in
 - Only one relationship or
 - In fixed number of relationships or
 - In a variable number of relationships



Vendor1 supplies
items i1,i3,i5

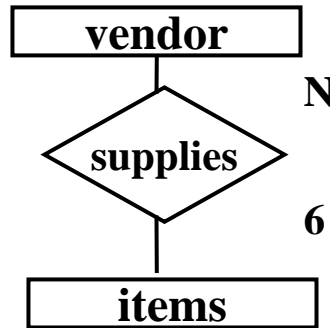
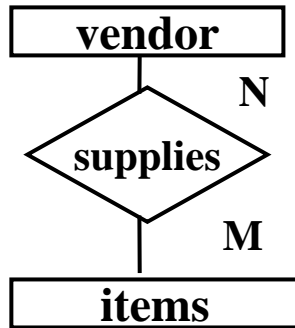
Vendor2 supplies
items i1 and i2

Vendor3 supplies
items i4 and i5

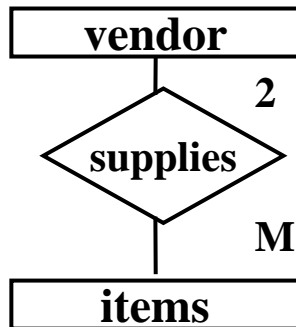
Observe a vendor can supply many items

Observe also that many vendors can supply the same item

RELATION CARDINALITY REPRESENTATION

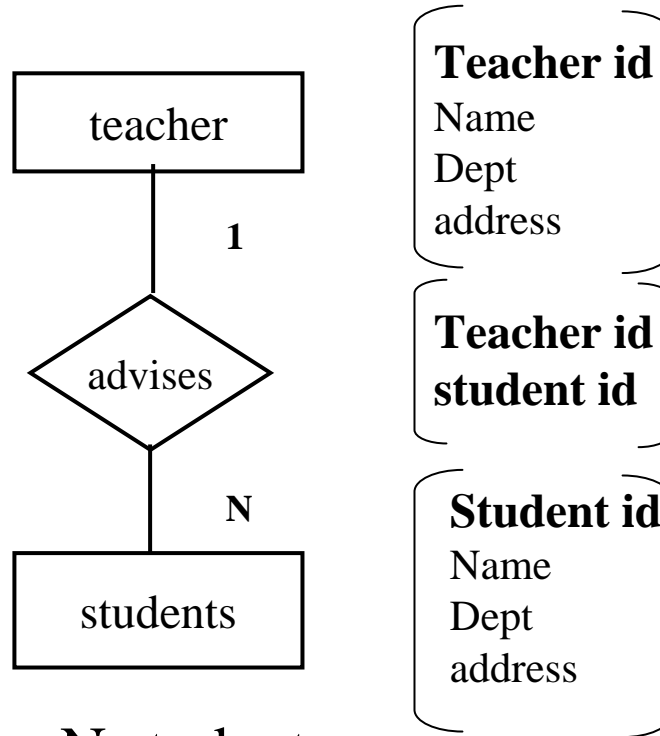


A vendor cannot supply more than 6 items
N vendors can supply a given item



An item cannot be supplied by more than 2 vendors

EXAMPLES



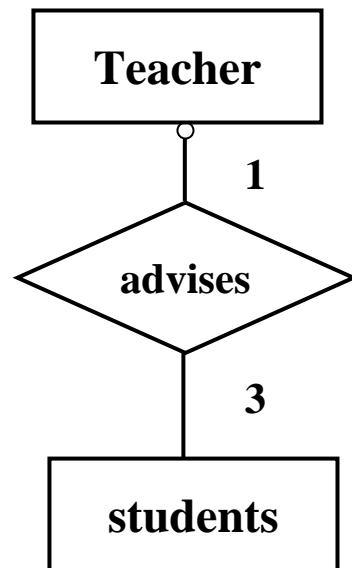
1 Teacher advises N students

- Observe that in the advises relationship the identifier need not be composite
- If it is N : M relationship then the relationship will have the identifier of both participating entities sets

EXAMPLES (CONTD)

- Not all teachers may be required to advise
- A teacher can advise not more than 3 students

Represented by ER diagram , small open circle specifies that some teachers may not participate in advises relationship



WHY IS FINDING CARDINALITY NECESSARY

- The identifier of the relationship will be composite if cardinality is N:M
- It will be single if cardinality is 1:M
- If an entity has ○ attached to it ,not all entities in the set may be present in the relationship
- Will be useful later in designing data base

RELATIONS

- Entity sets and relationship sets are useful in designing data bases
- Entity - relationship sets may be put as a table of values. This is called a relation
- Relation name is entity name
- A row of a relation has a member of an entity or a relationship set

EXAMPLES OF A RELATION

<u>VENDOR CODE</u>	<u>VENDOR NAME</u>	<u>ADDRESS</u>
1456	Ram & co	112, 1 st cross Bangalore-12
1685	Gopal & sons	452,4 th main, Delhi-8
1284	Sivaraj brother	368 M.G Road, Pune-8
1694	Gita ltd	495 N.S.C Road,Calicut

RELATION name:Vendor(same name as entity name)

RELATION ATTRIBUTES: vendor code, vendor name address

Row of relation is called a tuple

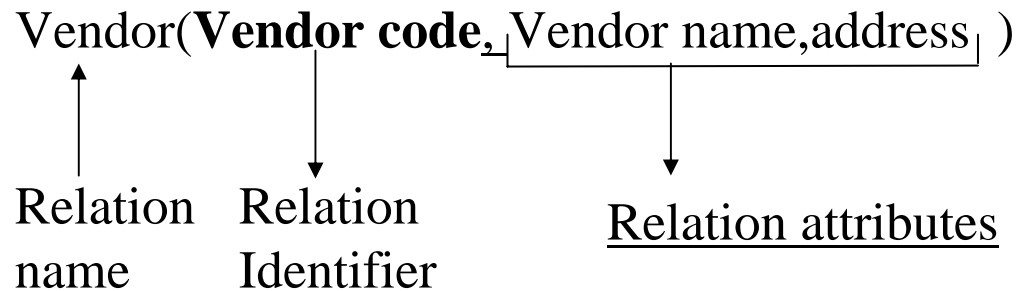
In a RELATION rows can be in any order
columns can be of any order

No two rows are identical

No two columns are identical

RELATION NOTATION

- Relation is an entire table
- However a concise notation used
- Notation uses: relation name and attributes
- Vendor relation:



EXAMPLES OF RELATIONS

Item (**Item code**, Item name)

Supplies (**vendor code**, **Item code** ,order no ,
qty supplied ,date of supply ,price/unit)

Relationship



Teacher (**Teacher_id** ,name ,dept ,address)

Advises (**Teacher_id** ,**student_id**)

Student (**Student_id** ,name ,dept ,address)

Bold faced attributes are key attributes

WHY RELATION ?

- Ease of storage of entity set as flat file in a computer's storage
- Sound theory of relations allows systematic design of relational data base
- Theory of normalizing relations
- Reduces duplication of data
- Tries to eliminate errors in adding, deleting, altering items in a data base
- Simplifies retrieval of data

NORMALIZING RELATIONS

- What is normalization of relations ?
- Why normalize relations ?
- How are relations normalized ?
- Normalizing is the process of restructuring relations to a form which:-
 - * Minimizes duplication of data in a database
 - * Operations of adding, deleting, modifying data in a database do not lead to inconsistent data in a database
 - * Retrieval of data simplified

WHY NORMALIZE ?

- A collection of relations relevant for an application constitute a relational database
- Relations are normalized to ensure that:
 - Collection of relations do not unnecessarily hold duplicate data
 - When a data item is modified it is modified in all relations where it appears - no inconsistency is there
 - When data is deleted accidentally, required data is not deleted
 - Simplifies retrieval of required data

HOW ARE RELATIONS NORMALIZED ?

UNNORMALIZED RELATION

Order no	order date	Item lines		
		<u>Item code</u>	<u>Qty</u>	<u>Price/unit</u>
1456	26021999	3687	52	50.40
		4627	38	60.20
		3214	20	17.50
1886	04031999	4629	45	20.25
		4627	30	60.20
1788	04111999	4627	40	60.20

1. Observe order for many items
2. Item lines has many attributes-called composite attributes
3. Each tuple has variable length
4. Difficult to store due to non-uniformity
5. Given item code difficult to find qty-ordered
6. Called Unnormalized relation

FIRST NORMAL FORM

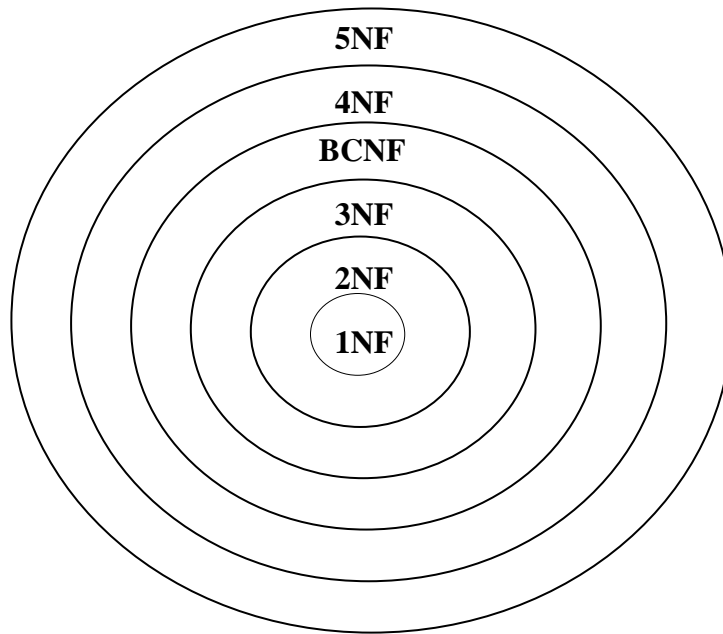
- Identify composite attributes
- Convert composite attributes to individual attributes
- Duplicate common attributes as many times as lines in composite attribute
- Every attribute describes single property -no composite attribute
- Some data duplicated
- This is called First normal form (1NF) also called flat file

FIRST NORMAL FORM – 1NF

<u>Order No</u>	<u>Order date</u>	<u>Item code</u>	<u>Qty</u>	<u>Price/unit</u>
1456	26021999	3687	52	50.40
1456	26021999	4627	38	60.20
1456	26021999	3214	20	17.50
1886	04031999	4629	45	20.25
1886	04031999	4627	30	60.20
1788	04111999	4627	40	60.20

HIGHER NORMAL FORMS

- First normal form is first essential step in normalization
- Higher normal forms known as 2NF,3NF,BCNF,4NF,5NF exist
- Each is an improvement of the preceding one
- A higher normal form also satisfies requirements of a lower normal form



HIGHER NORMAL FORMS

Higher normalization steps based on :

- Detecting dependence between attributes
- Identifying key attributes
- Detecting multivalued dependency between attributes

FUNCTIONAL DEPENDENCY

- Given X, Y as two attributes in a relation
- Given X if only one value of Y corresponds to it then Y is functionally dependent on X

$$X \longrightarrow Y$$

e.g. Given Item code - Item name known

Therefore Item code \longrightarrow Item name

- Functional dependence may be based on a composite attribute

$$\boxed{X, Z} \longrightarrow Y$$

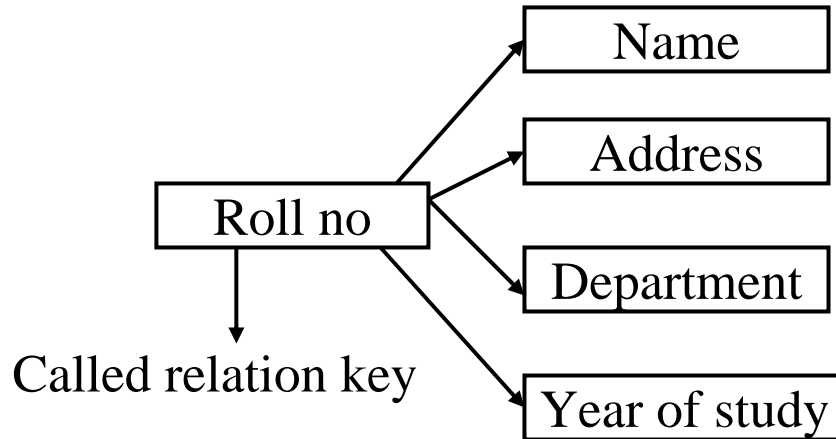
↓
composite attribute

$$\boxed{\text{Order no. , item code}} \longrightarrow \text{Qty , price}$$

↓
composite attribute

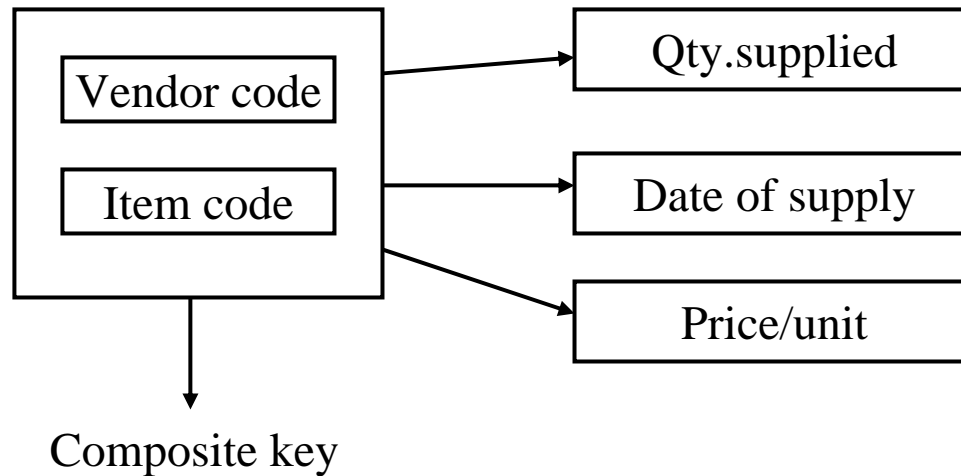
DEPENDENCY DIAGRAM

Student (Roll no, name, address, dept., year of study)



- Roll no. determines uniquely values of all other attributes in the relation
- Therefore it is called a key

DEPENDENCY DIAGRAM



Composite key enclosed in one box in diagram

WHY NORMALIZE RELATIONS-REVISITED

- To ensure that while operating on data base we do not
 - Lose data
 - Introduce inconsistencies
- Operations on data base
 - Insertion of new data should not force leaving blank fields for some attributes
 - Deletion of a tuple should not delete vital information
 - Updating - changing value of an attribute should be possible without exhaustively searching all tuples of relation

EXAMPLE TO SHOW NEED FOR NORMALIZATION

FIRST NORMAL FORM – 1NF

<u>Order No</u>	<u>Order date</u>	<u>Item code</u>	<u>Qty</u>	<u>Price/unit</u>
1456	26021999	3687	52	50.40
1456	26021999	4627	38	60.20
1456	26021999	3214	20	17.50
1886	04031999	4629	45	20.25
1886	04031999	4627	30	60.20
1788	04111999	4627	40	60.20

INSERTION : Enter new item with code 3945 and price 30.50 for which no order has been placed. Inserted tuple will have no values(i.e have to be left blank) for order no and order date

DELETION: If order no1886 is deleted the fact that item code 4629 costs 20.25 is lost

UPDATE: If price of item 4627 is changed,all instances of this item code have to be changed by exhaustive search-errors possible

IDEAL NORMALIZATION

At the end of normalization a normalized relation

- Should have no data values duplicated in rows
- Every attribute in a row must have a value
- Deletion of a row must not lead to accidental loss of information
- Adding a row should not affect other rows
- A value of an attribute in a row can be changed independent of other rows

SECOND NORMAL FORM (2NF)

A relation is in 2NF if

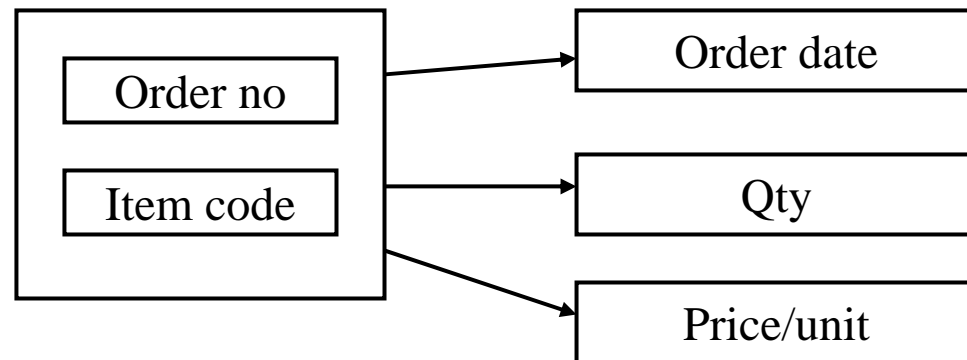
- It is in 1NF
- Non key attributes functionally dependent on key attribute
- If key composite then no non-key attribute can functionally depend on one part of the key.

2NF FORM NORMALIZATION-EXAMPLE

INF : itemorder (**Order no,Item code**, Order date,Qty,Price/unit)

Composite key

Dependency diagram



- Not in 2NF as non key attribute dependent on part of a key attribute(Price/unit dependent on Item code)
- 2NF relations are Order(**order no**,order date)
 Prices(**item code**,price/unit)
 Order details(**order no,item code**,qty)
- Observe a single 1NF relation split into three 2NF relations

2NF FORM

1 NF Orders Relation

Order No	Order date	Item code	Qty	Price/unit
1456	26021999	3687	52	50.40
1456	26021999	4627	38	60.20
1456	26021999	3214	20	17.50
1886	04031999	4629	45	20.25
1886	04031999	4627	30	60.20
1788	04041999	4627	40	60.20

2 NF Relations

<u>ORDERS</u>		<u>ORDER DETAILS</u>			<u>PRICES</u>	
<u>Order No</u>	<u>Order date</u>	<u>Order No</u>	<u>Item code</u>	<u>Qty</u>	<u>Item code</u>	<u>Price/unit</u>
1456	26021999	1456	3687	52	3687	50.40
1886	04031999	1456	4627	38	4627	60.20
1788	04041999	1456	3214	20	3214	17.50
		1886	4629	45	4629	20.25
		1886	4627	30		
		1886	4627	40		

ADVANTAGES OF 2NF

* NON KEY ATTRIBUTES WHOLLY DEPENDENT ON KEY

- Repetition of order date removed
- If order 1886 for item 4629 is cancelled the price/unit is lost in 1NF as the whole tuple would be deleted
- In 2NF item price not lost when order 1886 for item 4629 cancelled. Only row 4 in order details deleted
- Duplication of data in a relation is not there

THIRD NORMAL FORM

- Relation in 2NF
- There is functional dependence between some Non-key attributes

This needs further normalization as the non-keys being dependent leads to unnecessary duplication of data

EXAMPLE

Student(Roll no, name, dept, year, hostelname)

- If students in a given year are all put in one hostel then year and the hostel are functionally dependent
- Year implies hostel-hostel name unnecessarily duplicated
- If all students of the year 1 changed to another hostel many tuples need change

NORMALIZATION TO 3NF

Student(**Roll no**, name, dept, year)

Hostel (**year**, hostel)

This is in 3NF

Example : Employee (**empcode**,name,salary,project no,termination date of project)

* **termination date (non-key attribute)**

Dependent on project no. another non-key attribute

•**Thus needs normalization**

3NF relations : Employee(**empcode**,name,salary,projectno)

Project(**Projectno_**,termination date of project)

NORMALIZATION TO 3NF

Passenger(**Ticket code**, Passenger name, Train no, Departure time, Fare)

Train no. and departure time are non-key attributes and are functionally dependent

3NF Relations :

Passenger(**Ticket code** , Passenger name, Train no, Fare)

Train details (**Train no.**, departure time)

BOYCE-CODD NORMAL FORM

Assume

- * Relation has more than 1 possible key
- * Keys composite
- * Composite keys have common attribute
- * Non-key attributes not dependent on one another

Thus relation in 3NF, still there could be problems due to unnecessary duplication and loss of data accidentally

EXAMPLE

EXAMPLE

Professor (**Prof code**, **Dept**, Head of Dept, Percent time)

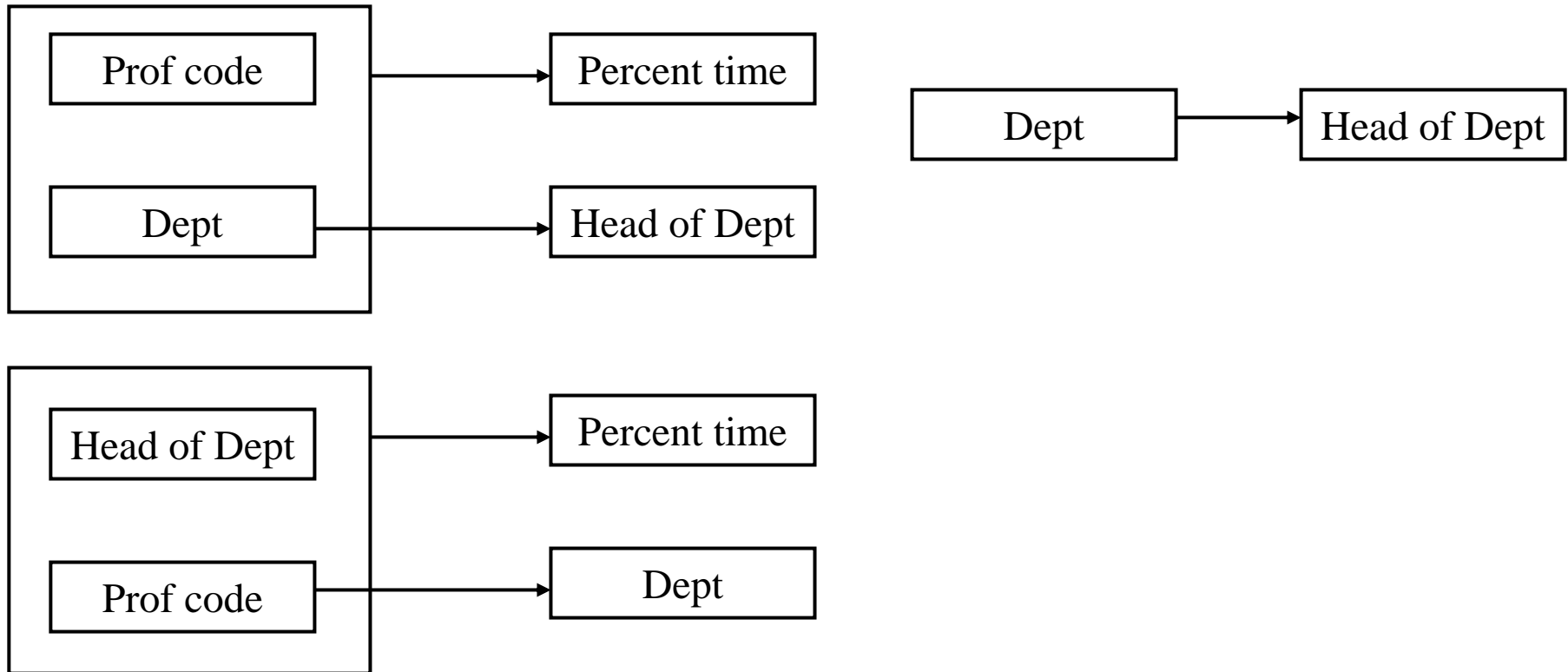
RELATION

Professor code	Dept	Head of dept	Percent time
P1	Physics	Ghosh	50
P1	Maths	Krishnan	50
P2	Chem	Rao	25
P2	Physics	Ghosh	75
P3	Maths	Krishnan	100
P4	Maths	Krishnan	30
P4	Physics	Ghosh	70

- Observe two possible composite keys (**Prof code**, **Dept**) or (**Prof code**, **Head of Dept**)
- Observe Head of dept name is repeated
- If professor P2 resigns the fact that Rao is Head of Chemistry is lost as lines 3 & 4 will be deleted

EXAMPLE (CONTD)

The dependency diagrams are:



- Percent time a Prof.spends in the department is dependent on Prof code and Department
- Head of Dept depends on department

NEED FOR BCNF

- Observe the given relation is in 3NF as non key attributes are independent of one another and wholly dependent on key
- However there are problems pointed out
- Problem due to the fact that there are two possible composite keys
- An attribute of one of the composite key depends on a attribute of the other possible composite key
- This leads to the problem indicated

NORMALIZING TO BCNF

- Identify the dependent attributes in the possible composite keys
- Remove them and create anew relation

EXAMPLE

Composite keys

1. **Prof code ,Dept**
2. **Prof code,Head of Dept**

Dependency : Dept \longrightarrow Head of dept

New relations

Professor (**Prof code, Dept**, Percent time)

Department (**Dept**, Head of Dept)

NORMALIZED BCNF RELATIONS

Professor code	Dept	Percent time
P1	Physics	50
P1	Maths	50
P2	Chem	25
P2	Physics	75
P3	Maths	100
P4	Maths	30
P4	Physics	70

Dept	Head of Dept
Physics	Ghosh
Maths	Krishnan
Chem	Rao

- Observe there is no redundancy
- If P2 resigns information of Head of dept of chemistry is not lost

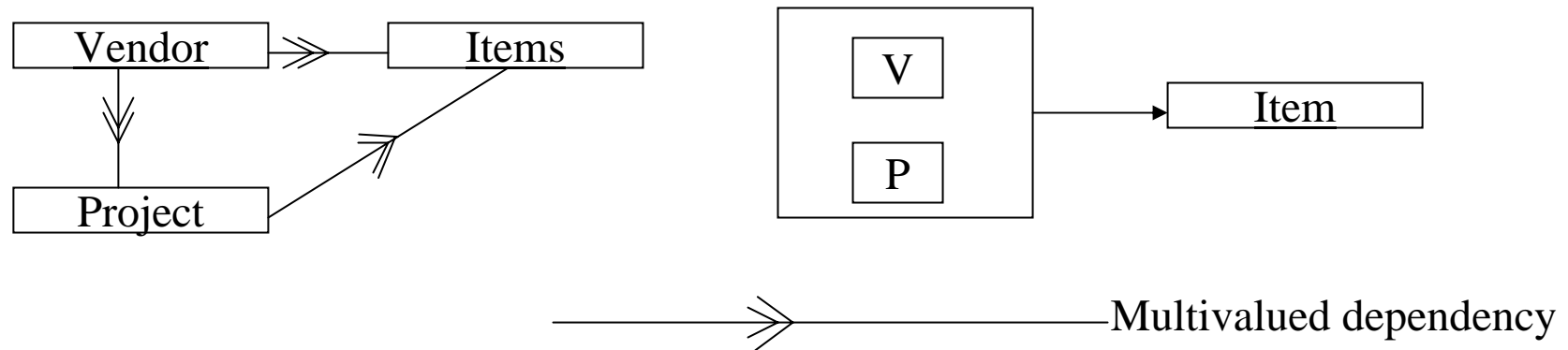
FOURTH NORMAL FORM

- Needed when there are multi-valued dependencies

Example : (Vendor, Project, Item) relations

Assumptions :

- A vendor capable of supplying many items to many projects
- A project needs many items
- Project may order the same item from many vendors



Vendor-Project-Item supply capability relation

FOURTH NORMAL FORM (CONTD)

<u>Vendor code</u>	<u>Project code</u>	<u>Item code</u>
VI	PI	I1
VI	PI	I2
VI	P3	I1
VI	P3	I2
V2	PI	I2
V2	PI	I3
V3	PI	I1
V3	P2	I1

Problems

- Item I1 duplicated for VI and also for V3
- If VI is to supply to project P2 but the item to be supplied is not decided there will be blank in item column

Relation in BCNF but still has above problem and need normalization to 4NF

NORMALIZATION TO 4NF

- Split vendor-project-item relation into two relations
- Resulting relation must have not more than one independent multivalued dependency

RESULTING RELATIONS

<u>Vendor</u>	<u>Item</u>
VI	11
VI	12
V2	12
V2	13
V3	11

<u>Project</u>	<u>Item</u>
P1	I1
P1	I2
P1	I3
P2	I1
P3	I1
P3	I2

OBSERVE NO UNNECESSARY DUPLICATION

NEED FOR 5NF

- In 4NF relations vendor capability to supply items and projects need for items are there.
- They are obtained by splitting the given relation
- Looking at relation project-item we see that project P2 requires item I1
- From vendor item relation we see that I1 is supplied by V1.
- This would lead us to infer that (V1,P1,I1) must be a tuple in the original relation but it is not. In other words V1 does not supply item I1 to project P2.
- This spurious tuple has occurred because vendor V1 may not be allowed to supply item I1 to project P2
- Similarly another spurious tuple is (V3,P3,I1)
- We thus need a third relation which specifies the vendors who are allowed to supply to projects

OBTAINING 5NF RELATIONS

- We add a third relation to the two 4NF relations
- This is vendor-project relation shown below

VENDOR CODE	PROJECT NO
V1	P1
V1	P3
V2	P1
V3	P1
V3	P2

- With this relation added we will not get the spurious tuples (V1,P2,I1),(V3,P3,I1)
- The two 4NF relations together with the vendor-project relation called 5NF relations obtained by decomposing the original vendor-project-item relation which has a BCNF relation

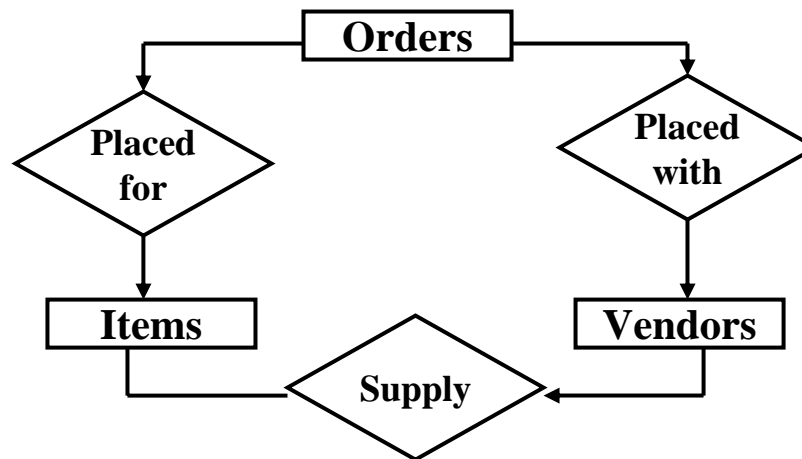
EXAMPLES OF DATA BASE DESIGN

ORDER - VENDOR - ITEMS ORDERED EXAMPLE IN CASE STUDY

Information on dependencies given :

- Orders for item placed with many vendors
- A given order no is only to one vendor
- Many items supplied against a given order no
- A vendor has capacity to supply many items but only some items may be ordered

ER - DIAGRAM



RELATIONS-UNNORMALIZED

RELATIONS-UNNORMALIZED

1. ORDERS(**Order no**, Order date)
2. ORDERS PLACED FOR(**Order no**, **item code**, qty ordered, delivery time allowed)
3. ORDERS PLACED WITH(**order no**, **vendor code**, **item code**)
4. VENDOR(**Vendor code**, vendor name, vendor address)
5. ITEM(**item code**, item name, price/unit)
6. SUPPLIES(**vendor code**, **item code**, **order no**, qty. supplied, date of supply)

NORMALIZATION:

Relation 1,4,5 are in 3NF and need no change

Relation 2 has a composite key, attributes of composite key not related.

Non key attributes dependent on composite key, need no change.

Relation 3: order no and item code have multivalued dependency. Relation 2 already has **order no**, **item code** as composite key. Relation 3 is reduced to:

7. ORDER PLACED WITH(**order no**, **vendor code**)

NORMALIZATION OF SUPPLIES RELATION

Consider relation 6 :

6. SUPPLIES (**vendor code, item code, order no**, qty supplied, date of supply)
- It has a composite key with three attributes
 - Attributes **item code** and **order no** have multi-valued dependency as many items can be supplied in one order
 - Need normalization to 4NF

Normalized to

8. ACTUAL SUPPLIES (**order no, item code**, qty supplied, date of supply)
9. VENDOR CAPABILITY (**vendor code, item code**)

The second relation may have items not yet ordered with a vendor but which could be supplied by vendor

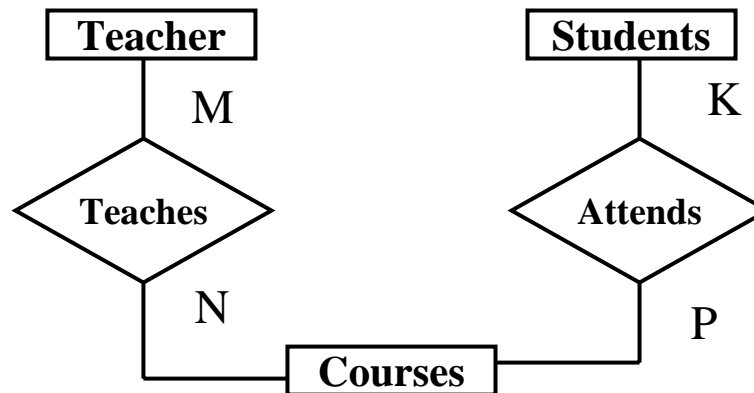
The Normalized relations are : 1,2,4,5,7,8,9

STUDENT-TEACHER-COURSES EXAMPLE

Information on dependence

- A teacher may teach more than one course in a semester
- A teacher belongs to only one dept.
- A student may take many courses in a semester
- A course may have many sections taught by different teachers

E-R Diagram



RELATION-UNNORMALIZED

- 1 TEACHER (**Teacher code**, teacher name, address, rank, dept)
- 2 TEACHER_COURSES (**Teacher code**, **Course no**, no of students, section no)
- 3 COURSE (**Course no** , **semester taught** , Course name, credits)
- 4 STUDENT (**Student no**, student name, dept, year)
- 5 STUDENT COURSES (**Student no**, **Course no**, semester no)

- a) Relations 1,3,4 in 3NF
- b) Relations 2 and 5 have multi-attribute key which has multi-valued dependency but do not need normalization
- c) However information on which teacher teaches a given student a specified course cannot be found from relations 1 to 5

Therefore Add relation

- 6 TEACHER_STUDENT (**Teacher code**, **Student no**, **Course no**)
- THIS SET NORMALIZED

CONCLUSIONS

- We have seen how data relevant to applications are organized logically into set of relations
- The process of normalization depends on the semantics, i.e, meanings of data and an understanding of how various data elements are related
- It is thus a human intensive activity-it cannot be automated

CONCLUSIONS (CONTD)

- In most problems in practice one is satisfied with 3NF. Higher normal forms are theoretically important and in some cases becomes essential.
- There is a mathematical theory which underpins the idea of relations and normalization giving it a sound basis. We have not discussed it in this module.
- A full fledged course in Data Base will describe in detail the mathematical basis and methods of querying a database

PROBLEMS WITH FILE BASED SYSTEMS

If programs and files independently developed for each application in the organization it leads to the following problems

- DATA REDUNDANCY-Some data may be duplicated in many files.
e.g.: Address of customer
- LACK OF DATA INTEGRITY- Duplicated data may be different in different files (New address in one file and old address in another file)
- DATA AVAILABILITY- May require search of number of files to access a specified data
- CONTROL BY MANAGEMENT-Difficult as data scattered across files.
All files should be accessed to find specified data

Aim of data base management systems is to reduce above problems

DATABASE AND DATABASE MANAGEMENT SYSTEM

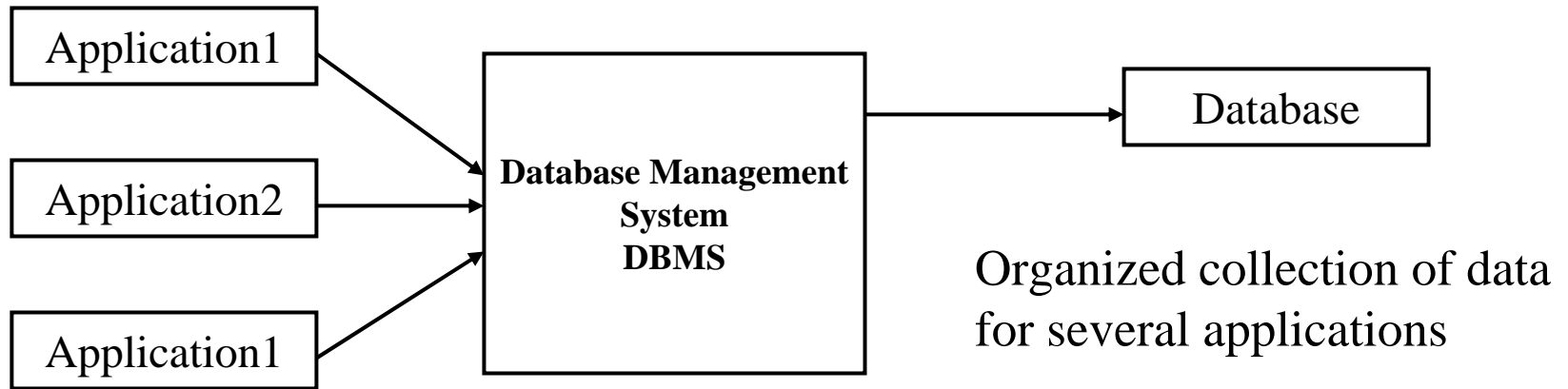
DATA BASE - A Collection of related data necessary to manage an organization (Excludes transient data)

- Models data resource and is independent of applications

DATA BASE MANAGEMENT-

- A set of procedures that manage the database and provides access to the database in the form required by the application program

DATABASE MANAGEMENT SYSTEM



Procedures to provide data in the form required by applications. Applications need not know physical organization of data

OBJECTIVES OF A DATABASE MANAGEMENT SYSTEM

- Data is an organizational resource. It should be collected, validated, protected, logically organized and stored with controlled redundancy. This is the organizational database
- One of the main objectives of DBMS is to facilitate sharing of a database by current and future applications
- DBMS must ensure data independence for programs

OBJECTIVES OF DBMS

- Data independence to be provided to application programs
- Data independence allows
 - Change of database without affecting application programs
 - Change of hardware or system software without affecting application programs
 - Sharing of data by different applications by providing views appropriate for the application

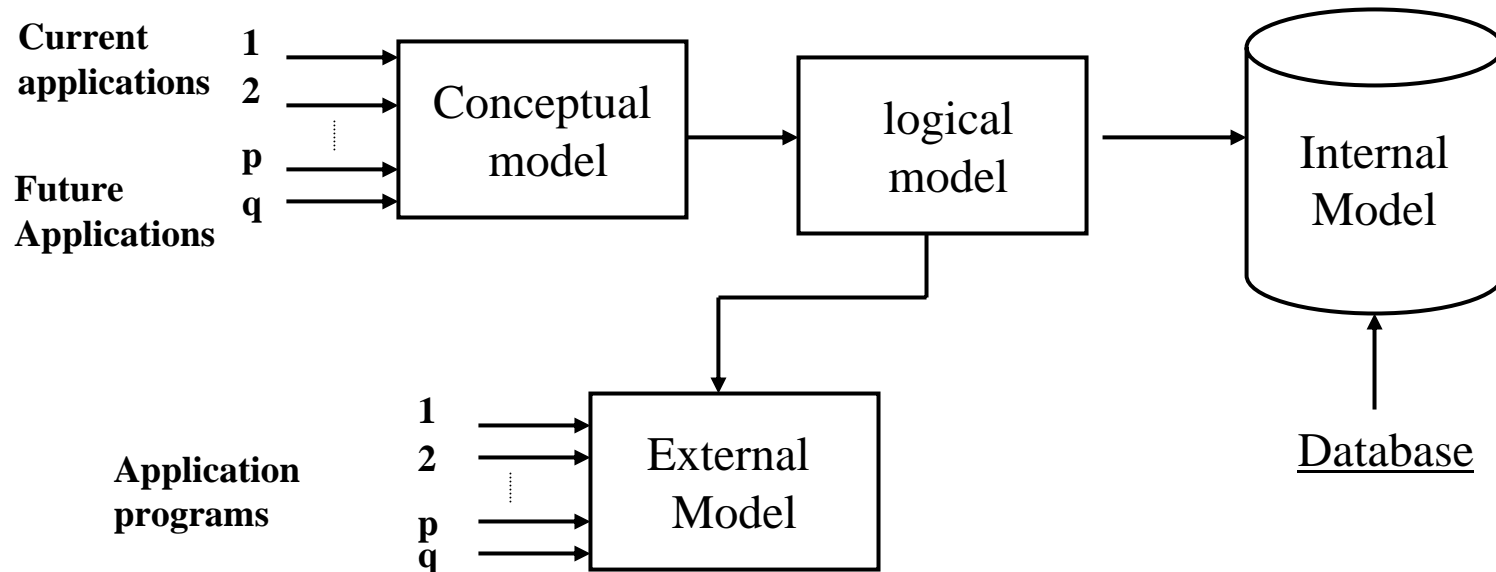
OBJECTIVES OF DBMS

- Control of Redundancy - Avoid unnecessary duplication
- Relations between data items specified
- Data integrity - Preserve consistency of data values
- Data Security - Permit access to data to authorized users only
- Performance - Provide timely information when needed
- Ensure management control on addition, deletion, change and dissemination of data

OVERVIEW OF DBMS

- Data needs of current and possible future applications determined
- Using E-R data modelling conceptual data model found
- Converted to relations if Relational DBMS used
 - called logical data model
- Application programmers access subset of logical data model
 - called external data model
- Logical model mapped to physical model for storage in disk store
 - called internal model
- External data model kept invariant

VARIOUS TERMS USED IN DESCRIBING DBMS



COMPONENTS OF DBMS

- Data Definition Languages (DDL) to define conceptual, logical and external models
- Data manipulation and query language called Structured Query Language (SQL)
- Features to implement security
- Checkpointing and roll back recovery procedures
- Record of accesses. Audit trail of changes

DATABASE ADMINISTRATOR

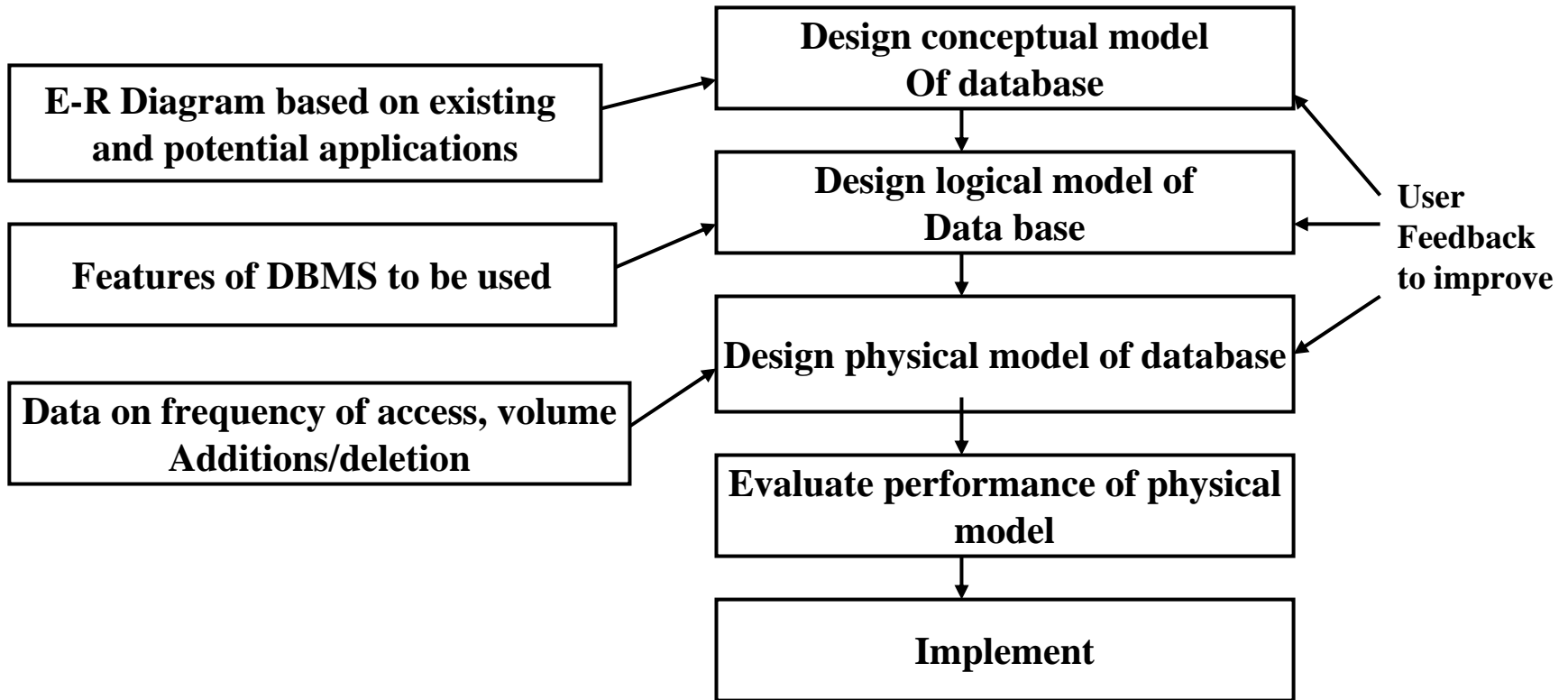
Database Administrator's responsibilities

- Controller of data recourse
- Ensure integrity, security and privacy
- Maintenance of data dictionary
- Coordination of development and maintenance of data base
- Determining access rights

STEPS IN DESIGNING DATABASE

INPUTS

DESIGN STEPS



CHARACTERISTICS OF A GOOD DATA BASE DESIGN

- Satisfy current and future needs of organization
- Cater to unanticipated user requirements in the best possible way
- Expandable with growth and changes in organization
- Easy to change when hardware and software change
- Ensure data security by allowing only authorized persons to access and modify database