CS403 Quiz 1 Mcqs

Lecture 1 to 10

Spring 2022 #Mid-Term

WITH PROVE ANSWER

ORANGE MONKEY TEAM

Important NOTE:-

Maybe toppers are successful in school or college carrier but backbenchers are successful in their life because toppers run according to books while backbenchers don't follow rules and regulation that is why backbenchers are successful. For life, there are no books and another reason why toppers can't succeed because they fear failure and don't try anything new. Backbenchers don't fear failure as they have faced the failure in their school or college carrier so that is the reason they try new things and at last gain success.

As Abdul Kalam nicely quoted, "The best brains of the nation may be found on the last benches of the classroom"

Translation into Urdu:

ہوسکتا ہے کہ ٹاپر اسکول یا کالج کیریئر میں کامیاب ہوں لیکن بیک بینچر اپنی زندگی میں کامیاب ہوتے ہیں کیونکہ ٹاپر کتابوں کے مطابق چلتے ہیں جبکہ بیک بینچر قواعد و ضوابط پر عمل نہیں کرتے ہیں اسی وجہ سے بیک بینچر کامیاب ہوتے ہیں۔ زندگی کے لیے، کوئی کتابیں نہیں ہیں اور ایک اور وجہ ہے کہ ٹاپرز کامیاب نہیں ہو پاتے کیونکہ وہ ناکامی سے ڈرتے ہیں اور کچھ نیا کرنے کی کوشش نہیں کرتے۔ بیک بینچر ناکامی سے نہیں ڈرتے کیونکہ انہوں نے اپنے اسکول یا کالج کیریئر میں ناکامی کا سامنا کیا ہے یہی وجہ ہے کہ وہ نئی چیزیں آزماتے ہیں اور آخر کار کامیابی حاصل کرتے ہیں۔

جیسا کہ عبدالکلام نے اچھی طرح سے نقل کیا ہے، "قوم کے بہترین دماغ کلاس روم کے آخری بینچوں پر مل سکتے ہیں"۔

1. **In cross reference matrix attributes are specified on _____.**

Y axis….confirm

55. In cross reference matrix transitions are specified on __ on _____.
    a. Y axis, X axis
    b. X axis, Y axis
    c. X axis, X axis
    d. Y axis, Y axis

2. **Which model operates at the lowest level of abstraction?**

Physical…..confirm

Which model operates at the lowest level of abstractio

Select correct option:

conceptual

internal

external

physical

3. **Entities enrolled in a relationship are called its _____.**

Participants…confirm

4. **A _____ is an abstraction of a relationship**

Relationship type….confirm

A ***relationship type*** is an abstraction of a relationship

5. **In which phase of database development process specific database management system (DBMS) is chosen?**

Logical data base design…..confirm

Logical Database Design

This is the next phase of database design, in which appropriate data model is chosen,

6. **DBMS select a specific file organization for the storage of data on disk, to implement that specific file system the DBMS needs to create specific_____.**

Indexes…….confirm

storage of data on disk, to implement that specific file system the DBMS needs to create specific indexes. Now whenever the DBMS will

7. **Data inconsistency leads to a number of problems such as_____.**

Loss of information and incorrect result……confirm

specific data item. Data inconsistency leads to a number of problems, including loss of information and incorrect results. In database approach it is controlled because data is

8. **Which of the following change can cause problem in database levels?**

Deleting an attribute from the database structure……confirm

inconsistency in the database levels. The changes which can be done transparently may include the following:

- Adding a file to the database
- Adding a new field in a file
- Changing the type of a specific field

9. **In data based development process which of the following is related to requirement gathering phase.**

Detailed study of system….confirm

flows between those sections. Detailed study of the system is started to find out the requirements of each section. This phase is the detailed study of the system and its functionality decisions made at this stage decide the overall activity of the organization. Requirements of one section of the organization are fulfilled in such a way that all the sections in the organization are supporting each other, for example we can say that the results produced by the processing taking place at one section are used as input for another section. All the users of the systems are interviewed and observed to pinpoint and precisely define the activities taking place in the different section of the organization.

Database Development Process

10. Which of the following is correct about database modeling?

Process of creating the logical structure of the database……confirm

The process of creating the logical structure of the database is called database modeling.

11. A relationship is an association between two entities or between an entity and itself. The diamond symbol for a relationship is _____ if one of the participants is dependent on the other.

Doubled…..confirm

Diamond is doubled if one of the participant is dependent on the other

12. Which of the following is incorrect related to secondary key?

Selected key must be unique identifier…..confirm

13. In database system the description and definition of data is stored in _____.

Schema….confirm

A database generally stores its schema in a data dictionary.
Although a schema is defined in text database language, the

14. A candidate key that does not have a null value and is selected to uniquely identify all other attribute values in any given row is called a _____.

Primary key….confirm

A certain value that may be associated with any attribute is NULL, that means "not given" or "not defined". A major characteristic of the PK is that it cannot have the NULL value. If PK is a composite, then none of the attributes included in the PK can have the NULL, for example, if we are using "name, fName" as PK of entity type STUDENT,

15. Which of the following is Not related to the management of data in the database?

How data will be transformed…..confirm

- Management of Data in the Database
- Management of Users associated with the database.

Management of the data means to specify that how data will be stored, structured and accessed in the database.

16. In entity-relationship diagram the entity type which has existence dependency constant is classified as:

Weak entity…..confirm

Que.	In entity-relationship, the entity type which has existence dependency constraint is classified as
a.	single entity
b.	foreign entity
c.	weak entity
d.	strong entity
	Answer: weak entity

17. Which of the following to the semantic Data Model?

E-R data model…..confirm

Developed in a semantic data model (generally E-R data model)

18. Detailed data flow diagram is created for only those processes from the _____ for which we want to show the details.

Level 0 diagram…..confirm

DFDs. The special features associated with this diagram are that, one, it is optional, that is, it is created for only those processes from the level 0 diagram for which we want to show the details. For a small sized system we may not need to develop even a single

19. The total participation by entities is represented in E-R diagram as:

Double line….confirm

20. Which of the following is the major drawback of file processing system?

Non-sharing of data

Another major drawback in the traditional file system environment is the non-sharing of data. It means if different systems of an organization are using some common data then rather than storing it once and sharing it, each system stores data in separate files. This

21. A relationship that involves three entities called.

Ternary relationship….confirm

- **Ternary Relationship**

  A **Ternary** relationship is the one that involves three entities e.g.

22. The entity relationship model consists of the following major constructs _____.

Entity, Attribute and relationship…..confirm

Constructs in E-R Data Model

The E-R data model supports following major constructs:

- Entity
- Attribute
- Relationship

23. The DBMS uses the _____ to access the database at each layer or model.

Data Dictionary…..confirm

information along with all the database design information files in the database st

The DBMS uses the data dictionary to access the database at each layer or model, f

24. Which of the following is the best description of the data security in DBMS?

Efficient Check on all users access…..confirm

- **Better Data Security:**

All application programs access data through DBMS, So DBMS can very efficiently check that which user is performing which action and accessing which part of data , So A DBMS is the most effectively control and maintain security of Data stored in a database.

25. Consider the following tables

EMP(empId, empName, qual, depId)

DEPT(depId, depName, numEmp)
The primary key of EMP table is _____ whereas foreign key of EMP table is _____.

EmpId, depId

26. Which is not correct according to Data Flow Diagram?

DFDs provide us a way expressing decision points….confirm

Data Flow Diagrams:
The most common tool used for deigning database systems is Data Flow Diagram. It is used to design systems graphically and expresses different system detail in different DFD levels.
DFDs show the flow of data between different processes o a specific system.
DFDs are simple, and hide complexities.
DFDs are Descriptive and links between processes describe the information flow.

27. Which of the following Greek letter is used to denote the select operation?

( **) Greek sigma symbol……confirm**

3. Which of the following is used to denote the selection operation in relational algebra?
a) Pi (Greek)
b) Sigma (Greek)
c) Lambda (Greek)
d) Omega (Greek)
View Answer

Answer: b

28. In relational data model, a record or an entity instance is represented as ____ in a relation.

Row…confirm

table consists of rows and columns. Rows of a table are also called tuples. A row or tuple of a table represents a record or an entity instance, where as the columns of the

29. While mapping many-to-many relationship _____ table(s) are created.

Three

30. Select and project are the examples of _____.

Unary Operation…..confirm

: "Select" and "project" are the examples of
Select correct option:

Unary operations
Binary operations

31. In natural join the common attribute in the output table appears _____.

Once…..confirm

In natural join the common attribute in the output table appears _____.

- Twice
- Three times
- Four times
- **Once**

32. RDM is simple there is just one structure and that is _____.

a Relation or a table….confirm

The RDM is simple, why, there is just one structure and that is a relation or a table.

33. RDM stands for _____.

Relational Data model….confirm

relational data model (RDM),

34. Which of the following constant states that if a foreign key exists in a relation either the foreign key value must match the primary key value of some tuple in its home relation or the foreign key value must be completely null?

Referential Integrity Constraint….confirm

Referential Integrity Constraint:
This constraint is applied to foreign keys. Foreign key is an attribute or attribute combination of a relation that is the primary key of another relation. This constraint states that if a foreign key exists in a relation, either the foreign key value must match the primary key value of some tuple in its home relation or the foreign key value must be completely null.

35. Which constraint is generally used for efficiency purpose in the data entry process.

Default value…..confirm

Default Value:
This constraint means that if we do not give any value to any particular attribute, it will be given a certain (default) value. This constraint is generally used for the efficiency purpose in the data entry process. Sometimes an attribute has a certain

36. The relationship between department to employees is .

One to many….confirm

In the second part of the Figure-1 we see a relationship between the Employee and project entities, the relationship describes one to many association between the project and the employees. It shows that there can be one project having a number of employees,

37. A row of a table represents a _____ whereas the column of the table represents the _____.

Record, Attribute….confirm

A row of a table represents a _____, whereas the column of the table represents the _____.

Answer: Record, attribute

38. Which is not the function of Database Management System?

Failure Response Service

39. The conceptual database design can be transformed into any .

Data model

40. Which of the following is the second stage in the database development process?

Requirement Analysis

41. Functional dependencies under interference rule reflexivity are called dependencies.

Trivial

42. If many instances of the "Book" entity are associated with the many instances of the "Student" entity, then the relationship between them is:

Many to many

43. Which of the following constraints enforces entity integrity?

PRIMARY KEY

44. Unary operations involves

Only one relation

45. A entity has a primary key that is partially or totally derived from the parent entity in the relationship.

Weak

46. Business

Relationship

47. In _____ appropriate data model is chosen.

Logical Database Design

48. Which of the following is drawback of standardization?

Lack of uniqueness

49. Recursive relationship is also called .

Unary

50. The foreign key attribute, which is present as a primary key in another relation is called as

of the foreign key attribute.

51. Binary relation

Home relation

52. DFD does not provide us a way to represent

Decisions

53. A _____ is used to maintain a connection between the users of the database system.

File server…confirm

54. Which of the following is an advantage of using the Traditional File Organization?

Simplicity

55. Which of the following is the input to Analysis Phase?

DFD

56. Name and _____ are part of the definitions of an attribute.

Domain

57. Which of the following most certainly implies the need for an entire table to implement?

A ternary relationship

58. Core of the Database Architecture is

Internal level

59. _____ is same as equi-join with a slight difference.

Natural join

60. Which of the following database environment causes a network overhead?

File server…..confirm

on the database he will have to send the whole file back to the server, thus causing a lot of network overhead. The Good thing

61. Conceptual database design is implemented using a .

Semantic data model….confirm

Conceptual database design
This design is implemented using a semantic data model,

62. Which constraint is generally used for efficiency purposes in the data entry process?

Default value

63. A 3 NF relation is converted to BCNF by

Dependent attributes of overlapping composite keys are put in a separate relation

64. Each _____ of a table contains atomic/single value.

Cell

65. A relation is said to be in BCNF when

It has no overlapping composite keys which have related attributes

66. Which one of the following is not a part of three-level architecture?

Physical level

67. Which of the following is INCORRECT about naming entities?

Always use abbreviations as they are convenient

68. In entity relationship diagram an instance is

Any particular entity

69. Which of the following is an example of metadata

Student

70. Level of data at which entities or objects exist in reality is called

Real world data

71. There are _____ basic properties of the database relations.

Six

72. The population of the data of the organization for which the database is created is called the _____ of the database.

Extension…confirm

62. The population of the data of the organization for which the database is created is called the _____ of the database.
a. Extension
b. Intension

73. A key which consist of two or more attributes is called_.

Composite key….confirm

or it could be composite which consists of two or more attributes.

74. How many types of dependencies are?

Three…(existence dependency, Identifier dependency, referential dependency)…….confirm

75. A mapping is one to one if each entity in X is associated with at most _____ entity in Y and vice versa.

One….confirm

A mapping R from X to Y is one-to-one if each entity in X is associated with at most one entity in Y and vice versa.

76. In _____ join, only selected rows of a relation are made cross product with second relation.

Theta

77. Which of the following is not a key?

Integrity constraint key….confirm

- Super Key
- Candidate Key
- Primary Key
- Alternate Key
- Secondary Key
- Foreign Key

78. A collection of concepts that can be used to describe the structure of a database

Data model

79. Any combination of attributes with super key is

Super key

80. Binary relationships are those, which are established between

Two entity type

81. When data stored onto a magnetic media it is stored in _____-.

Binary format……confirm

data when stored onto a magnetic media is stored in binary format,

82. Which type of Data flow diagram (DFD) explains the functionality of the processes?

Context level diagram

ı is Data Flow Diagram (DFD)
: have a context diagram of the s
:h the examination system, these
her entities.

83. In the employee table, we may store the name, father name, phone but age can be computed from the data of birth. SO age is the best example of the _____ attribute.

Derived…..confirm

stored in the database or obtained some other way. For example, we may store the name, father name, address of employees, but age can be computed from date of birth. The

advantage of declaring age as derived attribute is that whenever we will access the age,

84. _____ are called participants, when enrolled in a relationship.

Entities

85. Incase of Context-level Diagram, the system is represented by

One process only

86. Collection of raw facts is called .

Data

87. A software package designed to store and manage database

DBMS

88. In _____ join common attributes appear twice in the output with selected rows.

Equi join

89. Within a table, each primary key value .

Must be unique…..confirm

Within a table, each primary key value ____.
Select correct option:

is a minimal superkey
is always the first field in each table
must be numeric
must be unique

90. Must be numeric

Must be unique

91. _____ is not an operation of relational algebra.

Addition

92. Candidate keys which are not selected as primary key are called

Alternate keys

93. Internal schema/view defines the ___ of the data.

Structures…confirm

Internal schema or view defines the structures of the data

94. Major benefits of data sharing is:

Data of different application places at same place

95. Functional dependency between two attributes or sets of attributes i.e. Q and P is represented as .

Q → P

96. Which of the following is not the component of Relational Data Model?

Manipulation Language

97. The attribute that is calculated from other attribute is called .

Derived attribute

98. Consider the two attributes "CNIC_NO" and "Registration_NO". Both the attributes can be used for the unique identification. Now if the user selects the "CNIC_NO" as primary key, then "Registration_NO" is a_____ key.

Alternative

99. Which of the following constraint is included in the definition of the table?

Null

100. The database deals with large amount of data. This property is called

Scalability

101. Which one of the following is NOT a characteristic of meta data?

Includes user data…..confirm

102. Theta join is denoted mathematically as .

R X Θ S

103. In cross reference matrix transitions are specified on and database objects are specified on .

Y axis, Y axis

104. Each component of DBMS performs _____ functions.

Different

105. Which of the following data model is not record based?

Hierarchical Data Model

106. Network Data model

Semantic Data model

107. The basic purpose of diamond in ER diagram is to show the .

Relationships

108. Which of the following tool is used to design conceptual database design?

This model is independent of any tool

109. In "one to many" cardinality one instance of a relation is mapped.

Many instances of second entity type

110. **Cross reference matrix is developed at the _____ stage of the database.**

Designing

111. **User rights information is stored in**

Catalog

112. **Which of the following is incorrect about attributes?**

One entity type may have two attributes with same name……confirm

An attribute is identified by a name allocated to it and that has to be unique with respect to that entity type. It means one entity type cannot have two attributes with the same name. However, different entity types may have attributes with the same name. The guidelines for naming an attribute are similar to those of entity types. However, one

113. **Logical data independency provides independency to change _ model.**

Conceptual

114. **In natural join the common attribute in the output table appears .**

Once

115. **Which levels are mostly used for detailed DFD?**

Level-0, Level-1…..confirm

116. **By default, a non-key attribute in a relation can have _____ value.**

Null

117. **The three levels architecture is useful for .**

Hiding the details of internal systems

118. Which of the following is called a replacement of the manual file system?

File processing system…..confirm

computer-based approach of handling the commercial or business applications. That is why it is also called a replacement of the manual file system. Before the use computers,

119. In three level architecture of the database which level is responsible for storage of data on some storage media?

Internal/Physical view....confirm

120. With the help of _____ technique data values with the smaller sized codes can further reduce the space needed by the data for storage in the database.

Compression

121. If an entity is linked with itself, than it is called _____ relationship.

Recursive….confirm

An ENTITY TYPE linked with itself, also called recursive relationship.

122. Once the data has been transported to the physical level it is then managed by the _____.

Operating system…..confirm

enforced at this level. Once the data has been transported to the physical level it is then managed by the operating system. Operating

123. Duplication of data in controlled redundancy is controlled and .

Deliberate…..confirm

124. Teleprocessing is a sub-type of _____ .

Multi-user database

125. In an ERD, the focus is on the _____ and the relationships between them.

Entities

126. The most widely used conceptual model is the____ model.

ER

127. Student and book are two entities, what is the cardinality between the two?

Many to many

128. RDM is a simple there is just one structure and that is .

A relation or a table

129. A description on a particular collection of data using the given data model.

Database

130. Consider thy key "CNIC, Name" and choose the best option that describes the key most effectively.

The given key is a super key and candidate key as well.

131. Normalization is a process of restricting a relation to

Minimize duplication of data in a database

132. Unary relationship involves a single entity it is also called _ .

Recursive relationship….confirm

Unary Relationship

These are the relationships, which involve a single entity. These are also called recursive relationships. Unary relationships may have one to one, one to many and

133. Which of the following is a part of a DFD but not included in the ERD?

The external entity having a single instance

134. The database management system (DBMS) is used to

All of above

135. In cross reference matrix attributes are mention on _____ and reports are mention on

X axis, Y axis

136. A table can be logically connected to another table by defining a _____.

Primary key…..confirm

Explanation : A table can be logically connected to another table by defining a **Primary Key**.

137. In the relation model, the number of attributes and number of tuples in a relation are termed as _____ and _____respectively.

degree, cardinality

138. The function that an entity plays in a relationship is called that entity's _____.

Role

139. A database system is designed to meet the information needs of _____ in an organization.

Multiple users

140. Which of the following is the example of a strong entity?

Ahmad account of ABC bank has no existence if the ABC bank doesn't exist anymore

141. Which of the following is a completeness constraint?

Partial Completeness Constraint

142. In an employee entity which of the following attribute will work as a primary key?

Employee id

143. _____ is the first computer-based approach for handing the commercial or business applications

Database Systems

144. Which of the following is NOT the classification of entity type?

Dependent entity type

145. Which of the following is NOT the type of key?

Derived key

146. Data Redundancy means

Duplication of data

147. Which of the following is a type of dependency?

Referential

148. In your opinion, why relational database is widely acceptable?

Due to its dependencies

149. is the first comprehensive complete database design.

Conceptual database design

150. How data security can be assured in the database?

Implementing encryption algorithms

151. The attributes whose value is not stored in a database is known as _ attribute.

Derived….confirm

152. Logical database design, like conceptual database design is our

Database design

153. If a single instance of "student" is associated with the many instances of the "subjects" entity, then the relationship between them is:

One to many

154. The conceptual database design is drawn through .

ER model

155. A relational database is

One that consists of two or more tables that are joined in some way .

156. Data, information

Information, data

157. In which stage of the database development process logical design is created?

Implementation

158. Which of the following is NOT an entity?

Playground

159. After moving the determined columns from the original table to the new table while converting to 2NF, the determinate becomes the of the new table.

Primary key

160. Binary operator

Project operator

161. Free standing data dictionary is created by

CASE tools

162. A patient can be an outdoor or indoor. Analyze this statement and tell what type of constraint will be applied on patient?

Total completeness

163. If A → B and A → C, then A → BC. The inference Rule applies is:

Union

164. Which of the following is the correct abbreviation of ERD?

Entity relationship diagram

165. Database relation is also represented in a two dimensional structure called

Table

166. Which of the following specifies the number of instances of one entity that can or must be associated with each instance of another entity?

Cardinality view…..confirm

Recalling from the previous lecture we can say that that cardinality is just an expression which tells us about the number of instances of one entity which can be present in the second relation. Maximum cardinality tells us that how many instance of an entity can be

167. Data about data is metadata

True

168. Which of the following describes the job of a database administrator?

Monitoring and controlling database security and authorization: setting up controls to ensure the quality and integrity of data….confirm

169. Each table must have a _____ key.

Primary

170. A relational database system is based on the concept(s) of:

Tables, row and columns

171. A → BC and A → B, then A → C follow inference rule.

Projectivity

172. **Suppose, we have two entities "shape" and "triangle", then which of the following is a correct statement?**

Shape is a super type entity and triangle is a sub type entity

173. **The entity that is NOT dependent on any other entity for its existence is known as entity.**

Strong

174. **In cross reference matrix attributes are specified on**

X axis

175. **Which of the following clarify the semantics of a relationship?**

Roles

176. **How many types of cardinality is specified about relationship?**

Four

177. **Which of the following is NOT a component of a DFD?**

Relationship between external entities

178. **Which of the following provides us the facility to add properties of one entity to another entity automatically?**

Inheritance

179. **If there is more than one relationship between two entities and we mentioned role on the relationship link, then it is called:**

Multiple relationships

180. **Which one of the following is not an advantage of DBMS**

Recovery procedure is difficult

181. **In many to many relationship a third table is created for the relationship, which is also called as _**

Associative entity type

182. When one entity instance of another entity for its existence, then it is called

Existence dependency

183. Relational data model is widely accepted due to its mathematically proven foundation and its

Simplicity

184. If an entity "motor cycle" is associated with at most one "driver", then the relationship between these two entities is:

One to one

185. Database application is a program which is used for performing certain operations such as

Insertion of data, Extraction of data, Updating the data….All of above

186. Creating the relationships between new tables and their predecessors through the use of foreign keys is general requirements of

Second normal form

187. In the domains of attributes of a relation are atomic, that is they consist of single units that cannot be broken down further.

First normal form

188. Atomicity is a feature of .

1NF

189. The_____ can see entire information structure of the database.

DBA

190. The attribute whose value is not stored in a database is known as attribute.

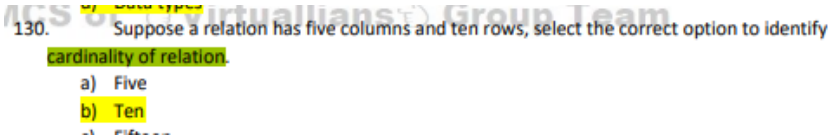
Derived

191. **are the structure defined for placing data in the attributes while designing physical design of database.**

Data types

192. **Suppose a relation has five columns and ten rows, select the correct option to identify cardinality of relation.**

Ten….confirm



193. **Which feature of database provides conversion from inconsistent state of DB to a consistent state ensuring minimum data loss?**

Recovery service

194. **Consider an employee is allotted an organization vehicle, which might soley be driven by that employee. What kind of relationship exists between employee and company?**

Many to one

195. **Which of the following is a feature of CHECK constant?**

used to enforce referential integrity

196. **Which of the following semantic model is used for the graphical representation to the conceptual database design?**

Entity Relationship (ER)

197. **Suppose we have two entities department and University. Which type of dependency exists between these two entities?**

Referential (Not sure)

198. **_____ is the next stage of conceptual database design.**

logical database design

199. **The _____ model is independent of any tool or data model.**

Entity Relationship (ER) (page 122)

200. While converting to 2NF all the columns are deleted from the original table except for the determinate which will serve as a _____ key.

Primary key

201. In relational algebra, which of the following operator is used to find tuples that are present in one relation but are not in another?

Union

202. Subject may be the prerequisite of another subject, is the example of _____ relationship.

Unary (google)

203. The maximum PL/SQL size of data type "VARCHAR2" in DBMS is bytes a)

32767

204. is state of database where a "course" cannot be inserted in the table, because this course has not been registered to any "student".

Insertion anomaly

205. The maximum PL/SQL size of data type "BLOG" in DBMS is

4 gigabytes

206. states that in a relation no attributes of a primary key (PK) can have a null value.

Entity integrity constraint

207. A database state where deletion of the information about the student record deletes the course information as well is called .

Deletion anomaly

208. An entity type is

A coherent set of similar objects that we want to store data on (e.g. STUDENTS, COURSE, CAR)

209. External scheme evolves as user needs are ____ over the time.

Modified

210. An attribute that is the collection of some other attribute(s) is known as _ attributes.

Composite

211. Which of the following is correct regarding Dataflow diagram?

Created at increasing levels of detail

212. Used to represent the relationships among the external entities are those attributes that are a combination of two or more than two attributes.

Composite attributes

213. Controlling redundancy in a database management system DOES NOT help to

avoid unauthorized access to data

214. Which of the following concepts is applicable with respect to 3NF?

Transitive dependency

215. Identify the operation which is NOT one of the parts of the five basic set operations in relational algebra?

Join

216. Making a change to the conceptual schema of a database but not affecting the existing external schemas is an example of

Logical data independence.

217. For a third normal form we concentrate on relations with one _____ key, and we eliminate transitive dependencies

Candidate

218. The entity relationship diagram is used to graphically represent the database model.

Conceptual

219. The _____ determines that the link between two entities is optional or mandatory.

Minimum cardinality

220. Select the most appropriate statement. Which of the following data model is used to design logical database design?

Relational

221. _____ are used by database administrator in situations where we have large

number of records and wastage of small amount of space in each record can lead to loss of huge amount of data storage space.

Codes

222. An attribute in a relation is foreign key If the _____ key from one relation is used as an attribute in that relation.

Primary….confirm

The foreign key attribute, which is present as a primary key in another relation is called as home relation of foreign key attribute, so in EMP table the depId is foreign key and its home relation is DEPT.
The foreign key attribute and the one present in another relation as primary key can

223. in data flow diagram which of the following operator is used, when data flows from a source process to one of the mentioned sinks?

Ring Sum Operator…confirm

- **Ring Sum Operator:**

This operator is used when data from a source process can flow to one of the mentioned sinks. For this purpose the symbol used is displayed in Figure: 11a and its presentation in

224. in data flow diagram which of the following operator is used, when data flows from a source process to all the connected sinks?

AND operator….confirm

- **AND Operator:**

This operator is used when data from a source process must flow to all the connected sinks. For this purpose the symbol used is displayed in Figure: 12a and its presentation in

225. The _____ rule specifies that an entity can be a member of only one subtype at a time.

disjoint page no 108

226. In relational algebra, which of the following Greek is used to represent the project operator?

∏ (PI) (page no 150)

227. In three level architecture of the database, which level is helpful to display data which is not actually placed in the database?

External Level….confirm

External View (Level, Schema or Model):

External views are also helpful when we want to display the data which is not place in the database or not stored at all.

228. Which of the following normal forms deals multivalued dependency?

4NF

229. While mapping _____ relationship, primary key of the entity type against one side of relationship will be included in the entity type on the many side of the relationship as foreign key.

one to many

230. In data flow diagram _____ symbol is used to represents data flows.

arrow (google)

231. In the below entity type which attribute can work as an alternate key?

Employee (Reg# Primary key,Name,CNIC,designation,address,phone)

CNIC

232. From the following who is responsible for proper working of the database and DBMS?

Database administrator

233. Suppose that "Program" is an entity which represents different courses are being offered by an institute, like MCS. BCS etc. Following are the major attributes of this entity. Identify the attribute which can be used as primary key.

program_ Code

234. The following is an of _____ inference rule.

If stId –> prName and prName –> credits then stId –> credits

Transitivity

235. Which of the following is the example of a weak entity?

The tire can exist without being attached to the car

236. In Relational Data Model, the _____ of a relation is the number of columns in that relation

Degree

237. Which of the following normal form deals with possible loss less decompositions?

5NF

238. Suppose we have a list of bank customers which availed loan facility from different branches of a bank. If we want to get a list of all customers and the amount of the loans excluding branch name. Which of the following operation allows us to generate this relation?

Union

239. Which of the following provides us data independency?

3-Level Architecture

240. Cross reference matrix is a tool available in the data dictionary, which helps us in finding ____.

Entities of the database and their associations

241. Conceptual database design, expressed in E-R data model can be implemented using any _____.

Database Management System (DBMS)

242. Which of the following might be represented with a single-valued attribute?

Computer's processor speed

243. Which of the following is a Unary operation?

Project

193. Which of the following data model has the following components?

1. Construct (Tables)

2. Manipulation language (SQL)

3. integrity constraints?

Relational

244. Which of the following Greek is used to represent the select operation?

σ (page 149)

245. Data about data is called metadata.

True (page no 10)

246. Normalization of database is used to _____.

Eliminate redundancy

247. In student entity which of the following is the best example of multivalued attribute?

Email address

248. Theta join is denoted mathematically as

R X theta S

249. In which normal form, we convert composite attributes into individual attributes?

1NF

250. Which of the following tool is used to graphically represent the components and design of the system?

DFD

251. Data type "BLOB" in database management system stands for _____.

Binary Large Object (page no 185)

252. In a three level DBMS architecture, the _____ level interacts directly with the users.

External

253. In the Database System _____ provides many features to ensure the data integrity.

Database Management System (DBMS)

basis of information produced from the database and the wrong information leads to wrong decisions, and business collapse. In the database system environment, DBMS provides many features to ensure the data integrity, hence provides more reliable data

254. The process of defining one or more subtypes of a super and forming relationships is known as

Generalization

255. External scheme evolves, as user needs are _____ over the time.

Switch

256. _____ of the Database is most difficult to change when a database is created.

Extension

257. In Database development process, which phase involves designing of any new application for the enhancement of the system?

Maintenance phase….confirm

258. An attribute of one entity is associated with another attribute of the same entity is called:

Recursive relationship….confirm

Recursive Relationship:
This is the situation when any attribute of one entity is associated with another attrib of the same entity. Such a link initiates from one entity and terminates on the same ent

259. Relational Data model is created in the _____ phase.

Analysis

260. Database system saves lots of efforts and finances, this advantage provides _____.

Time saving structure

261. Data inconsistency leads to a number of a problems such as _____.

loss of information and incorrect results

262. Every primary key (PK) will be a candidate key and every candidate key will be a _____.

Super key…confirm

also holds between candidate and primary keys, that is, every primary key (PK) is a candidate key and every candidate key is a super key.

263. If we want to include attributes from two different relations, which of the following operations will be used?

Union

264. When have two entities "Person" and "Doctor. Which of all is best suitable type of relationship between the mentioned entities?

Inheritance…..confirm

We have two (Raise For Success) entities "Person" and "Doctor". Which of the following is best suitable type of relationship between the mentioned entities?

- Inheritance Correct

Database Management System

(CS403)

Lecture No. 01

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	Chapter 1.

Overview of Lecture

- Introduction to the course
- Database definitions
- Importance of databases
- Introduction to File Processing Systems
- Advantages of the Database Approach

Introduction to the course

This course is first (fundamental) course on database management systems. The course discusses different topics of the databases. We will be covering both the theoretical and practical aspects of databases. As a student to have a better understanding of the subject, it is very necessary that you concentrate on the concepts discussed in the course.

Areas to be covered in this Course:

- Database design and application development: How do we represent a real-world system in the form of a database? This is one major topic covered in this course. It comprises of different stages, we will discuss all these stages one by one.

- Concurrency and robustness: How does a DBMS allow many users to access data concurrently, and how does it protect against failures?

- Efficiency and Scalability: How does the database cope with large amounts of data?

- Study of tools to manipulate databases: In order to practically implement, that is, to perform different operations on databases some tools are required. The operations on databases include right from creating them to add, remove and modify data in the database and to access by different ways. The tools that we will be studying are a manipulation language (SQL) and a DBMS (SQL Server).

Database definitions:

Definitions are important, especially in technical subjects because definition describes very comprehensively the purpose and the core idea behind the thing. Databases have been defined differently in literature. We are discussing different definitions here, if we concentrate on these definitions, we find that they support each other and as a result of the understanding of these definitions, we establish a better understanding of use, working and to some extent the components of a database.

Def 1: A shared collection of logically related data, designed to meet the information needs of multiple users in an organization. The term database is often erroneously referred to as a synonym for a "database management system (DBMS)". They are not equivalent and it will be explained in the next section.

Def 2: A collection of data: part numbers, product codes, customer information, etc. It usually refers to data organized and stored on a computer that can be searched and retrieved by a computer program.

Def 3: A data structure that stores metadata, i.e. data about data. More generally we can say an organized collection of information.

Def 4: A collection of information organized and presented to serve a specific purpose. (A telephone book is a common database.) A computerized database is an updated, organized file of machine readable information that is rapidly searched and retrieved by computer.

Def 5: An organized collection of information in computerized format.

Def 6: A collection of related information about a subject organized in a useful manner that provides a base or foundation for procedures such as retrieving information, drawing conclusions, and making decisions.

Def 7: A Computerized representation of any organizations flow of information and storage of data.

Each of the above given definition is correct, and describe database from slightly variant perspectives. From exam point of view, anyone will do. However, within this course, we will be referring first of the above definitions more frequently, and concepts discussed in the definition like, logically related data, shared collection should be clear. Another

important thing that you should be very clear about is the difference between database and the database management system (DBMS). See, the database is the collection of data about anything, could be anything. Like cricket teams, students, busses, movies, personalities, stars, seas, buildings, furniture, lab equipment, hobbies, hotels, pets, countries, and many more anything about which you want to store data. What we mean by data; simply the facts or figures. Following table shows the things and the data that we may want to store about them:

Thing	Data (Facts or figures)
Cricket Player	Country, name, date of birth, specialty, matches played, runs etc.
Scholars	Name, data of birth, age, country, field, books published etc.
Movies	Name, director, language (Punjabi is default in case of Pakistan) etc.
Food	Name, ingredients, taste, preferred time, origin, etc.
Vehicle	Registration number, make, owner, type, price, etc.

There could be infinite examples, and please note that the data that is listed about different things in the above table is not the only data that can be defined or stored about these things. As has been explained in the definition one above, there could be so many facts about each thing that we are storing data about; what exactly we will store depends on the perspective of the person or organization who wants to store the data. For example, if you consider food, data required to be stored about the food from the perspective of a cook is different from that of a person eating it. Think of a food, like, Karhahi Ghost, the facts about Karhahi ghosht that a cook will like to store may be, quantity of salt, green and red chilies, garlic, water, time required to cook and like that. Where as the customer is interested in chicken or meat, then black or red chilies, then weight, then price and like that. Well, definitely there are some things common but some are different as well. The thing is that the perspective or point of view creates the difference in what we store; however, the main thing is that the database stores the data.

The database management system (DBMS), on the other hand is the software or tool that is used to manage the database and its users. A DBMS consist of different components or subsystem that we will study about later. Each subsystem or component of the DBMS performs different function(s), so a DBMS is collection of different programs but they all work jointly to manage the data stored in the database and its users. In many books and may be in this course sometimes database and database management system are used interchangeably but there is a clear difference and we should be clear about them. Sometimes another term is used, that is, the database system, again, this term has been used differently by different people, however in this course we use the term database system as a combination of database and the database management system. So database is collection of data, DBMS is tool to manage this data, and both jointly are called database system.

Importance of the Databases

Databases are important; why? Traditionally computer applications are divided into commercial and scientific (or engineering) ones. Scientific applications involve more computations, that is, different type of calculations that vary from simple to very complex. Today such applications exist, like in the fields of space, nuclear, medicine that take hours or days of computations on even computers of the modern age. On the other hand, the applications that are termed as commercial or business applications do not involve much computations, rather minor computation but mainly they perform the input/output operations. That is, these applications mainly store the data in the computer storage, then access and present it to the users in different formats (also termed as data processing) for example, banks, shopping, production, utilities billing, customer services and many others. As is clear from the example systems mentioned, the commercial applications exist in the day to day life and are related directly with the lives of common people. In order to manage the commercial applications more efficiently databases are the ultimate choice because efficient management of data is the sole objective of the databases. So such applications are being managed by databases even in a developing country like Pakistan, yet to talk about the developed countries. This way databases are related directly or indirectly almost every person in society.

Databases are not only being used in the commercial applications rather today many of the scientific/engineering application are also using databases less or more. Databases are concerned of effectively latter form of applications which are more Commercialapplications. The goal of this course is to present an in-depth introduction to databases, with an emphasis on how to organize information in the database and to maintain it and retrieve it efficiently, that is, how to design a database and use it effectively.

Databases and Traditional File Processing Systems

Traditional file processing system or simple file processing system refers to the first computer-based approach of handling the commercial or business applications. That is why it is also called a replacement of the manual file system. Before the use computers, the data in the offices or business was maintained in the files (well in that perspective some offices may still be considered in the pre-computer age). Obviously, it was laborious, time consuming, inefficient, especially in case of large organizations. Computers, initially designed for the engineering purposes were though of as blessing, since they helped efficient management but file processing environment simply transformed manual file work to computers. So processing became very fast and efficient, but as file processing systems were used, their problems were also realized and some of them were very severe as discussed later.

It is not necessary that we understand the working of the file processing environment for the understanding of the database and its working. However, a comparison between the characteristics of the two definitely helps to understand the advantages of the databases

and their working approach. That is why the characteristics of the traditional file processing system environment have been discussed briefly here.

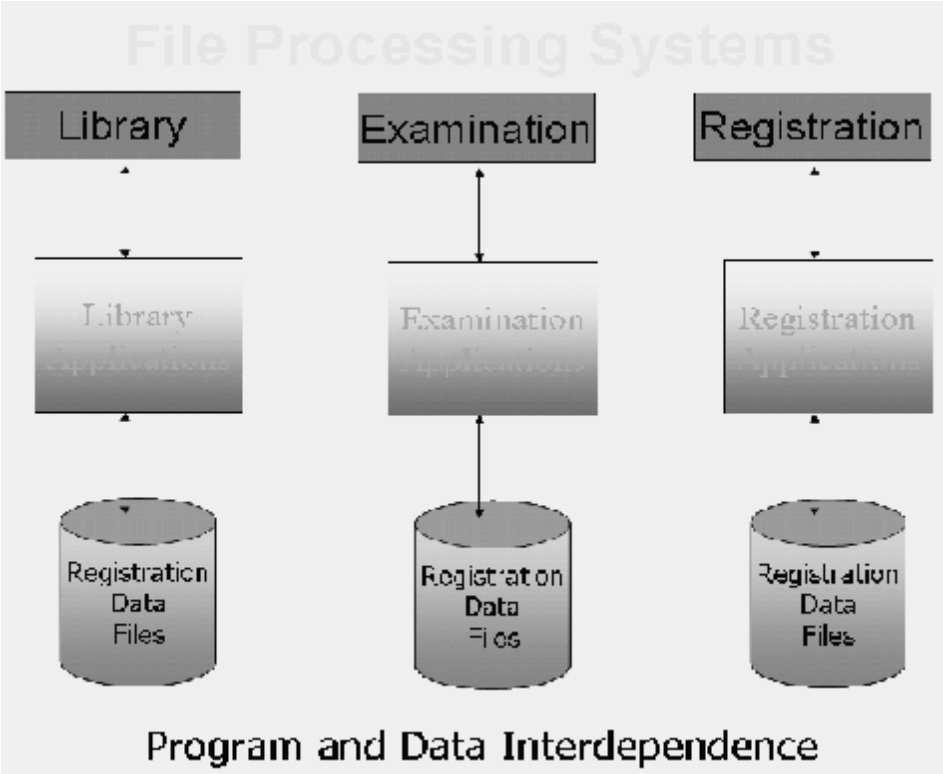


Fig. 1: A typical file processing environment

The diagram presents a typical traditional file processing environment. The main point being highlighted is the program and data interdependence, that is, program and data depend on each other, well they depend too much on each other. As a result any change in one affects the other as well. This is something that makes a change very painful or problematic for the designers or developers of the system. What do we mean by change and why do we need to change the system at all. These things are explained in the following.

The systems (even the file processing systems) are created after a very detailed analysis of the requirements of the organizations. But it is not possible to develop a system that does not need a change afterwards. There could be many reasons, mainly being that the users get the real taste of the system when it is established. That is, users tell the analysts or designers their requirements, the designers design and later develop the system based on those requirements, but when system is developed and presented to the users, it is only then they realize the outcome of the effort. Now it could be slightly and (unfortunately) sometimes very different from what they expected or wanted it to be. So the users ask changes, minor or major. Another reason for the change is the change in the requirements. For example, previously the billing was performed in an organization on the monthly

basis, now company has decided to bill the customers after every ten days. Since the bills are being generated from the computer (using file processing system), this change has to be incorporated in the system. Yet another example is that, initially bills did not contain the address of the customer, now the company wants the address to be placed on the bill, so here is change. There could be many more examples, and it is so common that we can say that almost all systems need changes, so system development is always an on-going process.

So we need changes in the system, but due to program-data interdependence these changes in the systems were very hard to make. A change in one will affect the other whether related or not. For example, suppose data about the customer bills is stored in the file, and different programs use this file for different purposes, like adding data into the bills file, to compute the bill and to print the bill. Now the company asks to add the customers' address in the bills, for this we have to change the structure of the bill file and also the program that prints the bill. Well, this was necessary, but the painful thing is that the other programs that are using these bills files but are not concerned with the printing of the bills or the change in the bill will also have to be changed, well; this is needless and causes extra, unnecessary effort.

Another major drawback in the traditional file system environment is the non-sharing of data. It means if different systems of an organization are using some common data then rather than storing it once and sharing it, each system stores data in separate files. This creates the problem of redundancy or wastage of storage and on the other hand the problem on inconsistency. The change in the data in one system sometimes is not reflected in the same data stored in other system. So different systems in organization; store different facts about same thing. This is inconsistency as is shown in figure below.

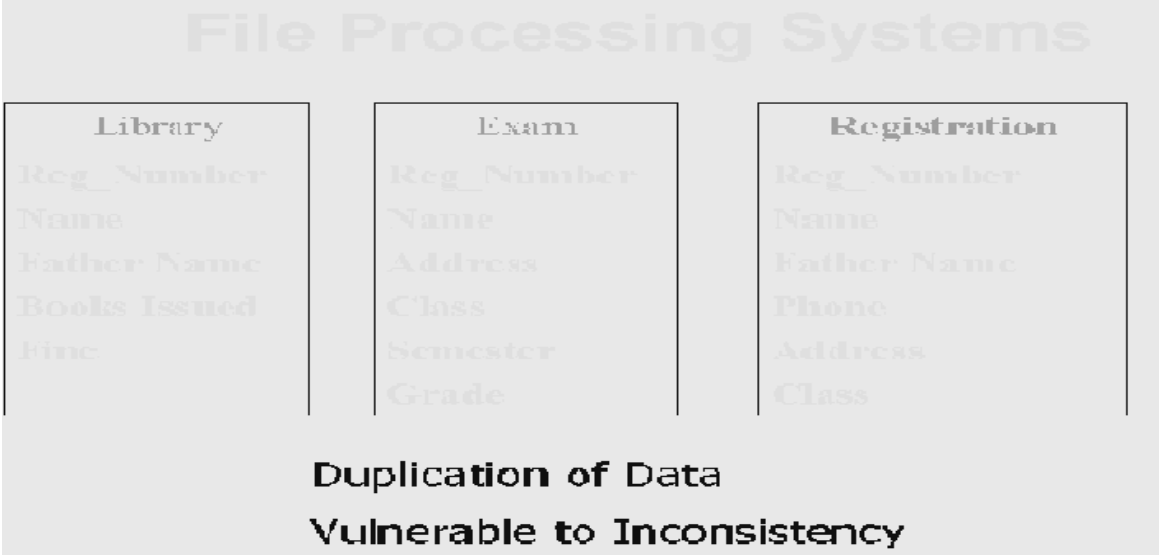


Fig. 2: Some more problems in File System Environment

Previous section highlighted the file processing system environment and major problems found there. The following section presents the benefits of the database systems.

Advantages of Databases

It will be helpful to reiterate our database definition here, that is, database is a shared collection of logically related data, designed to meet the information needs of multiple users in an organization. A typical database system environment is shown in the figure 3 below:

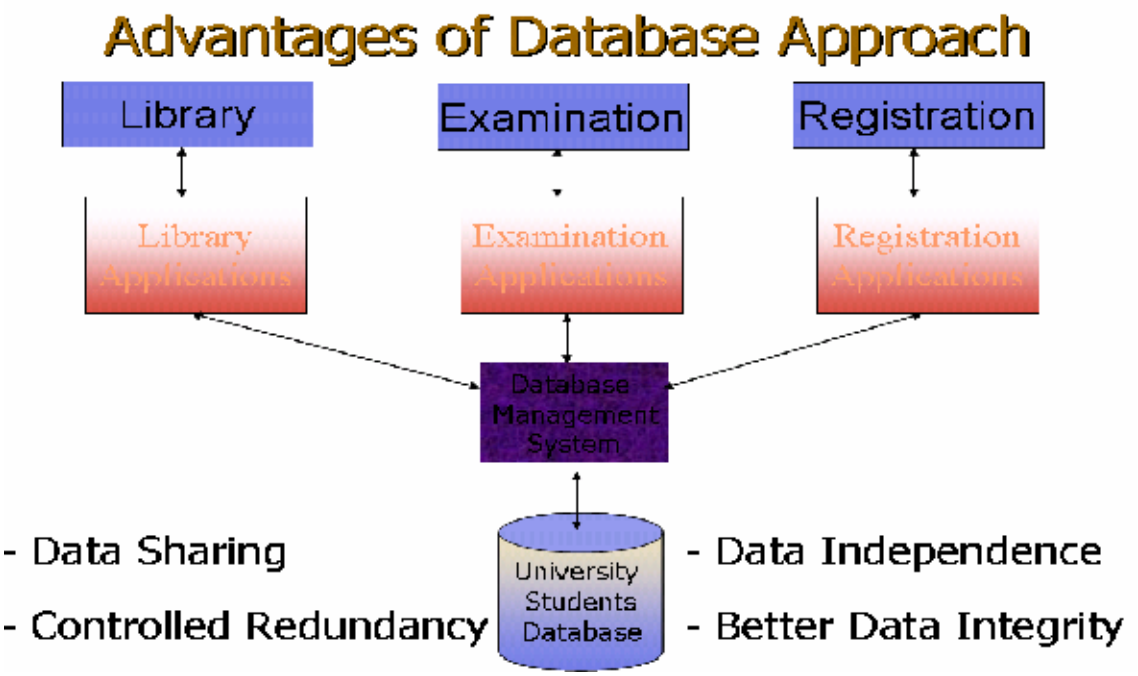


Fig. 3: A typical Database System environment

The figure shows different subsystem or applications in an educational institution, like library system, examination system, and registration system. There are separate, different application programs for every application or subsystem. However, the data for all applications is stored at the same place in the database and all application programs, relevant data and users are being managed by the DBMS. This is a typical database system environment and it introduces the following advantages:

- Data Sharing
  The data for different applications or subsystems is placed at the same place. This introduces the major benefit of data sharing. That is, data that is common among different applications need not to be stored repeatedly, as was the case in the file processing environment. For example, all three systems of an educational institution shown in figure 3 need to store the data about students. The example data can be seen

from figure 2. Now the data like registration number, name, address, father name that is common among different applications is being stored repeatedly in the file processing system environment, where as it is being stored just once in database system environment and is being shared by all applications. The interesting thing is that the individual applications do not know that the data is being shared and they do not need to. Each application gets the impression as if the data is being for stored for it. This brings the advantage of saving the storage along with others discussed later.

- Data Independence
  Data and programs are independent of each other, so change is once has no or minimum effect on other. Data and its structure is stored in the database where as application programs manipulating this data are stored separately, the change in one does not unnecessarily effect other.

- Controlled Redundancy
  Means that we do not need to duplicate data unnecessarily; we do duplicate data in the databases, however, this duplication is deliberate and controlled.

- Better Data Integrity
  Very important feature; means the validity of the data being entered in the database. Since the data is being placed at a central place and being managed by the DBMS, so it provides a very conducive to check or ensure that the data being entered into the database is actually valid. Integrity of data is very important, since all the processing and the information produced in return are based on the data. Now if the data entered is not valid, how can we be sure that the processing in the database is correct and the results or the information produced is valid? The businesses make decisions on the basis of information produced from the database and the wrong information leads to wrong decisions, and business collapse. In the database system environment, DBMS provides many features to ensure the data integrity, hence provides more reliable data processing environment.

Dear students, that is all for this lecture. Today we got the introduction of the course, importance of the databases. Then we saw different definitions of database and studied what is data processing then studied different features of the traditional file processing environment and database (DB) system environment. At the end of lecture we were discussing the advantages of the DB approach. There some others to be studied in the next lecture. Suggestions are welcome.

Exercises

- Think about the data that you may want to store about different things around you
- List the changes that may arise during the working of any system, lets say Railway Reservation System

Lecture No. 02

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	

Overview of Lecture

- Some Additional Advantages of Database Systems
- Costs involved in Database systems
- Levels of data
- Database users

Difference between Data and Information

Data is the collection of raw facts collected from any specific environment for a specific purpose. Data in itself does not show anything about its environment, so to get desired types of results from the data we transform it into information by applying certain processing on it. Once we have processed data using different methods data is converted into meaningful form and that form of the Data is called information

Example:

Example:

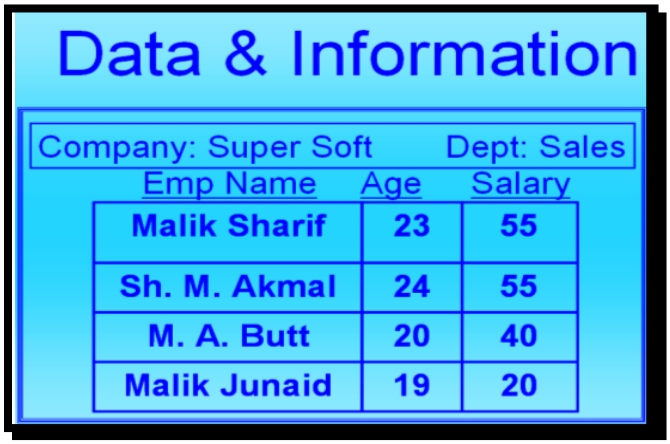


Fig. 1: Data and Information

If we consider the data in the above figure without the titles or the labels associated with the data (EmpName, age, salary) then it is not much useful. However, after attaching these labels it brings some meanings to us, this meaningfulness is further increased when we associate some other labels, like the company name and the department name etc. So this is a very simple example of processing that we can do on the data to make it information.

Once we have clear idea of what data and information is we proceed with another term knows as "schema" Schema is a repository or structure to express the format and other different information about data and database, as we can see from the database definition "Database is a self describing collection of interrelated records." The word self describing means that the data storage and retrieval mechanism and its format is described in the database, Actual place where these definitions and descriptions are performed is database schema.

- Database Application:

Database Application is a program or group of programs which is used for performing certain operations on the data stored in the database. These operations may contain insertion of data into a database or extracting some data from the database based on a certain condition, updating data in the database, producing the data as output on any device such as Screen, disk or printer.

- Database Management Systems:

Database management system is software of collection of small programs to perform certain operation on data and manage the data.

Two basic operations performed by the DBMS are:

- Management of Data in the Database
- Management of Users associated with the database.

Management of the data means to specify that how data will be stored, structured and accessed in the database.

Management of database users means to manage the users in such a way that they can perform any desired operations on the database. DBMS also ensures that a user can not perform any operation for which he is not allowed. And also an authorized user is not allowed to perform any action which is restricted to that user.

In General DBMS is a collection of Programs performing all necessary actions associated to a database.

Further Advantages of Database Systems:

Database systems are very much beneficent to enterprises and businesses, some of the advantages are listed below:

- Data consistency
- Better data security
- Faster development of new applications
- Economy of scale
- Better concurrency control
- Better backup and recovery procedures

Data Consistency:

Data consistency means that the changes made to different occurrence of data should be controlled and managed in such a way that all the occurrences have same value for any specific data item. Data inconsistency leads to a number of problems, including loss of information and incorrect results. In database approach it is controlled because data is shared and consistency is controlled and maintained.

Better Data Security:

All application programs access data through DBMS, So DBMS can very efficiently check that which user is performing which action and accessing which part of data , So A DBMS is the most effectively control and maintain security of Data stored in a database.

Faster Application Development:

The database environment allows us faster application development because of its many reasons. As we know that database is designed with the factor of future development in mind

So whenever we have to build a new application to meet the growing needs of the computerized environment, it may be easy due to the following reason:

- The data needed for the new application already resides in the database.

- The data might not already reside in the database but it could be derived from the data present in the database

Thus we can say that, to develop a new application for an existing database system less effort is required in terms of the system and database design.

o Economy of Scale:

Databases and database systems are designed to share data stored in one location for many different purposes, So it needs not be stored as many number of times in different forms as it is used, for example the data used by Admission Department of any education institution can be used to maintain the attendance record of the students as well as the examination records of the students. So it saves us lots of efforts and finances providing economy of scale.

o Better Concurrency Control:

Concurrency means the access of database form as number of points simultaneously. Concurrency control means to access the database in such a way that all the data accesses are completed correctly and transparently. One example of controlled concurrency is the use of ATM Machine for withdrawal of money (cash). All ATM machines of a bank are interconnected to a central database system worldwide, so that a user can access its account from anywhere in the world and can get cash from any ATM terminal. As there are thousands of ATM terminal across the world for a specific bank so as a result thousands of user process and access the bank's database. All this process is managed concurrently using the database systems and is done in such an efficient manner that no two user face any delay in the processing of their requests.

o Better Backup and Recovery Facility:

Data is a very important resource and is very much valuable for any organization, loss of such a valuable resource can result in a huge strategic disasters. As Data is stored on today's' storage devices like hard disks etc., It is necessary to take periodic backups of data so that in case a storage device looses the data due to any damage we should be able

to restore the data a nearest point, Database systems offer excellent facilities for taking backup of data and good mechanism of restoring those backups to get back the backed-up data.

It some time happens that a database which was in use and very important transactions were made after the last backup was made, all of a sudden due to any disastrous situation the database crashes (improper shutdown, invalid disk access, etc.) Now in such a situation the database management system should be able to recover the database to a consistent state so that the transactions made after the last backup are not lost.

Cost Involved:

Enjoying all these benefits of the database systems do have some additional costs on any organization which is going to adopt a database environment. These charges may also be known as the disadvantages of the database system. Different types of costs (Financial and Personnel) which an organization faces in adopting a database system are listed below:

- High Cost:

Database Systems have a number of inherent charges which are to be born by any organization that is going to adopt it. High Cost is one of these inherent charges, it includes the need for specialized software which is used to run database systems, Additional and specialized hardware and technically qualified staff are the requirements for adopting to the database system, all these requirements need an organization to invest handsome amount of money to have all the requirements of the database systems.

- Conversion Cost:

Once an organization has decided to adopt database system for its operations, it is not only the finance and technical man-power which is required for switching on to database system, it further has some conversion charges needed for adopting the database system, this is also a very important stage for making decision about the way the system will be converted to database system.

- Difficult Recovery Procedures:

Although the database systems and database management systems provide very efficient ways of data recovery in case of any disaster, still the process of recovering a crashed database is very much technical and needs good professional skills to perform a perfect recovery of the database.

Importance of Data

- Data as a Resource:

A resource is anything which is valuable for an organization. There can be a number of resources in any organization, for example, Buildings, Furniture, Vehicle, Technical Staff, Managers, supporting staff and Machinery etc. As all these are resources for organizations and are consumed very much carefully to get full benefit out of them. Data in the same way is a very important resource and needs to considered equally important as other resource are considered.

Why we call data as a resource?

Data is truly considered a resource because for an organization to make proper decisions at proper time it is only the data which can provide correct information and in-turn cause good utilization of other organizational resources. Organizations can not make good and effective decisions if the required data is not available in time or in the correct and desired format, such bad and miscalculated decisions ultimately lead to the failure of organizations or business.

Levels of Data

- Real World Data

The real world level of data means that level of data at which entities or objects exist in reality, it means that any object existing in reality have a name and other identifiable attributes through which we can identify that specific object or entity.

Example:

Any Student

- Meta Data:

For storage of the data related to any entity or object existing at real world level we define the way the data will be stored in the database. This is called Meta data. Meta data is also known as schema for the real world data. It tells that what type of data will be stored in the database, what will be size of a certain attribute of the real world data, how many and what attributes will be used to store the data about the entity in the database.

Example: Name , Character Type, 25 character size field,

Age, Date type, 8 bytes size

Class, Alpha Numeric, 8 byte size field

- Existence of Data:

Existence of the data level shows the actual data regarding the entities as real world level according to the rules define at the Meta Data level.

Example:

According to the definition given in the Meta data level the Actual data or Data occurance for the entity at real world level is shown below:

Name	Age	Class	
Ali	20/8/1979	MCS-I	
Amir	22/3/1978	MCS-II	etc…

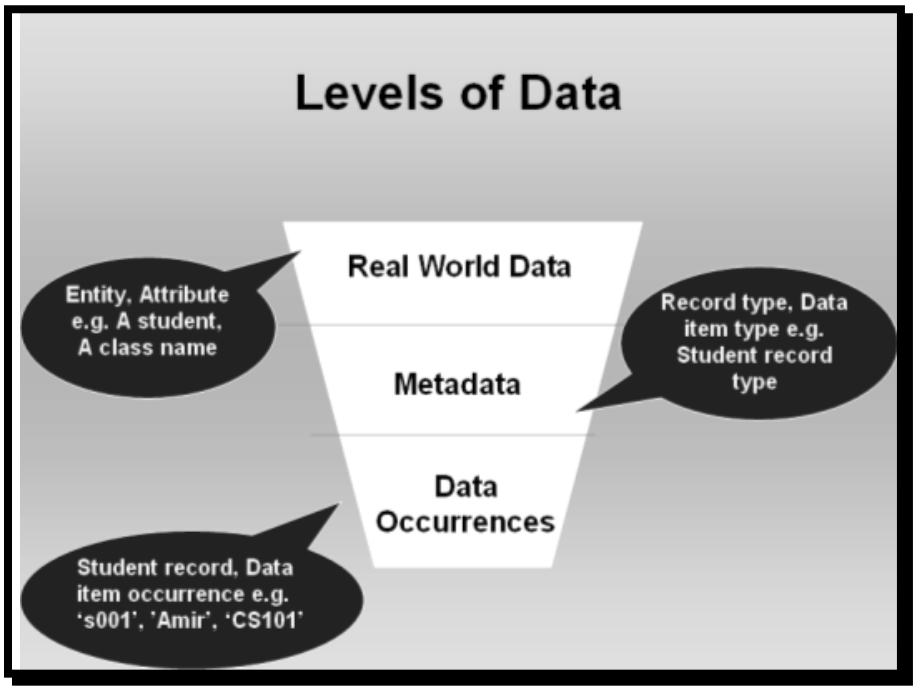


Fig. 2: Levels of Data

Users of Database Systems:

- Application Programmers
- End Users
  - Naïve
  - Sophisticated

- Application programmers:

This category of database users contains those people who create different types of database application programs that we have seen earlier. Application programmers design the application according to the needs of the other users of the database in a certain environment. Application programmers are skilled people who have clear idea of the structure of the database and know clearly about the needs of the organizations.

- End Users:

Second category of the Database users are the end users, this group of users contains the people who use the database application programs developed by the Application programmers. This category further contains two types of users:

- Naïve Users
- Sophisticated Users

- Naïve Users

This category of users is that category who simply use the application database programs created by the programmers. This group has no interaction with other parts of there database and only use the programs meant for them. They have not to worry about the further working of the database.

- Sophisticated Users:

This type of users has some additional rights over the Naïve users, which means that they can access the data stored in the database any of their desired way. They can access data using the application programs as well as other ways of accessing data. Although this type of users has more rights to access data, but these users have to take more responsibility and they need to be aware of the database structure. Moreover such users should be skilled enough to be able to get data from database with making and damage or loss to the data in database.

o Database Administrators (DBA):

This class of database users is the most technical class of db users. They need to have the knowledge of how to design and manage the database use as well as to manage the data in the database. DBA is a very responsible position in an organization. He is responsible for proper working of the database and DBMS, has the responsibility of making proper database backups and make necessary actions for recovering the database in case of a database crash. To fulfill the requirements of a DBA position a DBA needs vast experience and very elegant technical skills.

- Duties of the DBA

A Database administrator has some very precisely defined duties which need be performed by the DBA very religiously. A short account of these jobs is listed below:

- Schema definition
- Granting data access
- Routine Maintenance
  - Backups
  - Monitoring disk space
  - Monitoring jobs running

- Schema Design

DBA in some organization is responsible for designing the database schema, which means that DBA is the person who create all the meta data information for the organization on which the database is based. However in some very large scale organizations this job is performed by the database designer, which is hired for the purpose of database design and once the database system is installed and working, it is handed over to the DBA for further operation.

- Granting Access to Users:

DBA is also responsible for grant of access rights to the database users. Along with granting and revoking (taking back) the rights the DBA continuously monitors and ensure the legal use of these rights.

- Monitoring Disk Space :

When a new database is created it takes a limited space but as a result of daily activity the database acquires more data and grows in size very rapidly. The DBA has to monitor the disk space usage and statistics to ensure that no data over flow occurs at any stage.

- Monitoring Running Jobs:

To ensure the secure and proper functioning of the database system a DBA continuously monitors some associated activities also and ensure that all users are using their

authorities legally and different devices attached to the database system are functioning properly.

Typical Components of a Database Environment:

Different typical components of a database environment are shown in the figures below; they describe graphically the role of different types of users.

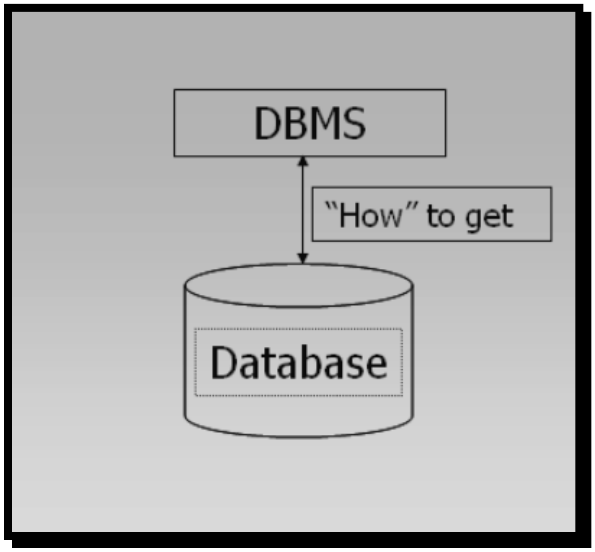


Fig. 3: DBMS and Database

Database is used to store data and DBMS uses mechanisms to get data from the database

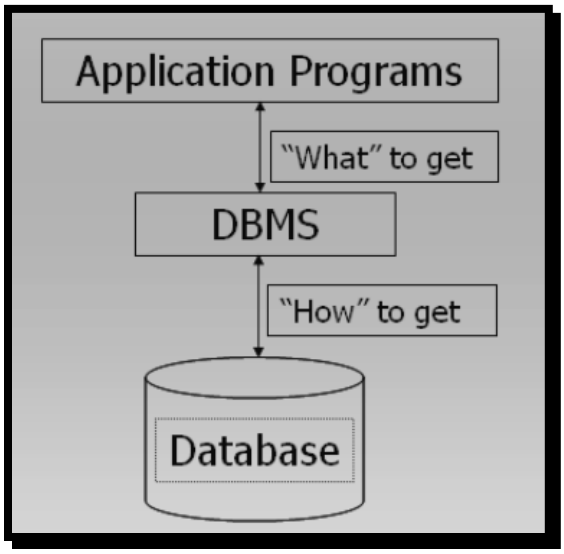


Fig. 4: Application Programs

Application programs talk to DBMS and ask for the data required

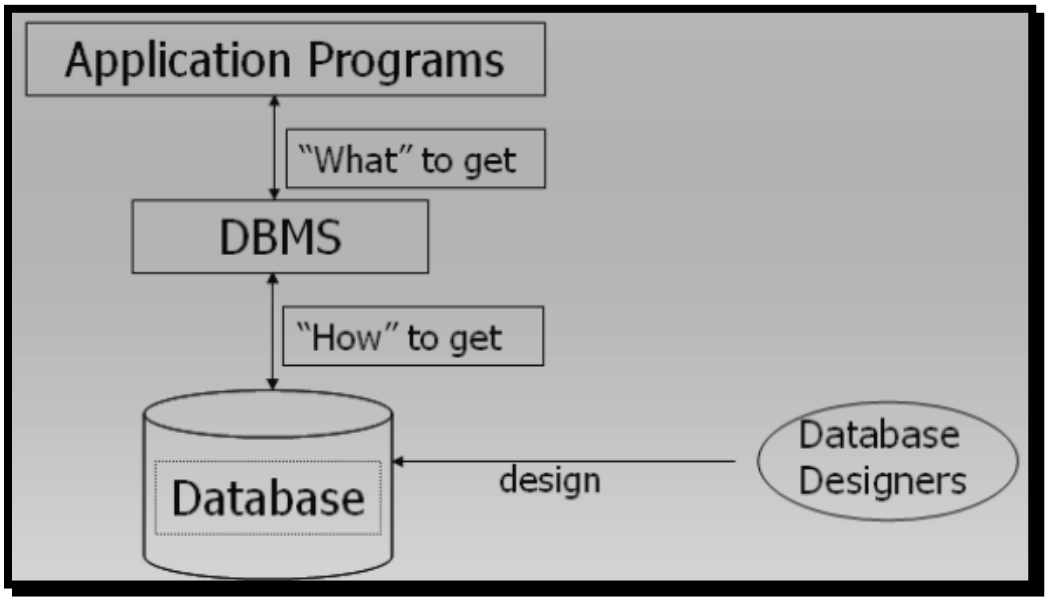


Fig. 5: Database Designers

Database designers design (for large organizations) the database and install the DBMS for use by the users of the database in any specific organization.

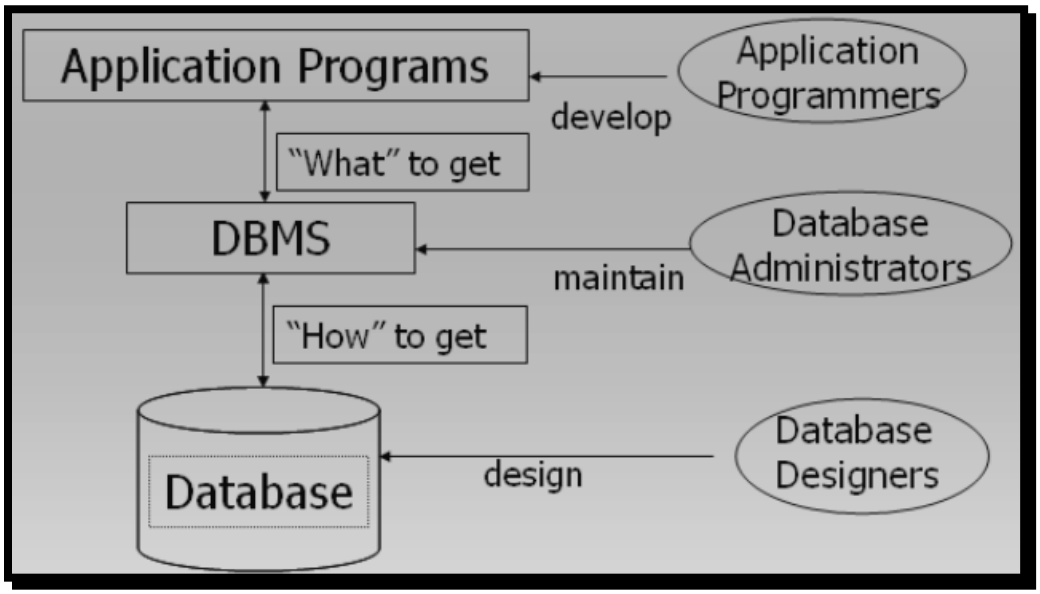


Fig. 6: Database Administrator

Once Database has been installed and is functioning properly in a production environment of an organization the Database Administrator takes over the charge and performs specific DBA related activities including:

- Database maintenance
- Database Backup
- Grant of rights to database users
- Monitoring of Running Jobs
- Managing Print jobs
- Ensuring quality of Service to all users

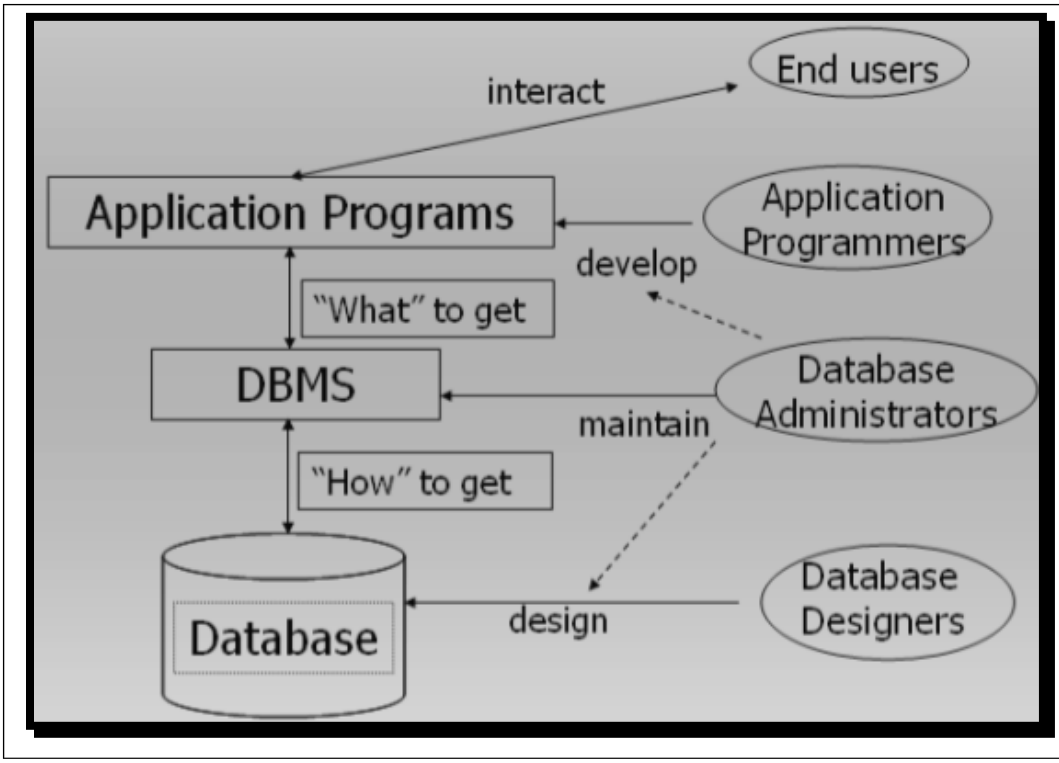


Fig. 7: Database Administration's interaction with other users

- Database administrator can interact with the database designer during database design phase so that he has a clear idea of the database structure for easy reference in future.
- This helps DBA perform different tasks related to the database structure.
- DBA also interacts with the application programmers during the application development process and provides his services for better design of applications.
- End users also interact with the system using application programs and other tools as specified in the description above.

This concludes lecture number 2, in case of any queries, please feel free to contact.

Lecture No. 03

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	Section: 4.1.1, 4.1.2

Overview of Lecture

- Database Architecture
- External View of the database
- Conceptual view of the database

Database Architecture:

Standardization of database systems is a very beneficent in terms of future growth, because once a system is defined to follow a specific standard, or is built on a specific standard, it provides us the ease of use in a number of aspects.

First if any organization is going to create a new system of the same usage shall create the system according to the standards and it will be easier to develop, because the standards which are already define will be used for developing the system.

Secondly if any organization wants to create and application software that will provide additional support to the system, it will be an easier task for them to develop such system and integrate them into existing database applications.

Users which will be using the system will be comfortable with the system because a system built on predefined standards is easy to understand and use, rather than understanding learning and using an altogether new system which is designed and built without following any standards.

Expansion to systems which are not built on standards is very hard and needs lots of efforts.

Technical staff working on a system built on standard has no problem to learn the use and architecture of the system and whenever there is a need in change of staff new staff members can be hired and put to work without any prior training for the use of system.

Database standard proposed by ANSI SPARK in 1975 is being used worldwide and is the only most popular agreed upon standard for database systems.

The Three Level Schema architecture provides us a number of benefits. For accessing data at different levels we have a number of users because not all users have to access data in database at all the database levels. The 3 levels architecture allows us to separate the physical representation of data from the users' views of data.

In the database, same data is stored in a specific feasible format and is available to different users in different formats as desired by different users. For example, consider we have stored the DOB (Date of Birth) in the database in a particular format, like in the form of dd-mm-yyyy (for example, 28-03-1987). However, the users from different departments may require to view the date of birth in different forms; the examination department may ask it to be displayed as month-day-yyyy (like march-28-1987) the Registrar's office may ask to display date of birth as mm/dd/yyyy, still the Library may need the in the form of dd/mm/yy. The Three Level Schema allows us to access the data in different formats at the external level, which is stored in a specific format at the internal level.

The Three levels architecture is useful for hiding the details of internal systems; it in-fact hides the details of underlying system views from the users at other levels and restricts the access of data and the system from any unauthorized intervention. It is the mechanism which allows us to store the data in the system in such a way that it can be provided to all users in their desired formats and with unveiling other details and information stored in the database. Moreover if there is a change to be done to the data stored in the database subject to the

requirements of a specific user it needs not be changed for that user specifically, we can create a change to the specific external view of that user and the internal details remain unchanged. Also if we want to change the underlying storage mechanism of the data stored on the disk we can do it without affecting the internal and conceptual view at the lowest level in the three levels architecture is the internal view or internal level which is shown below in the diagram and is illustrated in the coming lines.

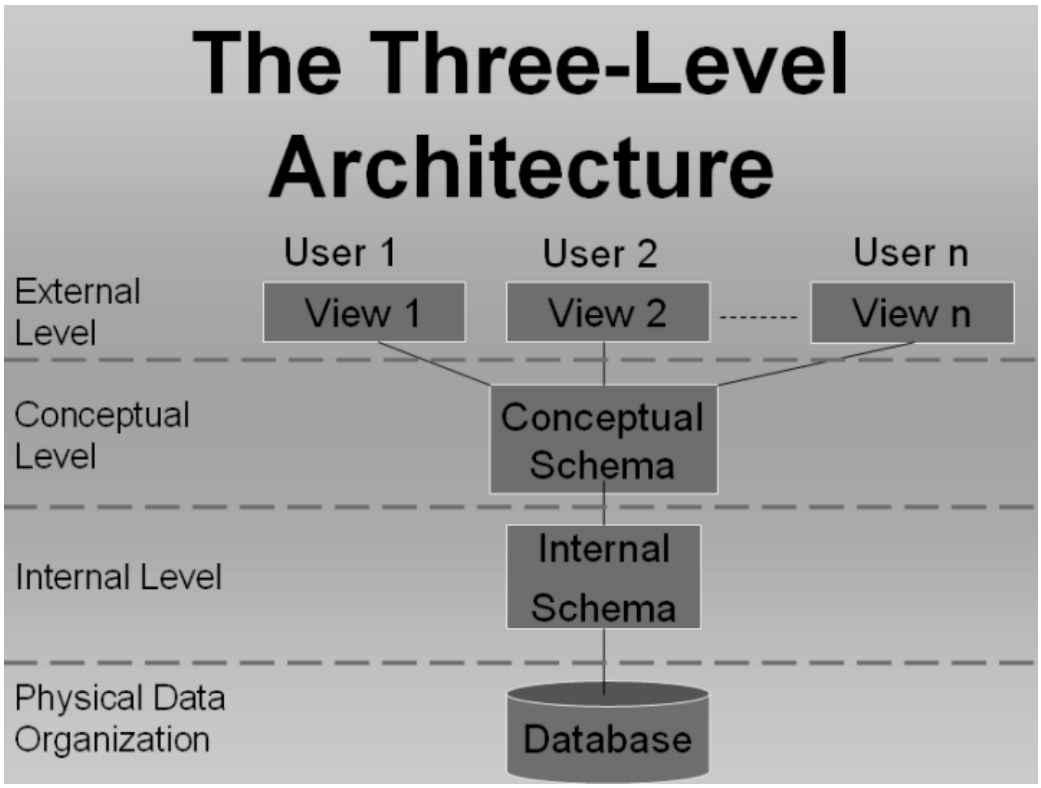


Fig. 1: Three level architecture of database

The Architecture:

The schemas as it has been defined already; is the repository used for storing definitions of the structures used in database, it can be anything from any entity to the whole organization. For this purpose the architecture defines different schemas stored at different levels for isolating the details one level from the other.

Different levels existing at different levels of the database architecture are expressed below with emphasis on the details of all the levels individually.

Core of the database architecture is the internal level of schema which is discussed a bit before getting into the details of each level individually. The internal level implements all the inner details and defines the intentions of the database. Internal schema or view defines the structures of the data and other data related activities in the database. For example it defines that for a student what data will be stored in terms of attributes of the student and it also defines how different values for these attributes will be stored, also it tells that who is allowed to make changes to the database and what changes he can make, etc.

These details give us the internal schema and are called the intention of the database. Intention for a database is almost permanent, because while designing the database it is ensured that no information is left behind which is important enough to be stored in the database and what information is important to be stored in the database from the future point of view.

Once the intention of the database has been defined then it is undesirable to change the intention for any reason. Because any small change in the intention of the database may need a lot of changes to be made to the data stored in the database. Extension of the database is performed on the bases of a complete intention, i-e once a database has been defined it is populated with the data of the organization for which the database is created. This population of the database is also called as the extension of the database. Extension is always done according to the rules defined in the internal schema design or the intention of the database.

Effects of changes made to different levels of the database architecture:

We can make changes to the different levels of the database but these changes need very serious consideration before they are made, Changes at different levels of database architecture need different levels of users attention for example a change to the data made for the extension of data will effect only a single record whereas when we make a change to the internal level of the

database the change effects all the stored records, similarly an invalid change in the extension of the database is not that fatal as a change in the intention of the database because a change in the extension of the database is not very hard to undo; incase of a mishap whereas a change of the same magnitude to the intention of the database might cause a large number of database errors (inconsistencies and data loss).

External View (Level, Schema or Model):

This level is explicitly an end user level and presents data as desired by the users of the database. As it is known that the database users are classified on two grounds

- Section of the organization
- Nature of Job of the users

The external level of the database caters to the needs of all the database users starting from a user who can view the data only which is of his concern up-to the users who can see all the data in the database and make all type of actions on that data.

External level of the database might contain a large number of user views, each user view providing the desired features and fulfilling requirements for the user or user group for which it is intended. The restriction or liberty a user or user groups get in his rights is the external view of that user groups and is decided very carefully.

External views are also helpful when we want to display the data which is not place in the database or not stored at all. Example of the first case can be a customer Phone number stored in the database. But when contacting the person it might appear that the area code for that specific user is not stored in the database, in that case we can simply pick up the area or city id of the customer and find the area code for that city from the corresponding Area Codes table.

Another situation may arise when we want to get a student enrolled in an institution and want to make sure that the student qualifies for the minimum required age limit, we will look the database, for the students age but if we have

stored only the date of birth of the student then the age of the student needs to be calculated at that very instance; this can be done very easily in the specific user view and age of the student can be calculated, even the user-view itself can tell use whether the student qualifies for the admission or not.

As the user view is the only entity or the interface through which a user will operate the database or use it so it must be designed in such a way that it is easy to use and easy to manage and self descriptive, also it is easy to navigate through. Also it should not allow the user to get or retrieve data which is not allowed to the user, so the user view should both be a facilitator and also a barrier for proper utilization of the database system.

As the system grows it is possible that a user view may change in structure, design and the access it provides to the users. SO External views are designed and create in way that they can be modified at a later stage without making any changes in the logical or internal views.

In the diagram below we can see two different users working as end users having their own external view; we can see that the same data record is displayed in two entirely different ways.

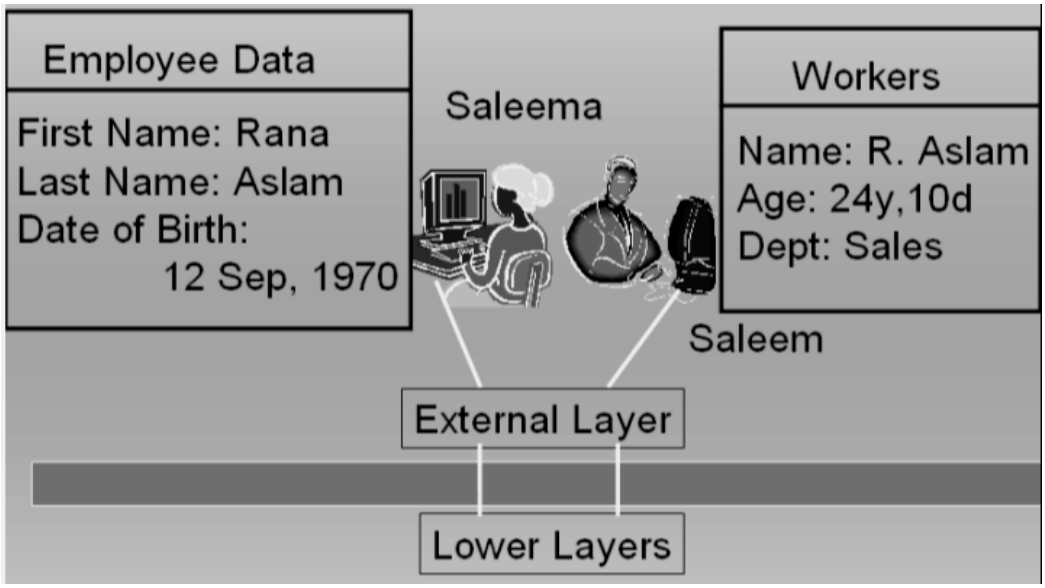


Fig. 2: Mapping between External layer and lower layers

Conceptual or Logical View:

This is the level of database architecture which contains the definition of all the data to be stored in the database and also contains rules and information about that structure and type of that data.

The conceptual view is the complete description of the data stored in the database. It stores the complete data of the organization that is why it is also known as the community view of the database. The conceptual view shows all the entities existing in the organization, attribute or characteristics associated with those entities and the relationships which exist among the entities of the organization.

We can take the example of the customers of a company. Now the conceptual schema will have all the details of the products of the company, retailing stores of the company, products present in the stock, products which are ready to be delivered, salespersons of the company, manager of the company and literally every other thing which is associated with the business of the company in any way.

Now after having all the information we know that the customers buy products from the outlets of the company, thus in such a case a specific customer has a relationship with that specific outlet of the company, or the customer may be represented as having association with the sales person which in-turn has association with the outlet., there may be a number of customers at a certain outlet and also to mange these salespersons there will be one or more managers.

We can see from the above given scenario that all the entities are logically related to each other in way or the other. The conceptual schema actually manages all such relationship and maps these relationships among the member entities. Conceptual schema along-with having all the information which is to be stored in the database stores the definition of the data to be stored. The definition may contain types of data, and constraints on data values etc.

Conceptual schema is also responsible for holding the authorization and authentication information, means that only those people can make use of the database whom we have allowed to make these changes, so therefore it is the task of the DBMS to ensure be checking the conceptual schema that he is authorized to check the data or make any changes to the data.

Conceptual schema as it describes the intention of the database; it is not changed often, because to make a change to the conceptual schema of the database requires lots of consideration and may involve changes to the other views/levels of the database also.

As in the previous example we saw two database users accessing the database and we saw that both of them are having totally different user views. Here when we see in the logical view of the data we can see that the data stored in the database is stored only once and two users get different data from the same copy of data at the underlying conceptual level.

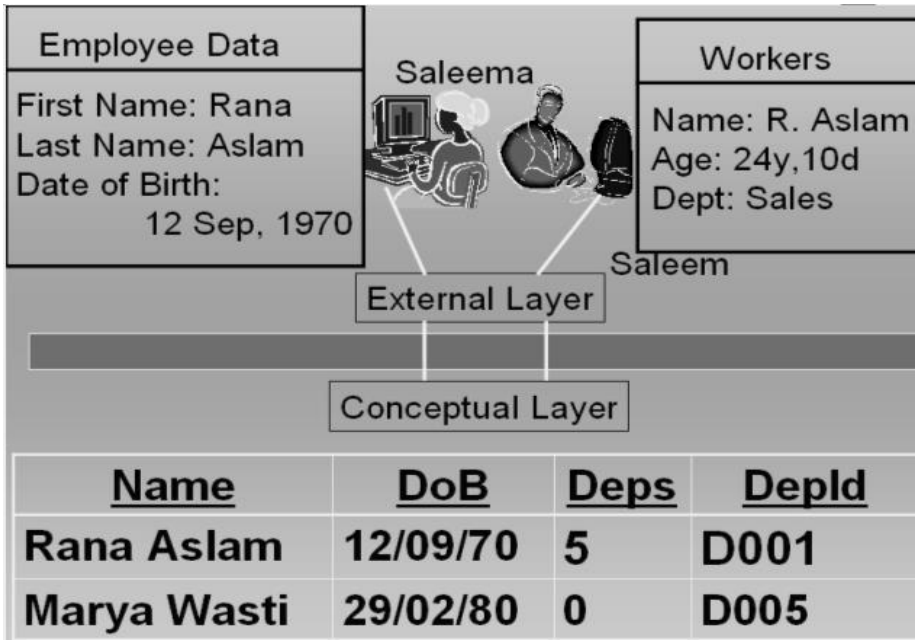


Fig. 3: External and conceptual layers

By summarizing it all we can say that the external view is the view of database system in which user get the data as they need and these database users need

not to worry about the underlying details of the data, all these users have to do is to provide correct requirement information to the DBA or the database designer whoever is designing the database for the system, so that the DBA or the database designer can create the database in such a way that they can fulfill the users requirements using the conceptual schema of the database.

Conceptual view/schema is that view of the database which holds all the information of the database system and provides basis for creating any type of the required user views and can accommodate any user fulfilling his/her requirements.

Exercise:

The data examples that you defined in the exercises of lecture 1, think of the different forms of data at the external and conceptual level. Also try to define mapping between them.

Lecture No. 04

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	4.1.3, 4.1.4
Hoffer	Chapter 2

Overview of Lecture

- Internal Schema of the Database Architecture
- Data Independence
- Different aspects of the DBMS

Internal or Physical View / Schema

This is the level of the database which is responsible for the storage of data on the storage media and places the data in such a format that i t is only readable by the DBMS. Although the internal view and the physical view are so close that they are generally referred to a single layer of the DBMS but there lays thin line which actually separated the internal view from the physical view. As we know that data when stored onto a magnetic media is stored in binary format, because this is the only data format which can be represented electronically, No matter what is the actual format of data, either text, images, audio or video. This binary storage mechanism is always implemented by the Operating System of the Computer. DBMS to some extent decides the way data is to be stored on the disk. This decision of the DBMS is based on the requirements specified by the DBA when implementing the database. Moreover the DBMS i tself adds information to the data which is to be stored. For example a DBMS has selected a specific File organization for the

storage of data on disk, to implement that specif ic fi le system the DBMS needs to create speci fic indexes. Now whenever the DBMS will attempt to retrieve the data back form the f ile organization system it will use the same indexes information for data retrieval. This index information is one example of additional information which DBMS places in the data when storing it on the disk. At the same level storage space utilization if performed so that the data can be stored by consuming minimum space, for this purpose the data compression can be performed, this space optimization is achieved in such a way that the performance of retrieval and storage process is not compromised. Another important consideration for the storage of data at the internal level is that the data should be stored in such a way that i t is secure and does not involve any security risks. For this purpose different data encryption algorithms may be used. Lines below detail further tidbits of the internal level.

The difference between the internal level and the external level demarcates a boundary between these two layers, now what is that difference, i t in fact is based on the access or responsibility of the DBMS for the representation of data. At the internal Level the records are presented in the format that are in match with schema definition of the records, whereas at the physical level the data is not strictly in record format, rather it is in character format., means the rules identified by the schema of the record are not enforced at this level. Once the data has been transported to the physical level i t is then managed by the operating system. Operating system at that level uses i ts own data storage utilities to place the data on disk.

Inter Schema Mapping:

The mechanism through which the records or data at one level is related to the changed format of the same data at another level is known as mapping. When we associate one form of data at the external level with the same data in another form is know as the external/conceptual mapping of the data. (We have seen examples of external/conceptual mapping in the previous lecture) In the same way when data at the conceptual level is correlated with the same data at the internal level, this is called the conceptual/Internal mapping.

Now the question arises that how this mapping is performed. Means how is it possible to have data at one level in date format and at a higher level the same data show us the age. This hidden mechanism, conversion system or the formula which converts the date of birth of an employee into age is performed by the mapping function and

it is defined in the specific ext/ con mapping, for example, when the data at the conceptual level is presented as the age of the employee is done by the external schema of that specific user. Now in this scenario the ext/ con mapping is performing the mapping with the internal view and is retrieving the data in desire format of the user. In the same way the mapping between an internal view and conceptual view is performed.
The figure below gives a clear picture of this mapping process and informs where the mapping between different levels of the database is performed.

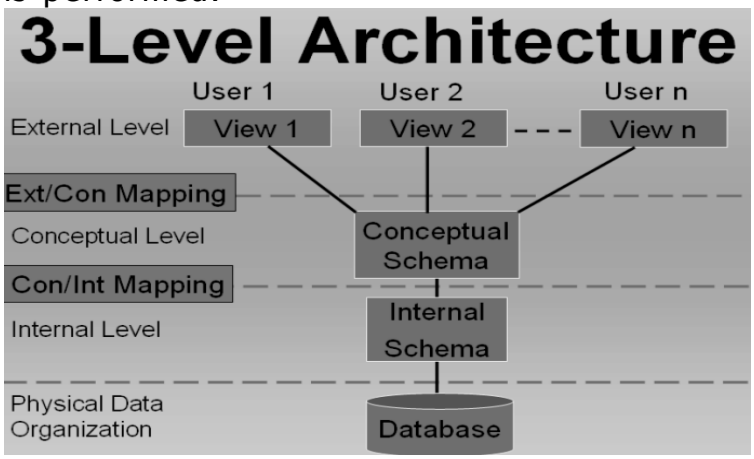


Fig:1: Mapping between External/Conceptual and Conceptual/Internal levels

In Figure-1 we can see clearly where the mapping or connectivity is performed between different levels of the database management system. Figure-1 is showing another very important concept that the internal layer and the physical layers lie separately the Physical layer is explicitly used for data storage on disk and is the responsibility of the Operating system. DBMS has almost no concern with the details of the physical level other than that i t passes on the data along- with necessary instructions required to the store that data to the operating system.

Figure-2 on the next page shows how data appears on different levels of the database architecture and also at that of physical level. We can clearly see that the data store on the physical level is in binary format and is separate from the internal view of data in location and format. Separation of the physical level from the internal level is of great use in terms of efficiency of storage and data retrieval.

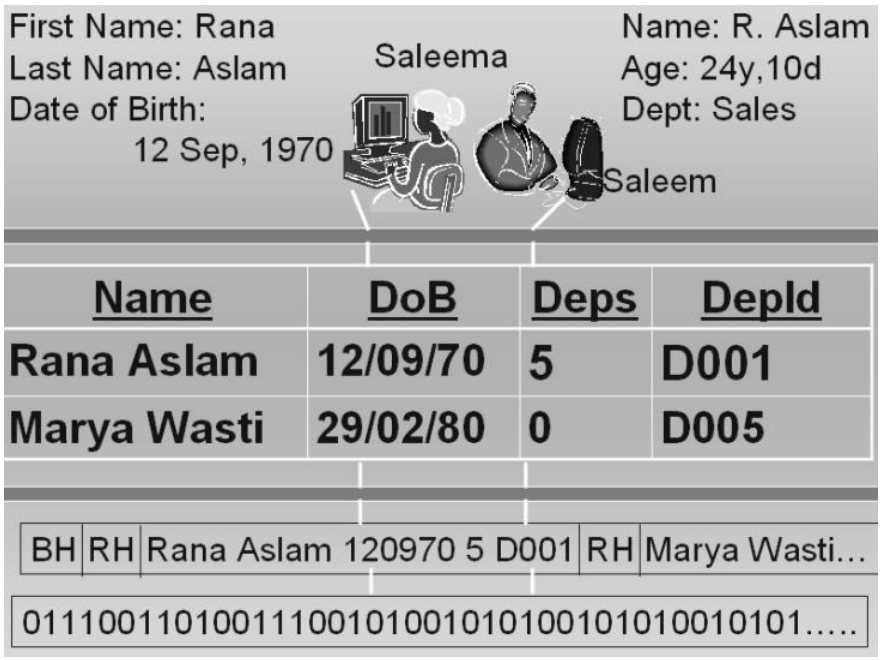


Fig: 2. Representation of data at different levels of data base Architecture and at the physical level at bottom

At the internal level we can see that data is prefixed with Block Header and Record header RH, the Record header is prefixed to every record and the block header is prefixed to a group of records; because the block size is generally larger than the record size, as a result when an application is producing data it is not stored record wise on the disk rather block wise which reduces the number of disk operations and in- turn improves the efficiency of writing process.

Data Independence:

Data Independence is a major feature of the database system and one of the most important advantages of the Three Level Database Architecture. As i t has been discussed already that the fi le processing system makes the application programs and the data dependent on each other, I-e if we want to make a change in the data we will have to make or reflect the corresponding change in the associated applications also.

The Three Level Architecture facilitates us in such a way that data independence is automatically introduced to the system. In other words we can say the data independence is major most objective of the Three Level Architecture. If we do not have data independence then whenever there will be a change made to the internal or

physical level or the data accessing strategy the applications running at the external level will demand to be changed because they will not be able to properly access the changed internal or physical levels any more. As a result these applications will stop working and ultimately the whole system may fail to operate.

The Data independence achieved as a result of the three level architecture proves to be very useful because once we have the data , database and data applications independent of each other we can easily make changes to any of the components of the system, without effecting the functionality and operation of other interrelated components.
Data and program independence is on advantage of the 3 -L architecture the other major advantage is that ant change in the lower level of the 3-L architecture does not effect the structure or the functionality on upper levels. I- e we get external/conceptual and conceptual/internal independence by the three levels Architecture.
Data independence can be classified into two type based on the level at which the independence is obtained.

- Logical Data Independence
- Physical Data Independence

Logical data independence
Logical data independence provides the independence in a way that changes in conceptual model do not affect the external views. Or simply i t can be stated at the Immunity of external level from changes at conceptual level.
Although we have data independence at different levels, but we should be careful before making a change to anything in database because not all changes are accepted transparently at different levels. There may be some changes which may cause damage or inconsistency in the database levels. The changes which can be done transparently may include the following:

- Adding a file to the database
- Adding a new field in a f ile
- Changing the type of a specific field

But a change which may look similar to that of the changes stated above could cause problems in the database; for example: Deleting an attribute from the database structure,

This could be serious because any application which is using this attribute may not be able to run any more. So having data independence available to us we still get problem after a certain change, i t means that before making a certain change its impact should also be kept in mind and the changes should be made while remaining in the limits of the data independence.

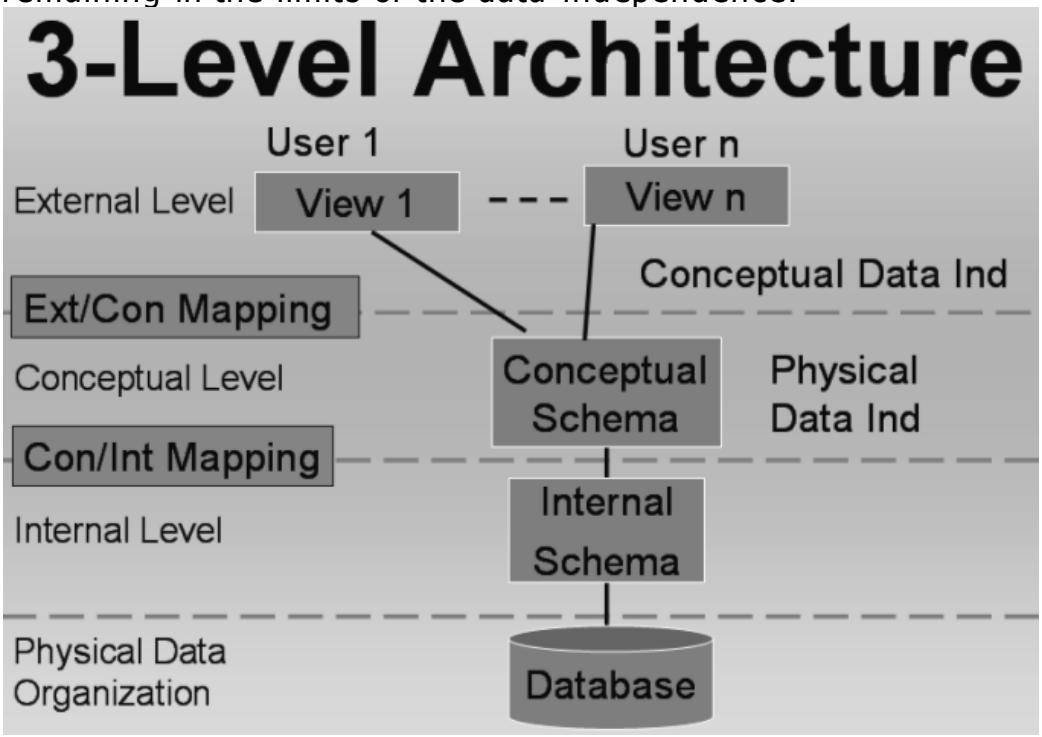


Fig:3 . The levels where the Conceptual and Physical data independence are effective

Physical Data Independence

Physical data independence is that type of independence that provides us changes transparency between the conceptual and internal levels. I-e the changes made to internal level shall not affect the conceptual level. Although the independence exist but as we saw in the previous case the changes made should belong to a specific domain and should not exceed the liberty offered by the physical data independence. For example the changes made to the file organization by implementing indexed or sequential or random access at a later stage, changing the storage media, or simply implement a different technique for managing fi le indexes or hashes.

Functions of DBMS

- Data Processing

- A user accessible Catalog
- Transaction Support
- Concurrency Control Services
- Recovery Services
- Authorization Services
- Support for Data Communication
- Integrity Services

DBMS lies at the heart of the course; i t is the most important component of a database system. To understand the functionality of DBMS it is necessary that we understand the relation of database and the DBMS and the dissection of the set of functions the DBMS performs on the data stored in the database.

Two important functions that the DBMS performs are:

User management
Data Management

The detailed description of the above two major activities of DBMS is given below;

- Data Processing

By Data management we mean a number of things i t may include certain operations on the data such as: creation of data, Storing of the data in the database, arrangement of the data in the databases and data-stores, providing access to the data in the database, and placing of the data in the appropriate storage devices. These action performed on the data can be classified as data processing.

- A User Accessible Catalog

DBMS has another very important task known as access proviso to catalog. Catalog is an object or a place in the DBMS which stores almost all of the information of the database, including schema information, user information right of the users, and many more things about the database. Modern relational DBMS require that the Administrative users of the database should have access to the catalog of the database.

- Transaction Support

DBMS is responsible for providing transaction support. Transaction is an action that is used to perform some manipulation on the data stored in the database. DBMS is responsible for supporting all the required operations on the database, and also manages the

transaction execution so that only the authorized and allowed actions are performed.

- Concurrency Support

Concurrency support means to support a number of transactions to be executed simultaneously, Concurrency of transactions is managed in such a way that if two or more transactions is making certain processing on the same set of data, in that case the result of all the transactions should be correct and no information should be lost.

- Recovery Services

Recovery services mean that in case a database gets an inconsistent state to get corrupted due to any invalid action of someone, the DBMS should be able to recover i tself to a consistent state, ensuring that the data loss during the recovery process of the database remains minimum.

- Authorization Services

The database is intended to be used by a number of users, who will perform a number of actions on the database and data stored in the database, The DBMS is used to allow or restrict different database users to interact with the database. It is the responsibility of the database to check whether a user intending to get access to database is authorized to do so or not. If the user is an authorized one than what actions can he/she perform on the data?

- Support for Data Communication

The DBMS should also have the support for communication of the data indifferent ways. For example if the system is working for such an organization which is spread across the country and i t is deployed over a number of offices throughout the country, then the DBMS should be able to communicate to the central database station. Or if the data regarding a product is to be sent to the customers worldwide i t should have the facility of sending the data of the product in the form of a report or offer to its valued customers.

- Integrity Services

Integrity means to maintain something in i ts truth or originality. The same concept applies to the integrity in the DBMS environment. Means the DBMS should allow the operation on the database which are real for the specific organization and i t should not allow the false information or incorrect facts.

DBMS Environments:

- Single User
- Multi-user
  - Teleprocessing
  - File Servers
  - Client-Server

Single User Database Environment

This is the database environment which supports only one user accessing the database at a specific time. The DBMS might have a number of users but at a certain time only one user can log into the database system and use i t. This type of DBMS systems are also called Desktop Database systems.

Multi-User Database systems

This is the type of DBMS which can support a number of users simultaneously interacting with the database in different ways. A number of environments exist for such DBMS.

- Teleprocessing

This type of Multi user database systems processes the user requests at a central computer, all requests are carried to the central computer where the database is residing, transactions are carried out and the results transported back to the terminals (literally dumb terminals). It has become obsolete now.

- File Servers

This type of multi-user database environment assumes another approach for sharing of data for different users. A fi le server is used to maintain a connection between the users of the database system. Each client of the network runs its own copy of the DBMS and the database resides on the f ile server. Now whenever a user needs data from the file server i t makes a request the whole file containing the required data was sent to the client. At this stage i t is important to see that the user has requested one or two records from the database but the server sends a complete file, which might contain hundreds of records. Now i f the client after making the desired operation on the desired data wants to write back the data on the database he will have to send the whole file back to the server, thus causing a lot of network overhead. The Good thing about this approach is that the server does not have lots of actions to do rather it remains idle for lots of the time in contrast with that of the teleprocessing systems approach.

- Client-Server

This type of multi- user environment is the best implementation of the network and DBMS environments. It has a DBMS server machine which runs the DBMS and to this machine are connected the clients having application programs running for each user. Once a users wants to perform a certain operation on data in the database i t sends i ts requests to the DBMS through its machine's application software; the request is forwarded to the DBMS server which performs the required operation on data in the database stored in the dame computer and then passes back the result to the user intending the result. This environment is best suited for large enterprises where bulk of data is processed and requests are very much frequent.

This concludes the topics discusses in the lecture No4 .In the next lecture Database application development process will be discussed

Exercises:

- Extend the format of data from the exercise of previous lecture to include the physical and internal levels. Complete your exercise by including data at all three levels
- Think of different nature of changes at all three levels of database architecture and see, which ones will have no effect on the existing applications, which will be adjusted in the inter-schema mapping and which will effect the existing applications.

Lecture No. 05

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	2.3.2, 2.4

Overview of Lecture

- Database Application Development Process
- Preliminary Study of System
- Tools used for Database system Designing
- Data Flow Diagrams
- Different types of Data flow Diagram

Database design and Database Application design are two almost similar concepts, form the course point of view it is worthwhile to mention that the course is mainly concerned with designing databases and it concentrates on the activities which are performed during the design of database and the inner working of the database. The process that will be discussed in this lecture for development of database is although not a very common one, but it specifies all the major steps of database development process very clearly. There exist many ways of system and database development which are not included in the scope of this course. But we will see only those portions of the other processes which are directly related with the design and development of database.

Database Application development Process includes the Following Stages or steps:

- Database Design
- Application Programs
- Implementation

These three steps cannot always be considered as three independent steps performed in a sequence or one after another. Rather, they occur in parallel, which means that from a certain point onward the application programs development may run in parallel with the database design stages, specially the last stages of the database design. Similarly while the design phases of the database are in progress, certain phases of the application programs can also be initiated, for example, the initial study of the screens' format or the reports layout. The database design process that we are going to discuss in this course does not take these steps independently and separately, and since the major concern of this course is the design stages of the database, it concentrate only on those.

- Database Design:

This part of the database application development process is most important process with respect to the database application development, because the database is something that will hold the organizations' data, in case the design of the database is not correct or is not correctly reflecting the situations or scenarios of the organization then it will not produce correct result, or even just produce errors in response to certain queries. So this portion of the database design is given great attention when designing a database application.

Database Development Process

The database development process means the same thing that we have mentioned as database application development process. Rather than discussing three stages of database application development separately, the steps given in the database development process include steps that cover all three phases mentioned for the database application development process.

Preliminary Study:

Design of database is carried out in a number of steps; these steps play important role in the design process and need to be given proper attention First Phase of the database development process is the Preliminary Stage, which is based on the proper study of the system. It means that all the parts of the systems, or the section of the subject

organization for which we intend to develop the system must be studied. We should find the relation or interaction of different section of the organization with each other and should understand the way information flows between different sections of the organization. Moreover it should also be made clear that what processing is performed at each stage of the system.

- Requirement Analysis:

Once we have investigated the organization for its different sections and the way data flows between those sections. Detailed study of the system is started to find out the requirements of each section. This phase is the detailed study of the system and its functionality decisions made at this stage decide the overall activity of the organization. Requirements of one section of the organization are fulfilled in such a way that all the sections in the organization are supporting each other, for example we can say that the results produced by the processing taking place at one section are used as input for another section. All the users of the systems are interviewed and observed to pinpoint and precisely define the activities taking place in the different section of the organization.

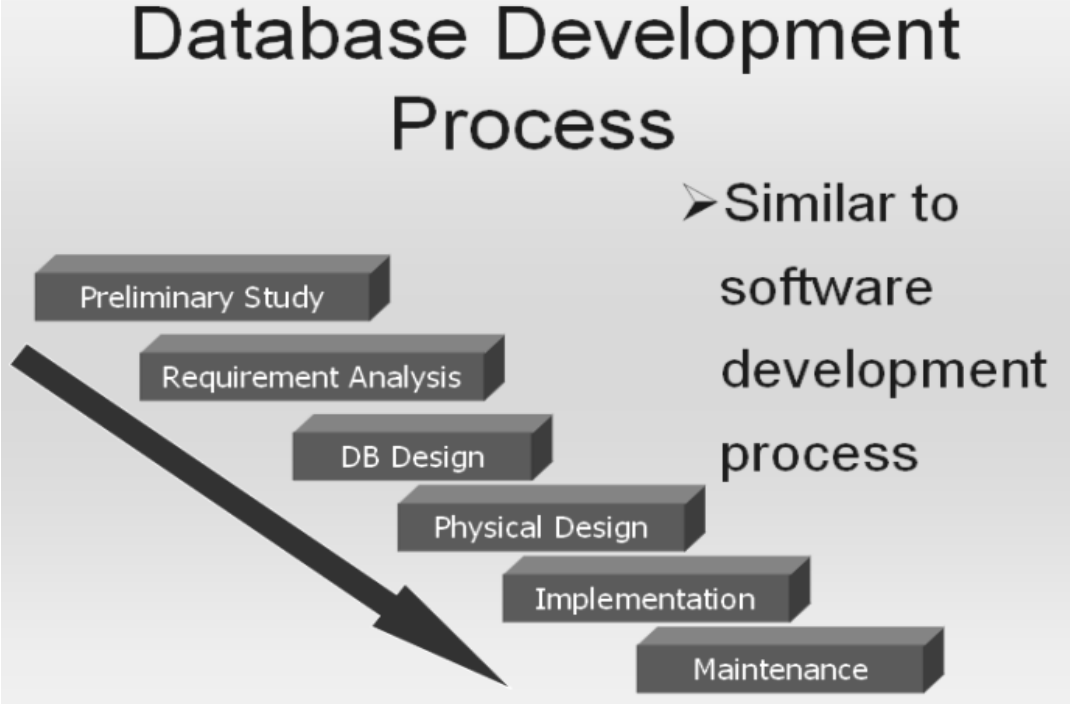


Fig: 1. Database Development process

- Database Design:

Third stage in the database development process is the database design; this is a rather technical phase of the process and need handsome skill as a Database Administrator. This is the phase where the logical design of the database is created and different schemas for the database are created logically. Entities are identified and given attributes, relationships are built and different types of entity mappings are performed.

- Physical Design

This is the phase where we transform our logical design into a Physical design by implementing the designed database onto a specific DBMS; the choice of the DBMS is made on the basis of requirements and the environment in which the system will operate. Implementing a database on a specific DBMS is very important because it involves the major financial investment of the organization, and can not be reverted in case a selected DBMS in not capable of providing the desired efficiency.

- Implementation:

This phase is specific to writing the application programs needed to carry out different activities according to use requirements. Different users may have different requirements of the data in the database, so the number of application programs is not known or fixed for all the organizations, it may vary for different organizations.

- Maintenance of the Database System:

Maintenance means to fine tune the system and check that the designed applications systems are fulfilling the purpose for which they are meant. Also this phase may involve designing any new application for the enhancement of the system. Or an already working application may need to be updated or modified to remove any errors or to add some functionality in the system. The phases involved in the development of the database application are expressed graphically in Figure-1.

All these stages are necessary and must be given the necessary attention at each level to get properly working and good system design and a better working environment.

Database Development Process: Approach 2

There are other development processes also with some of the stages or steps modified as compared to the model we have just studied. Such and alternative is given in the Figure-2 below. In this design process we see some of the design stages which existed in the previous designing steps but some of the stages are modified or merged with others to get more precise result or to distinguish different separate design phases. In this process of designing; the following steps exist:

- Analyze User Environment
- Develop Conceptual Model
- Map Conceptual Model to Logical
- Choose DBMS
- Develop Physical Design
- Implement System
- Test System
- Operational Maintenance

- Analyze User Environment

This is same step as we discussed while discussing the previous designing process

- Develop Conceptual Model

Next stage in this process model is the development of conceptual model or schema Here we actually transform the studied and analyzed information into the conceptual design of the database, this stage may also be connected with the requirement analysis phase, as expressed in the diagram by showing an arrow from this stage back to the first stage.

- Map Conceptual Model to Logical Model

Third stage is the mapping of the developed conceptual model to the logical model of the database, means at this stage the schema rules are defined and identified for general database structures.

- Choose DBMS

Once the mapping of the conceptual and logical model is done, the decision for the use of DBMS is made; again we refer to the previous model for selecting of the DBMS and will take care of all the necessary requirements of the environment before making a decision.

- Develop Physical Design

Once we have selected a DBMS, the logical design is then transformed into physical design. This also includes considering many other decisions, like, data type allocation, indexes to be created, file organizations, etc. Physical database design is achieved by using the DBMS specific rules for schema definition and all the facilities provided by the DBMS,

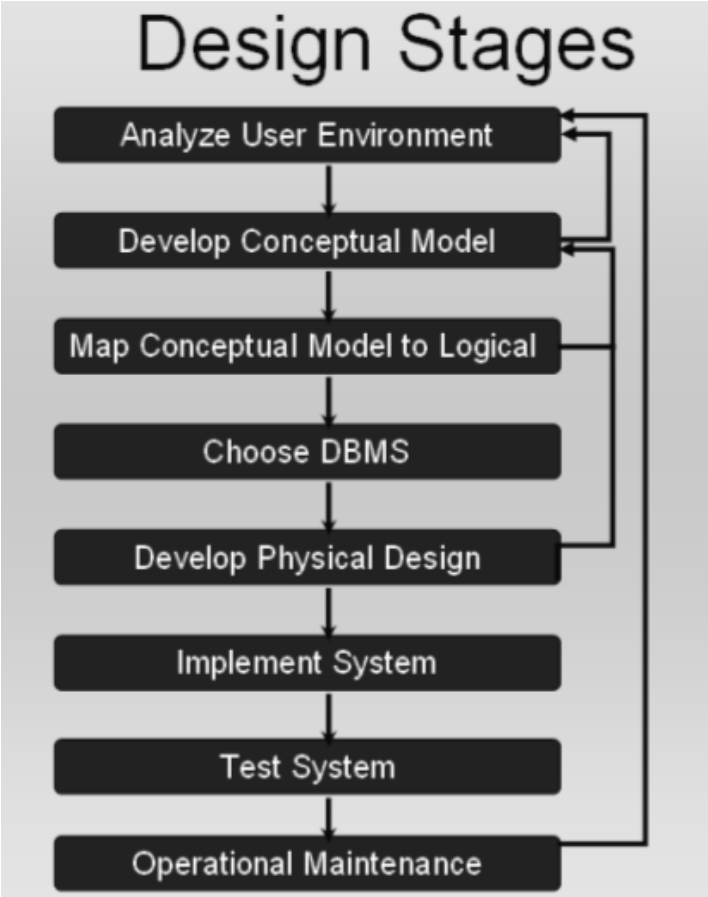


Fig: 3. Database Development Stages. (Second Approach)

- Implement System

This stage is also similar to the one described earlier, i.e., designing the application for different users and user groups of the organization.

- Test System

Testing is important in the sense that an application may be producing incorrect results, and this incorrectness may lead to the inconsistency of the system. So when a system

design is complete, once it is implemented it must be tested for proper operation and all the modules must be checked for their correctness. Whether the system modules are important or not because the result of the system is mostly dependent on the proper the functionality of all database applications and modules.

- Operational Maintenance:

Maintenance means to check that all parts of the system are working and once the testing of the system is completed the periodic maintenance measure are performed on the system to keep the system in working order.

Tools Used for Database System Development:

Why tools are used?

Tools are used for describing the design process in standard ways. If there is no standardized tool available for designing a specific systems; Then everyone will have to use its own design notation, and a notation used by one designer may not be understandable to the another one. This misunderstanding can be more drastic if both the designers are working for the development of the same system. Tools can also help the designer and the user to mutually agree on a specific design.

Data Flow Diagrams:

The most common tool used for deigning database systems is Data Flow Diagram. It is used to design systems graphically and expresses different system detail in different DFD levels.

DFDs show the flow of data between different processes o a specific system.

DFDs are simple, and hide complexities.

DFDs are Descriptive and links between processes describe the information flow.

- Limitation of DFDs

They do not provide us a way of expressing decision points.

DFDs are focused on flow of information only.

- Symbols used in DFD:

There are a limited number of symbols which are used for design process in DFDs.

- DATAFLOW:

The purpose of the dataflow in a DFD is to express the flow of information from one entity to another entity in the system

Data flows are pipelines through which packets of information flow.

Arrows are labeled with name of the data that moves through them. Figure-4 below show the Dataflow diagram

Fig: 4. Dataflow Symbol

- DATA STORE:

Data store is a repository for the storage of the data. When in a system the data is to be permanently stored somewhere for future reference or use the DATASTORE is used for this purpose. It is express with a rectangle open on right width and left width of the rectangle drawn with double lines.

Data in the DATASTORE is held sometimes for processing purposes also i-e it may not be a permanent data store.. Name of the DATASTORE is a noun which tells the storing location in the system. Or identifies the entity for which data is stored. Figure-5 shows a data store.

Fig: 5. Data store

- Processes:

Processes are expressed with ovals or rounded rectangles. Processes are used to express the transformation of incoming dataflow into outgoing dataflow. Process symbols are used for whatever is the action taking place and whatever is the magnitude or complexity of the action. Simply stating when data is transformed from one form into another the process symbol is used. Figure-6a and Figure-6b show two different shapes used for presenting process in DFD.

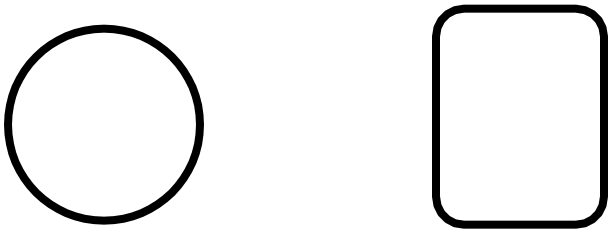


Fig: 6a Process Fig-6b

- DFD-Process:

In DFD processes are numbered for expressing their existence at a certain level in the system.

Fig: 7. Numbered DFD Processes

- External Entities:

These are the entities interacting with the system in any of two different ways. They may be either receiving the data from the system, or may be producing the data for the system to consume.

Shape used to express external entities is rectangle. The shape for external entity is shown in Figure-8.

Fig: 8. External Entity

- Collector:

This DFD shape is used to express several dataflow connections terminating at a single location. Collector is used to show the convergence of data to a single point. Fig 9a shows the Collector symbol and Fig 9b show a collector symbol acting as a sink for multiple data flows.

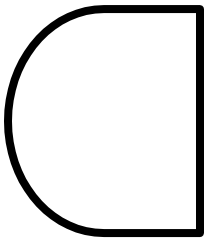


Fig: 9a Collector

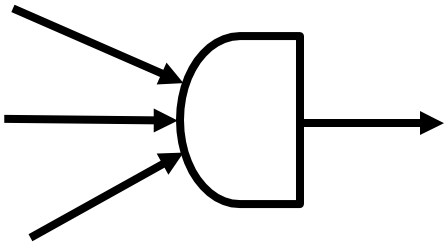


Fig 9b. Collector with Multiple Dataflow

- Separator:

The dataflow symbol which is used for separating data from a single source to multiple sinks is known as a separator.

Figure 10a show the presentation of separator and the figure 10b shows the separator as it may appear in a DFD.

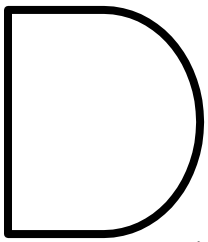


Fig: 10a Separator

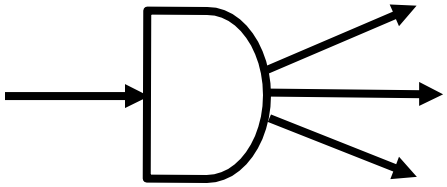


Fig 10b. Separator with Multiple Dataflow

- Ring Sum Operator:

This operator is used when data from a source process can flow to one of the mentioned sinks. For this purpose the symbol used is displayed in Figure: 11a and its presentation in a DFD is expressed in Figure-11b.

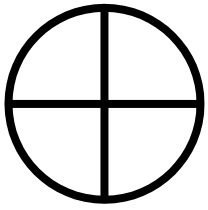


Fig: 11a Ring sum operator

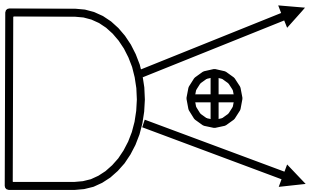


Fig 11b. Separator with Ring sum operator

- AND Operator:

This operator is used when data from a source process must flow to all the connected sinks. For this purpose the symbol used is displayed in Figure: 12a and its presentation in a DFD is expressed in Figure-12b.

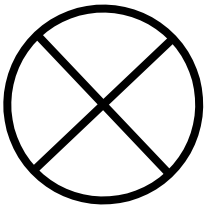


Fig: 12a AND operator

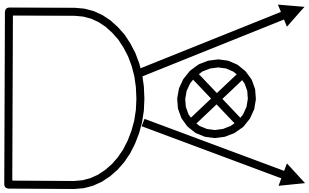


Fig 12b. Separator with AND operator

Types of DFD

- Context diagram
- Level 0 diagram
- Detailed diagram

- Context Diagram:

This is the level of DFD which provides the least amount of details about the working of the system. Context DFDs have the following properties:

They always consist of single process and describe the single system. The only process displayed in the CDFDs is the process/system being analyzed. Name of the CDFDs is generally a Noun Phrase.

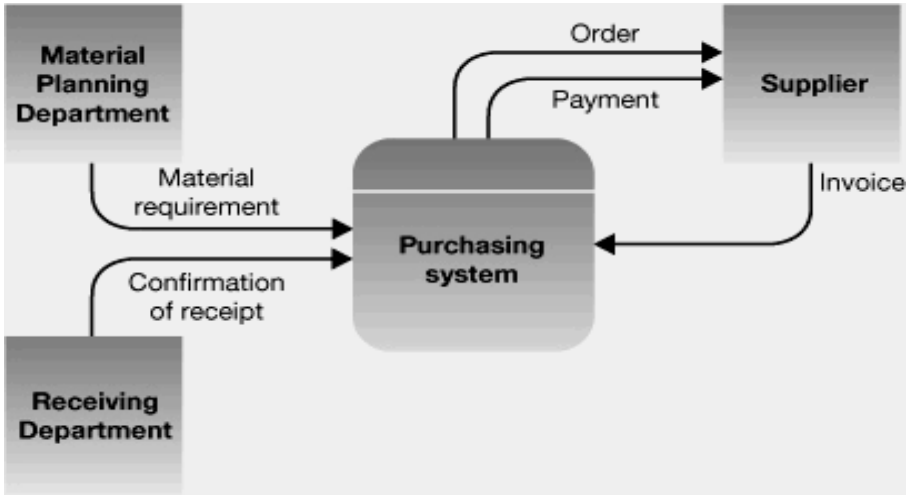


Fig: 13a. Example Context DFD Diagram

No System details are shown in the Contexts DFDs just context is shown. Input and output from and to the process are shown and interactions are shown only with the external entities. An example DFD at context level is shown in Figure: 13a and 13b.

In the context level DFDs no data stores are created. Ant dataflow from external entities are only directed toward the purported system and vice versa, no communication is show between external entities themselves.

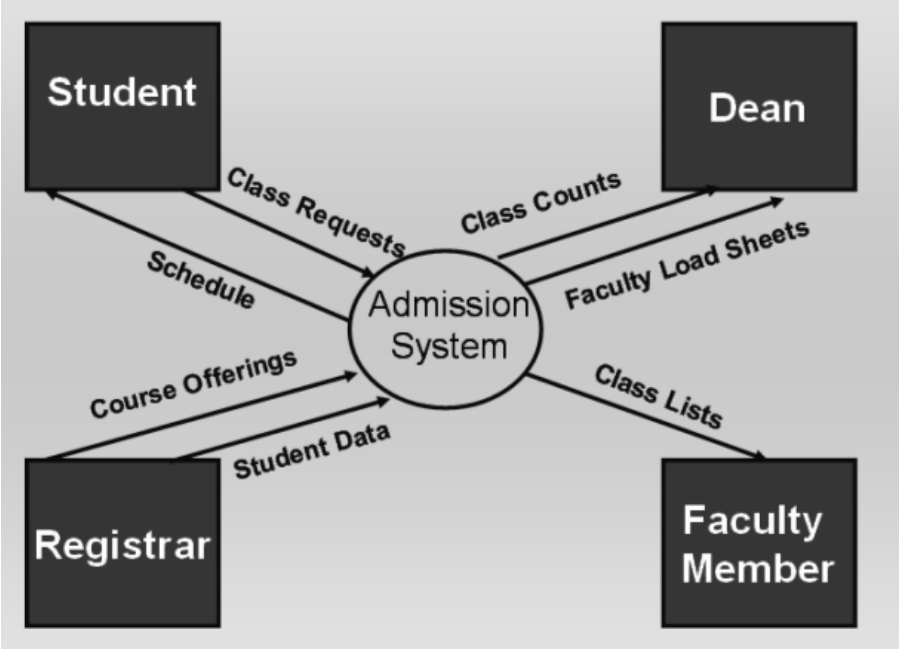


Fig: 13b. Example Context DFD Diagram

- Level 0 Data Flow Diagrams:

The level 0 Diagram in the DFD is used to describe the working of the whole system. Once a context DFD has been created the level zero diagram or level 'not' diagram is created. The level zero diagram contains all the apparent details of the system. It shows the interaction between a numbers of processes and may include a large number of external entities. At this level it is the duty of the designer to keep a balance in describing the system using the level 0 diagram. Balance means that he should give proper depth to the level 0 diagram processes. Because placing too much details and showing all of the miniature processes in the level 0 diagrams makes it too much complex. On the other hand it is also not recommended to just ignore even larger processes of the system, because in such a case although the level 0 DFD will become simple but now we will have to create large number of detail DFDs. So a balance in describing the system should be kept so that the depth of the Level 0 DFD is manageable.

- Steps in creating the level 0 DFD

1. Identify distinct modules of the system for which to create the DFD

2. Create DFDs for all the modules one by one to show the internal functionality of the system.
3. Once DFD for the distinct modules of the system have been created, establish link between different DFDs where required by either connecting the entities of the system, processes of the system or the data stores in different DFDs.
4. Now comes to the stage of placing the numbers on processes. As we know that the level 0 diagram encompasses a large number of smaller systems, ant is a combination of a number of context DFDs. In level 0 diagram a process when it has a lot of details, it is not explained further in the level 0, and rather it is postponed for the detailed diagram.
   In the detailed Data Flow and is given a number. Numbering processes is based on a specific notation, in the level 0 diagrams only left half or the portion before the decimal point is valid but in the detailed diagram when a complex process is expressed further its sub processes are number like 1.0, 1.1, and 1.2 and so on.

Lecture No. 06

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	Section 2.4

Overview of Lecture

- Detailed DFD Diagrams:
- Database Design Phase
- Data Models
- Types of Data Models
- Types of Database Designs

Detailed Data Flow Diagram:

This Type of the Data flow diagrams is used when we have to further explain the functionality of the processes that we showed briefly in the Level 0 Diagram. It means that generally detailed DFDS are expressed as the successive details of those processes for which we do not or could not provide enough details.

The symbols and other rules regarding the detailed DFD are same as are in other types of DFDs. The special features associated with this diagram are that, one, it is optional, that is, it is created for only those processes from the level 0 diagram for which we want to show the details. For a small sized system we may not need to develop even a single detailed DFD, since the level 0 diagram might be covering it sufficiently. Second specific characteristic of the detailed DFD is its processes' numbering. Numbering of processes in the detailed DFD is done on the basis of numbering of the particular process in level 0 diagrams whose sub-processes are being included in the detailed DFD. For example, a specific process which was numbered in the level 0 diagram as 1.0 or 1 may have a number of sub-processes since we did not represent the process 1.0 in detail in level 0 diagrams. So in the detailed dataflow diagram we create sub-processes of that process and then number all the sub processes of that specific process as the sublets of the process.

Numbering of such sub processes is done as 1.1, 1.2, and 1.3… for first second and third sub-processes of the process 1.0 respectively. The phenomenon of creating sub-processes does not end at creating a few sub-processes for a specific process shown at level 0 diagrams. Rather it may continue deeper if there is requirement for further explanation of the any process or sub-processes. In such a case when we create sub-process of a sub-process 1.2 then the numbering is done in further extension of that specific sub processes number and example of such a numbering process is 1.2.1, 1.2.2, 1.2.3,…

Another point that is worth mentioning here is that we call processes in the detailed DFDs as sub-processes, but they are sub-processes only in reference to the process whose details they are explaining otherwise they are just like processes; transforming some input data into some form of output. The sub-processes may be performing relatively small amount of operations, still they are processes.

Maximum Number of Process in a DFD should not be very huge. Having a moderate number for a detailed DFD is also recommended because it adds clarity to our detailed data flow diagram. For clarity propose it is good to have a maximum of 7 or 9 processes in one detailed DFD. Moreover all the processes, sub processes, data stores, entities data flows and all other components of the DFD must be named properly, so that anyone who is using this DFD should be able to understand the DFD easily.

In all the levels of DFD it must be considered that all the processes have data inputs as well as data outputs. Data being sent to one process should be processed so that it changes its form and transforms from one form to another.

When creating a detailed diagram the data inputs and data outputs must be in coincidence, mean in both the diagrams the data input to a process and data output in the form of data flows must be same.

Data Dictionary

A database that containing data about all the databases in the database system. Data dictionaries store all the various schema and file specifications and their locations. They also contain information about which programs use which data and which users are interested in which reports.

Types of Data Dictionaries:

- Integrated

There are basically two types of data dictionaries which are available for use by a DBMS, with respect to their existence.

The first type of data dictionary in this context is the integrated data dictionary. Such a data dictionary is place embedded into the database system, and is created by the DBMS for its usage under the directions and requirements provided by the DBA

As the DBMS needs to talk with the "three level architecture" of database and mapping information along with all the database design information lies in the database schema. The DBMS uses the data dictionary to access the database at each layer or model, for this purpose the data dictionary of any type can be used but the integrated data dictionary is far more efficient than any free standing data dictionary because an integrated data dictionary is created by the DBMS itself and uses the same data accessing techniques etc.

- Free Standing

Second type of data dictionary is free standing data dictionary create by any CASE tool and then attached to the database management systems. A number of case tools are available for this purpose and help user designing the database and the database applications as well in some modern forms of the CASE tools.

Cross Reference Matrix

This is a tool available in the data dictionary and helps us in finding entities of the database and their associations. CRM is developed at the designing stage of the database; we can say that at the time of creation of the user views of reports for certain users we identify the material required by the users. In the cross reference matrix, on the Y axis we specify the accessible components of the database such as transitions, reports, or database objects and on the x axis we specify the attributes that will be accessed in the corresponding accessed object.

Now the matrix gets a shape of two dimensional arrays on which we have accessible objects of the database and on the other hand we have the elements which are available for access through those objects. Then whichever data item is accessible through a certain object we place a tick on the intersection of that row and column and thus we can easily identify the deferent items accessed in different reports.

	Transcript	SemResCard	AttendSheet	ClassResSub	cClassResult
courseNan	√	√		√	
cumulative	√	√			√
date	√	√		√	√
fatherNam	√	√			
finalMarks				√	
grade				√	√
grdPoint	√	√		√	√
marks	√	√			
midTerm				√	
programNa	√	√	√	√	√
semesterG	√	√			√
semesterN	√	√	√	√	√
semName			√	√	√
session	√	√			
sessMarks				√	
stName	√	√			
stNames			√	√	
stRegistrat	√	√			

Table 1: An example cross reference matrix

The cross reference matrix shown in table 1 lists different attributes against different reports required by different user groups of an exam system. Rows in this matrix contain different attributes and the columns contain different reports. Now the tick mark in the cells represents the use or presence of attributes in different reports. This matrix represents, on one side, the relative importance or use of different attributes. On the other hand it also helps to identify different entity types and their defining attributes. The attributes that are represented collectively on one or more reports are candidates of combining into a single entity type. Although it is necessary that attributes appearing together should be grouped into same entity type, but still they are candidates for combining into the one.

Data Dictionary in not very necessary for using such a cross reference matrix, instead for relatively small systems it can be created manually.

Outcome of the Analysis Phase

In the preliminary study phase, database designers collect information about the existing system from the users of the system. For this purpose they may interview different users or concerned persons, or they may distribute questionnaires among different users and ask them to fill them in and later may use these questionnaires in the analysis phase. Designers represent their understanding of the working of existing system in the form of DFDs and discuss it with the users to make it sure that they have understood all details of the existing system and the requirements of different users groups.

The DFDs are input to the analysis phase, where designers analyze the requirements of the users and establish the procedure to meet those requirements. From the database perspective, in the analysis phase designers have to identify the facts or data that is required to be stored in order to fulfill the users' requirements. For this purpose they may use some CASE tools, like cross reference matrix. Generally, in the analysis phase, designers prepare a draft or initial database design that they ultimately finalize in the next phase, that is, the database design phase. So in short we can say, that DFDs are the output of the preliminary phase and are input to the analysis phase. The initial design or a draft form of design (generally in entity-relationship data model) is the output of the analysis phase and input to the design phase. In the design phase, then you finalize the design.

The sequence of the activities mentioned above is not much important, however, the activities mentioned are important and must be performed in order to have a correct database or database application design. In the following lectures, we are going to study different tools that are used in the design phase, that is, the data models. We will be studying, both, the data models and their implementation in the database design phase.

Database Design Phase

Database design phase follows the analysis phase. Before starting the discussion on the design activity, it will be wise if we clearly understand some basic concepts that are frequently used in this phase.

- Database Design /Database Model

These terms can be used interchangeably for the logical structure of the database. The database design/model stores the structure of the data and the links/relationships between data that should be stored to meet the users' requirements. Database design is stored in the database schema, which is in turn stored in the data dictionary.

- Database Modeling

The process of creating the logical structure of the database is called database modeling. It is a very important process because the designing of the application provides us the basis for running our database system. If the database is not designed properly the implementation of the system can not be done properly. Generally the design of the database is represented graphically because it provides an ease in design and adds flexibility for the understanding of the system easily.

Data Model

Data model is a set or collection of construct used for creating a database and producing designs for the databases. There are a few components of a data model:

- Structure:

What structures can be used to store the data is identified by the structures provided by the data model structures.

- Manipulation Language

For using a certain model certain data manipulations are performed using a specific language. This specific language is called data manipulation language.

- Integrity Constraints

These are the rules which ensure the correctness of data in the database and maintain the database in usable state so that correct information is portrayed in designing the database. Generally these components are not explicitly defined in data models, they may be available in some of the modern DBMSs but in traditional and general model, these may not be available.

Significance of the Data Model

Data model is very important tool because it is something which is sued for designing the database for a DBMS and no DBMS can exist independent of any data model, now if we use a specific DBMS but are not sure about the data model it uses for data abase usage, we can not create a proper database.

As a specific DBMS is base on the use of a specific data model so when using a DBMS it is of great use to know that what structures, manipulation languages and integrity constraints are implemented by a specific DBMS. As it is the only way to know the facilities and functionalities offered by the DBMS.

This is the reason whenever we get a specific DBMS, it is explicitly mentioned with that DBMS, that which data model this DBMS uses.

Types of Data Models

- Semantic Data Model

These are the data models which provide us better flexibility in implementing constraints, better language utilities and better data structure constructs. As a result actions performed using proper data and structure tools gives us better data designing and manipulation facilities. A better data model provides better opportunities to express multiple situations in the database design and as a result get better output from the tool or model in the form of a better database design.

- ER- Data Model
- Object oriented data model

- Record Based Data Model

This is the second type of data models available to use and has three basic types

- Hierarchical Data Model
- Network Data model
- Relational Data model

These models are records based and are not in similarity with those of semantic data models. These models handle the data at almost all the three level of the three layers of the database architecture. Semantic data models are generally used for designing the logical or conceptual model of the database system, once very common example of the semantic data model is ER-Data Model and is very much popular for designing databases. No DBMS is based on ER Data model because it is purely used for designing whereas a number of DBMS are available based on OO data model, network data model, relational data model l and hierarchical data model.

Types of Database Design

Conceptual database design
This design is implemented using a semantic data model, for example for creating a design for an organization database we can use and we do use the ER-Data model.

Logical Database design
This design is performed using a data model for which we have a DBMS available and we are planning to run our database system that DBMS.

Physical Database Design:
The Logical design created using a specific data model and created after the analysis of the organization, it needs to be implemented in a physical DBMS software so the Physical database design is performed and the design created so far in the logical form are implemented on that very DBMS.

By separating the three design levels we get the benefit of abstraction on one hand whereas on the other hand we can create our logical and conceptual designs using better design tools, which would have not been possible if we are using the same design-tool for al the three levels. Moreover if in future there is a need to make a change in the physical implementation of the data we will have to make no changes in the logical or conceptual level of the database design , rather the change can be achieved by only using the existing conceptual model and implementing it again on Physical model using a separate DBMS.

Lecture No. 07

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	Section: 5.2 5.3
Hoffer	Page: 85 - 95

Overview of Lecture

- Entity
- Different types of Entities
- Attribute and its different types
- In the previous lecture we discussed the importance and need of data models. From this lecture we are going to start detailed discussion on a data model, which is the entity relationship data model also known as E-R data model.

Entity-Relationship Data Model

It is a semantic data model that is used for the graphical representation of the conceptual database design. We have discussed in the previous lecture that semantic data models provide more constructs that is why a database design in a semantic data model can contain/represent more details. With a semantic data model, it becomes easier to design the database, at the first place, and secondly it is easier to understand later. We also know that conceptual database is our first comprehensive design. It is independent of any particular implementation of the database, that is, the conceptual database design expressed in E-R data model can be implemented using any DBMS. For that we will have to transform the conceptual database design from E-R data model to the data model of the particular DBMS. There is no DBMS based on the E-R data model, so we have to transform the conceptual database design anyway.

A question arises from the discussion in the previous paragraph, can we avoid this transformation process by designing our database directly using the data model of our selected DBMS. The answer is, yes we can but we do not do it, because most commercial DBMS are based on the record-based data models, like Hierarchical, Network or Relational. These data models do not provide too much constructs, so a database design

in these data models is not so expressive. Conceptual database design acts as a reference for many different purposes. Developing it in a semantic data model makes it much more expressive and easier to understand, that is why we first develop our conceptual database design in E-R data model and then later transform it into the data model of our DBMS.

Constructs in E-R Data Model

The E-R data model supports following major constructs:

- Entity
- Attribute
- Relationship

We are going to discuss each one of them in detail.

The Entity

Entity is basic building block of the E-R data model. The term entity is used in three different meanings or for three different terms and that are:

- Entity type
- Entity instance
- Entity set

In this course we will be using the precise term most of the time. However after knowing the meanings of these three terms it will not be difficult to judge from the context which particular meaning the term entity is being used in.

Entity Type

The entity type can be defined as a name/label assigned to items/objects that exist in an environment and that have similar properties. It could be person, place, event or even concept, that is, an entity type can be defined for physical as well as not-physical things. An entity type is distinguishable from other entity types on the basis of properties and the same thing provides the basis for the identification of an entity type. We analyze the things existing in any environment or place. We can identify or associate certain properties with each of the existing in that environment. Now the things that have common or similar properties are candidates of belonging to same group, if we assign a name to that group then we say that we have identified an entity type.

Generally, the entity types and their distinguishing properties are established by nature, by very existence of the things. For example, a bulb is an electric accessory, a cricket bat is a sports item, a computer is an electronic device, a shirt is a clothing item etc. So identification of entity types is guided by very nature of the things and then items having properties associated with an entity type are considered to be belonging to that entity type or instances of that entity type. However, many times the grouping of things in an environment is dictated by the specific interest of the organization or system that may supersede the natural classification of entity types. For example, in an organization, entity

types may be identified as donated items, purchased items, manufactured items; then the items of varying nature may belong to these entity types, like air conditioners, tables, frying pan, shoes, car; all these items are quite different from each other by their respective nature, still they may be considered the instances of the same entity type since they are all donated or purchased or manufactured.

What particular properties of an entity type should be considered or which particular properties jointly form an entity type? The answer to this question we have discussed in detail in our very first lecture, where we were discussing the definition of database. That is, the perspective or point of view of the organization and the system for which we are developing the database is going to guide us about the properties of interest for a particular group of things. For example, if you have a look around you in your bedroom, you might see tube light, a bulb, fan, air conditioner, carpet, bed, chair and other things. Now fan is an item that exists in your room, what properties of the fan we are interest in, because there could be so many different properties of the fan. If we are developing the database for a manufacturer, then we may be interested in type of material used for wings, then the thickness of the copper wire in the coil, is it locally manufactured or bought ready made, what individual item costs, what is the labor cost, what is the total cost, overhead, profit margin, net price etc. But if we are working for a shopkeeper he might be interested in the name of the company, dealer price, retail price, weight, color of fan etc. From the user perspective; company name, color, price, warranty, name of the dealer, purchase date and alike. So the perspective helps/guides the designer to associate or identify properties of things in an environment.

The process of identifying entity types, their properties and relationships between them is called abstraction. The abstraction process is also supported by the requirements gathered during initial study phase. For example, the external entities that we use in the DFDs provide us a platform to identify/locate the entity types from. Similarly, if we have created different cross reference matrices, they help us to identify different properties of the things that are of interest in this particular system and that we should the data about. Anyway, entity types are identified through abstraction process, then the items possessing the properties associated with a particular entity type are said to be belonging to that entity type or instances of that entity type.

While designing a system, you will find that most of the entity types are same as are the external entities that you identified for the DFDs. Sometimes they may be exactly the same. Technically, there is a minor difference between the two and that is evident from their definitions. Anything that receives or generates data from or to the system is an external entity, where as entity type is name assigned to a collection of properties of different things existing in an environment. Anything that receives or generates data is considered as external entity and is represented in the DFD, even if it is a single thing. On the other hand, things with a single instance are assumed to be on hand in the environment and they are not explicitly identified as entity type, so they are not represented in the E-R diagram. For example, a librarian is a single instance in a library system, (s)he plays certain role in the library system and at many places data is generated

from or to the librarian, so it will be represented at relevant places in the DFDs. But the librarian will not be explicitly represented in the E-R diagram of the library system and its existence or role is assumed to be there and generally it is hard-coded in the application programs.

Entity Instance

A particular object belonging to a particular entity type and how does an item becomes an instance of or belongs to an entity type? By possessing the defining properties associated with an entity type. For example, following table lists the entity types and their defining properties:

Entity Types	Properties	Instances
EMPLOYEE	Human being, has name, has father name, has a registration number, has qualification, designation	M. Sharif, Sh. Akmal and many others
FURNITURE	Used to sit or work on, different material, having legs, cost, purchased	Chair, table etc.
ELECTRIC APPLIANCES	Need electricity to work, purchased	Bulb, fan, AC
OFFICE EQUIPMENT	Used for office work, consumable or non-consumable,	Papers, pencil, paper weight etc.

Table 1: Entity types, their properties and instances

Each entity instance possesses certain values against the properties with the entity type to which it belongs. For example, in the above table we have identified that entity type EMPLOYEE has name, father name, registration number, qualification, designation. Now an instance of this entity type will have values against each of these properties, like (M. Sajjad, Abdul Rehman, EN-14289, BCS, and Programmer) may be one instance of entity type EMPLOYEE. There could be many others.

Entity Set

A group of entity instances of a particular entity type is called an entity set. For example, all employees of an organization form an entity set. Like all students, all courses, all of them form entity set of different entity types

As has been mentioned before that the term entity is used for all of the three terms mentioned above, and it is not wrong. Most of the time it is used to mention an entity type, next it is used for an entity instance and least times for entity set. We will be precise most of the time, but if otherwise you can judge the particular meaning from the context.

Classification of entity types

Entity types (ETs) can be classified into regular ETs or weak ETs. Regular ETs are also called strong or independent ETs, whereas weak ETs are also called dependent ETs. In the following we discuss them in detail.

Weak Entity Types

An entity type whose instances cannot exist without being linked with instances of some other entity type, i.e., they cannot exist independently. For example, in an organization we want to maintain data about the vehicles owned by the employees. Now a particular vehicle can exist in this organization only if the owner already exists there as employee. Similarly, if employee leaves the job and the organization decides to delete the record of the employee then the record of the vehicle will also be deleted since it cannot exist without being linked to an instance of employee.

Strong Entity Type

An entity type whose instances can exist independently, that is, without being linked to the instances of any other entity type is called strong entity type. A major property of the strong entity types is that they have their own identification, which is not always the case with weak entity types. For example, employee in the previous example is an independent or strong entity type, since its instances can exist independently.

Naming Entity Types

Following are some recommendations for naming entity types. But they are just recommendations; practices considered good in general. If one, some or all of them are ignored in a design, the design will still be valid if it satisfies the requirements otherwise, but good designs usually follow these practices:

- Singular noun recommended, but still plurals can also be used
- Organization specific names, like customer, client, gahak anything will work
- Write in capitals, yes, this is something that is generally followed, otherwise will also work.
- Abbreviations can be used, be consistent. Avoid using confusing abbreviations, if they are confusing for others today, tomorrow they will confuse you too.

Symbols for Entity Types

A rectangle is used to represent an entity type in E-R data model. For strong entity types rectangle with a single line is used whereas double lined rectangle is drawn to represent a weak entity type as is shown below:

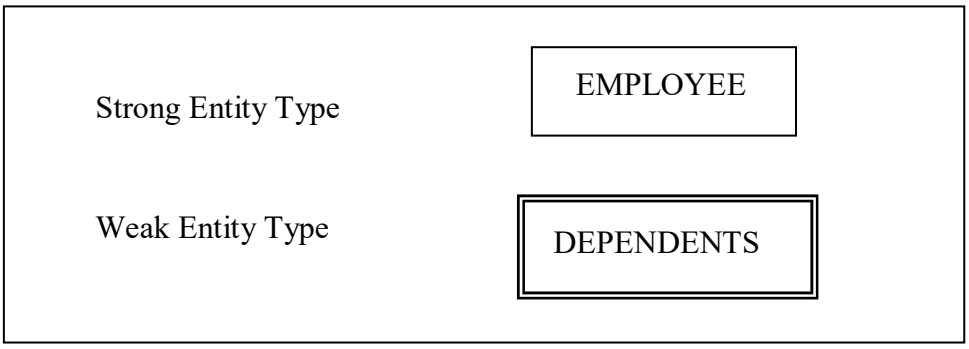


Figure 1: Symbols used for entity types

We have discussed different types of entity types; in the next section we are going to discuss another component or E-R data model, that is, the attribute.

Attribute

An attribute of an entity type is a defining property or quality of the instances of that entity type. Entity instances of same entity type have the same attributes. (E.g. Student Identification, Student Name). However, values of these attributes may be same or different. For example, all instances of the entity type STUDENT may have the attributes name, father name, age; but the values against each of these attributes for each instance may be different. Like, one instance may have the values (M. Hafeez, Noor Muhammad, 37) other may have others. Remember one thing, that the values of the attributes may be same among different entity instances. The thing to remember at this stage is that attributes are associated with an entity type and those attributes then become applicable /valid for all the instances of that entity type and instances have values against these attributes.

An attribute is identified by a name allocated to it and that has to be unique with respect to that entity type. It means one entity type cannot have two attributes with the same name. However, different entity types may have attributes with the same name. The guidelines for naming an attribute are similar to those of entity types. However, one difference is regarding writing the names of attributes. The notation that has been adopted in this course is that attribute name generally consists of two parts. The name is started in lower case, and usually consists of abbreviation of the entity types to which the attribute belongs. Second part of the attribute name describes the purpose of attribute and only first letter is capitalized. For example empName means name attribute of entity type EMPLOYEE, stAdrs means address attribute of the entity type STUDENT and alike. Others follow other notations, there is no restriction as such, and you can follow anyone that you feel convenient with. BUT be consistent.

Domain of an Attribute

We have discussed in the previous section that every attribute has got a name. Next thing is that a domain is also associated with an attribute. These two things, name and the domain, are part of the definitions of an attribute and we must provide them. Domain is the set of possible values that an attribute can have, that is, we specify a set of values either in the form of a range or some discrete values, and then attribute can have value out of those values. Domain is a form of a check or a constraint on attribute that it cannot have a value outside this set.

Associating domain with an attribute helps in maintaining the integrity of the database, since only legal values could be assigned to an attribute. Legal values mean the values that an attribute can have in an environment or system. For example, if we define a salary attribute of EMPLOYEE entity type to hold the salary of employees, the value assigned to this attribute should be numeric, it should not be assigned a value like 'Reema', or '10/10/2004', why, because they are not legal salary values [1]. It should be numeric. Further, even if we have declared it as numeric it will have numeric values, but about a value like 10000000000. This is a numeric value, but is it a legal salary value within an organization? You have to ask them. It means not only you will specify that the value of salary will be numeric but also associate a range, a lower and upper limit. It reduces the chances of mistake.

Domain is normally defined in form of data type and some additional constraints like the range constraint. Data type is defined as a set of values along with the operations that can be performed on those values. Some common data types are Integer, Float, Varchar, Char, String, etc. So domain associates certain possible values with an attribute and certain operations that can be performed on the values of the attribute. Another important thing that needs to be mentioned here is that once we associate a domain to an attribute, all the attributes in all entity instances of that entity type will have the values from the same domain. For example, it is not possible that in one entity instance the attribute salary has a value 15325.45 and in another instance the same attribute has a value 'Reema'. No. All attribute will have values from same domain, values may be different or same, whatever, but the domain will be the same.

[1] Sometimes when some coding has been adopted, then such strange values may be legal but here we are discussing the general conditions

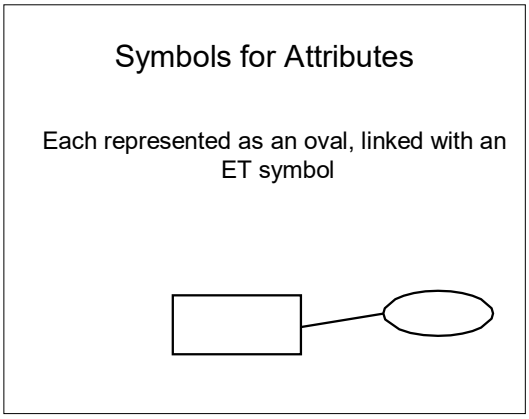


Figure 2: Symbol used for attribute in E-R diagram

Types of Attributes

Attributes may be of different types. They may be:

- Simple or Composite
- Single valued or multi-valued
- Stored or Derived

Simple or Composite Attributes:

An attribute that is a single whole is a simple attribute. The value of a simple attribute is considered as a whole, not as comprising of other attributes or components. For example, attributes stName, stFatherName, stDateOfBorth of an entity type STUDENT are example of simple attributes. On the other hand if an attribute consists of collection of other simple or composite attributes then it is called a composite attributes. For example, stAdres attribute may comprise of houseNo, streetNo, areaCode, city etc. In this case stAdres will be a composite attribute.

Single valued or multi-valued Attributes:

Some attribute have single value at a time, whereas some others may have multiple values. For example, hobby attribute of STUDENT or skills attribute of EMPLOYEE, since a student may have multiple hobbies, likewise an employee may have multiple skills so they are multi-valued attributes. On the other hand, name, father name, designation are generally single valued attributes.

Stored or Derived Attributes:

Normally attributes are stored attributes, that is, their values are stored and accessed as such from the database. However, sometimes attributes' values are not stored as such, rather they are computed or derived based on some other value. This other value may be stored in the database or obtained some other way. For example, we may store the name, father name, address of employees, but age can be computed from date of birth. The

advantage of declaring age as derived attribute is that whenever we will access the age, we will get the accurate, current age of employee since it will be computed right at the time when it is being accessed.

How a particular attribute is stored or defined, it is decided first by the environment and then it has to be designer's decision; your decision. Because, the organization or system will not object rather they will not even know the form in which you have defined an attribute. You have to make sure that the system works properly, it fulfills the requirement; after that you do it as per your convenience and in an efficient way.

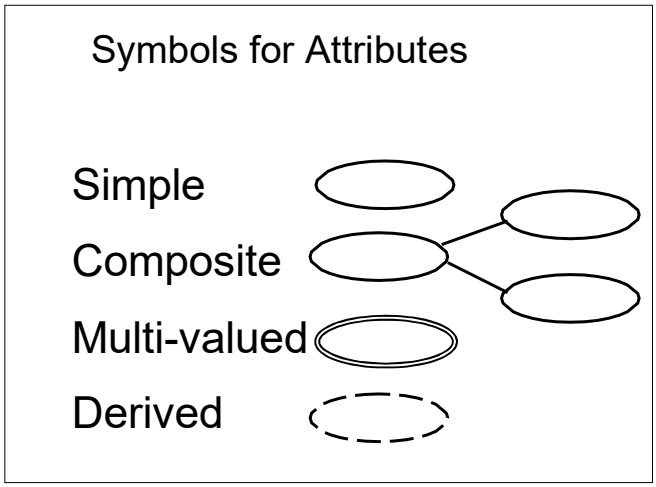


Figure 3: Symbol used for different types of attributes in E-R diagram

An example diagram representing all types of attributes is given below:

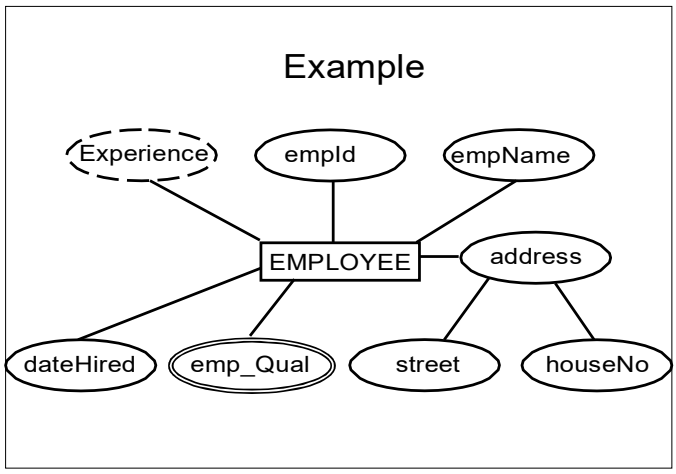


Figure 4: Example entity type with attributes of different types
This concludes this lecture.

Summary:

In this lecture we have discussed entity and attribute. We discussed that there are three different notions for which the term entity is used and we looked into these three terms in detail. They are entity type, entity instance and entity set. An entity type is name or label assigned to items or objects existing in an environment and having same or similar property. An entity instance is a particular item or instance that belongs to a particular entity type and a collection of entity instances is called an entity set. We also discussed in this lecture the attribute component of the E-R data model and its different types. The third component the E-R data model, that is, the relationship will be discussed in the next lecture.

Exercises:

- Take a look into the system where you work or study or live, identify different entity types in that environment. Associate different types of attributes with these entity types.
- Look at the same environment from different possible perspectives and realize the difference that the change of perspective makes in the abstraction process that results in establishing different entity types or/and their different properties.

Lecture No. 08

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	Section 5.4

Overview of Lecture

- Concept of Key and its importance
- Different types of keys

Attributes

Def 1:

An attribute is any detail that serves to identify, qualify, classify, quantify, or otherwise express the state of an entity occurrence or a relationship.

Def 2:

Attributes are data objects that either identify or describe entities.

Identifying entity type and then assigning attributes or other way round; it's an "egg or hen" first problem. It works both ways; differently for different people. It is possible that we first identify an entity type, and then we describe it in real terms, or through its attributes keeping in view the requirements of different users' groups. Or, it could be other way round; we enlist the attribute included in different users' requirements and then group different attributes to establish entity types. Attributes are specific pieces of information, which need to be known or held. An attribute is either required or optional. When it's required, we must have a value for it, a value must be known for each entity occurrence. When it's optional, we could have a value for it, a value may be known for each entity occurrence.

The Keys

Attributes act as differentiating agents among different entity types, that is, the differences between entity types must be expressed in terms of attributes. An entity type can have many instances; each instance has got a certain value against each attribute defined as part of that particular entity type. A *key* is a set of attributes that can be used to

identify or access a particular entity instance of an entity type (or out of an entity set). The concept of key is beautiful and very useful; why and how. An entity type may have many instances, from a few to several thousands and even more. Now out of many instances, when and if we want to pick a particular/single instance, and many times we do need it, then key is the solution. For example, think of whole population of Pakistan, the data of all Pakistanis lying at one place, say with NADRA people. Now if at sometime we need to identify a particular person out of all this data, how can we do that? Can we use name for that, well think of any name, like Mirza Zahir Iman Afroz, now we may find many people with this name in Pakistan. Another option is the combination of name and father name, then again, Amjad Malik s/o Mirza Zahir Iman Afroz, there could be so many such pairs. There could be many such examples. However, if you think about National ID Card number, then no matter whatever is the population of Pakistan, you will always be able to pick precisely a single person. That is the key. While defining an entity type we also generally define the key of that entity type. How do we select the key, from the study of the real-world system; key attribute(s) already exist there, sometimes they don't then the designer has to define one. A key can be simple, that is, consisting of single attribute, or it could be composite which consists of two or more attributes. Following are the major types of keys:

- Super Key
- Candidate Key
- Primary Key
- Alternate Key
- Secondary Key
- Foreign Key

The last one will be discussed later, remaining 5 are discussed in the following:

- Super key

  A super key is a set of one or more attributes which taken collectively, allow us to identify uniquely an entity instance in the entity set. This definition is same as of a key, it means that the super key is the most general type of key. For example, consider the entity type STUDENT with attributes registration number, name, father name, address, phone, class, admission date. Now which attribute can we use that can uniquely identify any instance of STUDENT entity type. Of course, none of the name, father name, address, phone number, class, admission date can be used for this purpose. Why? Because if we consider name as super key, and situation arises that we need to contact the parents of a particular student. Now if we say to our registration department that give us the phone number of the student whose name is Ilyas Hussain, the registration department conducts a search and comes up with 10 different Ilyas Hussain, could be anyone. So the value of the name attribute cannot be used to pick a particular instance. Same happens with other attributes. However, if we use the registration number, then it is 100% sure that with a particular value of registration number we will always find exactly a single unique entity instance. Once you identified the instance, you have all its attributes available, name, father name,

everything. The entity type STUDENT and its attributes are shown graphically in the figure 1 below, with its super key "regNo" underlined.

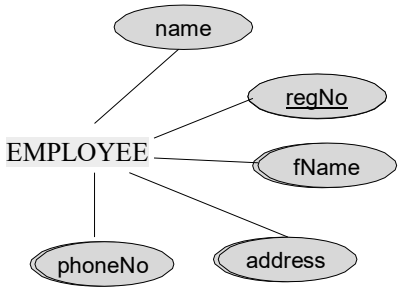


Fig. 1: An entity type, its defining attributes and super key (underlined)

Once specific characteristic with super key is that, as per its definition any combination of attributes with the super key is also a super key. Like, in the example just discussed where we have identified regNo as super key, now if we consider any combination of regNo with any other attribute of STUDENT entity type, the combination will also be a super key. For example, "regNo, name", "regNo, fName, address", "name, fName, regNo" and many others, all are super keys.

- Candidate key

  A super key for which no subset is a super key is called a candidate key, or the minimal super key is the candidate key. It means that there are two conditions for the candidate key, one, it identifies the entity instances uniquely, as is required in case of super key, second, it should be minimum, that is, no proper subset of candidate key is a key. So if we have a simple super key, that is, that consists of single attribute, it is definitely a candidate key, 100%. However, if we have a composite super key and if we take any attribute out of it and remaining part is not a super key anymore then that composite super key is also a candidate key since it is minimal super key. For example, one of the super keys that we identified from the entity type STUDENT of figure 1 is "regNo, name", this super key is not a candidate key, since if we remove the regNo attribute from this combination, name attribute alone is not able to identify the entity instances uniquely, so it does not satisfy the first condition of candidate key. On the other hand if we remove the attribute name from this composite key then the regNo alone is sufficient to identify the instances uniquely, so "regNo, name" does have a proper subset (regNo) that can act as a super key; violation of second condition. So the composite key "regNo, name" is a super key but it is not a candidate key. From here we can also establish a fact that every candidate key is a super key but not the other way round.

- Primary Key

A candidate key chosen by the database designer to act as key is the primary key. An entity type may have more than one candidate keys, in that case the database designer has

to designate one of them as primary key, since there is always only a single primary key in an entity type. If there is just one candidate key then obviously the same will be declared as primary key. The primary key can also be defined as the successful candidate key. Figure 2 below contains the entity type STUDENT of figure 1 but with an additional attribute nIdNumber (national ID card Number).

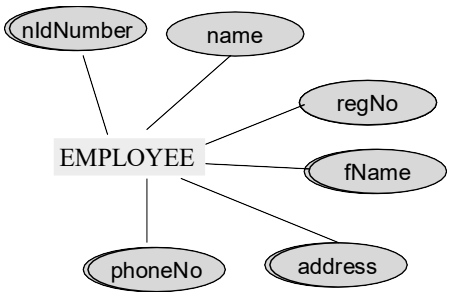


Fig. 2: An entity type, its defining attributes and two candidate keys

In figure 2, we can identify two different attributes that can individually identify the entity instances of STUDENT and they are regNo and nIdNumber, both are minimal super keys so both are candidate keys. Now in this situation we have got two candidate keys. The one that we choose will be declared as primary key, other will be the alternate key. Any of the candidate keys can be selected as primary key, it mainly depends on the database designer which choice he/she makes. There are certain things that are generally considered while making this decision, like the candidate key that is shorter, easier to remember, to type and is more meaningful is selected as primary key. These are general recommendations in this regard, but finally it is the decision of the designer and he/she may have his/her own reasons for a particular selection that may be entirely different from those mentioned above. The relation that holds between super and candidate keys also holds between candidate and primary keys, that is, every primary key (PK) is a candidate key and every candidate key is a super key.

A certain value that may be associated with any attribute is NULL, that means “not given” or “not defined”. A major characteristic of the PK is that it cannot have the NULL value. If PK is a composite, then none of the attributes included in the PK can have the NULL, for example, if we are using “name, fName” as PK of entity type STUDENT, then none of the instances may have NULL value in either of the name or fName or both.

- Alternate Keys
  Candidate keys which are not chosen as the primary key are known as alternate keys. For example, we have two candidate keys of STUDENT in figure 2, *regNo* and *nIdNumber*, if we select *regNo* as PK then the *nIdNumber* will be alternate key.
- Secondary Key

Many times we need to access certain instances of an entity type using the value(s) of one or more attributes other than the PK. The difference in accessing instances using the

value of a key or non-key attribute is that the search on the value of PK will always return a single instance (if it exists), where as uniqueness is not guaranteed in case of non-key attribute. Such attributes on which we need to access the instances of an entity type that may not necessarily return unique instance is called the secondary key. For example, we want to see how many of our students belong to Multan, in that case we will access those instances of the STUDENT entity type that contain "Multan" in their address. In this case address will be called secondary key, since we are accessing instances on the basis of its value, and there is no compulsion that we will get a single instance. Keep one thing in mind here, that a particular access on the value of a secondary key MAY return a single instance, but that will be considered as chance or due to that particular state of entity set. There is not the compulsion or it is not necessary for secondary key to return unique instance, where as in case of super, candidate, primary and alternate keys it is compulsion that they will always return unique instance against a particular value.

Summary

Keys are fundamental to the concept almost any data model including the E-R data model because they enable the unique identity of an entity instance. There are different type of keys that may exist in an entity type.

Exercises:

- Define attributes of the entity types CAR, BOOK, MOVIE; draw them graphically
- Identify different types of keys in each one of them

Lecture No. 09

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	

Overview of Lecture

- Relationships in E-R Data Model
- Types of Relationships

Relationships

After two or more entities are identified and defined with attributes, the participants determine if a *relationship* exists between the entities. A relationship is any association, linkage, or connection between the entities of interest to the business; it is a two-directional, significant association between two entities, or between an entity and itself. Each relationship has a name, an optionality (*optional* or *mandatory*), and a degree (how many). A relationship is described in real terms.

Assigning a name, optionality, and a degree to a relationship helps confirm the validity of that relationship. If you cannot give a relationship all these things, then perhaps there really is no relationship at all.

Relationship represents an association between two or more entities. An example of a relationship would be:

- Employees are assigned to projects
- Projects have subtasks
- Departments manage one or more projects

Relationships are the *connections and interactions* between the entities instances e.g. DEPT_EMP associates Department and Employee.

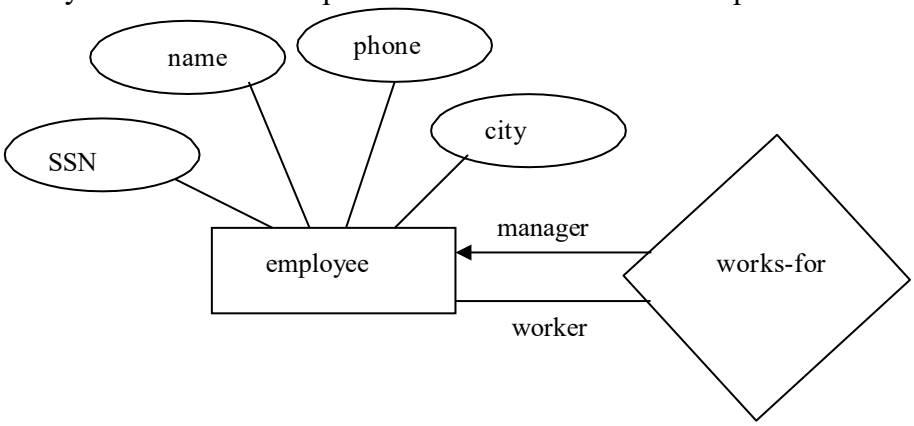
- A *relationship type* is an abstraction of a relationship i.e. a set of relationships instances sharing common attributes.

- Entities enrolled in a relationship are called its *participants*.

The participation of an entity in a relationship is *total* when all entities of that set might be participant in the relationship otherwise it is *partial* e.g. if every *Part* is supplied by a *Supplier* then the SUPP_PART relationship is total. If certain parts are available without a supplier than it is partial.

Naming Relationships:
If there is no proper name of the association in the system then participants' names of abbreviations are used. STUDENT and CLASS have ENROLL relationship. However, it can also be named as STD_CLS.

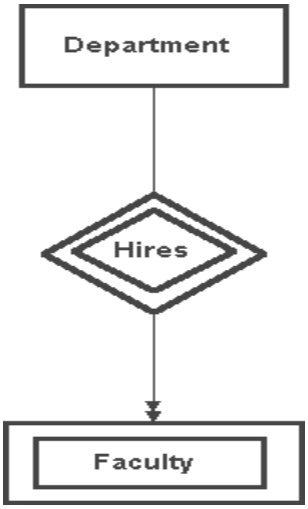
Roles:
Entity set of a relationship need not be distinct. For example



The labels "manager" and "worker" are called "roles". They specify how employee entities interact via the "works-for" relationship set. Roles are indicated in ER diagrams by labeling the lines that connect diamonds to rectangles. Roles are optional. They clarify semantics of a relationship.

Symbol for Relationships:

- Shown as a Diamond
- Diamond is doubled if one of the participant is dependent on the other
- Participants are connected by continuous lines, labeled to indicate cardinality.
- In partial relationships roles (if identifiable) are written on the line connecting the partially participating entity rectangle to the relationship diamond.
- Total participation is indicated by double lines

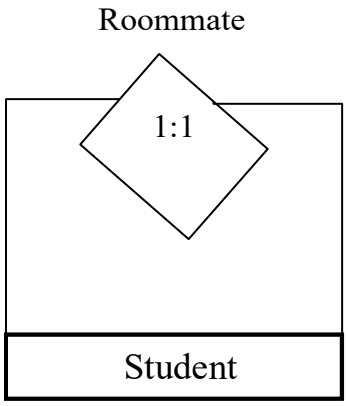


Types of Relationships

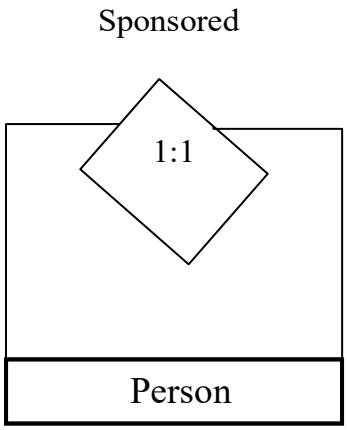
- Unary Relationship

  An ENTITY TYPE linked with itself, also called recursive relationship. Example Roommate, where STUDENT is linked with STUDENT

  Example 1:



Example
2:



- Binary relationship

  A Binary relationship is the one that links two entities sets e.g. STUDENT-CLASS. Relationships can be formally described in an ordered pair form.

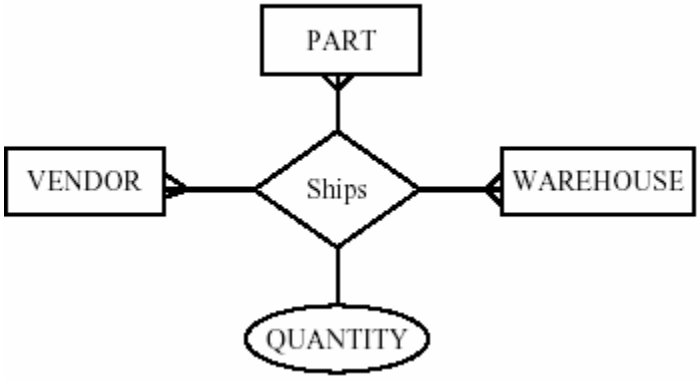
  Enroll = {(S1001, ART103A), (S1020, CS201A), (S1002, CSC201A)}

  Entire set is relationship set and each ordered pair is an instance of the relationship.

- Ternary Relationship

  A Ternary relationship is the one that involves three entities e.g.

  STUDENT-CLASS-FACULTY.



- N-ary Relationship

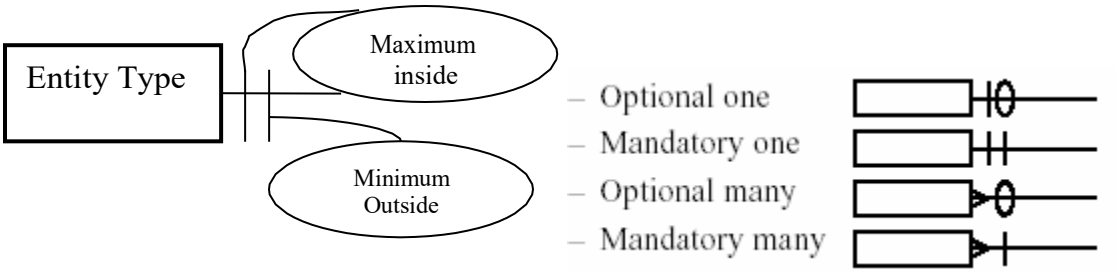
  Most relationships in data model are binary or at most ternary but we could define a relationship set linking any number of entity sets i.e. n-ary relationship

Entity sets involved in a relationship set need not be distinct. E.g.

Roommate = {(Student1, Student2) | Student1 ∈ Student Entity Set, Student2 ∈ Student Entity Set and Student 1 is the Roommate of Student2}

Relationship Cardinalities

The cardinality of a relationship is the number of entities to which another entity can map under that relationship. Symbols for maximum and minimum cardinalities are:

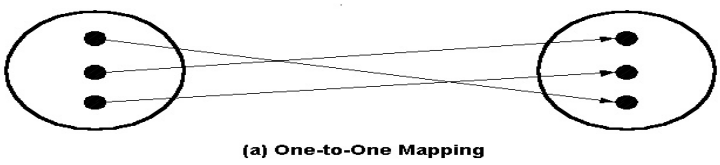


- One-to-One mapping:

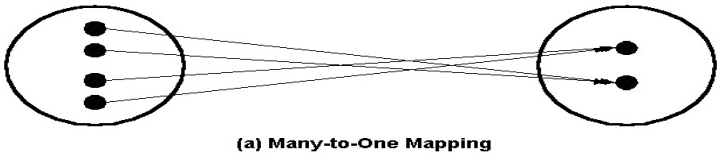
  A mapping R from X to Y is one-to-one if each entity in X is associated with at most one entity in Y and vice versa.

- Many-to-One mapping:

  A mapping R from X to Y is many-to-one if each entity in X is associated with at most one entity in Y but each entity in Y is associated with many entities in X.



(a) One-to-One Mapping



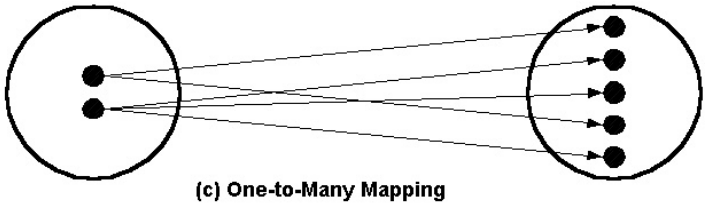
(a) Many-to-One Mapping

- One-to-Many mapping:

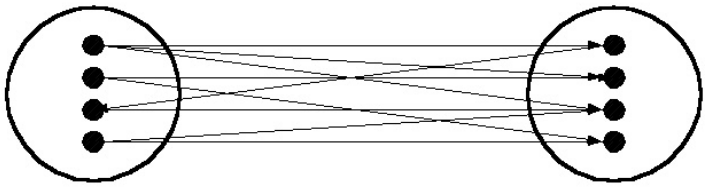
  A mapping R from X to Y is one-to-many if each entity in X is associated with many entities in Y but each entity in Y is associated with one entity in X.

- Many-to-Many mapping:

  A mapping R from X to Y is many-to-many if each entity from X is associated with many entities in Y and one entity in Y is associated with many entities in X.



(c) One-to-Many Mapping



(d) Many-to-Many Mapping

Lecture No. 10

Reading Material

"Database Systems Principles, Design and Implementation" written by Catherine Ricardo, Maxwell Macmillan.	Page: 155 – 160
Hoffer	Page: 103 – 111

Overview of Lecture

- Cardinality Types
- Roles in ER Data Model
- Expression of Relationship in ER Data Model
- Dependency
- Existence Dependency
- Referential Dependency
- Enhancements in the ER-Data Model
- Subtype and Supertype entities

Recalling from the previous lecture we can say that that cardinality is just an expression which tells us about the number of instances of one entity which can be present in the second relation. Maximum cardinality tells us that how many instance of an entity can be placed in the second relation at most. Now we move onto discuss that what the minimum cardinality is.

Minimum Cardinality:

As the name suggests that the minimum cardinality is the inverse of the maximum cardinality so we can say that the minimum cardinality show us that how many instance of one entity can be placed in another relation at least. In simple words it can be said that the minimum cardinality tells that whether the link between two relations is optional or compulsory. It is very important to determine the minimum cardinality when designing a database because it defines the way a database system will be implemented.

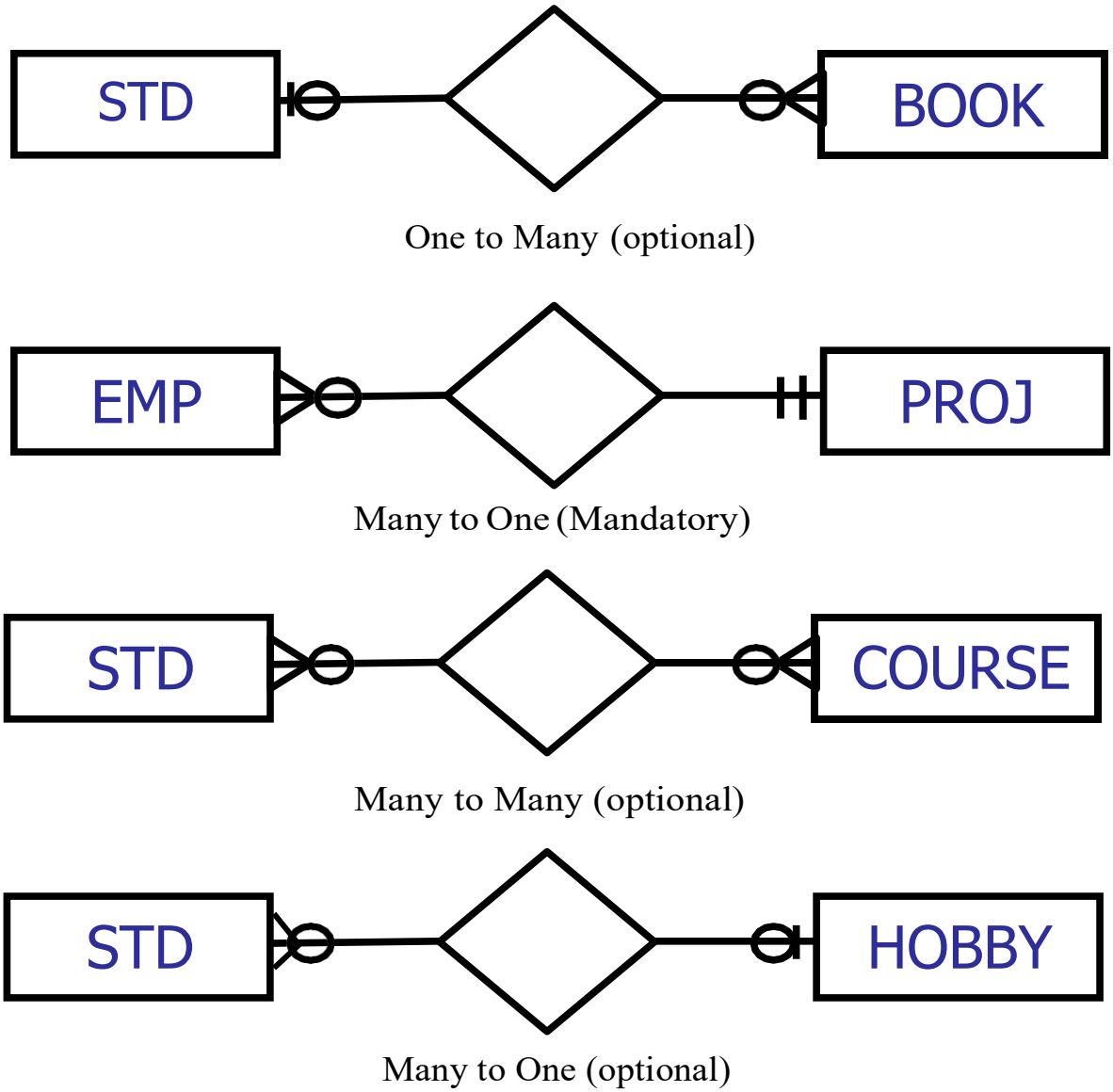


Fig 1: Different Cardinalities

In the figure-1 we have one to many cardinality between the entities. Maximum cardinalities are shown with the modifier that appears on the link and is adjacent to the entity rectangle. The other modifier which is next to the maximum cardinality modifier tells the minimum cardinality. The minimum cardinality modifier lies at more distance from the entity as compared to the maximum cardinality modifier.

Determination of the cardinalities is done by interviewing the users of the system and by the analysis of the organization.

The cardinality shown in First Part of the Figure-1 is shown using a relationship between a student and book; this can be a library scenario where students are borrowing books from the library. We can see in the diagram the shape adjacent to the student entity it shows that the minimum cardinality for the student relationship is zero and maximum cardinality is one. Where as on the other side of the diagram the shape  adjacent to

the book entity show that at most there can be many instances of the book associated with a single instance of student entity, and that there can be at-least no instance associated with the student entity. In general library scenario we can say that one student can borrow at least no and at most many books. Hence the minimum and maximum cardinality is shown.

In the second part of the Figure-1 we see a relationship between the Employee and project entities, the relationship describes one to many association between the project and the employees, It shows that there can be one project having a number of employees, but for the existence of one employee at one project is necessary. So the minimum and maximum cardinality on the project side of the relationship is one, and employees associated with each project can be many at most and none at-least.

Third part of the Figure-1 shows the association between the student and the course entities. Here we can see that the relationship between the student and the course is zero at least and many at most on both the sides of the relationship. The minimum cardinality with zero minimum is also called the optional cardinality. It also shows that one student can have registered more that one subjects and one subject can also be taken by many students. Also it is not necessary for a student to get registered one subject.

In the fourth part of the Figure-1 we can see the one to many cardinality between the student and hobby entities the cardinality descriptors show that a student may have no or at most one hobby, but it is worthwhile to notice that the cardinality of the hobby with the student in many but optional, now we can say that one hobby can be associated to nay student but there is a chance that no hobby is associated to one student at a certain time.

Other Notations:

The notation mentioned above is known as crow's foot notation for the expression of ER-Diagrams, there can be other notation as well which can be used for creating ER-Diagrams; one of these notations is shown in the Figure-2. We can see that the one to many cardinality shown in the first part of the diagram is expresses with single and double arrows. The Single arrow in this case shows the one and double arrow show the many cardinality.

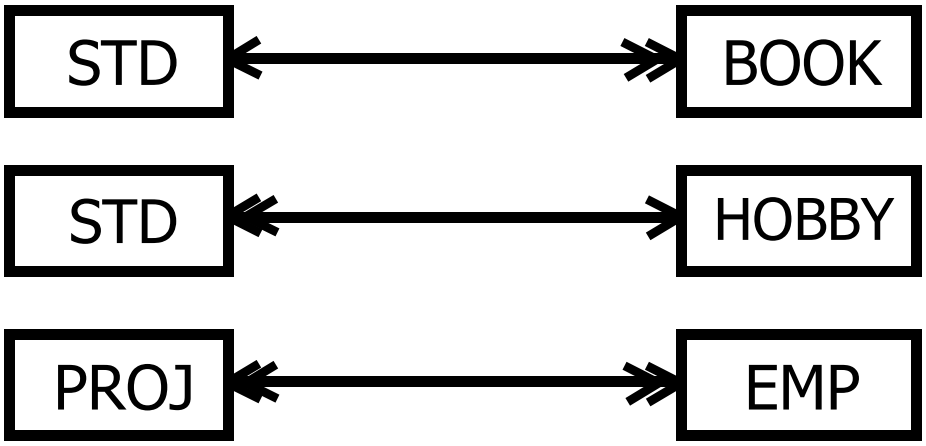


Fig. 2: Arrow-head notation

So the First part of the figure-2 show One to many cardinality, second part of the Figure shows many to one and the third part of the cardinality shows many to many cardinality between the entities involved.

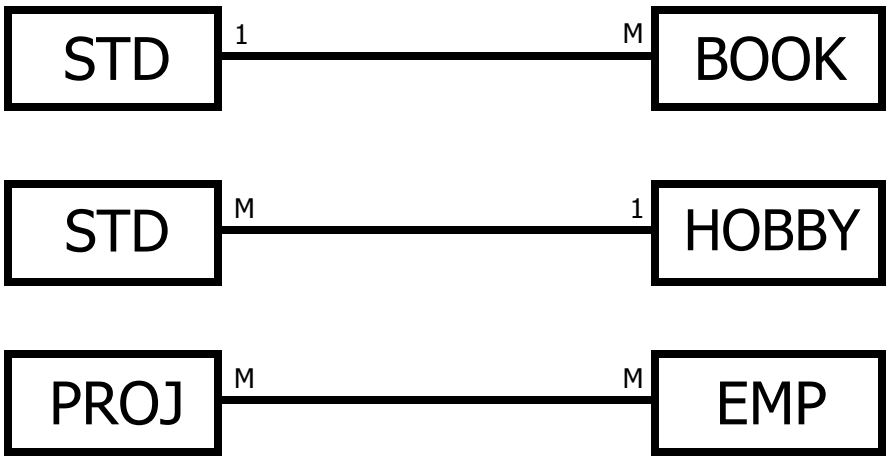


Fig. 3: Alphabetical notation

The above Figure shows another notation for creating ER-Diagrams which show that to show the one cardinality we have used 1 and for many cardinality M or N is used.

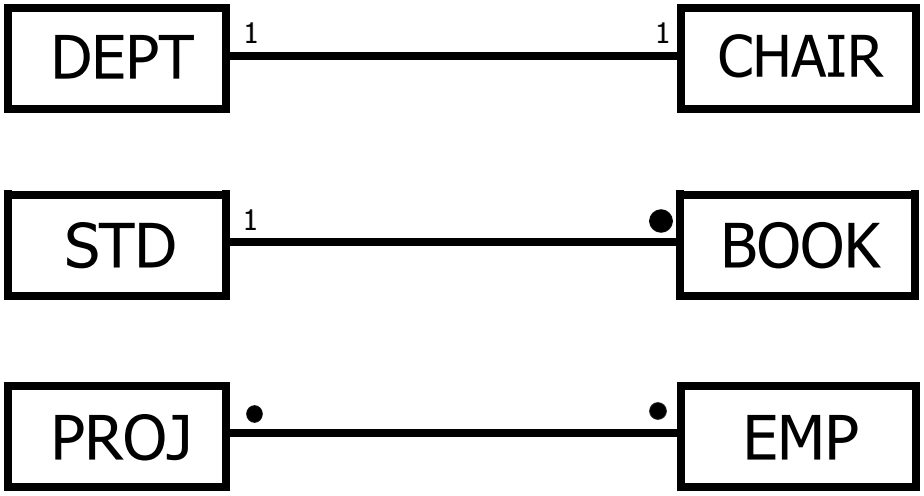


Fig. 4: Dot-based notation

Notations shown in the Figure-4 above as also used for creating ER-Diagrams where 1 is used for showing the single cardinality and the black filled Dot is used for showing many cardinality.

Roles in Relationships

The way an entity is involved in a relationship is called the role of the entity in the relationship. These details provide more semantics of the database. The role is generally clear from the relationship, but in some cases it is necessary to mention the role explicitly. Two situations to mention the role explicitly

Recursive Relationship:
This is the situation when any attribute of one entity is associated with another attribute of the same entity. Such a link initiates from one entity and terminates on the same entity.

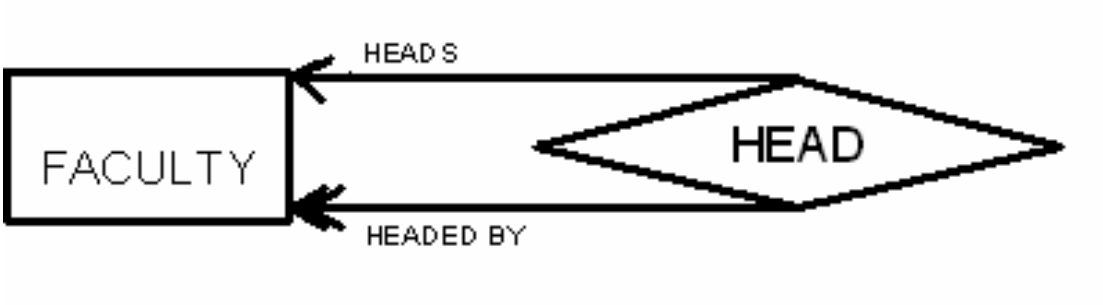


Fig-5: Roles in a unary relationship

Figure-5 above shows the recursive relationship which tells that in the faculty of a certain institute we can have one faculty member from among the same faculty as the head of the faculty. Now the role mentioned on the relationship tell that many Faculty instance are headed by one of the entity instance from the same faculty relation.

Multiple Relationships:
This is the second situation which needs the role to be mentioned on the relationship link when there is more than one relationship.

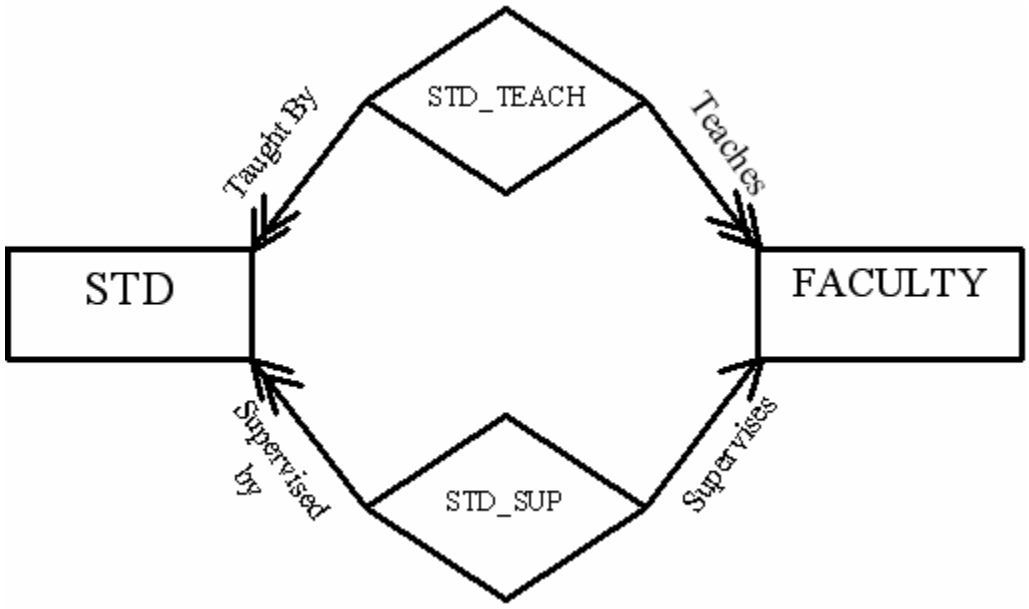


Fig. 6: Multiple relationships

As an example we can have a relationship of Faculty members and students as one faculty member may teach a number of students and at the same time one student may have been taught by a number of faculty members. This is one side of the picture. Now on the other side we can say that a faculty member may be supervising a number of students for their final projects. It shows two types of associations between the faculty and the students. So in this type of situation it is necessary to mention the role of the entities involved in the relationship.

Dependencies

Dependency is a type of constraint, for example once we define the cardinality or relationship among two entities it also is a constraint or check that tells that cardinality should be followed while populating data in relations. Similarly the dependency is a constraint. There are a number of dependency types which are expressed below:

The Existence dependency:
This is the type of dependency which exists when one entity instance needs instance of another entity for its existence. As we have seen earlier in case of employee of and organization and the projects associated with the employees there we see that employees are dependent on projects, it means that if no project is assigned to an employee it can not exist. In other words we can say that at a certain time an employee must be working on at least one project.

Identifier Dependency:
It means that the dependent entity incase of existence dependency does not have its own identifier and any external identifier is used to pick data for that entity. And to define a key in this entity the key of the parent entity is to be used in the key for this entity may be used as composite keys.

Referential Dependency:
This is the situation when the dependent entity has it own key for unique identification but the key used to show the reference with the parent entity is shown with the help of an attribute of the parent entity. Means to show the link of the parent entity with this entity there will be an attribute and a record in this entity will not exist without having a record in the parent entity. Despite of having its own identifier attribute.
This type of identifier or attribute in the weak entity is known as foreign key.

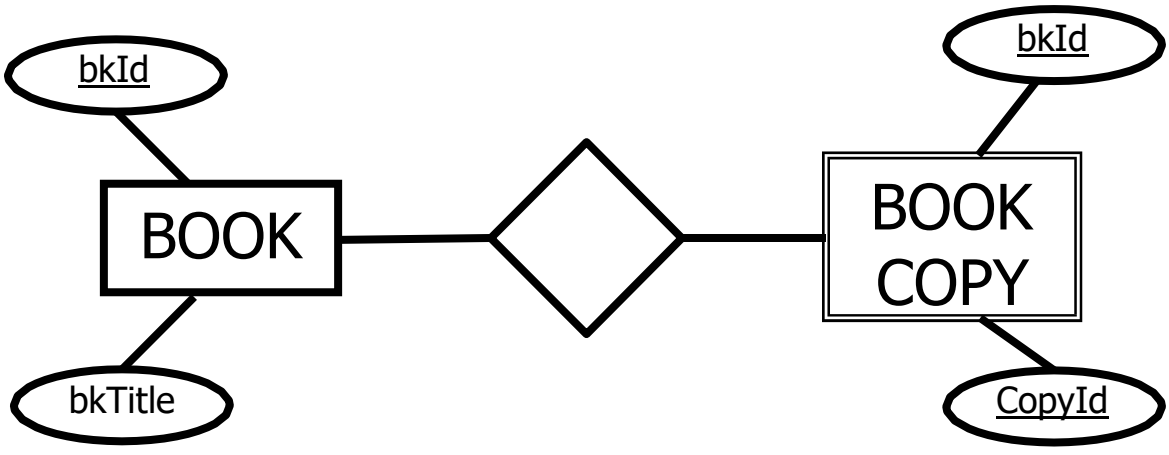


Fig-7

In the Figure-7 above the relation shown is expression the existence dependency where it is necessary for a book instance to exist if there exist the copies of the book with the same bkId.

Enhancements in E-R Data Model:

The topics that we have discussed so for constitute the basics of ER-Model. The model is further extended and strengthened with addition of some new concepts and modeling constructs, which are discussed below

Super-type and Subtypes

These are also relationships existing between entities, also referred to as generalized and specialized respectively let us examine the figure below to grasp the idea of super-type and subtype.

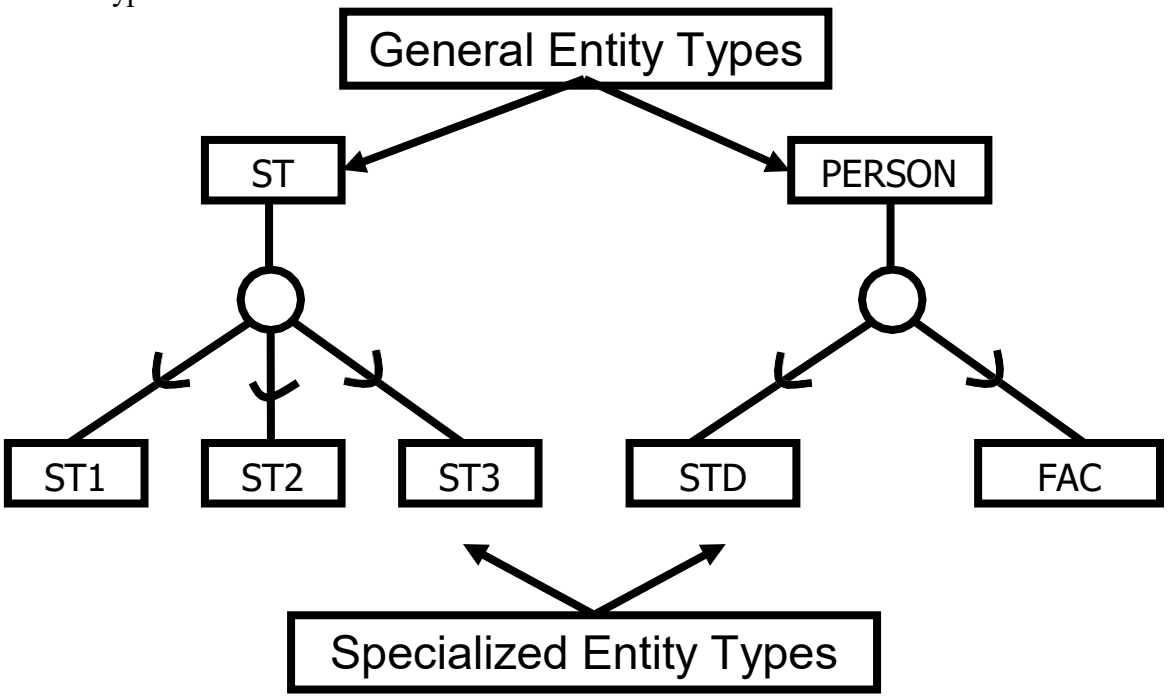


Fig-8 (Super-types and Subtypes)

In the Figure:8 show above there are different levels of existence of entities, at the top level we have general entity type, which are described as having a number of Subtype entities, these sub entities are in-turn acting as supertypes entities for a number of other entities. As we see in case of person supertype we can have further classify the person entity as Student (STD) and Teacher of Faculty member (FAC). Subtype entities are expressed with a link to the supertypes having an arc on the link—the arms of which

point to the supertype entity. As we move downward the distributed entities are known as specialized entities.

In the next Lecture the process of Generalization and Specialization will be discussed in detail.

Summary:

In this lecture we have discussed an important topic of cardinalities and their representation in the E-R data model. For a correct design the correct identification of cardinalities is important.