
Contents

1	Introduction	3
2	Entity-Relationship Model	5
2.1	ER model	5
2.2	Entity	5
2.3	Relationship	6
2.4	Weak entity	6
2.5	ER notation and diagrams	6
2.6	Relationship types of degree higher than two	7

Chapter 1

Introduction

Database is set of stored data that is persistent, integrated, interconnected with as little as possible redundancy, based on a data model, under control of a central system, used by multiple user, shared and concurrent.

Indexes are used to improve searching. Explain the following

1. B-tree and B⁺-tree
2. R-tree (for two dimension trees)
3. ISO and image
4. HDFS

Chapter 2

Entity-Relationship Model

In database design, database designers first need to know the data they are modeling. This includes knowing the *data requirements* and *functional requirements*. Functional requirements are the operations on the data. Once the requirements have been analyzed, the next step is designing a *conceptual schema/model* for the database. Note that this step is not dependent on the DBSM system and is done using high-level conceptual modeling. Then, the database designers must be this conceptual model to a DBSM system, this step is called *logical design* or *data mapping design*. The last step is *physical design* in which the goal is to design the underlying hardware, that is the storage structures, file organization, indices, access paths, and specifying physical parameters for the design.

2.1 ER model

Entity-Relationship model is a conceptual model of data.

2.2 Entity

An *entity* is an object or a thing that exists independently. Each entity has *attributes* that describe it. For example STUDENT is an entity that has a name and a student number. Each attribute has some value, based on which we can categorize the attributes into

Definition:

Simple/Atomic Attributes that are not divisible into other attributes. For example "First Name".

Composite Attributes that can be divided into other attributes. For example "Name" can be divided into "First Name", "Middle Name", and "Last Name".

Singlevalued Attributes that can only have one value. For example "Social Security Number".

Multivalued Attributes that can have many values. For example "Phone Number" might have multiple values.

Derived Attributes that can be determined from another attributes which is called the *stored attribute*. For example, "Area" can be determined from "Radius" in case of a CIRCLE entity.

Attributes can have *null* value. We can consider a domain of values – a set – for each attribute. To allow nullable and multivalued attribute we can consider an attribute to be a subset of this domain.

A set of entities that have similar attribute like STUDENT is called an *entity type*. An important attributes of the entities in an entity type is their *key/identifier attribute* which uniquely determines the specific entity. Some entity types might have more than one key attribute, For example a STUDENT can be determined via its "Student Number" or "Social Security Number".

2.3 Relationship

The interaction between two or more entities is a *relationship*. Mathematically, a relationship is a Cartesian product n entity types. E_1, \dots, E_n said to *participate* in a relationship type $R = E_1 \times \dots \times E_n$ and individual entities e_1, \dots, e_n are said to *participate* in a relationship instance $r = (e_1, \dots, e_n)$. The degree of a relationship is the number of entity types that participate in it. Binary relationship are relationship with two entity types. Binary relationship can have different *cardinality ratios*

Many-to-Many denoted as $M : N$. For example each STUDENT can enroll in multiple COURSES and each COURSE has many STUDENT enrolled in it.

One-to-Many denoted as $N : 1$. Each DEPARTMENT has many FACULTY but each FACULTY has only one DEPARTMENT.

One-to-One denoted as $1 : 1$. Each DEPARTMENT has one DEAN and each DEAN belongs to one DEPARTMENT.

Total participation or *existence dependency* is when each entity in the first entity type must be in a relationship with another entity in the second entity type. For example, each FACULTY must have a DEPARTMENT. In *partial participation* some of the entities in the first entity type are in a relationship with the entities in the second entity type. For example, not every FACULTY is a DEAN.

A relationship might have attributes for itself. For instance, a DEAN *MANAGES* a DEPARTMENT from a starting time to some other time.

2.4 Weak entity

An entity type that does not have a key attribute is called a *weak entity*. Those entities that have a key attribute are called *strong entity*. In real world, a weak entity is an entity that is dependent on the existence of another entity, called *identifying/owner entity type*. In ER model we can model this relationship by a total participation relationship which is called *identifying relationship*. A weak entity type normal has one or more *partial key* attributes that distinguish it from the other related weak entities to the same owner entity. For example, a COURSEGROUP entity has the "Semester", "Year", and the "Group Number" attributes of a COURSE entity. Every COURSEGROUP entity must be related to a COURSE entity and here, each of its attributes are partial keys.

By adding weak entities we can reduce the order of a relationship to two and even remove attributes of the relationship and give it to the weak entity.

2.5 ER notation and diagrams

Insert the corresponding diagrams :)

2.6 Enhanced/Extended ER model