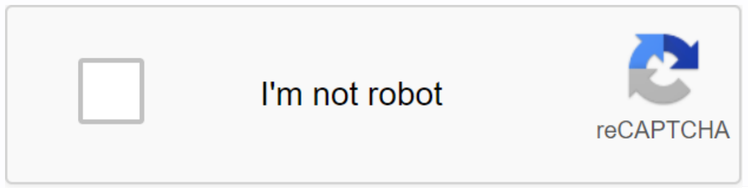

Continue

Specialization and generalization pdf

Table of Material Objectives At the end of this chapter you should be able to: describe the concepts of specialization/normalization, aggregation and structure. Illustrate how specialization/normalization, aggregation and structure are represented in ER diagrams. Map the specialization/generalization relationship for tables suitable for relational database implementation. In parallel to this chapter, you should read Chapter 12 of Thomas Connolly and Carolyn Baig, a practical approach to database system design, implementation and management, (5th edn.). This chapter builds on the previous chapter that addressed the basic concepts of entity-relationship (ER) modeling. The chapter discussed the concepts of an entity, partnership, recurring relations, weak institutions and strong institutions. It also clarifies how these concepts can be represented in the ER diagram. Improved computer speed and memory, in recent years, have started developing sophisticated software applications like geographic information systems (GIS). The basic features of ER modeling are not enough to represent all concepts in such applications. To meet these needs, many different semantic data models have been proposed and some of the most important semantic concepts have been successfully incorporated into the original ER model. This chapter discusses and shows advanced ER modeling concepts, namely expertise/generalization, aggregation and structure. Reference This chapter continues to address database design concepts from top to bottom. Like previous chapters, this database closely links with other chapters on design, normalization and other design topics. The module on performance tuning in the chapter also has considerable relevance to the content, such as chapters on sequencing, as decisions made during database design have a big impact on the performance of the application. Brief on previous concepts In the previous chapter, we discussed the basic concepts of ER modeling. This section revises some important concepts. Institutions can represent a range of people, things, events, places or concepts within an area under consideration. An entity can have one or more features or features. Two notations are common to represent an entity: box notation, and notation that employs ellipses to represent attributes related to an entity. Figure 7.1 Figure 7.2 Relationship types These express the number of entities with which another entity can be connected through a relationship. The relationships that exist between the two

entities can be classified by the following: • One-to-one Figure 7.3 • One-to-many Figure 7.4 • The many-to-many Figure 7.5 relationship partnership defines whether it is mandatory or optional for an entity to participate in a relationship. It is also known as the membership class of the relationship. There are two types Terms: Mandatory and optional. Most entities are involved in binary relationships, so it follows that there are four main types of subscription relationships: mandatory Figure 7.6 for both entities mandatory for one entity, optional for the other Figure 7.7 optional for one entity, Mandatory Figure 7.9 note for the other Figure 7.8 optional for both entities: We have used one-to-many relationship types to indicate participation. Refer to the previous chapter for more information on how to model partnerships for other relationship types. Specialization/Generalisation We have discussed a variety of relationships that can take place between the institutions. Some entities have relationships that form a hierarchy. For example, a shipping company may have a variety of ships for their business. The relationship that exists between the concept of the ship and the specific type of ships creates a hierarchy. The ship is called superclass. Specific types of vessels are called subclasses. Superclass: A unit type that represents a general concept at a higher level. Subclass: A unit type that represents a specific concept at lower levels. A subclass is called heirs from a superclass. A subclass can inherit from several superclasses in the hierarchy. When a subclass inherits from one or more superclasses, it inherits all their characteristics. In addition to inherited characteristics, a subclass can also define its distinctive characteristics. A subclass also inherits participation in the relationship set in which its superclass (high level unit) participates. The process of making superclasses from a group of subclasses is called normalization. The process of creating subclasses from the general concept is called specialization. Specialization: A means of identifying sub-groups within a unit set that have features that are not shared by all entities (up-down). Normalization: Many unit sets are synthesized in high-level unit sets based on common features (bottom-up). A diamond notation representing specialization/generalization in ER diagrams is a common representation of specialization/normalization relationships in ER diagrams. Figure 7.10 As an example, let's consider the following scenario: Africa holds many historical artefacts in different places. Each artifacts are placed in a specific place. A location can be a point, province, country or sub-region of Africa. The landscape is a specialization relationship between different specific types of location and locations (i.e. point, province, country and sub-region). This specialization relationship is represented in the ER diagram below. Figure 7.11 To demonstrate the generalization, let's imagine that an artifact is one of the examples of African cultural objects. Another type of a cultural item is an artist. It is clear to see that a cultural item is an artwork and a superclass of artist. This normalization relationship can be represented as a show in the ER diagram Figure 7.12 Constraints on specialization/normalization There are three constraints that may apply to specialization/normalization: constraints of membership, unrelated constraints and barriers to perfection. Membership constraints define condition: Membership requirements of a specialization/normalization relationship can be defined as a condition in a condition such as tanker is a ship where cargo = oil user defined: Sometimes designers can define superclass-subclass relationship. This can be done to simplify design models or represent a complex relationship that exists between entities. Unrelated constraints disjointed: The unrelated barrier applies only if a superclass has more than one subclass. If subclasses are disjointed, a unit event can only be a member of one of the subclasses, such as postgrads or undergrads – you can't have both. To represent an unconnected superclass/subclass relationship, 'or' is used. Figure 7.13 Overlapping: This applies when a unit event can be a member of more than one subclass, such as students and employees - some people are both. 'and' is used to represent overlapping expertise/normalization relationship in the ER diagram. Figure 7.14 Completion constraints total: Each superclass (high level unit) must belong to the subclass (lower level unit set), for example a student must be postgraded or undergraded. To represent perfection in the specialization/generalization relationship, the keyword 'mandatory' is used. Figure 7.15 Partial: Some superclasses may not belong to subclasses (lower-level unit sets), for example some people in UCT are neither students nor employees. The keyword 'optional' is used to represent a partial specialization/normalization relationship. Figure 7.16 We can show both unrelated and completeness constraints in the ER diagram. Following our examples, we can add disjointed and perfection barriers. Figure 7.17 Some members of a university are both students and employees. Not all members of the university are staff and students. Figure 7.18 A student at the university must be either a graduate or a postgraduate, but not both. Relational Table Specialization/Generalization Relationship Mapping Specialisation/Generalization Relationship can be mapped to relational table in three ways. To demonstrate the methods, we will take student, postgraduate and graduate relationships. The university has a student's registration number and a name. Only postgraduate students have supervisors. Graduation collects marks through his course. Figure 7.19 Method 1 All entities in the relationship are mapped to different tables. Students (Regno, Name) Posgrad (Regno, Supervisor) Undergrad (Regno, Digit) Method 2 are mapped only to subclass tables. The features in the superclass are duplicated in all subclasses. PosGrad (Regno, Name, Observer) UnderGrad (Regno, Name, Digit) This method is most preferred when the legacy is unrelated and complete, such as every Either Posgrad or Undergrad is there and no one has both. Method 3 Only superclass is mapped to a table. Properties in subclasses are moved to superclasses. Student (Regno, name, supervisor, marks) This method will introduce zero values. When we enter the graduation record in the table, the observer column value will be invalid. Similarly, when we put the PG record in the table, the points value will be zero. Review question 1 Discuss the specialization/generalization relationship in ER modeling. Review Question 2 Explain the three difficulties that can be applied to the specialization/generalization relationship. Aggregation represents a 'is-a' relationship between unit types, where one represents the 'whole' and the other 'part'. An example of aggregation is car and engine entities. A car is made of an engine. The car is complete and the engine is part. Aggregation does not represent strong ownership. That means, a part can exist on its own without the whole. There is no strong ownership between a car and an engine. The engine of a car can be transported to another car. Aggregation in ER diagrams is finally represented by a row with diamonds to represent aggregation. Figure 7.20 The 'whole' part should be placed at the end of the diamond. For example, the car-engine relationship will be represented as shown below: Figure 7.21 The structure structure is a form of aggregation that represents a collaboration between entities, where there is a strong ownership between 'whole' and 'part'. For example, a tree and a branch have a structure relationship. One branch is a 'whole' part of the tree - we cannot cut the branch and add it to another tree. A row with a diamond filled at the end is used to represent the structure representing the structure in the ER diagrams. Figure 7.22 The example of a tree-branch relationship can be shown as shown below: Figure 7.23 Review Question 3 Using an example, explain the concepts of aggregation and structure. Exercise 1 Draw er diagrams for a small database for a bookstore. The database will store information about books for sale. Each book has an ISBN, title, value and brief description. Each book is published by a publisher in a certain publishing year. For each publisher, the database keeps the name, address and phone number. Each book is written by one or more authors. For each author, the database maintains its OWN ID, name and a brief introduction. Each book is stored in exactly one warehouse with a special quantity. For each warehouse, the database maintains the warehouse name, location and phone number. Each book has one or more sellers, who can be either a companies (corporate seller) or individual (individual seller). For each company, the database maintains a name of the company, its address, its phone number (there can be more than one phone number, each with a number and a description) and its contact For each individual vendor, the database maintains a name, a phone number and an email address. A contact person whose company sells a book cannot sell the same book as an individual seller at the same time (he can sell other books as an individual seller). Additional Content - What is XML XML? In previous chapters, we introduced database technology and how it is used by businesses to store data in a structured format. XML (eXtensible Markup Language) has become a standard for structured data interchanges between businesses. It was formally ratified by the World Wide Web Consortium (W3C) in 1998. XML uses markup for plain text formatting. Markup refers to helpful information (tags) in text that give structure and meaning. We have shown how to use relational tables to represent entities and their characteristics. XML also supports the representation of entities and characteristics. In this section, we will introduce XML. Students are encouraged to study detailed books for more information. XML is a useful website for learning . Element is a building block of an element XML document. All elements are limited to <and=>. Element names are case sensitive and cannot include spaces. The representation of an element is shown below: <Element> .... An XML document can contain multiple elements, but must be a basic element. A root element is a basic element of all other elements. Figure 7.24 attribute elements can be properties. Attributes are specified by the name = value pairs inside the initial tag of an element: <Element attribute=value>.. </Element> All values of attributes are enclosed in double quotes. An element can have multiple features, but each attribute name can only be once. <Element attribute1=value1 attribute2=value2> Example to display XML to represent relational table records in XML, let's imagine we have a customer table that holds customer information. Figure 7.25 We can represent the information in XML: Figure 7.26 Explanation <?xml version=1.0 encoding=UTF-8?>: XML prolog. Prolog is used to specify the version of XML and the encoding used. This is optional, but if it appears in the document, it must be the first line in the document. Customer element: The customer is the basic element. Customer element: A customer element represents a tuple in the customers table. The table has three features, CUSTOMER_ID, name and location. In our XML, CUSTOMER_ID is represented as a characteristic of the customer element. The name and location are represented as the hair elements of the customer element. Note that we have repeated the customer element three times to capture three records in the Customer table. Document type definition XML technology sentence for writing well formed documents Specifies but does not apply the structure of the document. XML document authors are free to structure an XML document in any way they want. </Element> </Element> May be problematic when verifying the document. How many elements can the document contain? What elements should a document have? It is difficult to answer these questions unless we even specify the structure of the document. Document type definition (DTD) is used to define the structure of xml document. DTD specifies the following: Which elements can be. Can properties be an element/element? What sub elements/can be inside each element, and how often it should be. DTD Element Syntax: <! ELEMENT element (subelements-specification) > DTD specialty syntax: <! ATTLIST element (attributes) > DTD for XML we defined above can be defined as shown below: Figure 7.27 Explanation! DOCTYPE: Defines the basic element of the document that defines the customer element. <!ELEMENT>: Defines an XML element. The first element to be defined is the customer element. The customer element contains a hair element, the customer, indicated in parentheses. + Symbol means that the customer element can appear once or more under the customer element. The customer has two sub-elements, name and location. Name and location elements contain character data as a hair element. <! ATTLIST >: Defines the attribute. A feature of the typed character data in the customer element is the customer ID. Xml data should be exchanged between organizations. The name of the same element can have different meanings in different organizations, causing confusion over the exchange of documents. Specifying a unique string as an element name avoids confusion. A better solution is to use a unique name followed by an element name. Unique Name: Element-name adding a unique name to all element names can be cumbersome for long documents. To avoid using unique names on all documents for a long time, we can use xml namespaces. Figure 7.28 Namespace has been declared FB and has been launched for ' '. The namespace is URI. There are generic identifiers such as URI URLs. XQuery XQuery is a language for finding and removing elements and attributes from XML documents. The way SQL is for relational databases, XQuery is the query language for XML documents. For example, to display all the names of customers in the XML above, our XQuery will look like this: $x/Customer/Customer Return $x/Name Exercise 2 In Chapter 3, Introduction to SQL, we introduced the EMP table. Represent the record in the table in XML. XML. </IELEMENT>