**OODB: Ch. (25) Introduction to Object DBMSs**

The computer industry has seen signiﬁcant changes in the last decade. In database systems,

we have seen the widespread acceptance of RDBMSs for traditional business applications, such as order processing, inventory control, banking, and airline reservations.

However, existing RDBMSs have proven unsuitable for non-traditional business database applications. These applications include:

* computer-aided design (CAD);
* computer-aided manufacturing (CAM);
* computer-aided software engineering (CASE);
* network management systems;
* office information systems (OIS) and multimedia systems;
* digital publishing;
* geographic information systems (GIS);
* interactive and dynamic Web sites;

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

1. **Discuss the general characteristics of advanced database applications.**

**فى الكتاب مفيش اجابه صريحه للسؤال هوا بيعتمد على فهمك -- هو شارح الانواع كلها فأنا لخصتها علشان لو جة سؤال فى اى نوع منهم لوحده بس لو جة السؤال دة اكتب كل اللى تعرفه عن التطبيقات دى**

**☺**

**Computer-aided design (CAD):**

A CAD database stores data relating to mechanical and electrical design covering, for example, buildings, aircraft, and integrated circuit chips. Designs of this type have some common characteristics:  
 --> Design data is characterized by a large number of types.

--> Designs may be very large, perhaps consisting of millions of parts.

--> The design is not static but evolves through time.

--> Updates are far-reaching because of functional relationships, tolerances, and so on.

--> There may be hundreds of staff involved with the design and they may work in parallel on multiple versions of a large design.

**Computer-aided manufacturing (CAM):**

A CAM database stores similar data to a CAD system, in addition to data relating to discrete production (such as cars on an assembly line) and continuous production (such as chemical synthesis).  
There will also be applications that control various physical processes, such as opening valves and increasing the ﬂow of cooling systems.  
\*\* Characteristics:   
--> these applications are often organized in a hierarchy.  
 --> These applications must respond in real time.

--> These applications must be capable of adjusting processes to maintain optimum performance within tight tolerances.

--> The applications use a combination of standard algorithms and custom rules to respond to different conditions.

**Computer-aided software engineering (CASE):**

A CASE database stores data relating to the stages of the software development lifecycle: planning, requirements collection and analysis, design, implementation, testing, maintenance, and documentation.

\*\* Characteristics:

-->software conﬁguration management tools allow concurrent sharing of project design, code, and documentation.

**Network management systems:**

Network management systems coordinate the delivery of communication services across a computer network.

\*\* Characteristics:

--> These systems perform such tasks as network path management, problem, management, and network planning.

--> These systems also handle complex data and require real-time performance and   
continuous operation. For example, a telephone call might involve a chain of  
network switching devices that route a message from sender to receiver, such as:

Node ⇔ Link ⇔ Node ⇔ Link ⇔ Node ⇔ Link ⇔ Node

**Office information systems (OIS) and multimedia systems:**

An OIS database stores data relating to the computer control of information in a business including electronic mail, documents, invoices, and so on. To provide better support for this area, we need to handle a wider range of data types other than names, addresses, dates, and money. Modern systems now handle free-form text, photographs, diagrams, and audio and video sequences.

**Digital publishing:**

It is becoming possible to store books, journals, papers, and articles

electronically and deliver them over high-speed networks to consumers.

Digital publishing is being extended to handle multimedia documents consisting of text, audio, image, and video data and animation.

**Geographic information systems (GIS):**

A GIS database stores various types of spatial and temporal information, such as that used in land management and underwater exploration.

Much of the data in these systems is derived from survey and satellite photographs, and tends to be very large.

**Interactive and dynamic Web sites:**

Consider a Web site that has an online catalog for selling clothes. The Web site maintains a set of preferences for previous visitors to the site and allows a visitor to:

--> Browse through images of the items in the catalog and select one to obtain   
a full-size image with supporting details.

--> Search for items.

--> Obtain a 3D rendering of any item of clothing.

--> Select accessories from items presented in a sidebar.

--> view a running total of the bill.

There is a need to handle multimedia content (text, audio, image, video data, and animation) and to interactively modify the display based on user preferences and user selections.

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**2- Discuss why the weaknesses of the relational data model and relational DBMSs may   
 make them unsuitable for advanced database applications.**

***The weaknesses of the relational data model and relational DBMSs:***

1. **Poor representation of ‘real world’ entities:**

The fragmentation of a ‘real world’ entity into many relations is inefficient.

2. **Semantic overloading :**

There is no mechanism to distinguish between entities and relationships, or to distinguish between different kinds of relationship that exist between entities. If such distinctions could be made, then it might be possible to build the semantics into the operations. It is said that the relational model is semantically overloaded.

3. **Poor support for integrity and general constraints:**

Integrity refers to the validity and consistency of stored data. Integrity is usually expressed in terms of constraints, which are consistency rules which are consistency rules that the database is not permitted to violate (تنتهك). Unfortunately, many commercial systems do not fully support these constraints.

4. **Homogeneous data structure:**

The relational model assumes both horizontal and vertical homogeneity. Horizontal homogeneity means that each tuple of a relation must be composed of the same attributes. Vertical homogeneity means that the values in a particular column of a relation must all come from the same domain. Further, the intersection of a row and column must be an atomic value. This structure is too restrictive for many ‘real world’ objects that have a complex structure, and it leads to unnatural joins, which are inefficient as mentioned above (point 1).

5. **Limited operations:**

The relational model has only a ﬁxed set of operations, such as set and tuple-oriented operations.

6. **Difficulty handling recursive queries:**

Atomicity of data means that repeating groups are not allowed in the relational model. As a result, it is extremely difficult to handle recursive queries.

7. **Impedance mismatch:**

A set of problems encountered when using a relational database to store (the state of) objects from software written in an object-oriented programming language.

8. **No Concurrency:**

Concurrency control in Transactions in business processing is generally short-lived.

9. **Schema changes are difficult**:

Organizations are unable to make these changes because they cannot afford the time and expense required to modify their information systems.

10. **RDBMSs are poor at navigational access**:

Navigational access is access based on movement between individual records. Navigational access is important for many of the complex applications.

ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**3- Deﬁne each of the following concepts in the context of an object-oriented data model:**

**(a) Abstraction, encapsulation, and information hiding.**

**(b) Objects and attributes.**

**(c) Object identity.**

**(d) Methods and messages.**

**(e) Classes, Subclasses, Superclasses, and Inheritance.**

**(f) Overriding and overloading.**

**(g) Polymorphism and dynamic binding.**

**(h) Complex objects.**

The main concepts that occur in object-orientation data model**:**

**(a) abstraction, encapsulation, and information hiding:**

--> Abstraction is the process of identifying the essential aspects of an entity and ignoring the unimportant properties.

--> Encapsulation used to hide the values or state of an object inside a class to prevent unauthorized access to them.

--> Information hiding means that we separate the external aspects of an object from its internal details, which are hidden from the outside world. In other words information hiding provides a form of data independence.

-->There are two views of encapsulation: the object-oriented programming language (OOPL) view and the database adaptation of that view.

--> In OOPL view an object has an interface part and an implementation part.

* The interface part is the specification of the set of operations that can be performed on the object.
* The implementation part has a data part and a procedural part.

--> The data part is the representation or state of the object.

-- > The procedure part describes, in some programming language,  
 the implementation of each operation

--> In the database view, proper encapsulation is achieved by ensuring that programmers have access only to the interface part. In this way encapsulation provides a form of logical data independence.

**(b) Objects and attributes:**

* In object-oriented systems, each real world entity is represented by an object to which is associated a state and a behavior.
* **The state** is represented by the values of the object's attributes.
* **The behaviour** is defined by the methods acting on the state of the object.
* The current state of an object is described by one or more attributes (instance variables).

--> A simple attribute can be a primitive type such as integer, string, real, and so on.

--> A complex attribute can contain collections and/or references.

--> A reference attribute represents a relationship between objects and contains a  
 value, or collection of values, which are themselves objects.

* An object that contains one or more complex attributes is called a complex object

**(c) Object identity**

* Each object is identified by a single OlD (Object Identifier).
* By using the OlD, objects can share other objects.
* **example on object sharing:**
* a Person has a name, an age and a set of children. Assume Peter and Susan both have a 15-year-old child named John. In a system without identity, Peter is represented by: (peter, 40, {(john, 15, {})}) and Susan is represented by: (Susan, 41, {(john, 15, {})}). In an identity-based model, these two structures can share the common part (john, 15, {}).
* **example on object updating :**
* Object updates: assume that Peter and Susan are indeed parents of a child named John. In this case, all updates to Susan's son will be applied to the object John and also to Peter's son.
* Very often a relation can have several alternative keys, called **candidate keys**, and the key which is actually chosen as the key of the relation is known as the **primary key**. In order to maintain correlations between the tuples of different relations **external keys** are used.
* A key consists of the value of one or more attributes and can be modified, whereas an OlD is independent from the state of the object.
* a key is unique within a relation, whereas the OlD is unique within the entire database.
* **advantages to using OIDs as the mechanism for object identity:**

1. They are efficient: OIDs require minimal storage within a complex object.
2. They are fast: OIDs point to a location within a table quickly.
3. They cannot be modiﬁed by the user:
4. They are independent of content :
5. Since OlDs are implemented by the system, Better performance is obtained.

* **Disadvantages of OIDs**--> minimal semantic significance.
* **Identity and Equality**
* Two objects are identical if they have the same OID.
* '=', is identity equality.
* Two objects are equal if they have the same state.
* '==', is value equality.
* Shallow equality: if object states contain the same values
* Deep equality: if object states contain the same values and if related objects also contain the same values.

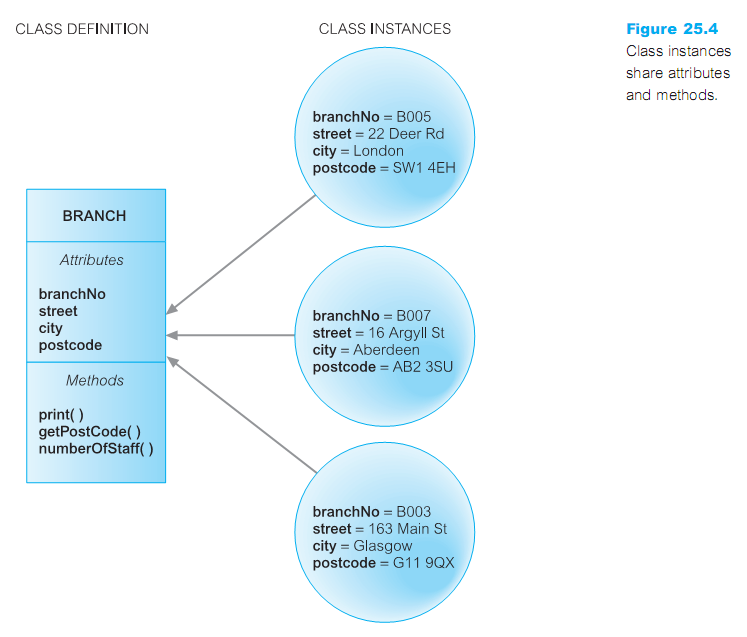
**(d) Methods and messages**

* **Methods** deﬁned the behavior of the object. They can be used to change the object’s state by modifying its attribute values
* A method consists of a signature which specifies the name of the method and a body which consists of a block of code that carries out the required functionality.
* **Message** is simply a request from one object (the sender) to another object (the receiver) asking the second object to execute one of its methods. The sender and receiver may be the same object.

**(e) Classes, subclasses, superclasses, and inheritance:**

* **Classes**

1. Objects that have the same attributes and respond to the same messages can be grouped together to form a class.
2. The objects in a class are called instances of the class.
3. Each instance has its own value(s) for each attribute, but shares the same attribute names.



* **Subclasses, Superclasses and inheritance:**

1. The concept of inheritance is "reusability".
2. Inheritance allows one class to be deﬁned as a special case of a more general class. These special cases are known as subclasses and the more general cases are known as superclasses.
3. With inheritance:

--> Objects of subclass belong automatically to superclass.

--> Attributes and methods are inherited from superclass.

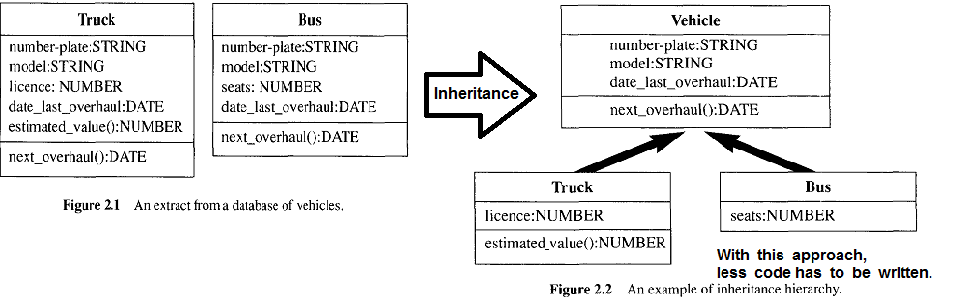
--> Subclass can introduce new attributes and methods.

1. Advantages:

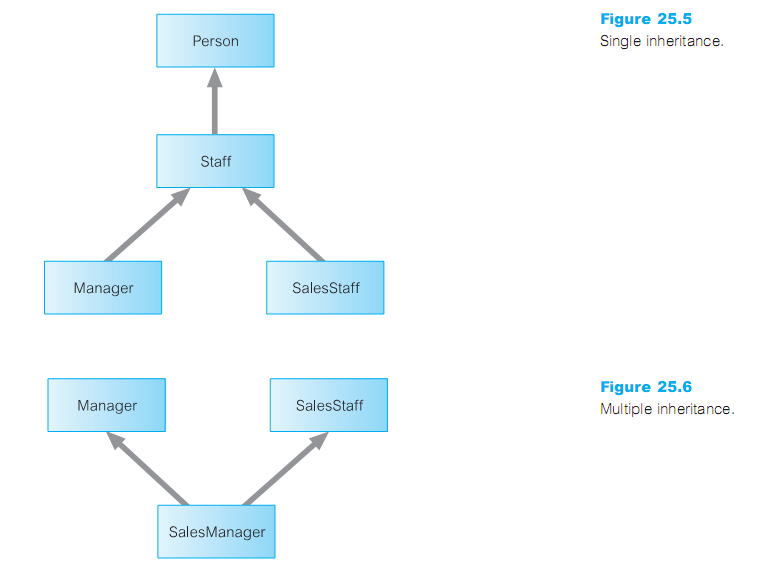
--> Powerful modeling tool because it gives a concise and precise

description of the world.  
 --> Reuse of specification and implementation.

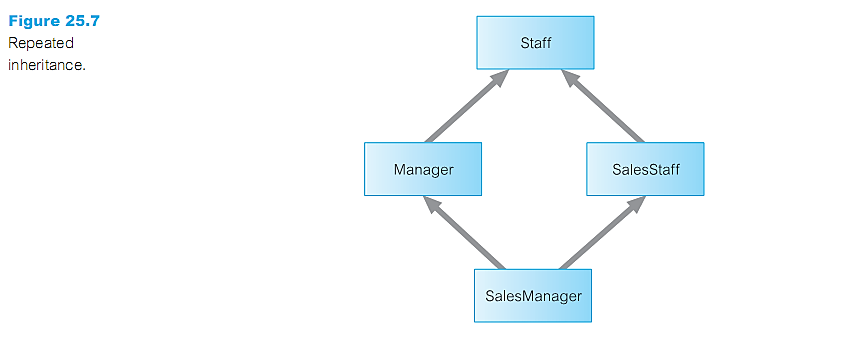
--> With inheritance, less code has to be written.



1. There are several forms of inheritance: single inheritance, multiple inheritance, repeated inheritance, and selective inheritance.
2. **Single inheritance**: the subclasses inherit from no more than one superclass.
3. **Multiple inheritance:** the subclasses inherit from more than one superclass.



1. **Repeated inheritance**: is a special case of multiple inheritances where the superclasses inherit from a common superclass.



1. **Selective inheritance**: allows a subclass to inherit a limited number of properties from the superclass.

**(f) Overriding and overloading:**

* Properties (attributes and methods) are automatically inherited by subclasses from their superclasses. However, it is possible to redeﬁne a property in the subclass. In this case, the deﬁnition of the property in the subclass is the one used. This process is called **overriding**.

**overloading**

--> Effect caused by method overriding   
--> let the system pick the appropriate implementation at run time

**(g) polymorphism and dynamic (late) binding:**

* Overloading is a special case of the more general concept of polymorphism.
* Polymorphism in Greek meaning ‘having many forms’.
* There are three types of polymorphism: operation, inclusion, and parametric.
* Overloading is a type of operation polymorphism.
* A method deﬁned in a superclass and inherited in its subclasses is an example of inclusion polymorphism.
* Parametric polymorphism, or generosity as it is sometimes called, uses types as parameters in generic type, or class, declarations.
* **Dynamic (Late) Binding**
* -->The system cannot bind operation names to programs at compile time. Therefore, operation names must be resolved at run-time. This delayed translation is called "late binding".

**(h) Complex objects:**

**Complex object**

Formed from simpler ones by constructors.

* Constructor in a class: a sequence of program instructions used to create an object.
* The simplest objects are objects such as integers, characters, strings, Booleans and floats.

**The complex object**

Constructors such as tuples, sets, bags, lists, and arrays.

* The object constructors must be orthogonal: any constructor should apply to any object For example, we should be able to use not only SET(TUPLE()) and LIST(TUPLE()) but also TUPLE(SET()) and TUPLE(LIST())..
* The main disadvantage of using complex values is that they mean that the data model is conceptually more complicated.

ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**4- Discuss the difﬁculties involved in mapping objects created in an object-oriented   
 programming language to a relational database. (السؤال دا فيه رغى كتير واكواااااااد الناس تبقى تبص عليه فى**

**الكتااااب )**

* **Mapping Classes to Relations:**
* There are a number of strategies for mapping classes to relations.

(1) Map each class or subclass to a relation  
 (2) Map each subclass to a relation.

(3) Map the hierarchy to a single relation.

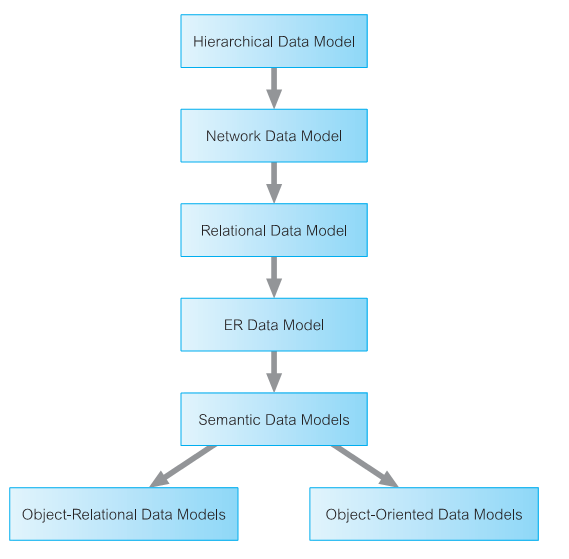
**--> the difficult of using these strategies are that we have lost semantic information.**

* **Accessing Objects in the Relational Database:**
* Having designed the structure of the relational database, we now need to insert objects into the database and then provide a mechanism to read, update, and delete the objects.
* **Object-Relational Impedance Mismatch:**
* a set of problems encountered when using a relational database to store (the state of) objects from software written in an object-oriented programming language.
* Code to map between models is considerable overhead, costly and hard to maintain

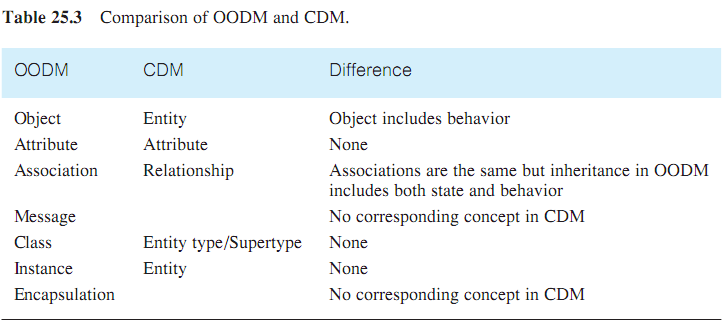
ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**5- Describe the three generations of DBMSs.**

* **First Generation:**
* The ﬁrst approach was based on the hierarchical data model by IBM.
* The second approach was based on the network data model.
* disadvantages of the two model:
* Complex programs had to be written to answer even simple queries.
* There was minimal data independence.
* There was no widely accepted theoretical foundation.
* **Second Generation:**
* Relational DBMSs are referred to as second-generation DBMSs
* RDBMSs have their failings, particularly their limited modeling capabilities.
* The attempts to provide a data model that represents the ‘real world’ more closely have been loosely classiﬁed as semantic data modeling. Some of the more famous models are:
* The Semantic Data Model.
* The Functional Data Model
* The Semantic Association Model
* **Third Generation:**
* In response to the increasing complexity of database applications, two ‘new’ data models have emerged: the Object-Oriented Data Model (OODM), and the Object-Relational Data Model (ORDM), previously referred to as the Extended Relational Data Model (ERDM).



**\*\*\* Comparison of Object-Oriented Data Modeling (OODM) and Conceptual Data   
 Modeling (CDM). الجزئيه دى مش موجوده فى الاسئله اللى ف اخر الشابتر بس مهمه و ممكن تيجى**

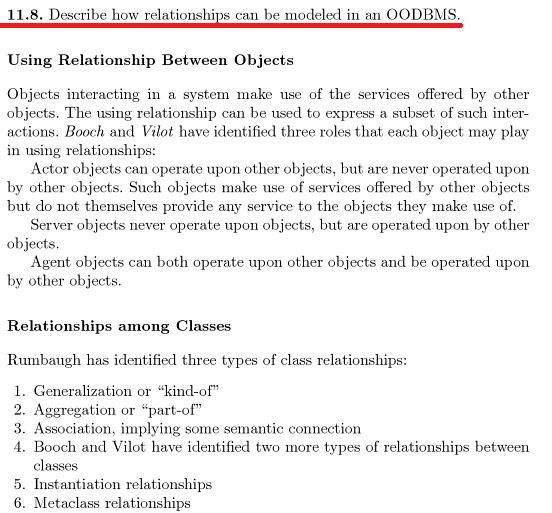
****

* The main difference is the encapsulation of both state and behavior in an object, whereas CDM captures only state and has no knowledge of behavior. Thus, CDM has no concept of messages.

**ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ**

**6- Describe how relationships can be modeled in an OODBMS.**

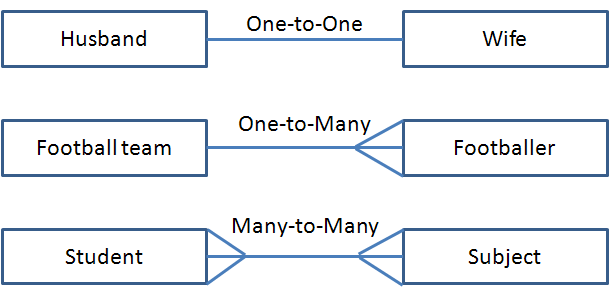
الاجابه دى مش موجوده فى الكتاب موجوده فى كتاب تانى لقيته ع النت



**و دى اجابه الكتاب علشان لو السؤال جيه مباااشر**

* Relationships are represented in an object-oriented data model using reference attributes. typically implemented using OIDs.
* relationships types:

**-->** One-to-one (1:1) **-->** One-to-many (1 :\*) **-->** many-to-many (\* :\*)



* **Referential integrity** (التكامل المرجعي)

Is a database concept that ensures that relationships between tables remain consistent.

* There are several techniques that can be used to handle referential integrity:
* *Do not allow the user to explicitly (بشكل صريح) delete objects*. the system automatically deletes objects when they are no longer accessible by the user.
* *Allow the user to delete objects when they are no longer required*. the system may detect an invalid reference automatically and set the reference to NULL.
* *Allow the user to modify and delete objects and relationships when they are no longer required*. the system automatically maintains the integrity of objects.
* **Behavioral Design:** is a technique for analysis of the processing requirements of the enterprise (project).
* In object-oriented analysis, the processing requirements are mapped on to a set of methods.
* The methods that are visible to the user or to other objects **(public methods)** must be distinguished from methods that are purely internal to a class **(private methods).**
* The three types of public and private method:   
  1- constructors and destructors. 2- Access. 3- Transform.
* *Constructors and destructors:* Constructor methods generate new instances of a class and each new instance is given a unique OID. Destructor methods delete class instances that are no longer required.
* *Access methods:* Access methods return the value of an attribute or set of attributes of a class instance. It may return a single attribute value, multiple attribute values, or a collection of values.
* *Transform methods:* Transform methods transform the state of a class instance.
* **Identifying methods:** There are several methodologies for identifying methods
* Identify the classes and determine the methods that may be usefully provided for each class.
* Decompose (حلل) the application in a top-down fashion and determine the methods that are required to provide the required functionality.

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**7- Describe the different modeling notations in the UML.**

* The UML (Uniﬁed Modeling Language) is commonly deﬁned as ‘a standard language for specifying, constructing, visualizing, and documenting the artifacts of a software system**.**
* The primary goals in the design of the UML were to:
* Provide users with a ready-to-use, expressive visual modeling language.
* Provide extensibility and specialization mechanisms to extend the core concepts.
* Be independent of particular programming languages and development processes.
* Provide a formal basis for understanding the modeling language.
* Encourage the growth of the object-oriented tools market.
* Support higher-level development concepts such as collaborations, frameworks, patterns, and components.
* Integrate best practices.
* **UML Diagrams :** UML deﬁnes a number of diagrams, of which the main ones can be divided into the following two categories:
* Structural diagrams, which describe the static relationships between components. These include:
* Class diagrams.
* Object diagrams.
* Component diagrams.
* Deployment diagrams.
* Behavioral diagrams, which describe the dynamic relationships between components. These include:
* Use case diagrams.
* Sequence diagrams.
* Collaboration diagrams.
* State chart diagrams.
* Activity diagrams.

في الكتاب شارح كل ديجرام لوحده   
مش عارف بقي الدكتور ممكن يجيب في الامتحان ارسم ديجرام ولا لا !!!

ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**With My Best Wishes   
 "Mohamed Noaman"**