

COMP67321

Introduction to Object-oriented Databases

Goran Nenadic

School of Computer Science
University of Manchester

1

Brief overview of OO concepts

2

Brief overview of OO concepts

- Classes and objects
- Object identity
- Encapsulation and information hiding
- Inheritance
- Polymorphism

3

Classes and objects

- A class is a construct used to describe the structure of objects generated from the class
 - can be an abstract data type equipped with a possible partial implementation
- An object is an instance of a class
 - objects are referred to as class instances
- Note
 - some languages make distinction between types (*data types*) and classes (*collections*)

4

Object identity (OID)

- An object has an existence and identity independent of its value
 - Object Identity (OID) is unique
 - not visible to the users, but used internally to identify each object
- Note
 - there are also *values*, which do not have OIDs
 - cannot be referenced (e.g. constants)

5

Type constructors

- Used to construct state (current value) of a complex object from other objects
- object = (OID, type constructor, current state)
- Basic constructors
 - atom: represent all basic atomic values (numbers, strings, Booleans, etc)
 - tuple (i.e. *struct*)
 - collection types: set, list, bag, array

6

Type constructors – example

Student (OID3, tuple<name:OID1, student-id: OID2>)

name (OID1, atom, John)

student-id (OID2, atom, S007)

Note: **identical** object states vs. **equal** object states

name (OID4, atom, John)

7

Encapsulation

- Define the behavior of objects based on the operations that can be externally applied to them
 - internal structure of objects is hidden
- Encapsulation: separation of the objects of a class from anything outside of the class
 - information hiding
- Stipulate a *public interface* to a class and hide the “secrets” in the private part

8

Encapsulation

- Public interface is a collection of *signatures* of the functions provided by a class
 - signature specifies the operation’s name, parameters, and return value class
 - implementation (*method*) of operations is hidden
- Objects of a class can only be used through the public interface of the class
 - an object is known to the outside world by its name and signatures published in its public interface

9

Encapsulation

- Visible and hidden attributes and methods
 - hidden attributes completely encapsulated and can be accessed only through methods
- Update is a typical encapsulated method
 - create, delete, update
- Benefits
 - implementations of class functions can be changed without affecting the clients who use the class functions

10

Inheritance

- Class hierarchy
 - definition of new class based on previously defined ones
 - PERSON: Name, Address, Birthdate, Age
- Subtype
 - a new type that is similar but not identical to an already defined type
 - STUDENT subtype-of PERSON: Program, Average
- Supertype
 - inherits all the functions of the subtype

11

Inheritance

- Single inheritance
 - each type has a single supertype
 - type hierarchy is a tree
- Multiple inheritance
 - types can have multiple supertypes
 - problems: name conflicts, semantics, exception handling
- Inheritance can be used to support code resusage

12

Polymorphism

- Operator overloading
 - the same *operator name* or *symbol* bound to two or more different *implementations* of the operator, depending on the type of objects to which the operator is applied
 - example: operator '+' for numbers, strings, sets
- Binding of an object to a function
 - static (early): during the compile time
 - dynamic (or late or runtime) otherwise
 - Power of dynamic binding is that an object can have a dynamic form whose functions can be determined at runtime

13
