DatabaseDescription with SDM: A Semantic Database Model
M. Hammer and D. McLeod
1. Summarise the various forms of interclass connections supported in SDM and show how each of these could be used in a university database example with information about students, professors, courses etc.

   a) Subclass connection

   Members of a nonbase class S are of the same basic entity type as those in the parent class. The subclass contains some, but not necessarily all of the members of the parent class.

   A subclass S is defined by specifying the parent class and a predicate.

   Predicates:

   1) Attribute defined subclass

   Predicate on the member attributes of the parent class to indicate which of its members are also members of the subclass

   COURSES
   
   member attributes:
   
   Name
   Specialization
   Language

   MATH_ENGLISH_COURSES
   
   interclass connection: subclass of COURSES where Specialization='maths' and Language='english'
2) **User controllable subclass**
The subclass contains at all times only entities members of the parent class.

CANCELLED_COURSES
   interclass connection: subclass of COURSES where specified

3) **Set operator defined subclasses** (intersection, union, difference)

PHD_STUDENTS
   interclass connection: subclass of PERSONS where is in STUDENTS and is in PROFESSORS

4) **Existence subclass**
A subclass S consists of the members of C that are values of some attribute of another class

DEPARTMENTS
   member attributes
   Dean
   value class: PROFESSORS

DEANS
   interclass connection: subclass of PROFESSORS where is a value of Dean of DEPARTMENTS
b) **Grouping Connection**
A grouping class is a class whose members are taken from an underlying class

1) **expression-defined grouping class**
The grouping expression specifies how the members of the underlying class are placed into groups. For instance, a grouping on a common value of an attribute.

**COURSES**
interclass connection: grouping of COURSES on common value of Specialization

2) **Enumerating grouping class**
Consists in providing a list of classes members of the grouping class
Classes: INFORMATIK_STUDENTS, NACHDIPLOM_STUDENTS, IIIE_STUDENTS

**TYPES**
interclass connection: grouping of STUDENTS consisting of classes INFORMATIK_STUDENTS, NACHDIPLOM_STUDENTS, IIIE_STUDENTS

3) **User-controllable grouping class**
consists of a collection of user-controllable subclasses

**SA**
interclass connection: grouping of STUDENTS as specified
2. The paper presents the various features of attributes (member and class) that can be represented in SDM. Again, use the example of a university database to show the use of these features in an application other than that presented in the paper.

a) **Attribute name** – must be unique with respect to the set of all attributenames in the class, base class and subclasses of the base class
   Attribute Last_name of class PERSON

b) **Attribute value** – either an entity or a collection of entities. The value is taken from the value class
   Value class of attribute Dean in DEPARTMENTS is TEACHER

c) **Applicability**
   - *Member attribute* applies to each member of the class
     Office for PROFESSOR
   - *Class attribute* applies to the class as a whole
     Number for DEPARTMENTS
d) Attribute description
DEPARTMENTS
  member attributes
  Dean
    description: the dean of the department
    value class: PROFESSORS

e) single/multivalued
STUDENTS
  member attributes
  Assistants
    description: the assistants of the student
    value class: STUDENTS  //can be a Phd or a normal student

f) Mandatory – not null

g) Not changeable – once set to a nonnull value cannot be altered

h) Non overlapping – the values of the attribute for 2 different entities have no entities in common
STUDENTS
   member attributes
   Legi_Number
      value class: STRINGS
      may not be null
      not changeable
      non overlap in values

Member Attribute Interrelationships
Inversion
STUDENTS
   member attributes
   Courses
      value class: COURSES
      inverse: Attendants

COURSES
   member attributes
   Attendants
      value class: STUDENTS
      inverse: Courses
Matching

COURSES
member attributes
Participants
  value class: STUDENTS
  match: Student of PARTICIPATION on Course

PARTICIPATION
member attributes
Student
  value class: STUDENTS
Course
  value class: COURSES
identifiers: Student +Course
Mappings
The concatenation of attribute names that allows a user to reference the value of an attribute of an attribute.

DEPARTMENTS
  member attributes
    Dean
      description: the dean of the department
      value class: PROFESSORS

“Dean.Last_name”

Member attribute derivation
– the value of an attribute can be computed from other information in the database

Primitives:
- ordering, existence, recursive derivation, contents, same as, subvalue of, intersection, union, difference, arithmetic expressions, maximum, minimum, average, sum, number of
Class Attribute Interrelationships
- Attribute relation primitives can be used to define derived class attributes
Not deriving the value of the member attributes, but the value of class attributes

Attribute Predicates for Subclass Definition
MAPPING SCALAR_COMPARATOR CONSTANT
MAPPING SCALAR_COMPARATOR MAPPING
MAPPING SET_COMPARATOR CONSTANT
MAPPING SET_COMPARATOR CLASS_NAME
MAPPING SET_COMPARATOR MAPPING

Subclass GOOD_STUDENTS of STUDENTS where Average_mark>=5
3. Describe the model of subtyping and inheritance supported in SDM and how it relates to that of object-oriented programming with which you are familiar.

The notions of base and subclass is similar to these concepts in OO programming languages.
The overriding of attributes in subclasses is the same as in Java and C++.
Same as in Java, there is no multiple inheritance of classes. In Java, multiple inheritance is allowed by means of interfaces. In C++ multiple inheritance is allowed.

4. The paper states that “A database model must support a relativist view of the meaning of a database”. Explain what this means. What is the reason that the authors put forward for this? What features of SDM support this?

It means that the structure of the database should support alternative ways of viewing the same information. The schemata has to be flexible, logically redundant and integrated.

*Flexible* – allowing multiple and coequal views

*Logically redundant* – values of some components can be derived from others

advantages:- statical embedding of data values that would otherwise have to be repeatedly computed
- ease of development of new applications
Integration – schema describes the relationships and similarities between multiple views

Features of SDM supporting multiple views:
- inheritance
- different ways of modelling relationships (association between entities can be viewed as attribute of first entity, of second entity or as an entity itself).