**Data Models**

o A data model contains:

 o a set of notations to describe the data, and

 o a set of operations to manipulate the data.

o A design model contains:

 o a set of notation to model the applications, and

 o a set of design processes to follow (method).

 **The Entity-Relationship Model and Diagrams**

o ER Model is an object-oriented model.

oThere are many variations and extensions of the ER models.

o Approximately:

 Entities -- objects.

 Entity types -- classes.

 Entity set -- a collection of entities of the same type.

o Intension of an entity -- entity type and the definition of the entity set.

 Extension of an entity -- the entity set.

o Attributes of an entity -- representable properties and qualities.

 Domain of an attribute -- allowable values.

 o Same as range in Maths.

An attribute may be multi-valued. Some authors use double ovals for those attributes.

o Example:

 Entity: Employee.

 Type: A person.

 Entity set definition: A person currently employed at UHCL.

 Attributes:

 Name

 SS#

 Weight

 Skill (multi-valued)

 ...

 Domain of Weight: integer? accuracy? constraint?

 Domain of SS#: string of length 9? integer?

 A well-defined domain can help to detect errors.

o Basic ER Diagram notation:

 entities: rectangle.

 attributes: oval attached to the rectangle.

o An attribute may be multi-valued: represented as a double oval.

**Keys**

o Superkey: a set of attributes that uniquely identify an entity.

oSuperkeys depend on the semantics (assumptions) of the application, not a particular instance.

o Superkey does not need to be minimal.

o Candidate keys:

 (a) Superkeys

 (b) Minimal: no proper subset of a candidate is a superkey.

o There may be more than one candidate keys in an entities.

o Primary key:

 o A selected candidate key for the database.

 o May be underscored.

 o Some rules for selecting primary keys:

 (1)That will not change within the lifetime of the instance of the entity. (e.g. {NAME, ADDRESS} -- bad.)

 (2) That will not have null values. (e.g. {NAME, ADDRESS} -- bad.)

 (3)Avoid "smart" keys that indicates locations, classification, etc. They may change. Example: don't use the first two digit to indicate branches.

 (4)Consider substituting composite keys by surrogate key. e.g. Game\_No instead of {Date, Home\_Team, Visiting\_Team}.

o Example:

 Entity: student.

Attributes: SS#, Student\_#, Name, Street, City, State, Zip, Major\_Code, Phone, GPA.

 Superkeys: Any superset of {SS#}, any superset of {Student\_#}

 Candidate Keys: {SS#}, {Student\_#}

 Primary key: {SS#}

 Foreign key: {Major\_Code}

 Entity: major

 Attributes: Major\_Code, Name, School

Superkeys: {Major\_Code}, {Name}, {Major\_Code, School}, {Major\_Code, Name}, {Name, School}, {Major\_Code, Name, School}

 Candidate Keys: {Name}, {Major\_Code}

o Foreign keys:

 oA set of attributes that is a candidate key in another entity.

o Time stamping: including a timestamp.

 e.g. entity: Price\_History:

 attributes: effective\_date, item\_id, price.

**Relationships**

o Relationship (not relation) -- association between entities.

o Represented by a diamond connecting the entities involved.

o The role of every entity should usually be mentioned.

O Attributes may be included. A composite entity (gerund) is an entity representing a relationship. Symbol: a diamond inside a rectangle.

o Types:

 o binary: most common

 e.g. parent-child

 o ternary: less frequent

 e.g. Project-Employee-Resource (shared resource).

 e.g. Student-Class-Faculty in p153 of Ricardo -- may be replaced by 2 binary relationship: Student-Class, Faculty-Class.

 is not the same as three binary relationships.

e.g. Supplier-Part-Warehouse with attribute quantity. (all three are many)

 o n-ary (n>3): quite uncommon

o Cardinality of relationships:

 o Depend on the semantic of the relationship.

 Note that there may be more than one relationship between two entities.

 o1-1 (Husband-Wife, ), 1-many (Faculty-Class, Department-Faculty), many-1 (Student-Major, Employee-Employee (Manager)), many-many (Suppliers-Parts, Classes-Students, Employee-Department, etc)

 oOther variations: exact numbers (2 arms), minimum and maximum, possible values, mandatory (vertical bar) or not (small circle).

o Dependencies:

 o Existence dependency:

 o Total mapping.

 oEvery instance of the dependent entities must be involved in the relationship.

 oRicardo: double rectangles on the weak (dependent) entity (poor choice). Instead, specify the strong entity with a mandatory cardinality.

 oExamples:

 Employee-Department

 Patient-Patient\_History

Student-Major (existence dependency if every student must have a major; no undecided major).

 o Identifier dependency:

 o The weak entity has no candidate key (no identifier of its own).

 o Special case of existence dependency.

 o Used in OOD but not the relational model.

 o Referential dependency

 o The weak entity contains a foreign key of the strong entity.

 o Special case of existence dependency.

Example on p164 of Ricardo:

 o Study and ask simple questions.

 o Referential dependency between faculty (weak) and department.

 oNote the ambiguity of the double rectangle for faculty (Which relationship does it belong to? There are three.)

 oNote that Class-Faculty should also have a referential dependency.

 oNote that Chair-Member is a relationship between Faculty. It may better be between Department and Faculty. In the current setting: what happen to a department with one faculty? Who is the chair of the chair?

**Other ER diagram extensions:**

o Aggregation

 o A-Part-Of relationship.

 o Tree structure -- may be multi-levels.

 o ER diagram's representation: a rectangle (part) within a rectangle.

 Example: Car - (Engine, Carburetor, ...)

 Textbook

 Faculty - Evaluation (instead of identifier dependency)

 o A relationship may be a part of an entity.

 Example: Ricardo.

 Department -- Class, Student.

o Generalization and specialization

 o A-Kind-Of (ISA) relationship.

 o Graph Inheritance.

 o ER: inverted triangle.

 o Information about exclusiveness, disjointness may be added.

 o Generalization and aggregation are othrogonal.

**Classwork**

(1)The entity Student in an elementary school has the following attributes SS#, Name, Class and Status. It is known that there are two candidate keys: {SS#} and {Name, Class}.

 (a) Is it possible for two students to have the same name?

 (b) Is it possible for two students to have the same SS#?

 (c) List all superkeys of the entity Student?

(2) Draw the ER diagram for the following company.

 (a)The company owns several divisions. Each division belongs to the same company.

 (b)A division may operate several projects. Each project belongs to only one division. A division may operate no project.

 (c)A division has many employees. Each employee belongs to only one division. To be classified as a division, at least 25 employees must be assigned to it.

 (d)An employee may be assigned to many projects. Each project has at least one employee.

 (e)Each of the divisions is managed by one of its employees.

 (f)An employee may or may not have dependents.

 (g)An employee may or may not have specific technical job skills. Many employees may have the same skills.

 Have you made any additional assumptions in constructing your ER diagram?

**Solution To Classwork**

(1) (a) Yes

 (b) No

 (c) {SS#}, {Name, Class}, {SS#, Name}, {SS#, Class}, {SS#, Status},

 {SS#, Name, Class}, {SS#, Name, Status}, {SS#, Class, Status},

 {Name, Class, Status}, {SS#, Name, Class, Status}.

**Keys**

o Superkey: a set of attributes that uniquely *identify* an entity.

oSuperkeys depend on the semantics (assumptions) of the application, not a particular instance.

o Superkey does not need to be minimal.

o Candidate keys:

 (a) Superkeys

 (b) Minimal: no proper subset of a candidate is a superkey.

o There may be more than one candidate keys in an entities.

o Primary key:

 o A selected candidate key for the database.

 o May be underscored.

 o Some rules for selecting primary keys:

 (1)That will not change within the lifetime of the instance of the entity. (e.g. {NAME, ADDRESS} -- bad.)

 (2) That will not have null values. (e.g. {NAME, ADDRESS} -- bad.)

 (3)Avoid "smart" keys that indicates locations, classification, etc. They may change. Example: don't use the first two digit to indicate branches.

 (4)Consider substituting composite keys by surrogate key. e.g. Game\_No instead of {Date, Home\_Team, Visiting\_Team}.