THE ONTOLOGICAL WORLD-VIEW

Objects:

- An world is made of objects; objects have *static* (structural) and and *dynamic* (behavioral, time-dependent) properties. A database can represent both static and dynamic views of the world.
- Objects may be physical or abstract.

Structural Properties:

- Each object is described by ≥ 1 attributes, which form its structure. The attribute-value pairs describe a valid internal *state* of the object.

Relational Properties:

- An object may participate in (interact with) zero or more relationships with other objects of the same kind or different kinds.
- This may give rise to a larger complex of "emergent" objects, with zero or more *emergent* attributes (and behaviors) of the composition that are not present in any of the component objects.
- The relationships may change with time and these emegent objects then take the form of dynamic objects.

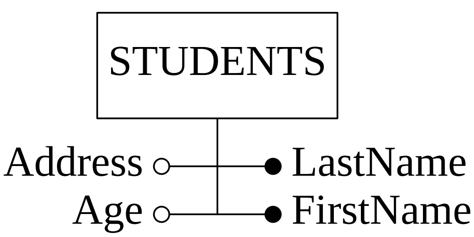
AN ENTITY-SET IN AN ER-MODEL

Entity-set E**:** Represents a set of entities of some type E.

- Each individual entity $e \in E$ is described by a common set of attributes *Attrb*(E) associated with E.
- A subset of *Attrb*(E), called the (primary) *key* or *key-attributes*, distinguishes each entity $e \in E$ from other $e' \in E$, $e' \neq e$. (The key is assumed to be minimal; no subset of it does the job.)
- Dom(E) is the cartesian product of Dom(A_i), $A_i \in$ Attrb(E).

Example 1.

- An entity-set STUDENTS, with 4 attributes and key = {LastName, FirstName}. An individual student is an element of STUDENTS.



STUDENTS
(*LastName, Firstname*, Address, Age)

(key attributes shown in italics)

STUDENTS

LastName	*FirstName*	Address	Age
Adams	John	Computer Sc., LSU	21
Lee	Peter	45 Parkins Rd., BR 70802	23
...	...	...	...

(No two students have the same {LastName, FirstName}.)

Question:

•? Why is {Age} not a suitable key? What can prevent {Address} being a key? What can prevent {Address, Age} being a key? Are there some kind of "Address" that can be taken as a key?

Example 2.

- An entity-set STUDENTS with 5 attributes and key = {S#}.

STUDENTS
(*S#*, LastName, FirstName, Address, Age)

STUDENTS

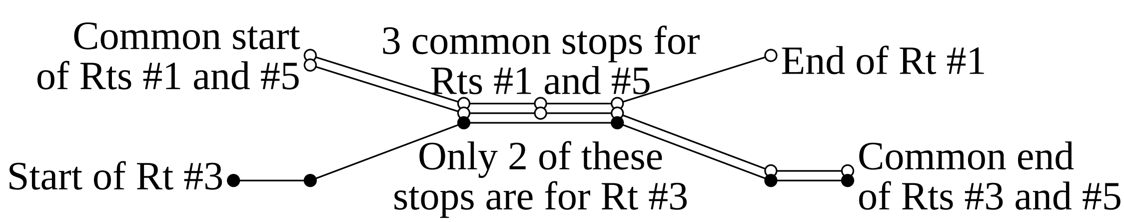
S#	LastName	FirstName	Address	Age
1023	Adams	John	Computer Sc., LSU	21
2115	Lee	Peter	45 Parkins Rd., BR 70802	23
...	...	...	...	...

(No two students have the same S#.)

> The choice of entity-sets and their attrbutes depend on the application.

EXERCISE

1. What is common to all elements e of an entity set E? What is different for any two e, $e' \in E$? What determines the key of E?
2. Design an entity-set for books in a typical University Library.
3. Consider bus routes in a city, which may have overlapping segments as shown in the example below. A *route-segment* is the part between two successive stops on it. Rt #1 has 4 segments and Rts #3 and #5 have 5 segments. Give a suitable entity-set (its attributes and the key) to represent such routes.



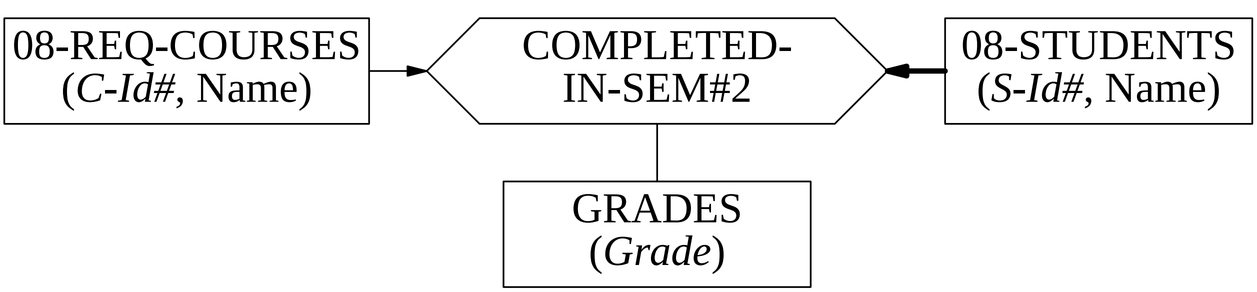
A RELATIONSHIP-SET IN AN ER-MODEL

Relationship-set R:

- Represents a particular type of connection among $n \geq 2$ entity-sets E_i (which need not be distinct). Each element $r = (e_1, e_2, \cdots, e_n)$ of the relationship-set is a combination of an $e_i \in E_i$ and represents a connection of type R among the entities e_i.
- There can be several (different) relationsship types connecting the same group of E_i's
- Not every combination $r = (e_1, e_2, \cdots, e_n)$ may be valid, i.e., $R \subseteq E_1 \times E_2 \times \cdots \times E_n$. In particular, an $e_i \in E_i$ may not participate in any $r \in R$. A subset $K = \{E_1, E_2, \cdots, E_k\}$ is called the (primary) *key-entities* of R if each $(e_1, e_2, \cdots, e_n) \in R$ can be distinguished by its components $(e_1, e_2, \cdots, e_k)$.
- R may have its own attributes to represent the result of the interaction of e_i's participating in $r = (e_1, e_2, \cdots, e_n) \in R$. If the same combination of e_i's interact more than once, then they must be distinguished by a subset of these attributes, called the (primary) *key-attributes* of R.

Example.

- Shown below is an ER-diagram, with a (ternary) relationship-set. Thick lines indicate *onto* relationship, the arrow indicates (primary) *key* entities of the relationship-set. None of 08-REQ-COURSES and 08-STUDENTS by itself can be a key (why).



EXERCISE

1. Consider a city-wide-bus-transportation system. There are any groups of users here:

    (1) Bus-drivers. They need to know their driving-schedule, that is, the routes they will serve during each work-day and the paricular bus they will drive. (You may assume for simplicity that each driver works on 8:00am-5:00pm schedule; and their is no bus-service on week-ends and outside of 8:00am-5:00pm.)

    (2) Bus-passengers. They need to know the bus-schedule for each stop on each route. (Here, passengers are abstract objects, which interact with the transporation system in the sense that their interests are served but no specific physical person is of interest to us.)

    (3) Bus-repair shops. (The managment needs to know which repair works are carried out on which bus, the repair costs, the out-of-service time periods for the busses due to the repair-works, etc)

    (4) Managers. They do the work-assignment of drivers, assign busses to routes, create bus-time-schedules for different stops for different routes, control repair works and costs, etc.

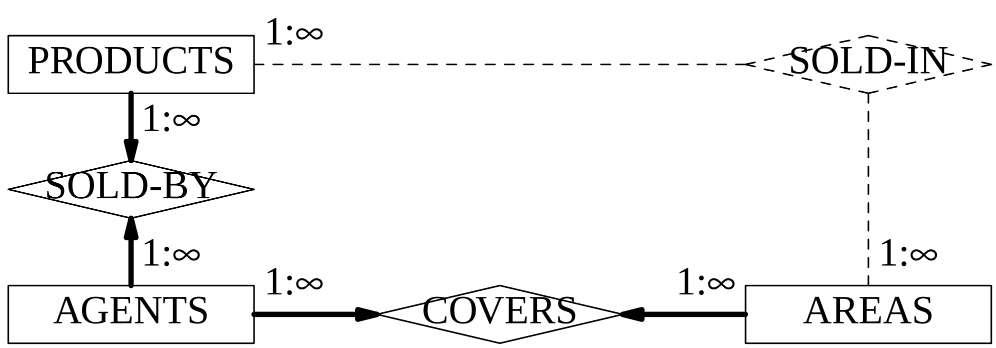
    Determine the entities (their attribuites and keys) in this world-of-discourse. Also, determine the relationships and the participating entities for each of them.

2. Do the same for the world of a University like LSU. (You can make suitable simplifications as needed.)

ANOTHER EXAMPLE OF ER-DIAGRAM

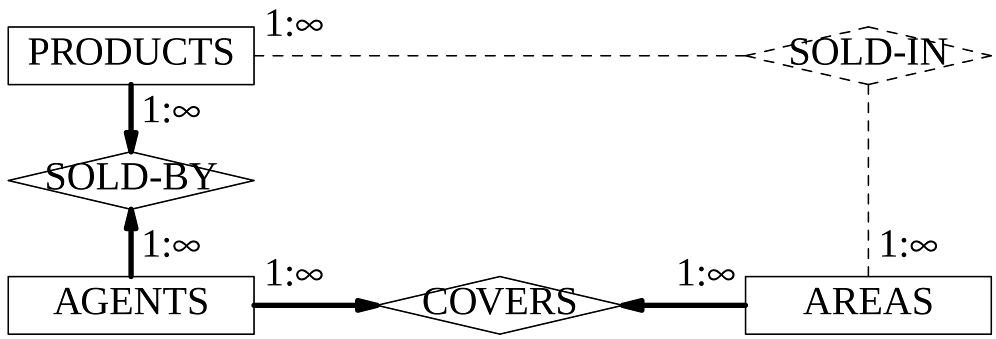
Cardinality Constraint:

- Shows the min and max number of tuples in the relationship-set for each entity in an entity-set.
- In the example below, the minimum number of tuples in SOLD--BY for each PRODUCTS entity is 1, and there is no upper limit on the maximum. (Min = 1 is equivalent to "onto".)
- The only way to represent the fact "every product is sold in every area" is by explicitly stating it as a constraint: SOLD-IN = $\Pi_{Attrb(PRODUCTS) \cup Attrb(AREA)}$ [SOLD-BY ⊗ COVERS]
  - Adding a relationship "SOLD-IN" in the E-R diagram, as shown in dotted lines, does not help.
- Not all constraints can be expressed in ER-diagram.



Not shown here are the attributes of entity-sets and relationship-sets and their keys.

CONVERTING ER-SCHEMA TO RELATION-SCHEMAS



Relation-Schemas:

- The attributes of a relation-schema corresponding to an entity-set consists of the attributes of the entity-set.
- The attributes of a relation-schema corresponding to a relationship-set consists of the key-attributes of each participating entity-set and the attributes of the relationship-set.
  - TotalQuant in SOLD-BY = total quantity sold by an agent by product (for different areas that he covers).
  - TotalQuant in COVERS = total quantity (of all different products) sold by an agent by areas. (This has a different meanining than above.)
- The key of the relation for a relationship-set is the combination of keys of entity-sets which have arrows into the relationship-node in the ER-diagram.

PRODUCTS

<u>*P#*</u>	Pname

AGENTS

<u>*Ag#*</u>	Ag-Name	Ag-Addr

SOLD-BY

<u>*P#*</u>	<u>*Ag#*</u>	Total-Quant

AREAS

<u>*Ar#*</u>	Ar-Name

COVERS

<u>*Ag#*</u>	<u>*Ar#*</u>	Prio-rity	Quan-tity

EXERCISE

1. Consider the relation-schemas below; key attributes are shown underlined/italics. Determine the ones that might have come from Relationship-sets (others came from Entity-sets). Show the ER-diagram.

BUSES

Lic#	Model	Make	Cost	PurchDate	DealerName

ROUTES

Rt#	*fromStop*	toStop

WORK-SHIFTS

Sh#	StartTime	EndTime

REPAIR-WORKS

Lic#	*RepShop#*	*Bill#*	RepCost	RepType	ServOffDate	ServRetDate

REPAIR-SHOPS

RepShop#	Address	Tel

DRIVERS

Dr#	DrName	Address	Tel	DrLicInfo

SCHEDULES

Dr#	*Lic#*	*Rt#*	*Sh#*	startDate	EndDate

2. If we wanted to break SCHEDULES into two relations, one for schedule of drivers and the other for schedule of busses, then what should be their attributes? What would be an advantage or disdvantage of this breaking?

MORE ON BORROWS-RELATIONSHIP

Example. An instance of BORROWS may look like the following.

BORROWS

PERSON	BOOK	IssueDate
p_1	b_1	d_1
p_1	b_1	d_2
p_1	b_2	d_2
p_2	b_2	d_2

PERSONS

Id#	Name	Address
110	J. Adams	Comp.Sc, LSU
112	S. Kundu	Comp.Sc, LSU
113	P. Lee	Libr.Sc, LSU

BOOKS

Call#	Title	Author	Year	Publ
QA70.1	Algorithm Design	Aho	1978	PrenticeHall
QA76.2	Soft. Engg.	Jalote	1998	Springer
QA76.9	UML Distilled	Fowler	1999	Addison-Wesley
QA76.7	UML in Nutshell	Albir	1998	O'Reilley

BORROWS

Call#	**IssueDate**	Id#
QA76.2	05Jun01	112
QA76.7	07Jun01	112
QA76.2	10Jul01	113

Assumptions in the Model:

Each book is issued, say, for two weeks; thus we store only the IssueDate (the books are assumed returned on time and thus ReturnedDate is not stored). Also, when a book is returned, the associated BORROW-tuple is not deleted; otherwise, we do not need a key attribute for BORROW (why?). (Why can't we make PERSONS a key entity of BORROWS instead of BOOKS?)

Question: If there are multiple copies of a book, how will the BOOK-entity change?

Problem in using a single entity for all attributes of PERSONS, BOOKS, and BORROWS:

PERSONS-BORROWS-BOOKS
(**Id#**, Name, Address, IssueDate, **Call#**, Title, Author, Year, Publ.)

PERSONS-BORROWS-BOOKS

Id#	Name	Address	**IssueDate**	**Call#**	Title	Author	Year	Publ
110	J. Adams	Comp.Sc, LSU	–	–	–	–	–	–
112	S. Kundu	Comp.Sc, LSU	05Jun01	QA76.2	Soft. Engg.	Jalote	1998	Springer
112	S. Kundu	Comp.Sc, LSU	07Jun01	QA76.7	UML in Nutshell	Albir	1998	O'Reilley
113	P. Lee	Libr.Sc, LSU	10Jul01	QA76.2	Soft. Engg.	Jalote	1998	Springer
–	–	–	–	QA70.1	Algorithm Design	Aho	1978	PrenticeHall
–	–	–	–	QA76.7	UML Distilled	Fowler	1999	Addison-Wesley

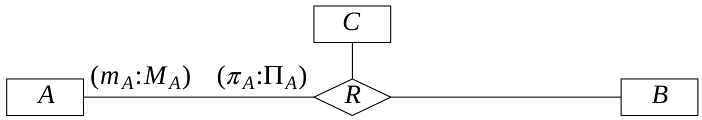
(1) *Data storage:* Too much redundant information and missing-data (both causing waste of memory).

(2) *Data manipulation:* The determination of books issued to a person requires scanning more rows, unless we store the rows with missing-data, say, at the end (but it increases data movement on a book return).

EXERCISE

1. How will you modify the BOOK-entity if some books have multiple copies? Can we model issue-dates as an entity-set and make BORROWS a ternary (3-way) relationship among PERSONS, BOOKS, and ISSUE-DATES? Give the attributes of ISSUE-DATES and key for BORROWS.
2. Determine the entities and relationships for modeling family-relationships.
3. What goes wrong if we consider STUDENTS and GRADES as entity-sets and COURSE as a relationship-set between them to indicate which student got what grade for what course?

CARDINALITY-CONSTRAINTS IN A RELATIONSHIP



Constraint imposed by one entity-set on the combination of others:

m_A: For each $a \in A$, there are *at least* m_A combinations $(b_i, c_i) \in B \times C$ such that $(a, b_i, c_i) \in R$, together with suitable values for the attributes of R for each (a, b_i, c_i).

M_A: For each $a \in A$, there are *at most* M_A combinations $(b_i, c_i) \in B \times C$ such that $(a, b_i, c_i) \in R$.

Constraint imposed by the combination of other entity-sets on one:

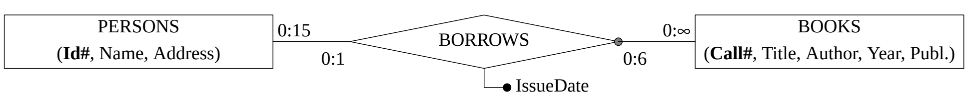
π_A: For each combination $(b, c) \in B \times C$ and the attribute-values for R, there are *at least* π_A many a_i such that $(a_i, b, c) \in R$.

Π_A: For each combination $(b, c) \in B \times C$ etc, there are *at most* Π_A many a_i such that $(a_i, b, c) \in R$.

Example. The cardinalities in the ER-model below correspond to the assumptions:

(1) The maximum number of books that a person may have borrowed at any given time is 15; but on any day, he can borrow only a maximum of 6 books.

(2) A book is borrowed by at most one person at a time and retained for at least one time-unit (for IssueDate). (If IssueDate is not a key-attribute of BORROWS so that when a book is returned the associated tuple in BORROWS is deleted and BORROWS keeps track of only the current borrowings, then M_{BOOKS} would be $1 = \Pi_{PERSONS}$ and likewise Π_{BOOKS} would be $15 = M_{BOOKS}$.)



Special Cases: Binary-relationships without its own attributes: $\pi_A = m_B$ and $\Pi_A = M_B$. For unary-relationships, π and Π are not defined.

The entities, relationships, their attributes, and the various cardinalities (m, M) and (π, Π) depend on the situation that you are modeling.

Two other cardinalities:

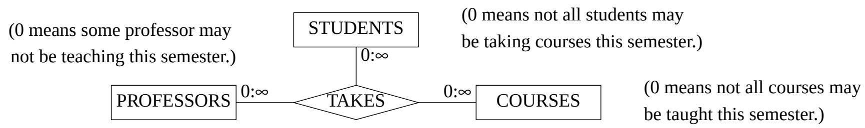
n_A: The total number of elements of A that are involved in the relationship R; $n_A \geq \pi_A$.

N_A: The total size $|A|$ of A (some $a \in A$ may not be involved in the relationship R;) $N_A \geq n_A$.

EXERCISE

1. Which of $m_A = 1$, $M_A = 1$, $\pi_A = 1$, and $\Pi_A = 1$ corresponds to the 1-1 property of R (which has no attributes of its own) from A to $B \times C$, the property of being onto $B \times C$, the property of being total on A, and the property of being a function from A to $B \times C$?

2. Show some reasonable cardinalities π and Π for each entity in the relationship below; state any assumptions that you make. A tuple $(p_i, s_j, c_k) \in$ TAKES means the student s_j takes the course c_k from professor p_i in some particular (fixed) semester.


(0 means some professor may
not be teaching this semester.)
STUDENTS
0:∞
(0 means not all students may
be taking courses this semester.)
PROFESSORS
0:∞
TAKES
0:∞
COURSES
(0 means not all courses may
be taught this semester.)

IMPORATNCE OF MODELING THE CARDINALITIES

Query answering:

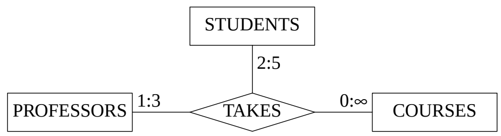
- Query: "Find Call#'s of all books that are currently borrowed."
  Scan the elements of BORROWS-relationship and get their book-part. The cardinalities ($\underline{m}BOOKS$: $\bar{m}BOOKS$) = (0:1) implies that there is no need to attempt elimination of duplicates, as would be the case for the query "Find all persons who have currrently borrowed $\geq k$ ($1 \leq k \leq 15$) books".
- Query: "Find the address of the person who borrowed the book with a given Call#."
  Use the index (hashing) Call# to find the book (avoiding expensive search), and use the pointer to the relationship-tuple in BORROWS for that book. If this pointer is not null, then from the relationship-tuple in BORROWS follow the pointer to the person-tuple in PERSONS and then get his address.

Implementation:

- As indicated above, the implementation may depend on the cardinalities.
- If the results of one query is to be passed to another query, then the cardinalities also affect the data-structure for the outputs.

A BIG RELATIONSHIP HIDES MANY SMALLER RELATIONSHIPS

Example. Consider the relationship below representing a particular (regular) semester in a department; each student is required to take ≥ 2 but no more than 5 courses (no matter who teaches them) in a semester and each professor is required to teach at least 1 and at most 3 courses in a semester. {STUDENT} is not a key-entity for TAKES because a student may take many courses from different professors in a semester; similarly, none of {PROFESSORS} and {COURSES} is a key.



TAKES

PROFESSORS	STUDENTS	COURSES
p1	s1	c1
p1	s2	c1
p1	s1	c2
p1	s3	c2
p2	s2	c3
p3	s3	c4
p3	s1	c4

Projection: Ignoring the data in column COURSES (and removing duplicate rows) gives $\Pi_{PROFESSORS \times STUDENTS}$ = Projection of TAKES on PROFESSORS×STUDENTS.

$\Pi_{PROFESSORS \times STUDENTS}(TAKES)$

PROFESSORS	STUDENTS
p1	s1
p1	s2
p1	s3
p2	s2
p3	s3
p3	s1

This relationship shows only those professor-student combinations (p_i, s_j) such that s_j is taking some course from p_i. In particular, it involves only those professors who are teaching at least one course and the students who are taking at least one course.

Different kinds of information-loss:

- that s1 is taking two courses (c1 and c2) from p1.
- that s2 and s3 are taking different courses from p1.
- that s1 is taking different courses from p1 and p3.

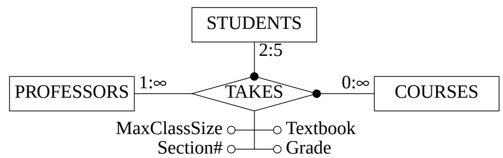
EXERCISE

1. Show the projection of TAKES on STUDENTS×COURSES using the above data. What are some of the information-losses occurring in this projection?
2. What are some reasonable entity-keys for TAKES? (Verify your answers against the sample data.)

3. Show an appropriate cardinality to represent (if possible) in the ER-model shown above: (i) a professor teaches at most 3 courses in a semester, (ii) the maximum number of courses offered (available) in the dept. is 20, and (iii) the number of professors in the department is p.

USE OF COMPOUND ENTITIES FOR A BETTER MODEL

Example. Consider the relationship "TAKES" shown earlier, with three added attributes here.



(MaxClassSize and section# can depend on the course.)

Problems with the model: It does not capture the following relevant facts:

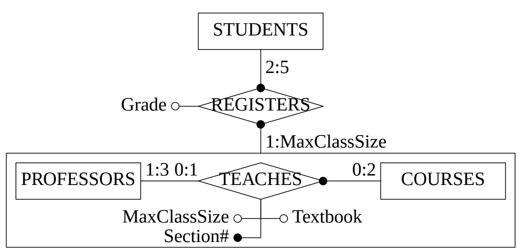
(a) all students use the same textbook for a given professor-course combination (the same textbook may be used for different courses by the same or different professors).

(b) each section of a course is taught by one professor, and each course has at most 2 sections.

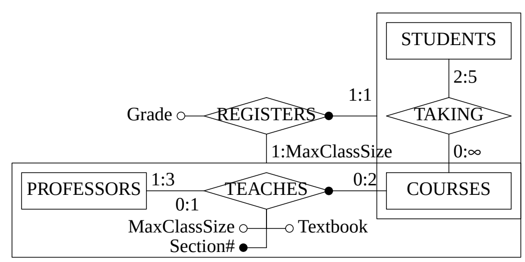
(c) a professor teaches ≤ 3 courses (i.e., course-sections).

(d) MaxClassSize and Textbook for a course-section depends on the professor teaching it.

(e) a student is not allowed to take more than one section of a course.



(i) Solves the above problems except for Note (2) below; Section# ≤ numSections.



(ii) A final model that corrects the problem in (i); Section# is not a key-attribute. See Note (3) below.

Notes:

(1) The box around PROFESSORS, TEACHES, and COURSES indicate a complex entity-set made out of the professor-course pairs of the tuples in TAKES-relationship, including the associated attributes {Textbook, MaxClassSize, Section#}.

(2) The entity-key of REGISTERS includes both STUDENTS and the compound-entity; neither one by itself will suffice. However, since the compound-entities are identified by COURSE and Section#, we get the entities {STUDENT, COURSE} together with the attribute {Section#} identifies the tuples of REGISTERS. The only problem here is then is the unnecessary role of Section# in identifying the tuples REGISTERS.

(3) If $q_1 = (p_i, c_k)$ participates in TEACHES and $q_2 = (s_j, c_{k'})$ participates in TAKING in (ii), then (q_1, q_2) together with the associated attributes for q_1 in TEACHES participates in REGISTERS if and only if $c_k = c_{k'}$. The tuples represented by REGISTERS in (ii) therefore correspond exactly to those of TAKES in the initial model at the top of the page.

(4) In (ii), the cardinality 1:1 on STUDENTS-TAKING-COURSES means "Grade" can be thought of as an attribute of TAKING instead of REGISTERED. (Why is it right to put the $(m\!: M)$-cardinalities of COURSE in TAKING-relationship (0:∞) instead of (1:MaxClassSize)?)

EXERCISE

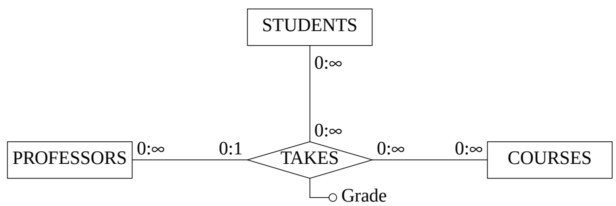
1. Modify the ER-model in (ii) to allow different textbooks to be used for different sections of the same course. Also show a modified form of (ii) to indicate that the max. number of courses that a student can take (in a semester) depends on his GPA.

AN EXAMPLE OF WEAK-COMPOSITE ENTITY

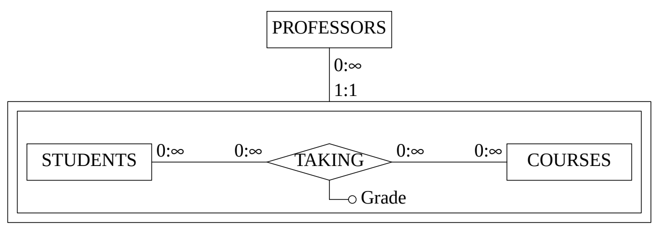
Example. Consider the ER-model shown below, which was also considered earlier and was corrected by using two composite entity sets TAKING(STUDENT, COURSES) and TEACHES(PROFESSORS, COURSES).

Shown here also the (π, Π)-cardinalities that were not considered earlier.

The attribute Textbook is eliminated here; the alternative mode based on weak-entity is not suitable in presence of this attribute.



(i) An initial model.



(ii) An alternative model.

Note that if $(m_A, M_A) = (m_B, M_B) = (0, 1)$, then $(\pi_C, \Pi_C) = (0, 1)$ for a ternary relation $R(A, B, C)$.

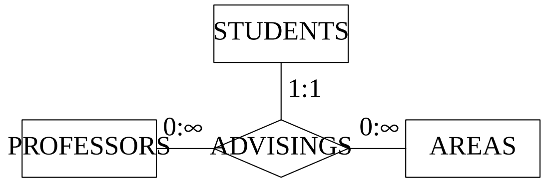
EXERCISE

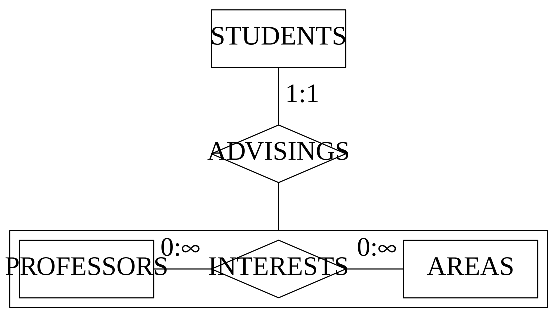
1. Modify the ER-model in (ii) on the previous page by adding a weak-entity HOMEWORKS to model that a professor gives ≥ 0 homeworks for a course. Use Hwk# as an attribute-key (with values Hwk#1, Hwk#2, etc. for HOMEWORKS. Different sections of a course may have different homeworks (Hwk#1 in one section may be different from Hwk#2 in another section irrespective of whether they are taught by the same of different professors).

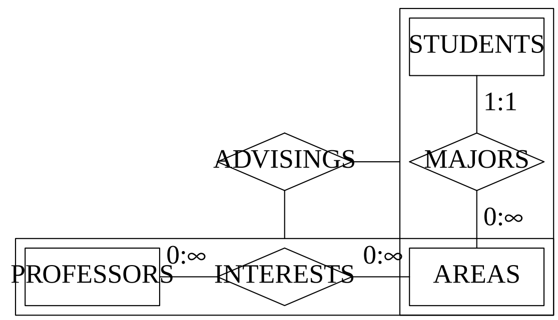
   How will the model change if we insist that all sections of the same course will have the same homeworks?

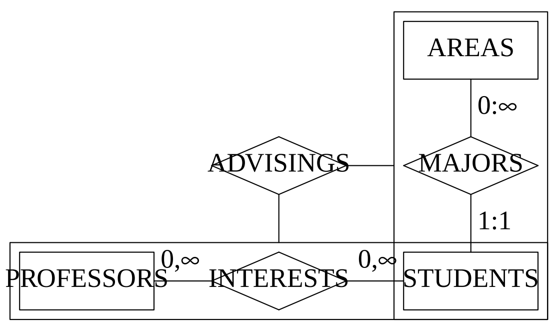
2. Modify your model in Problem 2 by adding CLASSPERIOD as a new entity (and not an attribute of some other entity or relationship, why?).

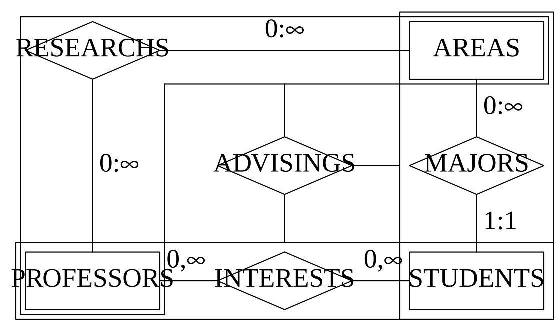
GENERALIZING AN ER-MODEL MORE PRECISE



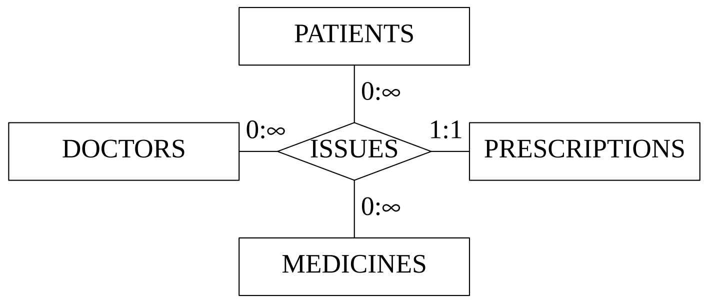








DISSECTING AND CORRECTING AN ER-MODEL

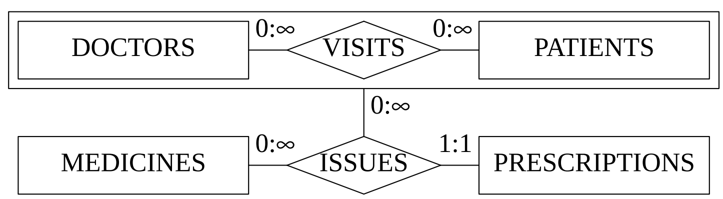


Problems:

(1) It is not possible for a patient to visit a doctor without having to get stuck with a prescription.

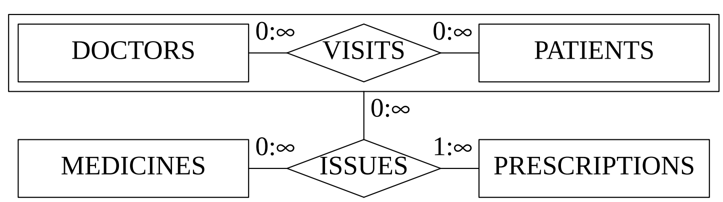
(2) He has to get a different prescription for each medicine he is prescribed in a visit.

After correction of (1):

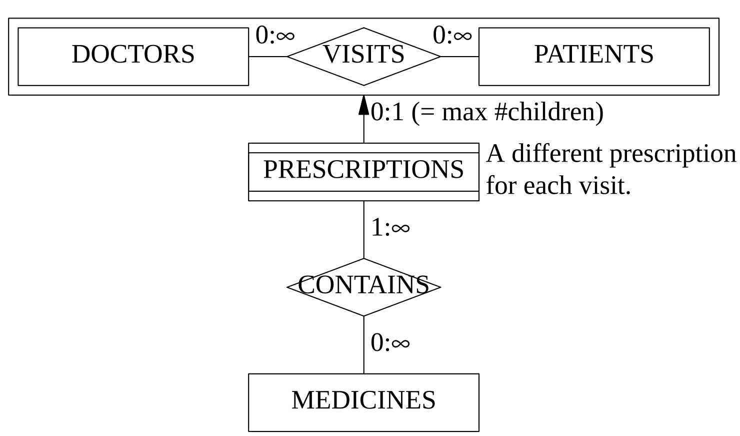


Use **Visit#** and **Date** as attribute-keys to distinguish different visits by a patient to different doctors on the same day.

After correction of (1) and partial correction of (2):

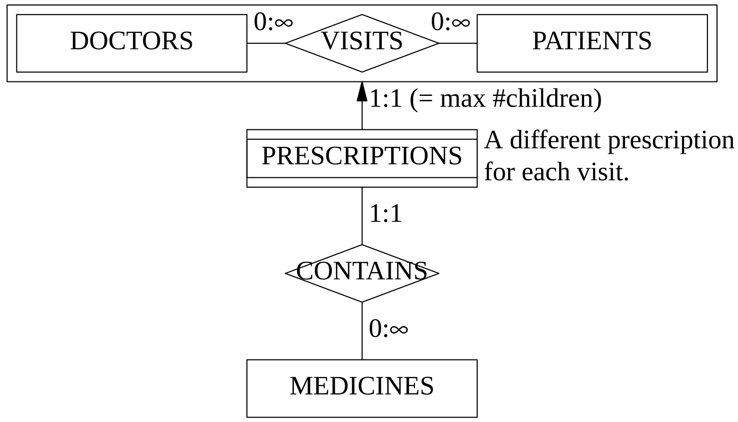


After correction of both (1)-(2):



Restricting the final model to obtain the original model:

Here, since each patient-doctor visit creates a unique prescription, the **Prescription#** can be used as a key to distinguish the visits and also the prescriptions.



EXERCISE

1. If we let k ($2 \leq k < \infty$) to be the maximum number of children weak-entity PRESCRIPTION for a given parent entity VISIT, then argue that this situation cannot be expressed by modifying the cardinalities in the original model. (You can do this by giving sample data instances that fi ts the new model but not the original model.)

COMBINING TWO DESIGNS INTO ONE

General multi-valued dependency:

$X \xrightarrow{m:\,n} Y$:	a tuple of values for the attributes in X determines at least m and at most n distinct tuples of values for the attributes in Y.
$X \xrightarrow{1:\,1} Y$:	ordinary functional dependency $X \to Y$; here, the function is *total*.
$X \xrightarrow{0:\,1} Y$:	ordinary functional dependency $X \to Y$, except that the function may be *partial*, i.e., not defined for all tuples of values of the attributes X.

Some Key Properties: Here, X, X', Y, etc. are non-empty subsets of attributes.

1. $\Pi_{X'}(I_{X'})$: $X' \xrightarrow{1:\,1} X$ for all X and $X' \subseteq X$, and R: $X \xrightarrow{0:\,\infty} Y$ for all X, Y, and R.

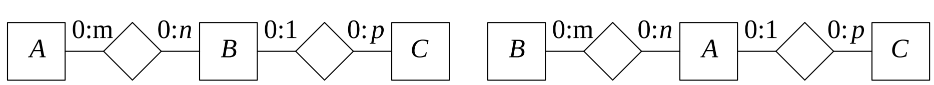
2. If R: $X \xrightarrow{m:\,n} Y$, then R: $X \xrightarrow{p:\,q} Y$ for $0 \le p \le m \le n \le q$.

3. If R: $X \xrightarrow{m:\,n} Y$ and R: $X \xrightarrow{p:\,q} Y$, then R: $X \xrightarrow{\max(m,\ p):\ \min(n,\ q)} Y$.

4. If R: $X \xrightarrow{m:\,n} Y$ and S: $Y \xrightarrow{p:\,q} Z$ and X, Y, Z are mutually disjoint, then $R \circ S$: $X \xrightarrow{[(m > 0)?\ p; 0]:\ nq} Z$.

5. If R: $X \xrightarrow{m:\,n} Y$ and $Y' \subset Y$ and $X \cap Y = \emptyset$, then $\Pi_{XY'}(R)$: $X \xrightarrow{[(m > 0)?\ 1; 0]:\ n} Y'$ (obtained by combining (1) and (4)).

6. If R: $X \xrightarrow{m:\,n} Y$ and S: $X \xrightarrow{p:\,q} Z$ and $Y \cap Z = \emptyset$, then there is an T: $X \xrightarrow{\max(m,\ p):\ \max(n,\ q)} Y \cup Z$ such that $R = \Pi_{XY}(T)$ and $S = \Pi_{XZ}(T)$.



$B \to C$ $\qquad\qquad$ $A \to C$

CONNECTIVITY PROPERTY: like a high level cohesion

(1)	The set of relations and entities used in a function should form a connected set. (There may be zero or more relations which connect the entity-sets used in the function that are not used in that function.)
(2)	If condition (1) is not true, then decompose the function into smaller parts, one for each connected component. (One maybe able to decompose these smaller functions even further.)

EXERCISE

1. Is there any advantage of including the "DueDate" in the ER-model?

2. Give an algorithm for generating the mailing-report for a given date and a person's Id#. **EXERCISE**

3. Consider a set of Authors who have written one book each (perhaps jointly with other authors), i.e., we have a functional dependency f: *Authors* $\to$ *Books*. A book may have multiple Authors. Assume that Authors have name and address and Books have publishers, year of publication, number of pages, and an ISBN#. Shown below is an ER-models and a sample of possible data-items in the associated relation for a book jointly written by two people and a book with a single author. Obtain an alternative ER-model which is better in some sense (explain it).

AUTHOR-BOOK
AuthorName, BookName, Publ, Year, NumPages, ISBN#

AuthorName	BookName	Publ	Year	NumPages	ISBN#
J. Adams	A tour of the World	Phantom Publishers	1990	258	1-56592-446-7
H. Brooks	A tour of the World	Phantom Publishers	1990	258	1-56592-446-7
S. Lee	Magic power of thought	Popular Publishers	1989	175	3-22824-334-6
...	...	...	...	...	...

4. What is meant by "a software correctly implements a data-model"?

5. Is the following statement true:

   - If (a software correctly implements a data-model) and (the software is found to be correct) then the data-model is correct.

   Give another such true statement connecting the same three components (two in the if-part and the one in then-part).

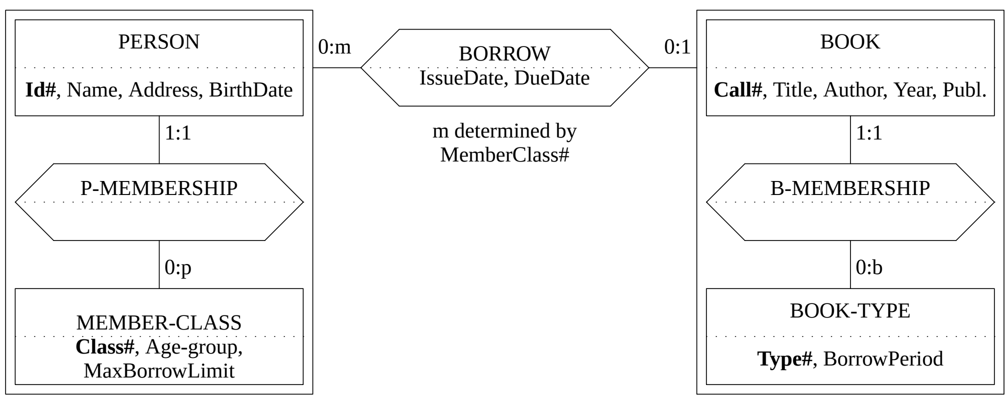
Forming the ER-model:

- Collect all detailed pieces of basic input data needed to provide the services by IS. (Include data resulting from intermediate computations that may be reused.)
- Identify all functional dependencies.
- Create the ER-model.

A MODIFIED ER-MODEL TO SUPPORT ADDITIONAL FUNCTIONALITIES

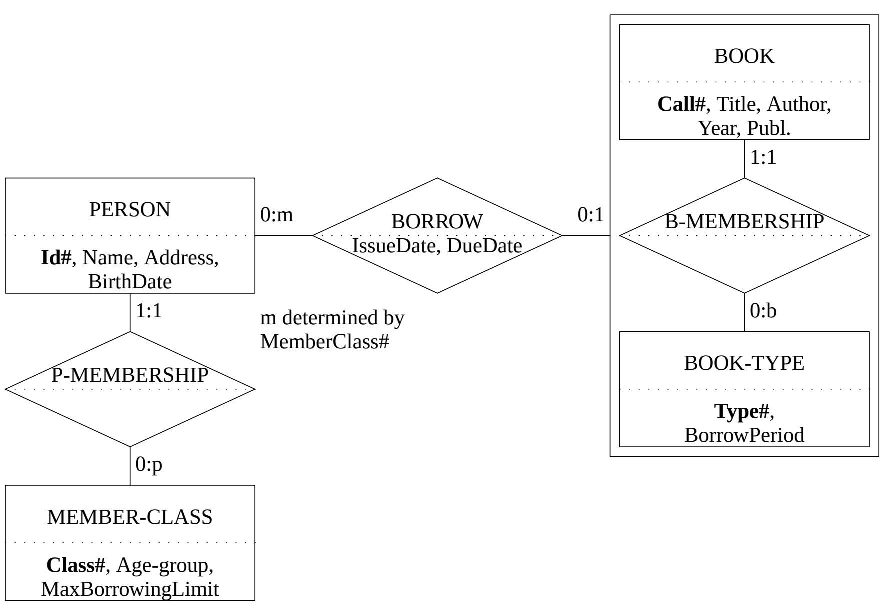
Additional functionalities:

- The number of books that may be borrowed at a time depends on the membership-class of a person, based on his age. The books may have different due dates, resulting from different borrow-periods or issue-date.
- The books are also classified into different types, and each type has a specified borrowing period (which may be reset from time to time by the management). We assume that once a book has been borrowed and its due date is set, a change in borrowing period does not change its due date. The change affects only the books to be borrowed in the future.



More Additional functionalities:

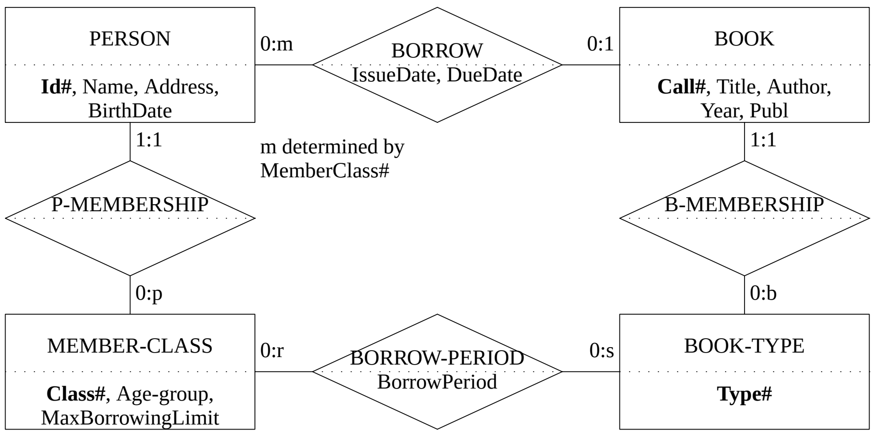
- The borrowing period of a book depends on the book-type, but not on the membership-type of the person.



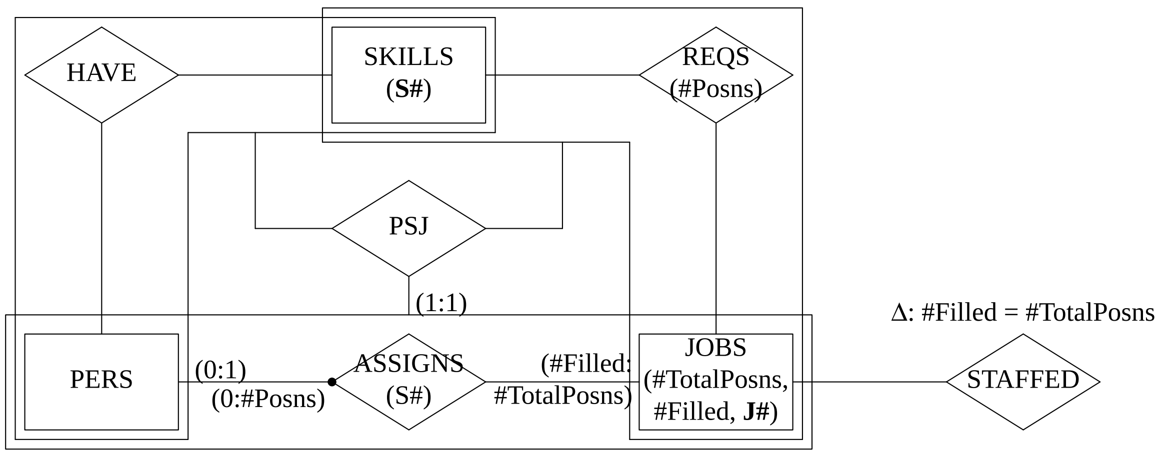
FURTHER MODIFICATION TO THE ER-MODEL TO SUPPORT ADDITIONAL FUNCTIONALITIES

Additional functionalities:

- The borrowing period of a book depends on its type and the membership-class of the borrower.



A COMPLEX EXAMPLE



Meanings of various cardinalities:

(a) (0:1) from PERS to ASSIGNS:
At most one person is assigned to a job for a given skill. This is same as saying that Person is a key of ASSIGNS

(b) (0:#Posns) from ASSIGNS to PERS:
At most #Posns persons have assigned to a given (Job, Skill) pair.

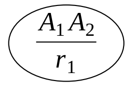
(c) (1:1) from ASSIGNS to PSJ:
An assignment of a person to a (Job, Skill) is made only if the person has the skill and that skill is needed for that job.

(d) (#Filled: #TotalPosns) from JOBS to ASSIGNS:
For a given job, there are at most #TotalPosns many (Persons, Skill) pairs , or equivalently, persons assigned to it. (This prevents over-staffi ng; we can replace the upper bound by $p \times$#TotalPosns to allow p% over-staffi ng.)

THE CASES INVOLVING TWO ATTRIBUTES

Case 1.

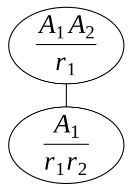
	A_1	A_2
r_1	1	1



$E_1(A_1,\ A_2)$

Case 2.

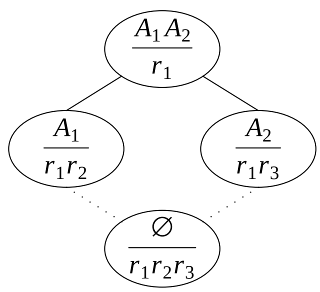
	A_1	A_2
r_1	1	1
r_2	1	0



$E_1(A_1)$
$R_1(E_1;\ A_2)$

Case 3.

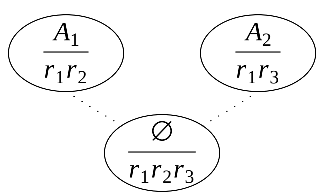
	A_1	A_2
r_1	1	1
r_2	1	0
r_3	0	1



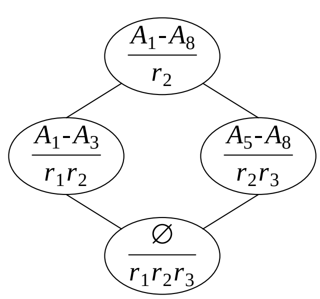
$E_1(A_1)$
$E_2(A_2)$
$R_1(E_1,\ E_2)$

Case 4. Disconnected.

	A_1	A_2
r_2	1	0
r_3	0	1



$E_1(A_1)$
$E_2(A_2)$



A THREE-LEVEL APPROACH TO ER-MODELING

(1) Collect all attributes (basic elements of information) relevant (in terms of being used in answering some queries) to the "world" being modeled.

(2) Imagine all possible combination of attributes that may be used to answer various queries. Then create a 0/1 vector for each such case, with 1 representing the use of that attribute-value. Finally, let M be the 0/1-matrix of all such row-vectors.

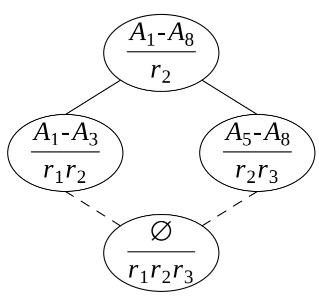
(3) Create a more detailed form of the matrix M in (2) which show for each row the subset of its attributes whose values may vary while the remaining attributes in that row are kept at some fixed value.

Example 1. Consider the simple library-world with books, persons (borrowers), and borrowing. The rows in the table below correspond to the following three descriptions about various situations that arise in the library-world.

1. A person who has no book issued to him (query: find all borrowers).
2. A person and a book borrowed by him (query: find the person who borroed a particular book which is to be recalle
3. A book not issued out to any person (query: find books that are not issued out).

The matrix M; '×' corresponds to 1 and '–' to 0.

	Id# A_1	Name A_2	Address A_3	IssueDate A_4	Call# A_5	Title A_6	Author A_7	Year A_8
r_1	×	×	×	–	–	–	–	–
r_2	×	×	×	×	×	×	×	×
r_3	–	–	–	–	×	×	×	×



E_1 with the attributes $\{A_1, A_2, A_3\}$.
E_2 with the attributes $\{A_5, A_6, A_7, A_8\}$.
$R(E_1, E_2)$ with the attribute $\{A_4\}$.

Entity-set: One entity-set for each lowest-level node in the concept-lattice above the bottom-node.

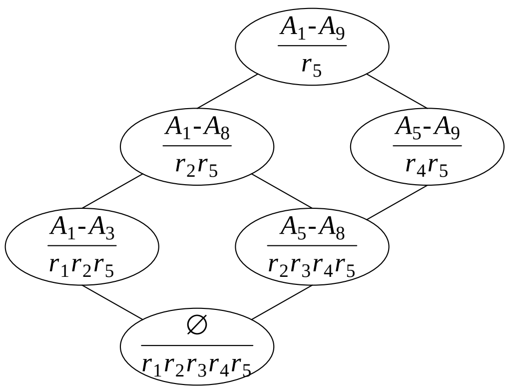
Relationship: One relationship-set for each level ≥ 2 node in the concept-lattice (except perhaps the top node), whose entitiy-sets correspond to the children of that node and whose attributes correspond to the attributes in the node that are not in any of the children. (Some of these relationships may be suprious, particularly, those which do not have their own attributes.)

EXERCISE

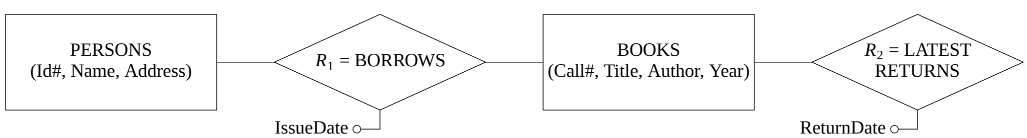
1. Extend the library-world with the additional attribute A_9 = ReturnDate, which is set when a book is returned. We intend to use it for identifying books that have not been used in last k (= 5, say) years so that they can be eliminated. Show the new matrix, the concept-lattice, and the resulting entity-sets and relationships.

1. A person who has no book issued to him.
2. A person and a book borrowed by him (person information used in recalling).
3. A book not issued out to any person ever.
4. A book that has been used but is not currently issued out.
5. A book that has been returned before and is also currently issued out.

	Id# A_1	Name A_2	Address A_3	IssueDate A_4	Call# A_5	Title A_6	Author A_7	Year A_8	RetDate A_9
r_1	×	×	×	–	–	–	–	–	–
r_2	×	×	×	×	×	×	×	×	–
r_3	–	–	–	–	×	×	×	×	–
r_4	–	–	–	–	×	×	×	×	×
r_5	×	×	×	×	×	×	×	×	×



E_1 with the attributes $\{A_1, A_2, A_3\}$.
E_2 with the attributes $\{A_5, A_6, \cdots, A_8\}$.
$R_1(E_1, E_2)$ with the attribute $\{A_4\}$.
$R_2(E_2)$ with the attribute $\{A_5, A_6, \cdots, A_9\}$.



The top node of the concept-lattice here does not give any useful relationship.

INTEGRITY CONSTRAINTS

Two Types:

(1) Constraints and properties that can be verified only by looking two or more columns of a single row.

(2) Constraints and properties that can be verified only by looking two or more rows of a relation.

(3) Constraints and properties that can be verified only by looking two or more relations.

The fact that the attribute "Amount" in a SALES-relation is ≥ 0 is not an integrity constraint; this is modeled by simply saying that the domain of Amount is *positive* numbers.

Question:

- What makes the cardinality constraints an integrity constraint?
- What about functional dependency?

CITY-WIDE BUS-TRANSPORTATION PROBLEM