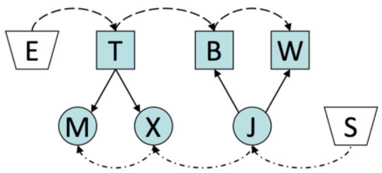
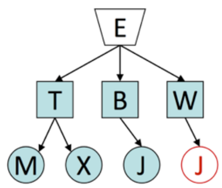
RELATIONAL MODEL

- Data Models – a notation for describing data, including
  - **Structure** of the data
  - **Constraints** on the content of data
  - **Operations** on the data
- Comparing data Models
  - Ex. Student w/ jobs
    - Mary (M), Xiao (X) → Tim Hortons (T)
    - Jaspreet (J) → Bookstore (B), Wind (W)
  - **Network** (graph) data model
    - Employers (E) = head of linked list of employers
    - Students (S) = head of linked list of students



  - **Hierarchical** (tree) data model
    - Employers (E) = parent node of employers
    - Double nodes needed to maintain tree



  - **Relational** (table) data model
    - Tables may store relations between attributes
    - Advantages:
      - Matches how we think about data
      - Allows data independence
    - Models allows:
      - Declarative access to data (system optimizes for you)
      - Relationships specified by queries
      - Develop, maintain apps and data layout separately

E

B	…
T	…
W	…

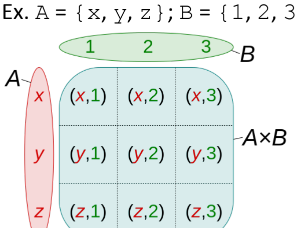
S

J	…
M	…
X	…

R

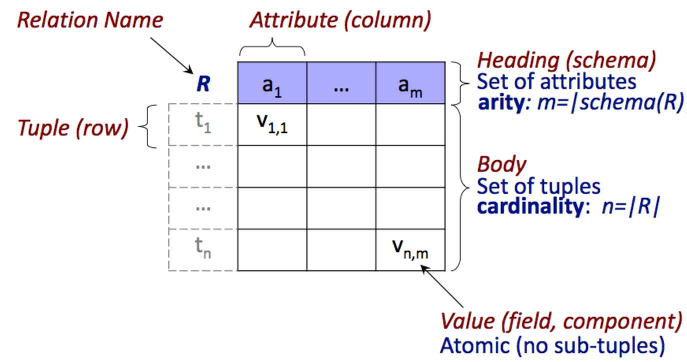
M	T
X	T
J	B
J	W

- Relational Model
  - Logical representation of data
    - Two-dimensional tables (relations)
  - Formal system for manipulating relations
    - Relational algebra
  - Result:
    - High-level (logical, declarative) description of data
    - Mechanical rules for rewriting/optimizing low-level access
    - Formal methods to reason about soundness
  - History:
    - Proposed by Edgar F. Codd in 1970 as a data model that strongly supports data independence
      - Commercialized in 1981
    - Based on (a variant of) the mathematical notion of relation → represented as tables
- Mathematical Relations
  - **Cartesian product**
    - Given sets D_1, D_2, …, D_n (does not have to be distinct)
    - $D_1 \times D_2 \times \ldots \times D_n$ = set of all possible ordered n-tuples $<d_1, d_2, \ldots, d_n>$ such that $d_1 \in D_1$, $d_2 \in D_2$, …, $d_n \in D_n$
    - Ex. $A = \{x, y, z\}$; $B = \{1, 2, 3\}$



Mathematical Relation on D 1 , D 2 , …, D n is a subset of the Cartesian

  - **Mathematical Relation** on D_1, D_2, …, D_n is a subset of the Cartesian product $D_1 \times D_2 \times \ldots \times D_n$
    - **Domains** of the relation are D_1, D_2, …, D_n
    - **Degree** of the relation is n
    - **Cardinality** of the relation is the number of n-tuples
    - **Arity** of the relation is the number of attributes
  - **Attributes**
    - Associate an attribute (unique name) w/ each domain that describes its role in the relation
      - Make the structure of a relation non-positional
    - Represented in tables by column headings
    - Notation:
      - `t[A]` or `t.a` = value on attribute `A` for a tuple `t`
      - more generally if `X = A, B, …, N` (a sequence of attributes) then `t[X]` → `<t[A], t[B], …, t[N]>`
  - **Tuples** = rows, no duplicates



- Value-based references
  - Lead to independence from physical data structures, such as pointers

Students

RegNum	Surname	FirstName	BirthDate
6554	Rossi	Mario	5/12/1978
8765	Neri	Paolo	3/11/1976
9283	Verdi	Luisa	12/11/1979
3456	Rossi	Maria	1/2/1978

Exams

Student	Grade	Course
3456	30	04
3456	24	02
9283	28	01
6554	26	01

Courses

Code	Title	Tutor
01	Analisi	Neri
02	Chimica	Bruni
04	Chimica	Verdi

Students

RegNum	Surname	FirstName	BirthDate
6554	Rossi	Mario	5/12/1978
8765	Neri	Paolo	3/11/1976
9283	Verdi	Luisa	12/11/1979
3456	Rossi	Maria	1/2/1978

Exams

Student	Grade	Course
	30	
	24	
	28	
	26	

Courses

Code	Title	Tutor
01	Analisi	Neri
02	Chimica	Bruni
04	Chimica	Verdi

- Definitions
  - **Relation schema**: relation name R with a set of attributes $A_1, \ldots, A_n$
    - Ex. $R(A_1, \ldots, A_n)$
  - **Database schema** (a.k.a **relational schema**): A set of relation schemas with different names
    - Ex. $D = \{R_1(X_1), \ldots, R_n(X_n)\}$
  - Relation (instance) on a relation schema
    - Ex. $R(X)$ = set of tuples on the set of attributes X
- Example
  - Data

Da Mario		
Receipt No: 1357		
Date: 5/5/92		
3	covers	3.00
2	hors d'oeuvre	5.00
3	first course	9.00
2	steak	12.00
	Total:	29.00

Da Mario		
Receipt No: 2334		
Date: 4/7/92		
2	covers	2.00
2	hors d'oeuvre	2.50
2	first course	6.00
2	bream	15.00
2	coffee	2.00
	Total:	27.50

Da Mario		
Receipt No: 3007		
Date: 4/8/92		
2	covers	3.00
2	hors d'oeuvre	6.00
3	first course	8.00
1	bream	7.50
1	salad	3.00
2	coffee	2.00
	Total:	29.50

  - Table representation A
    - Does not consider line of order
    - A duplicate entry would not show up in the database

Details

Number	Quantity	Description	Cost
1357	3	Covers	3.00
1357	2	Hors d'oeuvre	5.00
1357	3	First course	9.00
1357	2	Steak	12.00
2334	2	Covers	2.00
2334	2	Hors d'oeuvre	2.50
2334	2	First course	6.00
2334	2	Bream	15.00
2334	2	Coffee	2.00
3007	2	Covers	3.00
3007	2	Hors d'oeuvre	6.00
3007	3	First course	8.00
3007	1	Bream	7.50
3007	1	Salad	3.00
3007	2	Coffee	2.00

Receipts

Number	Date	Total
1357	5/5/92	29.00
2334	4/7/92	27.50
3007	4/8/92	29.50

  - Table representation B
    - Add line attribute allows duplicate entries to show up b/c line number is unique per order number

Details

Number	Line	Quantity	Description	Cost
1357	1	3	Covers	3.00
1357	2	2	Hors d'oeuvre	5.00
1357	3	3	First course	9.00
1357	4	2	Steak	12.00
2334	1	2	Covers	2.00
2334	2	2	Hors d'oeuvre	2.50
2334	3	2	First course	6.00
2334	4	2	Bream	15.00
2334	5	2	Coffee	2.00
3007	1	2	Covers	3.00
3007	2	2	Hors d'oeuvre	6.00
3007	3	3	First course	8.00
3007	4	1	Bream	7.50
3007	5	1	Salad	3.00
3007	6	2	Coffee	2.00

Receipts

Number	Date	Total
1357	5/5/92	29.00
2334	4/7/92	27.50
3007	4/8/92	29.50

- Incomplete information
  - Ex. table of county towns with its government office address
    - Other towns do not have government offices
  - Problem:
    - Florence is a county town, but address unknown
    - Tivoli is not a county town
    - Prato recently became a county town, government office may not have been existablished

City	GovtAddress
Roma	Via IV novembre
Florence	?
Tivoli	??
Prato	???

Do not use unused/unlikely domain values (0, 999, etc.) or typical values

  - Do not use unused/unlikely domain values (0, 999, etc.) or typical values to represent lack of information
    - B/c may lead of ambiguity (unused values may become meaningful)
    - Typical values can lead to trouble
      - Ex. calculate the age spread of a set of people
    - Need to distinguish btwn actual values and placeholders
  - Solution: **Null value**
    - Special value (not a value of any domain) which denotes the absence of a value
    - **Unknown value** → domain value exists, but unknown (Florence)
    - **Non-existent value** → attribute is not applicable for tuple (Tivoli)
    - **No-information value** → don't know if a value exists or not (Prato)
  - Database Management Systems do not distinguish between these types
    - Implicitly adopt no-information value

Integrity Constraints – properties

- **Integrity Constraints** – properties that must be satisfied by all meaningful database instances
  - Database = legal iff. It satisfies all integrity constraints
  - Reason:
    - Describe the application in greater detail
    - Contribute to data quality
    - Used by the system in choosing a strategy for query processing
  - **Intra-relational constraints**
    - **Tuple Constraint** – expresses conditions on the values of each tuple, independently of other tuples
      - Ex. Honors iff. Grade is A
        `NOT((Honors = "honors") OR (Grade = "A"))`
      - Ex. Finding the net value
        `Net = Gross – Deduction`
    - **Domain constraint** – tuple constraint that involves single attribute
      - Ex. Valid grade value is btwn A – F
        `(Grade <= "A") AND (Grade >= F)`
  - **Inter-relational constraints** → Referential Constraints
- **Keys** – a set of attributes that uniquely identifies tuples in a relation
  - A set of attributes `K` is a **superkey** for a relation `r` if `r` can not contain two distinct tuples `t1` and `t2` such that `t1.K` = `t2.K`
  - `K` is a key for relation `r` iff. `K` is a minimal superkey
    - **Minimal superkey** → no other superkey `K'` such that `K'`⊂`K`
  - Ex. students registeration 1

RegNum	Surname	FirstName	BirthDate	DegreeProg
284328	Smith	Luigi	29/04/59	Computing
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Fine Art

    - `RegNum` is a key → registration number identifies students
      - No pair of tuples w/ the same value for `RegNum`
    - `Surname`, `FirstName`, `BirthDay` is a superkey
      - No pair of tuples w/ the same values for all of `Surname`, `FirstName`, `BirthDate`
  - Ex. students registeration 2

RegNum	Surname	FirstName	BirthDate	DegreeProg
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Engineering

    - No pair of tuples w/ same values on both Surname and DegreeProg → can't conclude Surname and DegreeProg form key b/c there could be students w/ same surname in same program
  - Existence of keys
    - Relations are sets → each relation is composed of distinct tuples
    - Therefore, whole set of attributes for a relation = superkey
    - Existence of keys guarantees that each piece of data in the database can be accessed
  - If there are Null values, keys do not work well
    - 1. No guarantee of unique identification
    - 2. Do not help in establishing correspondences between data in different relations
    - Solution: primary keys
- **Primary Keys**
  - Presence of Null in keys has to be limited
  - Each relation must have a primary key → no Null value
    - Notation: attributes of primary keys are underlined
  - References btwn realtions are realized through primary keys
  - Ex. RegNum is the primary key

RegNum	Surname	FirstName	BirthDate	DegreeProg
643976	Smith	John	NULL	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	NULL	NULL
735591	Black	Lucy	05/03/58	Engineering

References Between relations

- References Between relations
  - Data in different relations referenced through (primary) key values

Students

RegNum	Surname	FirstName	BirthDate
6554	Rossi	Mario	5/12/1978
8765	Neri	Paolo	3/11/1976
9283	Verdi	Luisa	12/11/1979
3456	Rossi	Maria	1/2/1978

Exams

Student	Grade	Course
3456	30	04
3456	24	02
9283	28	01
6554	26	01

Courses

Code	Title	Tutor
01	Analisi	Neri
02	Chimica	Bruni
04	Chimica	Verdi

  - **Referential Constraints** – values on a set of attributes `X` of a relation `R1` must appear as values in relation `R2` where the `X` is the primary key of
    - X is a **foreign key** of relation R1
    - Referential integrity constraints are imposed in order to guarantee that the values are refer to existing tuples in the referenced relation
  - Ex. Referential constraints exists btwn:
    - Attributes Officer (of the relation Offences) and the RegNum (of the relation Officers)
    - Attributes Registration (of the relation Offences) and the Registration (of the relation Cars)

Offences

Code	Date	Officer	Dept	Registration
143256	25/10/1992	567	75	5694 FR
987554	26/10/1992	456	75	5694 FR
987557	26/10/1992	456	75	6544 XY
630876	15/10/1992	456	47	6544 XY
539856	12/10/1992	567	47	6544 XY

Officers

RegNum	Surname	FirstName
567	Brun	Jean
456	Larue	Henri
638	Larue	Jacques

Cars

Registration	Dept	Owner	...
6544 XY	75	Cordon Edouard	...
7122 HT	75	Cordon Edouard	...
5694 FR	75	Latour Hortense	...
6544 XY	47	Mimault Bernard	...

    - Violation of Referential Constraints
      - Officer 456 in relation Offences DNE in relation Officers

Offences

Code	Date	Officer	Dept	Registration
987554	26/10/1992	456	75	5694 FR
630876	15/10/1992	456	47	6544 XY

Officers

RegNum	Surname	FirstName
567	Brun	Jean
638	Larue	Jacques

Cars

Registration	Dept	Owner	...
7122 HT	75	Cordon Edouard	...
5694 FR	93	Latour Hortense	...
6544 XY	47	Mimault Bernard	...