relational model

Relational Model

- A database consists of several tables (relations)

Customer

CustID	Name	Street	City	State

Account

AccountNum	Balance

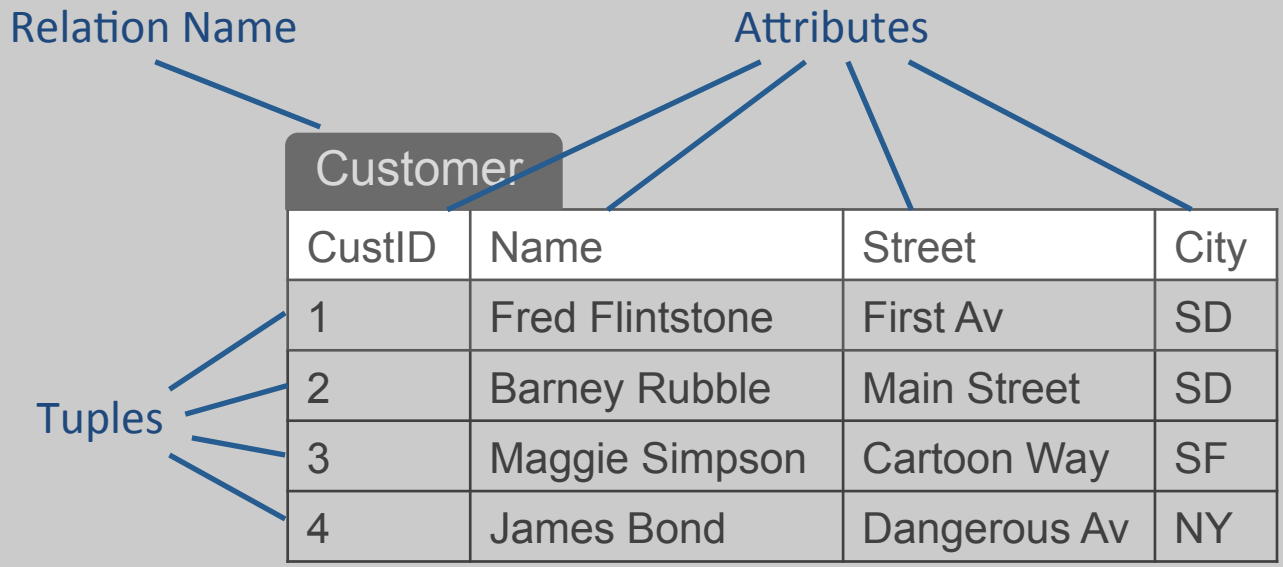
Depositor

CustID	AccountNum

- Columns in the tables are named by attributes
- Each attribute has an associated domain
  (set of allowed values)

  e.g. for Customer.State: {CA, NY, WA, …}
- Data in a table consist of a set of rows (tuples)
  providing values for the attributes

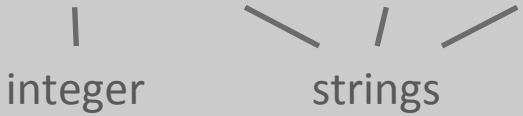
Relational Model Example



Relational Schema

- "Type declaration"
- Consists of:
  - Relation name
  - Set of attributes
  - Domain of each attribute
  - Integrity constraints

e.g. CUSTOMER(CustID, Name, Street, City)



Relational Schema

Attribute Types

- Each attribute of a relation has a:
  - Name
  - Domain: Set of allowed values
- Attribute values are (normally) required to be atomic; that is indivisible
- Sometimes, the special value null is considered a member of every domain

Relational Instance

- "The current content of the relation"
- Consists of:
  - A set of rows (tuples) over the attributes with values from the attribute domains

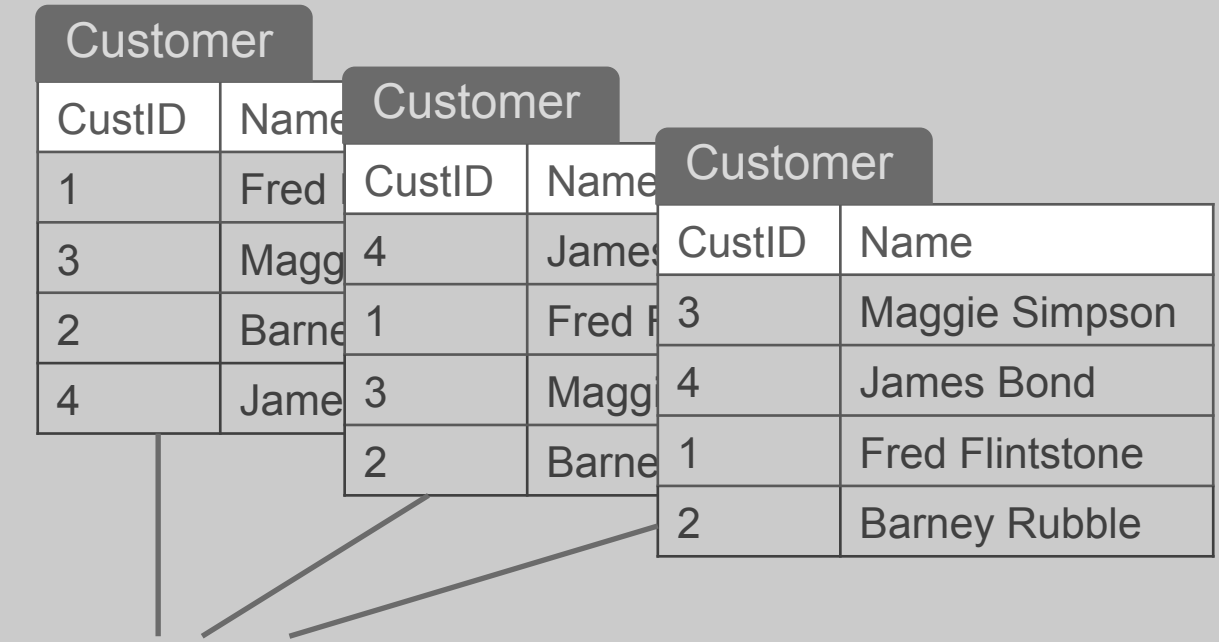
e.g.

Customer

CustID	Name	Street	City
1	Fred Flintstone	First Av	SD
2	Barney Rubble	Main Street	SD
3	Maggie Simpson	Cartoon Way	SF
4	James Bond	Dangerous Av	NY

Relations are Unordered

- The tuples are not considered to be ordered, even though they appear to be so when displayed in tabular form



Visual representations of the same relational instance

Tuples: Some notation

- Component values/coordinates of a tuple t: $t(A_i)$
  The value of attribute A_i for tuple t
- Subtuple of a tuple t: $t(A_i, A_j, \ldots, A_k)$
  The subtuple of t containing the values of attributes A_i, A_j, …, A_k

Tuples: Some notation

e.g.

Customer

	CustID	Name	Street	City
t →	1	Fred Flintstone	First Av	SD
	2	Barney Rubble	Main Street	SD
	3	Maggie Simpson	Cartoon Way	SF
	4	James Bond	Dangerous Av	NY

t = <4, “Fred Flintstone”, “First Av”, “SD”>

t(Name) = “Fred Flintstone”

t(Street) = “First Av”

Attribute and tuple values are generally assumed to be ordered

Database

- A database consists of multiple relations
- Information about an application is broken up into parts, with each relation storing one part of the information

account: stores information about accounts
depositor: stores information about which customer owns which account
customer: stores information about customers

Database

- Why not store all information as a single relation?
- It is possible
  e.g., *bank* (*accountNum, balance, customerName*, ..)
- But not desirable
  Results in repetition of information and the need for null values

Relational Integrity Constraints

- Constraints are conditions that must hold on all valid relation instances of a database
- Some common types of constraints:
  - Key constraints
  - Entity integrity constraints
  - Referential integrity constraints

Key Constraints

- Superkey of relation R:

  A set of attributes SK of R such that no two tuples *in any valid relation instance r(R)* will have the same value for SK. That is, for any distinct tuples t1 and t2 in r(R), t1(SK) ≠ t2(SK).

- Key of relation R:

  A "minimal" superkey; that is, a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey.

  e.g., the CAR relation schema:

  CAR(State, Reg#, SerialNo, Make, Model, Year)

  has two keys Key1 = {State, Reg#}, Key2 = {SerialNo}.

  {SerialNo, Make} is a superkey but not a key.

Key Constraints

- If a relation has *several* candidate keys, one is chosen arbitrarily to be the primary key.

Key Constraint Examples

- The primary key attributes are underlined

CAR	LicenseNumber	EngineSerialNumber	Make	Model	Year
	Texas ABC-739	A69352	Ford	Mustang	96
	Florida TVP-347	B43696	Oldsmobile	Cutlass	99
	New York MPO-22	X83554	Oldsmobile	Delta	95
	California 432-TFY	C43742	Mercedes	190-D	93
	California RSK-629	Y82935	Toyota	Camry	98
	Texas RSK-629	U028365	Jaguar	XJS	98

EMPLOYEE

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO

DEPARTMENT

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATIONS

DNUMBER	DLOCATION

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM

WORKS_ON

ESSN	PNO	HOURS

DEPENDENT

ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John		Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin		Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia		Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer		Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh		Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce		English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad		Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James		Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

WORKS_ON	ESSN	PNO	HOURS
	123456789	1	32.5
	123456789	2	7.5
	666884444	3	40.0
	453453453	1	20.0
	453453453	2	20.0
	333445555	2	10.0
	333445555	3	10.0
	333445555	10	10.0
	333445555	20	10.0
	999887777	30	30.0
	999887777	10	10.0
	987987987	10	35.0
	987987987	30	5.0
	987654321	30	20.0
	987654321	20	15.0
	888665555	20	null

DEPENDENT	ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP
	333445555	Alice	F	1986-04-05	DAUGHTER
	333445555	Theodore	M	1983-10-25	SON
	333445555	Joy	F	1958-05-03	SPOUSE
	987654321	Abner	M	1942-02-28	SPOUSE
	123456789	Michael	M	1988-01-04	SON
	123456789	Alice	F	1988-12-30	DAUGHTER
	123456789	Elizabeth	F	1967-05-05	SPOUSE

Entity Integrity

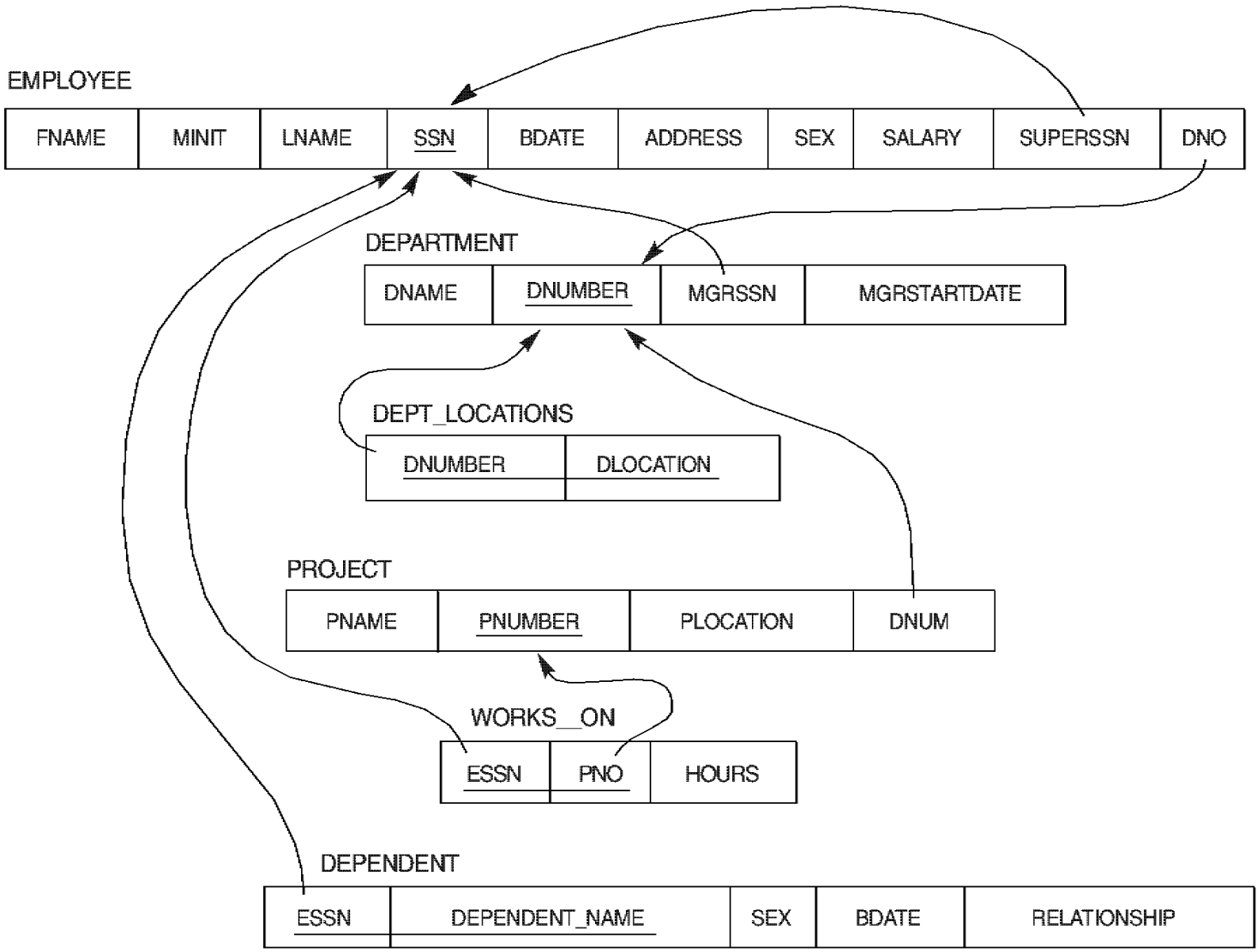
- The *primary key attributes* PK of each relation schema R in S cannot have null values in any tuple. This is because PK values are used to *identify* the individual tuples.

$$t(A) \neq \text{null for any tuple t in}$$
$$\text{a valid instance of R, where A is in PK}$$

Note: Other attributes of R may be similarly constrained to disallow null values, even though they are not members of the primary key.

Referential Integrity

- A constraint involving *two* relations of the database (the previous constraints involve a *single* relation).
- Used to specify a *relationship* among tuples in two relations: the referencing relation and the referenced relation.
- Tuples in the *referencing relation* R_1 have attributes FK (called foreign key attributes) that reference the primary key attributes PK of the *referenced relation* R_2. A tuple t_1 in R_1 is said to reference a tuple t_2 in R_2 if $t_1(FK) = t_2(PK)$.
- A referential integrity constraint can be displayed in a relational database schema as a directed arc from $R_1.FK$ to $R_2.PK$.


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS_ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

Referential Integrity Constraint

Statement of the constraint

The value in the foreign key column(s) FK of the **referencing relation** R_1 can be either (1) a value of a primary key PK in the **referenced relation** R_2 or (2) null.

In case (2), the FK in R_1 should not intersect its own primary key (or else entity integrity is violated)

Other types of constraints

- Semantic Integrity Constraints

  based on application semantics and cannot be expressed by the model per se
- Example

  - e.g., “the max. no. of hours per employee for all projects he or she works on is 56 hrs per week”
- A constraint specification language may have to be used to express these
- SQL-99 allows triggers and ASSERTIONS to support some of these