Database Systems

Relational Model

Prof. Dr. Agnès Voisard Muhammed-Ugur Karagülle

Institute of Computer Science, Databases and Information Systems Group

Fraunhofer FOKUS

2025

Notes

Notes

Outline

The relational model

- Introduced by Codd in 1970
- Based on a simple concept: relation
- Solid theoretical foundations
- Supports powerful - yet simple and declarative query languages

- Database: collection of relations
- Relation: resembles a table

Example:

The HOBBY relation

HOBBY

PersonName	Age	HobbyName
Jane	24	Fly fishing
Melony	25	Singing
Brian	29	Tennis
Brian	29	Jogging
Charlie	31	Dancing
Steve	24	Singing

Concepts (cont'd)

- Row: **tuple**; order of attributes matters
- Header: **attribute**. Has a certain type. All values in a column have a certain type.
- Data type describing the types of values of each column: **domain**.

Domains

Definition:

- A **domain** *D* is a set of atomic values.
- **Atomic value**: An indivisible unit within a domain.

Examples of Domains:

- $\{0, 1\}$: Binary digits.
- $\mathbb{N}$: Set of natural numbers.
- **Strings**: General set of strings, set of strings of length 12.
- **Names**: Set of names of individuals.
- **Hobbies**: Set of hobbies.
- **Social Security Numbers**: Set of 9-digit U.S. social security numbers.
- **Employee Ages**: Range of possible ages for employees, typically between 16 and 80.
- **Academic Departments**: Specific fields such as Linguistics, Mathematics, Economics, Physics, Computer Science.

Data type or **format** associated with a domain.

Example:

Data type for Employee-ages: integer number between 16 and 80.
For Academic-departments: set of all character strings representing valid departments, etc.

=> **Domain: name, data type**

Example:

FamilyName, string(20)

Attribute and relation schema

Attribute A_i

- Takes its values in a domain called domain of A_i, denoted by $dom(A_i)$.
- Several attributes can have the same domain.

A **relation schema** R

- Denoted by $R(A_1, A_2, \ldots A_n)$
  R: relation name and $A_1, A_2, \ldots A_n$: list of attributes.
- Used to describe a relation.
- **Degree** (**arity**) of a relation: number of attributes of its relation schema.

Attribute and relation schema (cont'd)

Example:

2 relation schemas:

HOBBY(PersonName, Age, HobbyName)
dom(PersonName) = Names

STUDENT(Name, SSN, HomePhone, Address, OfficePhone, Age, AverageGrade)

dom(Name) = Names,
dom(SSN) = Social-security-numbers, …

Relation (or **relation instance**) r of the relation schema $R(A_1, A_2, \ldots A_n)$ denoted by $r(R)$:

set of n-tuples $r = \{t_1, t_2, \ldots t_m\}$.

Each n-tuple: ordered list of values $t =< v_1, v_2, \ldots, v_n >$,
where each value v_i is an element of $dom(A_i)$ or a special null value.

Relation (cont'd)

Relation intension: schema R
Relation extension (state): relation instance $r(R)$

Notations

$R(A_1 : D_1, A_2 : D_2, \ldots A_n : D_n)$, where $D_i = dom(A_i)$
$\equiv R(A_1, A_2, \ldots A_n)$

$t[A_i]$ = value a_i in t for attribute A_i.

Example:

HOBBY(PersonName: FamilyName, Age: integer, Hobby-Name: Hobbies)
If $t =< Melony, 25, Singing >$ in the *HOBBY* relation,
$t[Age] =< 25 >$
$t[Name, Age] =< Melony, 25 >$.

Definition:

- A **relation** is a **set** of tuples.
- Tuples must be distinct.
- Impossible for two tuples to have the same combination of all their attributes.

Superkey:

- A **superkey** (SK) is a subset of attributes of a relation schema R, such that no two tuples have the same value for these attributes.
- Examples in a relation instance $r(R)$:
  - Consider tuples t_1 and t_2.
  - If $t_1[SK] \neq t_2[SK]$, then SK is a superkey.
- Every relation has at least one superkey.

A **key** K of R is a superkey of R such that
if any attribute A from K is removed,
the set of attribute K' (which is left) is not a superkey of R.

Example:

Relation *STUDENT*(Name, SSN, HomePhone, Address, Age, AverageGrade).
The attribute set $\{SSN\}$ is a key of *STUDENT* because no two students tuples can have the the same value for *SSN*.

Any set of attributes that includes *SSN* is a superkey (e.g., $\{SSN, Name, Age\}$).

Key (cont'd)

The value of a key uniquely identifies a tuple in a relation.

Key determined from the meaning of the attributes in the relation schema, not by the instances.

Relation schema has more than one key: **candidate keys**.

One is the **primary key**, used to identify tuples in a relation. (underlined).

Foreign key

Relation schema R_1.
Relation schema R_2 with primary key attributes *PK*.
A set of attributes *FK* in relation schema R_1 is a **foreign key** of R_1 if it satisfies:

- The attributes of *FK* have the same domain as the ones in *PK*.
- A value of *FK* in t_1 in R_1:
  - either occurs as a value of *PK* for t_2 in R_2 i.e., $t_1[FK] = t_2[PK]$
  - or is null.

Example:

R_1: *EMPLOYEE*(Name, Address, DName), FK: DName
R_2: *DEPARTMENT* (DName, Manager, NumberOfEmployees), PK: DName
DName is a foreign key of *EMPLOYEE* (from DEPARTMENT).

Relational database schema S: set of relation schemas.
$S = \{R_1, R_2, \ldots, R_m\}$

and a set of **integrity constraints** IC.

Simplest type of integrity constraints: data type of each data item.

Integrity constraints (cont'd)

Example:

In the *STUDENT* relation, value of AverageGrade must be an integer between 0 and 5. The value of Name must be a string of no more than 30 characters.

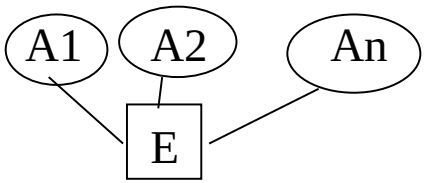
+ semantics aspects under the database designer's responsibility.

Relational database instance *DB* of *S* is a set of relation instances r_i
$DB = \{r_1, r_2, \ldots, r_m\}$ such that

- each r_i is an instance of R_i
- each r_i satisfies the integrity constraints defined in *IC*.

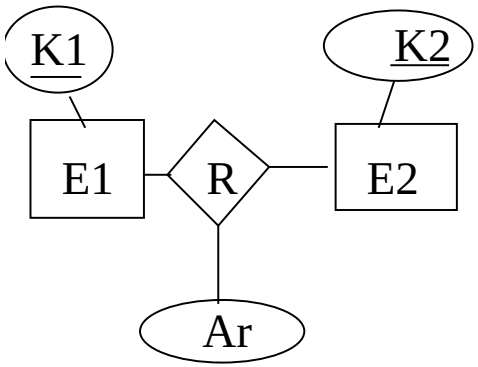
From ER to relational

- Each entity type E with attributes $A_1, \ldots, A_n$: relation schema R_E with attributes $A_1, \ldots, A_n$.
- Each relationship R between $E_1, E_2, \ldots, E_n$: Relation schema R_R whose attributes are the key of $E_1, E_2, \ldots, E_n$.



RE

A1	A2	...	An

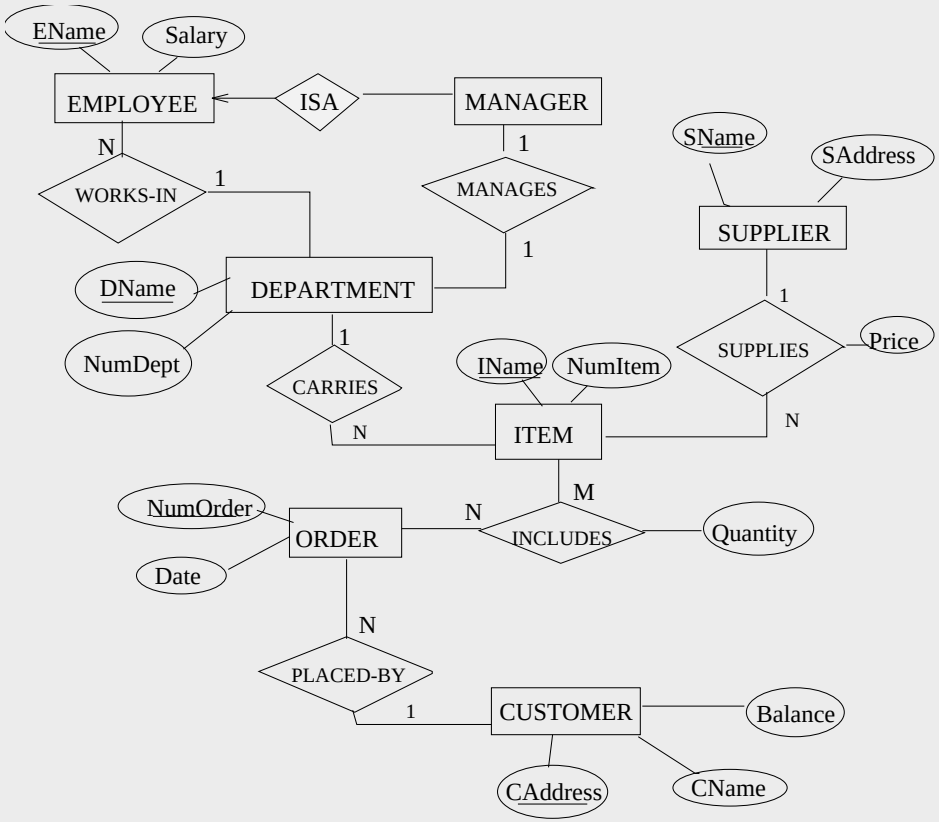


RR

K1	K2		Ar

Example:

Entity-relationship diagram:



Examples (cont'd)

Example:

Relational database schema:

EMPLOYEE(EName, Salary)
MANAGER(EName)
WORKS-IN(EName, DName)
MANAGES(EName, DName)
DEPARTMENT(DName, NumDept)

Examples (cont'd)

Example:

SUPPLIER(SName, SAddress)
ITEM(IName, NumItem)
CARRIES(IName, DName)
ORDER(NumOrder, Date)
CUSTOMER(CName, CAddress, Balance)
SUPPLIES(SName, IName, Price)
PLACED-BY(NumOrder, CName)
INCLUDES(NumOrder, IName, Quantity)

Remarks on the relational model

- Relations are sets: no duplicates
- Tuples identified through the value of attributes: relational model value-oriented
- No notion of order within a relation: no first, no last, etc.
- Attribute values are atomic.

1. **DDL** (for defining relation schemas):
   - creation, deletion of:
     - a relation schema
     - a database
   - Addition, deletion of an attribute

Operations on a relational database (cont'd)

2 **DML** (fixed schema):
- input of tuples
- display of a relation
- modification of a relation:
  - insertion, suppression, update
- queries
  - consultation of a relation
  - computation of a new relation

3 Transaction management

4 View management

Questions?

What will come next?

1 Welcome to Database Systems
2 Introduction to Database Systems
3 Entity Relationship Design Diagram (ERM)
4 Relational Model
5 Relational Algebra
6 Structured Query Language (SQL)
7 Relational Database Design - Functional Dependencies
8 Relational Database Design - Normalization
9 Online Analytical Processing + Embedded SQL
10 Data Mining
11 Physical Representation - Storage and File Structure
12 Physical Representation - Indexing and Hashing
13 Transactions
14 Concurrency Control Techniques
15 Recovery Techniques
16 Query Processing and Optimization