Relational Database Model

Introduced by E.F. Codd (1970) http://www.acm.org/classics/nov95/

Based on relational algebra and logic developed by

Schröder (1880s) Charles Peirce (1890s) Russell and Whitehead (1900s)

Codd's Twelve Rules Information represented at the logical level in tables. Data is determined by table, primary key, and column. Missing information is modeled as null values. Metadata is part of the database. Single language for all tasks in DBMS. Views and tables must change simultaneously. Single operations for retrieve, insert, delete, update. Operations independent of physical storage and access.

9. Database modifiable without affecting applications.

10. Constraints are part of database.

11. DML independent of physical layer (distributed, etc.)

12. Row-processing obeys same rules as set-processing.

Relations

Extensional versus intensional

Extensional Representation:

table of values rows = records columns = attributes

Note: tables are ordered, relations are not

Domains

```
Set of atomic values for an attribute
atomic = indivisible
(e.g. IT 240 = IT + 240 is divisible)
```

Examples age: {x: x is a valid age} sex: {male, female}

Physical Level: data type + format

Relation Schema R(A₁, A₂, ..., A_n) relational schema R: Name of Relation A₁, A₂, ..., A_n: Attributes A_i has domain D_i N: degree (arity) of R Movie(movieID, title, genre, length, rating) dom(movieID) = MovieIDs = {x: x is a number} dom(title) = Titles = {x: x is a number} dom(genre) = Genres = {Musical, Horror, ...} dom(length) = Lengths = {x: x is valid time} dom(rating) = Ratings = {NR, G, PG, PG-13, R, NC-17}

Relational Schemas Example

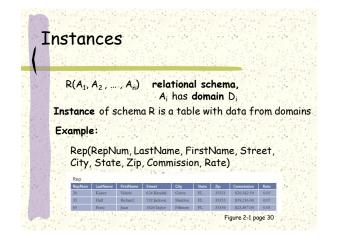
CUSTOMER(Customer_ID, Customer_Name, City, State, Postal_Code)

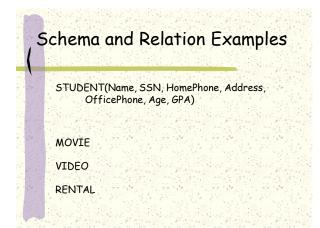
ORDER(Order_ID, Order_Date, Customer_ID).

ORDERLINE(Order_ID, Product_ID, Ordered_Quantity)

PRODUCT(Product_ID, Product_Description, Product_Finish, Standard_Price, Product_Line_ID)

More Examples: page 32





Records

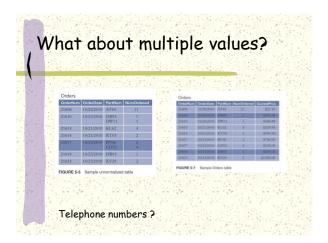
<101, "Thirty-Nine Steps", mystery, 101, R> <510, "Monkey Business", comedy, null, null>

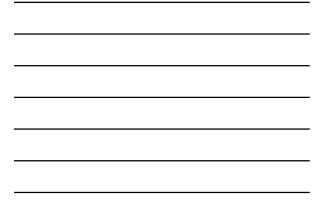
are possible records (or tuples) in

MOVIE(movieID, title, genre, length, rating).

null: value unknown, or attribute does not apply

values **atomic**: no multiple values (first normal form) (e.g. several genres) indivisible (name = first name + last name)





Functional Dependencies I

ORDER_DETAIL(Product, Quantity, PricePerUnit, Subtotal)

Example <Soymilk, 3, \$3.19, \$9.57> in ORDER_DETAIL

(Quantity, PricePerUnit) \rightarrow Subtotal

Quantity and PricePerUnit determine Subtotal

Functional Dependencies II

A→B (A determines B) if setting the values of attribute(s) of A determines the values of B

Example

SID \rightarrow Name in STUDENT(SID, Name)

Functional Dependencies III

EMPLOYEE(EmpID, FirstName, LastName, Salary, Sex)

STUDENT(Name, SID, HomePhone, Address, OfficePhone, Age, GPA)

PRODUCT(ProductID, Name, Description, PricePerUnit, UnitSize)

CAR(OwnerName, Vehicle#, Engine#, Color)

CUSTOMER(accountID, lastName, firstName, street, city, state, zipcode)

Constraints

- Domain constraints
- Key (uniqueness) constraints
- Entity integrity constraints
- Referential integrity constraints
- Data dependencies (functional dependencies, etc.)

Domain Constraints

Restriction on values of attributes (domain).

Specified as **data-type**: integer, char, etc., or user-defined type

Operations on data-types: +, *, <, =, ...

not null constraint for an attribute

Key: smallest set of attributes that determines all attributes in a relation. composite: more than one attribute

Example

Keys

STUDENT(Name, SID, HomePhone, Address, OfficePhone, Age, GPA)

{SID}: key {SID, Name, Address}: not key

Key Examples

MOVIE(Title, Year, Director, Length, Rating)

PRODUCT(ProductID, Name, Description, PricePerUnit, UnitSize)

ACTIVITY(StudentID, Activity, Fee)

COURSE(CourseID, Title, Enrolment)

ENROLED(StudentID, CourseID, CourseTitle, Grade)

Candidate Keys

If a relation has more than one key, these keys are called **candidate keys**.

Examples

• • •

- EMPLOYEE(EmpID, FirstName, LastName, Salary, Gender)
 - DePaul students: peoplesoftID and SSN
- COURSE(Department, Number, Name, Instructor)
- CAR(OwnerName, Vehicle#, Engine#, Color)

One candidate key is declared the **primary key** of the relation (underlined in schema)

	23.3			1.21.2			States .	1
Relatio	nal r	ataba	se Sche	m	a: Coll	ection of	Relation	ns
						ction of I		
Relatio		Jurubu			conce		instance	
ACTI	VITI	ES = {	STUDE	N	Т, АСТ	IVITY}		1.4
ACTI Student	VITI	ES = {	STUDE	N	T, ACT	IVITY}	Activity	Fe
	1 21		1. St. 1.		1. 5.	" and the s	<u>Activity</u> Piano	Fe \$2

Foreign Key 1

54. 5 . a.a.

A set of attributes in one relation (R $_1$) refering to a unique tuple in a second relation (R $_2$) through R $_2$'s primary key.

Student	SID	FName	LName	1	Activity	StudentID	Activity	Fee
Level.	101	Mark	Spencer			971	Piano	\$20
1.246	353	Gil	Ryle		Sec. 94	353	Reading	\$5
	971	Charles	Loeffler	1		971	Swimming	\$10

Terminology $\begin{array}{c} R_1 \mbox{ referencing relation} \\ R_2 \mbox{ referenced relation} \end{array}$

Foreign Key 2

Examples

REGISTRATION = {STUDENT, ENROLMENT, COURSE} COMPANY = {EMPLOYEE, WORKS_ON, PROJECT} SUPPLY = {SUPPLIES, SUPPLIER, PART, COMPANY}

Note R1 = R2 is possible

Example

EMPLOYEE (with supervisor)

MOVIE (remakes)

Referential Integrity

declaration of foreign keys in a database schema

STUDENT(SID, FName, LName) ACTIVITIES(StudentID references STUDENT, Activity, Fee)

or visually, by an arrow from foreign key to primary key

STUDENT(SID, FName, LName)

ACTIVITIES(<u>StudentID</u>, <u>Activity</u>, Fee)

Integrity Constraints

Domain Constraints: declaration of domains
Not Null Constraints: attribute values can not be null
Key Constraints: candidate keys (uniqueness)
Entity Integrity Constraint: primary key is not null
Referential Integrity Constraint: declaring foreign keys

A valid state is a database state fulfilling all integrity constraints Integrity Constraints defined by DDL

Semantic constraints (transitions) later