Contents

DBMS Classification 1

Hierarchical Database . . . 2

Network Database . . . 2

Object-Orientated Database . . . 2

Graph Database . . . 2

Relational Database . . . 2

The Relational Model 3

Formal Definition . . . 4

Characteristics of Relations . . . 5

Relational Model Constaints . . . 5

Keys and Integrity Constraints . . . 6

Primary Key . . . 6

Entity Integrity Constraint . . . 6

Referential Integrity . . . 7

Foreign Keys . . . 7

Table Relationships . . . 7

DBMS Classification

- Main method of DBMS classification is via the conceptual data model used
- The choice of model affects virtually all other components in the system
  - Particularly the external schemas and associated DML
- Examples
  - Hierarchical
  - Network
  - Relational
  - Object-oriented and Object-relational
  - Graph, Columnar, In Memory, NoSQL…

Hierarchical Database

- One of the oldest database models
    - Commonly used in Mainframe computing
- Organised hierarchically with parent and child nodes
    - Like a family tree!

Network Database

- Also have a hierarchical structure
- Uses “members” and “owners” rather than “parents” and “children”
- Each member can have more than one owner

Object-Orientated Database

- Attempts to Model Data Storage in a similar fashion to application programs
    - Persistent storage of program objects such as class definitions
    - Objects can survive part the end of program execution
- Impedance Mismatch Problem
    - Data Strutures incompatible with the programming language’s Data Structures

Graph Database

- Uses a graph stracture with:
    - Nodes
    - Edges
    - Properties
- Graph databases treat the relationship between things as equally important to the things themselves

Relational Database

- Differs from previous models as it is not Hierarchical, but Relational
- More flexible than either the hierarchical or network database models
- Uses notions of:

  - Relations (Tables)
  - Tuples (Rows)
  - Attributes (Columns)

- The Relational Model
  - First introduced in 1970
  - Theoretical Bases
    - Set Theory
    - First-Order Predicate Logic
- Database represented as a collection of *mathematical relations*
  - Informally, relations resemble tables of values
- The *table*, or *relation*, is the basic storage structure of a Relational Database
  - Tables are "Two-Dimensional"
- Each *row*, or *tuple*, in a table represents a collection of related values
  - A row represents a fact that corresponds to an entity or relationship in the real world
- Each *column*, or *attribute*, contains values of the same data type

STUDENT

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2431	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.2.1
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	433-11-2320	NULL	3452 Elgin Road	(817)749-6492	25	3.53

STUDENT = Relation Name

Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa = Attributes

Rows = Tuples

The Relational Model

- Domain

  - The data type describing the values that can appear in each column is represented by a *domain* of possible values
  - mobile_phone_number: The set of 10 digit phone numbers valid in Ireland
  - PPS_number: 9 charaters in length. 7 numeric characters in positions 1 to 7, followed by 1 alphabetic check character in position 8, and either a space or the letter “W” in position 9

Formal Definition

- A relational schema `R`, denoted by $R(A_1, A_2, \ldots, A_n)$ is made up of:
  - *relation* name `R`
  - List of *attributes* $A1 \ldots A_n$
  - Each *attribute* A_i is the name of the role played by *domain* D_i in the *relation* `R`
    - D_i is the *domain* or A_i and is denoted by $dom(A_i)$
  - The *degree* of a schema, is equal to the number of *attributes*, `n`

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2431	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.2.1
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	433-11-2320	NULL	3452 Elgin Road	(817)749-6492	25	3.53

- `STUDENT(Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa)`
- The *degree* of the relation **STUDENT** is
- `dom(Ssn) = ...`
- A *relation state* `r` of a relational schema $R(A_1, A_2, \ldots, A_n)$ also denoted `r(R)` is:
  - A *set of tuples* $r = \langle t_1, t_2, \ldots, t_m \rangle$
  - Each *tuple* `t` is an ordered list of `n` values $t = \langle v_1, v_2, \ldots, v_n \rangle$
    - where each value $v_i, 1 \leq i \leq n$, is an element of $dom(A_i)$
  - The i^{th} value of tuple t_n, which corresponds to attribute A^i, if referred to as $t_n[A_i]$ or $t_n[i]$

$t_3 = t_2[A_3] =$ 381-62-1245

Characteristics of Relations

- Ordering of tuples in a relation
  - A Relation defined as a *set* of tuples
  - Elements of a *set* have no order among them
- Ordering of values within a tuple
  - Each *tuple* `t` is an ordered list of `n` values $t = \langle v_1, v_2, \ldots, v_n \rangle$
  - Order can change as long as correspondence between attributes and values is maintained
- Values in tuples
  - Each value in a tuple is *atomic*
    - For example: Student Age
    - Composite and multivalued attributes not allowed in the "Flat" Relational Model
  - Multivalued attributes
    - For example: College Degree
    - Must be represented by separate relations
  - Composite attributes
    - For example: Address
    - Represented only by simple component attributes in basic relational model
- NULL values
  - Represent the values of attributes that may be unknown or may not apply to a tuple
  - Meanings for NULL values
    - Value unknown
    - Value exists but is not available
    - Attributes does not apply to this tuple (also known as value undefined)
  - The NULL value is defined for each domain and there are restrictions

Relational Model Constaints

- Restrictions on the actual values that can be placed in a database state
- These rules are derived from the rules of the world that the database represents
- Constraints can generally be divided into three categories:
  - Constraints inherent in the data model

        - *Inherent model-based* or *implicit constaints*
    - Constraints expressed in the schemas of the data model i.e. DDL
        - *Schema-based* or *explicit constraints*
    - Constraints that cannot be expressed in the DDL
        - Must be enforced by the application programs
        - *Application-based* or *semantic constraints*, *Business Rules*

Keys and Integrity Constraints

- A Relational DB consists of many relations
    - tuples of those relations can be related in various ways
- Every relation and every attribute has a name
    - As a result, can be uniquely identified
- Attrbute names are often qualifed by relation name
    - Resolves ambiguity
        - PATIENT.name
        - DOCTOR.name

Primary Key

- Most relations have one attribute whose values uniquely identify its tuples
    - e.g. student_names in the relation STUDENT
    - no two students can have the same student number
- This attribute is known as a *key*
    - More specifically, this type of *key* is called a *Primary Key*
- Not every relation uses a single attribute as its Primary Key
- When multiple Candidate Keys exist, they may be combined, or one chosen, to form a Primary Key

Entity Integrity Constraint

- Specifies that there may not be any duplicate entries in the Primary Key attribute
- NULL values are not permitted in Primary Key fields
    - Primary Key is used to identify a tuple
    - Having a NULL in a Primary Key implies that we cannot identify some tuples
- Once defined, Key and Entity Constraints are enforced by the DBMS

Referential Integrity

- Key and Entity Constraints are specified on individual relations
- Referential Integrity Constraints are specifies between two relations
  - Maintains consistency among tuples in the two relations
- Informally
  - A tuple in one relation that refers to another relation, must refer to an *existing tuple* in tha relation

Foreign Keys

- A Foreign Key formally specifies a Referential Integrity Constraint between two relations
- Consider two relation schemas R_1 and R_2
- A set of attributes `FK` in R_1 is a Foreign Key of R_1 that references R_2 if:
  - The attributes of `FK` have the same domains as the Primary Key attributes `PK` of R_2
    - `FK` is said to reference or refer to R_2
  - A value of `FK` is a tuple `t1` either occurs as a value of `PK` for some tuple `t2`, or is NULL
    - tuple t_1 is said to reference or refer to tuple t_2

Table Relationships

EMPNO	NAME	JOB	DEPTNO
7856	MCNULTY	OFFICER	30
7710	DANIELS	LIEUTENANT	40
7992	GREGGS	DETECTIVE	10
7428	MORELAND	DETECTIVE	20

DEPTNO	NAME	LOCATION
10	NARCOTICS	TOWER 221
20	HOMICIDE	CITY CENTER
30	MARINE	DOCKS
40	EVIDENCE	DOWNTOWN

DEPTNO	NAME	LOCATION

DEPTNO in first table is the Foreign Key

EMPNO and DEPTNO are Primary Keys