Data Models & Relational Model

Mirunalini.P

SSNCE

March 8, 2022

Session Objective

- To learn about Data models
- To learn about Relational databases

Session Outcome

At the end of this session, participants will be able to:

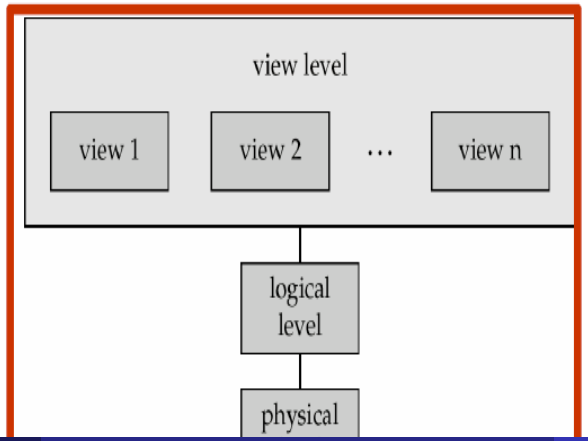
- Understand the different data models
- Understand Relational database Basic

Table of Contents

Data abstraction

Fundamental characteristic of database approach is to provide some level of data abstraction

- Refers to the suppression of details of data organization and storage
- Highlighting the essential features for improved understanding of data.
- By data abstraction different users can perceive data at their preferred level of detail.



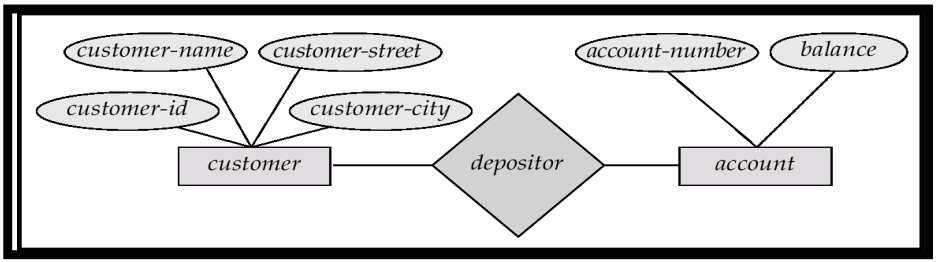
Data Model

A **data model** is a collection of concepts to describe:

- **Structure of a database** - elements, group of elements and relationship between them
- **Operations** for manipulating these structures - to specify dynamic behaviour apart from basic operations of the db application
- **Constraints** the db should obey - specify some restrictions on valid data must be enforced at all times

Categories of Data Models

- **Conceptual (high-level, semantic) data models**:
  1. Provide concepts for presenting data that are close to the way many users perceive data.
  2. Also called entity-based or object based data models
  3. Use concepts such as entities, attributes, and relationships.



- **Object-Based Data Model** : Extending the E-R model with notions of encapsulation, methods (functions) and object identity.

Categories of Data Models

Physical (low-level, internal) data models:

- Provide concepts that describe details of how data is stored in the computer.
- Provides information such as record formats, record orderings and acess paths.
- Specifies the indexing and hashing techniques that makes the search for particular database efficient.

Implementation (representational) data models:

- Provide concepts that fall between the two.
- Eg: Relational, Network and Hierarchical models
- Uses a collection of tables to represent both data and the relationships among those data.
- Each table has multiple columns, and each column has unique name
- Widely-used in commercial DBMS

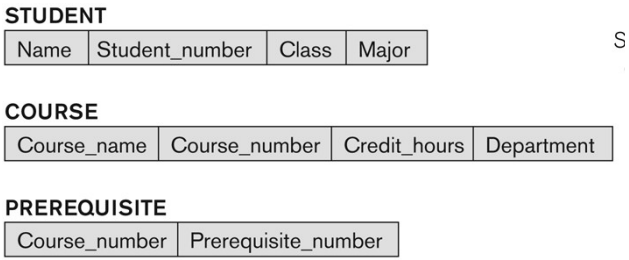
Relational data model

Customer-id	*customer-name*	*customer-street*	*customer-city*	*account-number*
192-83-7465	Johnson	Alma	Palo Alto	A-101
019-28-3746	Smith	North	Rye	A-215
192-83-7465	Johnson	Alma	Palo Alto	A-201
321-12-3123	Jones	Main	Harrison	A-217
019-28-3746	Smith	North	Rye	A-201

Schema

Schema

- The description of a database is called the database schema
- Specified during database design and is not expected to change frequently.
- A displayed schema is called a **schema diagram**.
- Object in the schema are called as **schema construct**
- A schema diagram displays only some aspects of a schema, such as the names of record types and data items, and some types of constraints.
- The database schema changes very infrequently.



Database State or Instance:

- The actual data in a database may change frequently.
- **Data Instance:**The actual data stored in a database at a particular moment in time.
- The database state changes every time the database is updated.
- DBMS is responsible is responsible for ensuring every state of the db is a valid state.

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

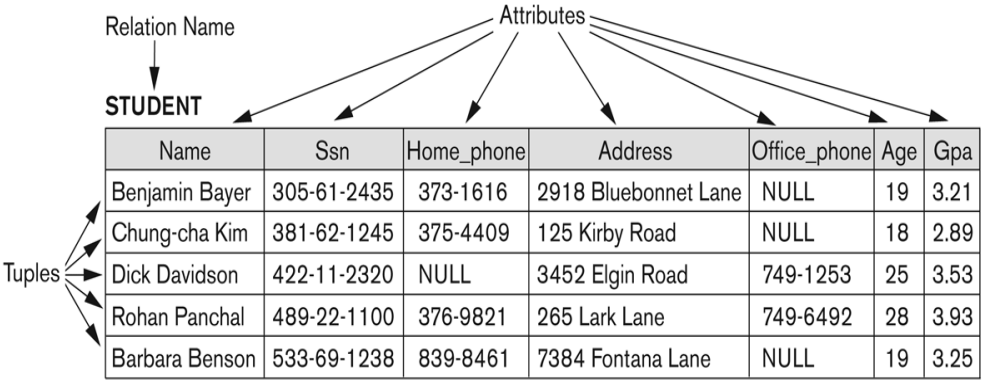
PREREQUISITE

Course_number	Prerequisite_number

Relational Model

- The relational Model of Data is based on the concept of a Relation which is based on the ideas of sets
- A relation looks like a table of values.
- A relation typically contains a **set of rows.**
- The data elements in each row represent certain facts, rows are called as **tuples**
- Each column has a column header called as **attribute name**

Relational Model - Example



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25

Formal Definition - Relational Schema

The Schema (or description) of a Relation:

- Denoted by $R(A1, A2, ...An)$
- R is the name of the relation
- The attributes of the relation are $A1, A2, ..., An$
- Each attribute A_i is role played by some domain D in the relation schema R

Formal Definition - Domain

- A domain D is set of atomic values.
- D is called domain of A_i denoted by $dom(A_i)$
- A domain has a logical definition and also has a data-type or a format defined for it.
- Domain is thus given a name, data type and format
- Degree of a relation is the number attributes n in the relation
- Ex:CUSTOMER (Custid, Custname, Address, Phone)
- For example, the domain of Custid is 6 digit numbers.

Formal Definitions - Relation State

Given $R(A1, A2, ..., An)$

- The relation r of the relation schema $R(A1, A2, ..., An)$
- Also denoted as $r(R)$: a **specific state**
- $r = \{t1, t2, ..., tn\}$ set of n-tuple
- Each tuple t_i is ordered list of n values
- $ti =< v1, v2, ...vn >$ where each vj element-of $dom(Aj)$
- The relation state is a subset of the **Cartesian product** of the domains of its attributes

$$r(R) \subset dom(A1) * dom(A2) * .... * dom(An)$$

Characteristics of Relational

- **Ordering of tuples in a relation**: The tuples are not considered to be ordered, even though they appear to be in the tabular form
- **Ordering of attributes**: We will consider the attributes in $R(A1, A2, ..., An)$ and the values in $t =< v1, v2, ..., vn >$ to be ordered.
- **Values in a tuple**: All values are considered atomic (indivisible). A special null is used to represent **unknown or inapplicable values**

Relational Model Constraints

There are many restrictions or constraints on the actual values in a database state.

- **Model-based constraints or implicit constraints:**
  - Constraints inherent in the data model
  - Eg: Relation cannot have duplicate tuples (or) no two tuples have the same conbination of values
- **Schema-based constraints or explicit constraints**
  - Constraints directly expressed in the schemas of the data model
  - Specified using DDL statements
  - Eg: Key constraints, Domain constraints etc.,
- **Application-based or semantic constraints or business rules.**
  - Constraints enforced by the application pro- grams
  - Eg: Triggers, views, stored procedures

Domain Constraints

- Domain constraints specify that within each tuple, the value of each attribute A must be an atomic value from the domain dom(A).
- Domain constraints can be violated if the attribute value is not from corresponding domain.

Keys

Constraints are conditions that must hold on all valid relation instances.
Three types of keys:

- Candidate keys
- Primary keys and Alternate keys
- Foreign keys

Candidate Keys

A realtion is defined as a set of tuples, by definition all elements in a set are distinct all tuples are distinct

At any given time, no two tuples in any valid relation instance $r(R)$ will have the same value for the set of attributes.

$$t1[SK] \neq t2[SK] \tag{1}$$

Any such subsets are called as **superkey**

A superkey specifies a uniquness constraint such that no two distinct tuples in any state of r of R can have same value

Every relation should have one default superkey - The set of all its attributes

Candidate Keys

A key k of a relation schema R is a superkey of R with an additional property that removing any attribute A from K leaves a set of attributes K^+ that is not a superkey of R any more

Let K be the set of attributes of R.

Then K is the candidate Key or Key for R if it has following properties:

- **Uniqueness**: No value of two distinct tuples of R contains the same value for K
- **Minimal superkey**: A superkey from which we cannot remove any attributes and still have the uniqueness constraint hold. This minimality property is required for a key but is optional for a superkey.
- **A key is a superkey but not vice versa**
- A superkey must be key (if it is minimal) or may not be a key

Candidate Keys

- {First} and {Last} do not give a unique identifier for each row but in {ID} uniqueness hold.
- Any set of attributes that includes {ID} uniquness hold.
- {ID, First}, {ID, Last} and {ID, First, Last} satisfy uniqueness, but are not minimal
- Candidate keys are {ID} and {First, Last}

Table 1: Candidate Keys

ID	First	Last
S139	John	Smith
S140	Mary	Jones
s141	John	Brown
S142	Jane	Smith

Primary Key

A database has following implicit properties:

- A given relation may have two or more candidate keys
- Exactly one of those keys be chosen as the primary key, the others are then called alternate keys
- We could choose either **{**{ID} or {First, Last}} as the Primary Key.
- **{ ID}** is more convenient as it is a single column and we know it will always be unique (what happens if another John Smith is added?)

Entity Integrity

Entity integrity constraint states that:

- No Primary Key (PK) value can be NULL.in any tuple of r(R)
- This is because primary key values are used to identify the individual tuples in a relation.
- If two or more tuples had NULL for their primary keys we are not able to distinguish them
- Other attributes of R may be constrained to disallow null, even though they are not members of the PK

Foreign Keys

A constraint involving two relations

- Used to specify a relationship among tuples in two relations: the referencing relation and the referenced relation
- A foreign key is a set of attributes, say FK, of referencing relation whose values are required to match values of candidate key CK of referenced relation
- Referential integrity – the database must not contain any unmatched foreign key values
- The value in the foreign key column FK of the referencing relation can be :
  - Same domain value of the referenced relation
  - a value of an existing primary key PK in the referenced relation, or
  - a null
- In other words, the constraint says simply: If B references A, then A must exist

Referential Integrity

Table 2: Department

DID	DName
13	Marketing
14	Accounts
15	Personnel

{DID} is a Foreign Key in Employee - each Employee's DID value is either NULL, or matches an entry in the Department relation.

Table 3: Employee

EID	EName	DID
15	John Smith	13
16	Mary Jones	14
17	John Brown	13
18	Jane Smith	Null

Referential Integrity

Table 4: Employee

ID	Name	Manager
E1496	John Smith	E1499
E1497	Mary Brown	E1498
E1498	Mark Jones	E1499
E1499	Jane Smith	NULL

{ID} is a Candidate Key for Employee, and {Manager} is a Foreign Key, which refers to the same relation – every tuple's Manager value is either NULL or matches an ID value.

FOREIGN KEY Manager REFERENCES Employee(ID)

Referential Integrity - violations

- When relations are updated, referential integrity can be violated
- When a referenced tuple is updated / deleted, an exception will be raised
- To compensate this, the options are:
  - **Cascade:** Actions cascaded to the matching tuples
  - **Restrict:** Restricted to the case where there are no matching tuples
  - **Nullify:** Make values NULL

Referential Actions – An Example

What happens if

- Marketing's DID is changed to 16 in Department?
- The entry for Accounts is deleted from Department?

Table 5: Department

DID	DName
~~13~~ 16	Marketing
~~14~~	~~Accounts~~
15	Personnel

Table 6: Employee

EID	EName	DID
15	John Smith	13
16	Mary Jones	14
17	John Brown	13
18	Jane Smith	Null

Referential Actions – CASCADE

- CASCADE allows the changes made to flow through
- If Marketing's DID is changed to 16 in Department, then the DIDs for John Smith and John Brown also changes
- If Accounts is deleted then so is Mary Jones

DID	DName
~~13~~ **16**	Marketing
~~14~~	~~Accounts~~
15	Personnel

EID	EName	DID
15	John Smith	**16**
~~16~~	~~Mary Jones~~	~~14~~
17	John Brown	**16**
18	Jane Smith	Null

Referential Actions - Nullify

- NULLIFY sets referenced values to NULL
- If Marketing's DID changes then "John Smith and John Brown " DIDs are set to NULL
- If Accounts is deleted, Mary Brown's DID becomes NULL

DID	DName
~~13~~ **16**	Marketing
~~14~~	~~Accounts~~
15	Personnel

EID	EName	DID
15	John Smith	~~13~~ NULL
~~16~~	~~Mary Jones~~	~~14~~ NULL
17	John Brown	~~13~~ NULL
18	Jane Smith	NULL

Reference

 Fundamentals of Database systems 7^{th} Edition by Ramez Elmasri.