Principles of Database Systems

Introduction to the Relational Model

Review Data Model

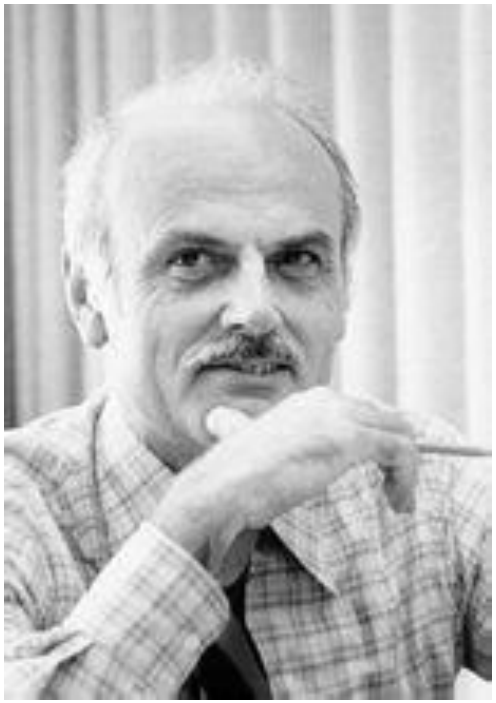
- Data Model
  - a collection of conceptual tools for describing
    - data
    - data relationships (数据联系)
    - data semantics (数据语义)
    - consistency constraints (一致性约束)

- A data model provides a way to describe the design of a database at the physical, logical, and view levels.

Review Data Model

- In the course, data models can be classified as
  - *Conceptual Data Model*
    - *Entity-Relationship Model (实体-联系模型)*
  - *Logical Data Model*
    - ***Relational model (关系模型)***
    - *network data model (网状模型)*
    - *hierarchical data model (层次模型)*
    - *Object-based data model (基于对象的数据模型)*
    - *Semistructured data model (半结构化数据模型)*
  - *Physical Data Model*
    - *B* tree model…*

Relational Model



埃德加·弗兰克·科德（Edgar Frank Codd，1923－2003），IBM公司研究员，被誉为“关系数据库之父”，并因为在数据库管理系统的理论和实践方面的杰出贡献于1981年获图灵奖。1970年，科德发表题为“**A Relational Model of Data for Large Shared Data Banks(大型共享数据库的关系模型)**”的论文，文中首次提出了数据库的关系模型。由于关系模型简单明了、具有坚实的数学理论基础，所以一经推出就受到了学术界和产业界的高度重视和广泛响应，并很快成为数据库市场的主流。20世纪80年代以来，计算机厂商推出的数据库管理系统几乎都支持关系模型，数据库领域当前的研究工作大都以关系模型为基础。

Relational Model

- Relational data structure
- Integrity constraints
  - constraints on attributes of schemas, e.g. value domain, type （域完整性）
  - constraints on dependencies among attributes of a schema （实体完整性）
  - constraints on dependencies among attributes of different schemas （参照完整性）
- Operations on the model

Structure of Relational Databases

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Structure of Relational Databases

- A relational database consists of a collection of relational tables
  - a row in a table represents a **relationship** among a set of values
  - a table is such a collection of relationships.

- As an example, we consider the table instructor, a row in the table can be thought of as representing the relationship between a specified ID and the corresponding values for name, dept_name, and salary values.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000

关系/元组/属性/域/笛卡尔积

元组/记录/行

姓名	生日	身高	项目	时间	国家号
博尔特	1986.821	196	100米跑	9’58	1
苏炳添	1989.8.29	172	100米跑	9’99	2
宁泽涛	1993.3.6	191	100米自	47’65	2

属性/字段/列

编号	国家名
1	牙买加
2	中国
3	美国

Structure of Relational Databases

- Since a table is a collection of such relationships, there is a close correspondence between the concept of **table** and the mathematical concept of **relation**.

- A relationship between n values is represented mathematically by an n-**tuple** of values, i.e., a tuple with n values, which corresponds to a **row** in a table.

Example of a Relation

attributes
(or columns)

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

tuples
(or rows)

Terms-Formal Definitions

- The set of allowed values for each attribute is called the **domain** (域) of the attribute

- Given sets (domain) D_1, D_2, …. D_n , a **relation** r is a subset of **Cartesian product** (笛卡尔积)
  $D_1 \times D_2 \times \ldots \times D_n$

  $r = \{(a_1, a_2, \ldots, a_n)\} \subseteq D_1 \times D_2 \ldots\ldots \times D_n$

Terms-Formal Definitions

- Example of relation
  - customer-name = {Jones, Smith, Curry, Lindsay}
    customer-street = {Main, North, Park}
    customer-city = {Harrison, Rye, Pittsfield}

  - then r = { (Jones, Main, Harrison),
    (Smith, North, Rye),
    (Curry, North, Rye),
    (Lindsay, Park, Pittsfield)}

    is a relation over
    customer-name $\times$ customer-street $\times$ customer-city

Terms-Formal Definitions

- Note:
  - Relation *r* in Database field is the limited set (有限集合)
  - Attributes in tuples are non-ordered (无序性)
    - e.g. $(d_1, d_2, \ldots, d_n) = (d_2, d_1, \ldots, d_n)$
  - the order in which the tuples appear in a relations is irrelevant
  - several attributes may have the same domain

Terms-Formal Definitions

- Note:
  - A domain is **atomic** (原子的) if its elements are considered to be indivisible
  - all domains of R **should be atomic**, first formal norm
    - e.g. multivalued attribute values are not atomic, { {1,2}, {4}, {4,6,7} }
    - e.g. composite attribute address={city, street, zipcode} is not atomic(or atomic?)
  - **The important issue is not what the domain itself is, but rather how we use domain elements in our database.**

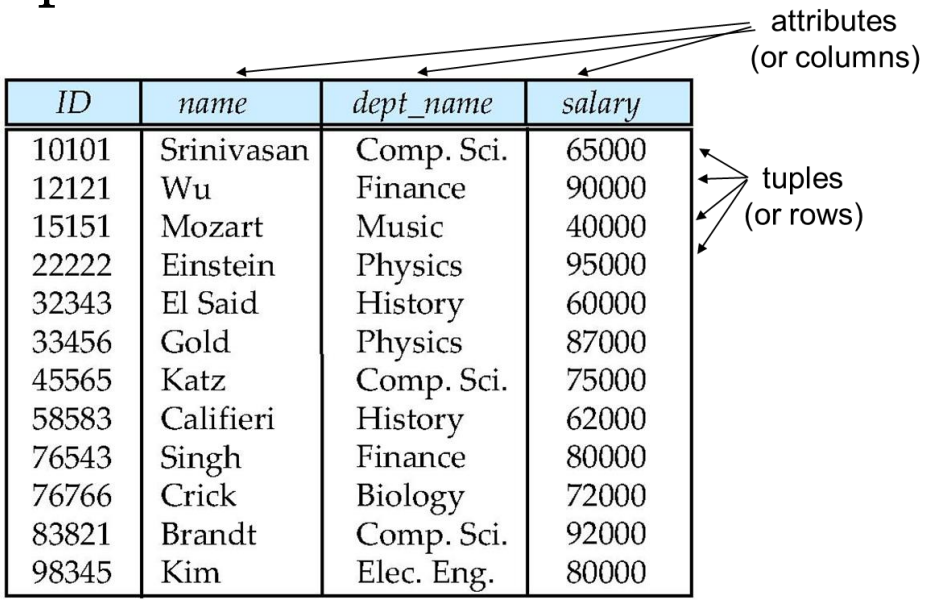
Terms-Formal Definitions

- Null
  - the special value *null* is a member of every domain.
  - the null value causes complications in the definition of many operations

Database Schema

Review the Definition of Relation

- Relation
  - Limited subset of Cartesian product
  - If relation is regarded as the counterpart of variable in the programming language, then, what is analogous to data type? And, what is the counterpart of variable's value?



ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Relation Schema

- The concept of a **relation schema** corresponds to the programming-language notion of type definition.

- In general, a relation schema consists of a list of attributes and their corresponding domains. For attributes A_1, A_2, …, A_n , $R = (A_1, A_2, \ldots, A_n)$ is a relation schema

  – e.g. Customer-schema =(customer-name, customer-street, customer-city)

  – orders of attributes is irrelevant

Relation Instance

- The concept of a relation instance corresponds to the programming-language notion of a value of a variable.

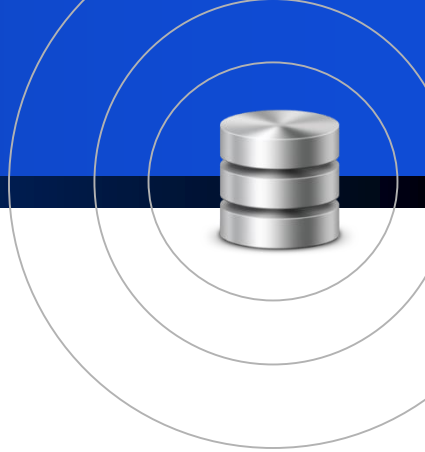
- **Relation instance**: the current values of a relation *r* at a particular time, where *r*=*r*(*R*) is a relation on the relation schema *R*.

- the contents of a relation instance ***(value of variable)*** may change with time, but the schema of a relation ***(data type of variable)*** does not generally change

Database Schema & Database Instance

- **Database schema**
  - logical design of the database

- **Database instance**
  - a snapshot of the data in the database at a given instant in time.



Keys

Preliminary Definitions

- Tabular representation of $r(R)$:
  - the current values of a relation $\boldsymbol{r(R)}$ (i.e. relation instance of the relation $\boldsymbol{r}$) are specified by a table
  - an element $\boldsymbol{t}$ in set r is a tuple, represented by a row in a table
  - tuple variable t, $\boldsymbol{t[A_i]}$ = the value of t on the attribute A_i , or $\boldsymbol{t[\{ A_i \}]}$ = the value set of t on the attribute set $\{ A_i\}$
    - e.g. $t_2[customer\text{-}name]$= 'Smith'

Preliminary Definitions

- Exercise: what is t, $t[A_i]$, and $t[\{A_i\}]$ respectively?

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Superkey

- A **superkey** (超码) is a set of one or more attributes that, taken collectively, can be used to identify uniquely a tuple in the relation

- **Def.** Given relation schema $R(A_1, A_2, \ldots, A_n)$, a subset K of R (i.e. $K \subseteq R$) is the *superkey* of R , if no two distinct tuples in relation $r(R)$ have the same values on all attributes in K, that is, if $t_1$and t_2 are in r and $t_1 \neq t_2$, then $t_1[K] \neq t_2[K]$ (or $t_1.K \neq t_2.K$)

Candidate Key

- A superkey may contain extraneous attributes. If *K* is a superkey, then so is any superset of *K*
- We are often interested in superkeys for which no proper subset is a superkey.
- Such ***minimal*** superkeys are called **candidate keys**(候选码)

Primary Key

- **Primary key** (主码) denotes a candidate key that is chosen by the database designer as the principal means of identifying tuples within a relation.

- Primary keys must be chosen with care. The primary key should be chosen such that its attribute values are never, or very rarely, changed

Key

- A key (whether primary, candidate, or super) is a property of the entire relation, rather than of the individual tuples.

- Any two individual tuples in the relation are prohibited from having the same value on the key attributes at the same time.

- The designation of a key represents a constraint in the real-world enterprise being modeled. (**entity integrity constraint**)

Prime Attribute & Non-Prime Attribute

- The constituent attributes of any candidate key are called **prime attributes**(主属性).

- Conversely, an attribute that does not occur in ANY candidate key is called a **non-prime attribute**(非主属性).

Foreign Key

- A relation, say r_1, may include among its attributes the primary key of another relation, say r_2. This attribute is called a **foreign key** (外码) from r_1, referencing r_2.

- The relation r_1 is also called the **referencing relation** of the foreign key dependency, and r_2 is called the **referenced relation** of the foreign key.

Referential Integrity Constraint

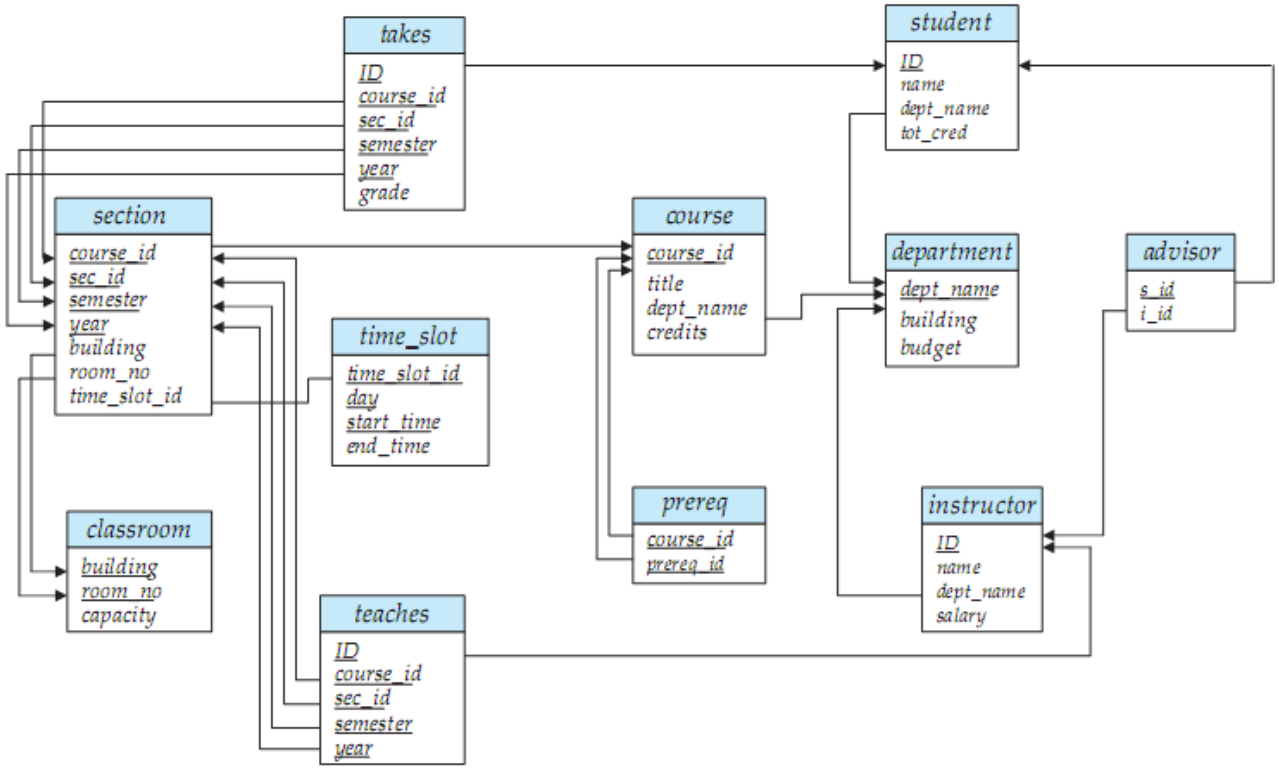
- a **referential integrity constraint** (参照完整性) requires that
  the values appearing in *specified attributes* of any tuple in the referencing relation
  also
  appear in *specified attributes* of at least one tuple in the referenced relation.

- Actually, referential integrity constraint is often used to restrict the value of foreign key: ***Value of the foreign key in referencing relation must appear in primary key of the referenced relation, null otherwise.***

Schema Diagrams

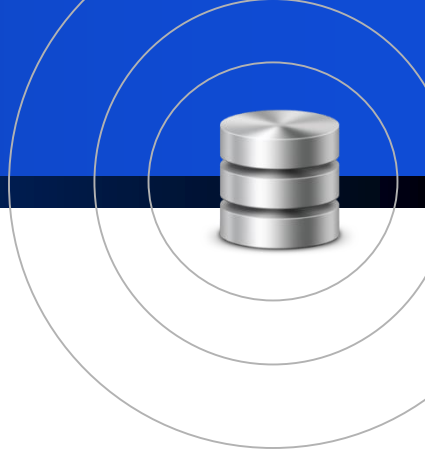
Schema Diagram

- A database schema, along with primary key and foreign key dependencies, can be depicted by **schema diagrams**



Other contents

- Relational Query Languages
  - Refer to chapter 3,4,6(SQL & relational algebra)
- Relational Operations
  - Refer to chapter 3,4,6(SQL & relational algebra)



Review

Review

- Content of Relational Model
  - Relational data structure
    - Table,relation,row,tuple
  - Integrity constraints
  - Operations on the model

- Terms(Formal Definitions) of Relational data structure
  - Domain→Cartesian product→Relation
  - null

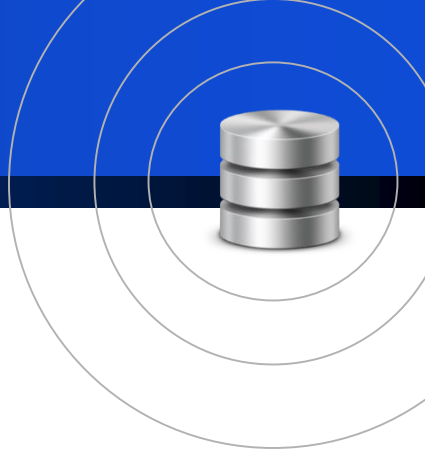
Review

- Database Schema
  - Relation → variable
  - Relation Schema → data type
  - Relation Instance → value

  - Database Schema and Database Instance

Review

- Keys
  - Superkey
  - Candidate key
  - Primary key
  - Prime attribute and non-prime attribute
  - Foreign key
  - Referential integrity constraint
  - Entity integrity constraint
- Schema Diagrams



Thanks