ICS2607: Information Management

Module 3: The Relational Database Model

Relational model

* Views data logically rather than physically

Table

* Structural and data independence
* Resembles a file conceptually
* Logical view of relational database that is based on relation
* Two-dimensional structure composed of rows and columns
* Contains a group of related entities

Note: Relational database models are easier to understand than hierarchical and network models.

Characteristics of a Relational Table

* A table is perceived as a two-dimensional structure composed of rows and columns
* Each table row (tuple) represents a single entity occurrence within the entity set.
* Each table column represents an attribute, and each column has a distinct name.
* Each intersection os a row and column represents a single data value
* All values in a column must conform to the same data format
* Each column has a specific range of values known as the attribute domain.
* The order of the rows and columns is immaterial to the DBMS
* Each table must have an attribute or combination of attributes that uniquely identifies each row.

Key

* Each row in a table must be uniquely identifiable
* One or more attributes that determines other attributes
* Determination and functional dependence
* Types of keys can represent
  + An unknown attribute value
  + A know, but missing, attribute value
  + A “not applicable” condition
* It can create problems when functions such as COUNT, AVERAGE, and SUM are used
* Can create logical problems when relational tables are linked
* Controlled Redundancy
  + Makes the relational database work
  + Tables within the database share common attributes
    - Enable tables to be linked together
  + Multiple occurrences of values not redundant when required to make the relationship work
  + Redundancy exists only when there is unnecessary duplication of attributes values.

Types of Keys

* Composite Key
  + Composed of more than one attribute
* Key Attribute
  + Any attribute that is part of a key
* Superkey
  + Any key that uniquely identifies each row
* Candidate Key
  + A superkey without unnecessary attributes
* Entity Integrity
  + Each row (entity instance) in the table has its own unique identity
* Nulls
  + No data entry
  + Not permitted in primary key
  + Should be avoided in other attributes
* Foreign Key
  + An attribute whose values match primary key values in the related table
* Referential Integrity
  + FK contains a value that refers to an existing valid tuple (row) in another relation.
* Secondary Key
  + Key used strictly for data retrieval purposes

Relational Database Keys

* Superkey
  + An attribute or combination of attributes that uniquely identifies each row in a table
* Candidate Key
  + A minimal (irreducible) superkey; a superkey that does not contain a subset of attributes that is itself a superkey
* Primary Key
  + A candidate key selected to uniquely identify all other attribute values in any given row; cannot contain null entries
* Foreign Key
  + An attribute or combination of attributes in one table whose values must either match the primary key in another table or be null
* Secondary Key
  + An attribute or combination of attributes used strictly for data retrieval purposes.

Integrity Rules

* Many RDBMs enforce integrity rules automatically \
* Safer to ensure that application design conforms to entity and referential integrity rules
* Designers use flags to avoid nulls
  + Flags indicate absence of some value

Types of Integrity Rules

* Entity Integrity
  + It requires all primary key entries to be unique and no part of the primary key may be null
  + Each row will have a unique identity, and foreign key values can properly reference primary key values.
* Referential integrity
  + It requires a foreign key to have either a null entry, as long as it is not a prt of its table;s primary key or an entry that matches the primary key value in a table to which it is related.
  + It is possible for an attribute to not have a corresponding value, but it will be impossible to have an invalid entry. The enforcement of the referential integrity rule makes it possible to delete a row in one table whose primary key has a mandatory matching foreign key values in another table.

Data Dictionary

* Provides detailed accounting of all tables found within the user/designer-created database
* Contains (at least) all the attribute names and characteristics for each table in the system.
* Contains metadata

System Catalog

* Contains metadata
* Detailed system data dictionary that describes all objects within the database.

Homonym

* Indicates the use of the same name to label different attributes

Synonym

* Opposite of a homonym
* Indicates the use of different names to describe the same attribute

Relationships within the Relational Database

* 1:M relationship
  + Indicates the use of the same name to label different attributes
  + Should be the norm in any relational database design
* 1:1 relationship
  + Indicates the use of the same name to label different attributes
  + One entity related to only one other entity and vice versa
  + Sometimes means that entity components were not defined properly
  + Could indicate the two entities that belong in the same table
  + Certain conditions absolutely require their use
* M:N relationships
  + Cannot be implemented as such in the relational model
  + M:N relationships can be changed into 1:M relationships
  + Implemented by breaking it up to produce a set of 1:M relationships
  + Avoid problems inherent to M:N relationship by creating a composite entity
    - Includes as foreign keys the primary keys of tables to be linked.
* 1:M relationships
  + Relational database norm
  + Found in any database environment

Data Redundancy Revisited

* Data redundancy leads to data anomalies
  + Can destroy the effectiveness of the database
* Foreign Keys
  + Control data redundancies by using common attributed shared by tables
  + Crucial to exercising data redundancy control
* Sometimes, data redundancy is necessary

Module 4: Entity Relationship Modeling

Entity Relationship Model

* ER model forms the basis of an ER diagram
* ERD represents the conceptual database as viewed by the end-user
* ERDs depict database’s main components
  + Entities
    - File systems or table as part of the database that we are developing
  + Attributes
    - Characteristics of entities
    - Field name/column names that introduce one to the file structure.
  + Relationships
    - relationship/interaction of one table to another

Entities

* Refers to **entity set** and not to single entity occurrence
  + Kung anong bumubuo sa dbms
* Corresponds to **table** and not to row in relational environment
* In Chen and Crow’s Foot models, entity is represented by rectangle with entity’s name
* The entity name, a noun, is written in capital letters.

Attributes

* Chen’s notation: attributes are represented by ovals connected to entity rectangle with a line
  + Each oval contains the name of attribute it represents
* Crow’s Foot notation: attributes written in attribute box below entity rectangle
* Required attribute: must have a value
* Optional attribute: may be left empty
* Domain: set of possible values for an attribute (main subject)
  + Attributes may share a domain
* Identifiers: one or more attributes that uniquely identify each entity instance (attribute leader)
* Composite identifier: primary key composed of more than one attribute (whatever new file system is created, the primary key is there; may sometimes bye considered a foreign key because it visits other keys)
* Composite attributes can be subdivided
* Simple attribute cannot be subdivided
* Single-value attribute can have only a single value
* Multivalued attributes can have many values
* M:N relationships and multivalued attributes should not be implemented
  + Create several new attributes for each of the original multivalued attributes’ components
  + Create new entity components of original multivalued attributes’ components
* Derived attribute: value may be calculated from other attributes

Advantages and Disadvantages of Storing Derived Attributes

|  | Stored | Not Stored |
| --- | --- | --- |
| Advantage | Saves CPU processing cycles  Saves data access time  Can be used to keep track of historical data | Saves storage space  Computation always yields current value |
| Disadvantage | Requires constant maintenance to ensure derived value is current, especially if any values used in the calculation change | Uses CPU processing cycles  Increases data access time  Adds coding complexity to queries |