

Appendix

Relational Databases and Database Design



Relational Theory


- A data management theory that prescribes:
 - data **structure**
 - data **integrity**
 - data **manipulation**
- Based on mathematical **set theory** & first formalized by E.F. Codd in 1970
- Many terms and concepts
 - relation = table = entity
 - tuple = row = record
 - attribute = column = field



Relations

- Table
 - a 2-dimensional structure of rows and columns
- Table Characteristics:
 - the columns in a relation are characteristics of an **entity**
 - each column has a unique attribute name
 - field values contain a single value (**atomic**)
 - column order is unimportant
 - row order is unimportant
 - each row is unique
 - entries in a column are from the same **domain**
 - what would be the domain for gender?
 - what would be the domain for hire date?
 - what would be the domain for an article's writerid?

3



Keys

- Primary key
 - an attribute (or collection of attributes) whose value uniquely identifies each row in a relation
 - primary key values must be **unique**, good if **minimal** and **static**
- Candidate key
- Alternate key
- Foreign key
 - an attribute (or collection of attributes) in one relation whose values must match the values for the primary key of another relation
- Composite (concatenated) key
- Nonkey attribute
 - an attribute that is not part of the relation's primary key

4



1:Many Relationships

- Each record in Table A can have **many** (zero or more) matching records in Table B but each record in Table B has only **one** matching record in Table A
 - e.g.: Each Customer can have many (zero or more) Orders but each Order is for one Customer
 - e.g.: Each Department can have many Employees but each Employee works in only one Department
- Implementation
 - have the primary key of the **one** table appear as a foreign key in the **many** table

5



1:1 Relationships

- Each row in one table has at most one matching row in the other table
- Entity subtype
 - a special type of 1:1 relationship
 - a relation whose primary key is a foreign key to a second relation & whose attributes are additional attributes for the second relation
- Commonly used to:
 - avoid **nulls**
 - control field access
 - overcome Access' 255 fields/table limit
- Can use a query to reunite the fields even though they are stored in separate tables

6



Many:Many Relationships

- Each record in Table A can have **many** matching records in table B and each record in Table B can have **many** matching records in Table A.
 - e.g.: each **Student** can enroll in many **Sections** and each **Section** can enroll many **Students**
 - e.g.: each **Pitcher** can pitch in many **Games** and each **Game** can use many **Pitchers**
- Implementation:
 - create a third table (**junction table**, intersection table, composite entity) which includes the primary keys from table A and table B as foreign keys
 - consider using them as a composite primary key in the new table
 - this breaks the Many:Many relationship into two separate 1:Many relationships

7



Tools used to Describe Relations and Relationships

- Shorthand Notation
 - used to describe a relation's name, attributes, primary key & foreign key(s)

```
Issue25

Writer(WriterID, LastName, FirstName, Phone, LastContact, Freelancer, ReprintAmount, Contact, Salutation,
Gender, HomePage)
Foreign key: Contact to Writer relation

Payment(TransactionNumber, Amount, DateIssued, Purpose, TaxDeductible, AuthorizedBy, PayeeWriterID)
Foreign key: PayeeWriterID to Writer relation

Article(ArticleNum, Title, Type, Issue, Length, WriterID)
Foreign key: WriterID to Writer relation
Type to Type relation

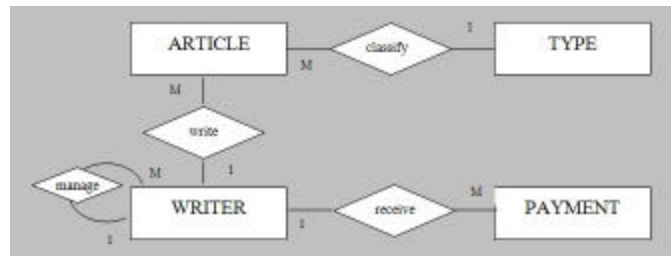
Type(Type, Description)
```

8

Tools used to Describe Relations and Relationships

- Entity-Relationship Diagrams

- graphically shows a database's entities and the relationships among the entities
- rectangles
- diamonds
- diamond in rectangle
- double rectangle



9

Integrity Constraints

- Rules that prevent contaminated data from appearing in the database
- A database has **integrity** when it is both correct and complete

10



Domain Integrity

- An attribute's values must come from its **domain**
- The set of legitimate values for a column
- Examples that violate domain integrity:
 - a negative weight
 - a gender code other than M or F
 - a nonnumeric quantity
 - violations of specific business rules (5-digit PartNumber)

11



Entity Integrity

- No part of the primary key is allowed to be null
 - must have a value
 - no duplicate values
 - Access creates/uses an Index to enforce uniqueness
- Access automatically enforces for each table's primary key

12



Referential Integrity

- Each **foreign key** value must match a **primary key** value in the related table
 - when a row in one table references a row in another table, the referenced row must actually exist
 - each Article's WriterID must match the WriterID of an existing Writer
- Prevents **orphan** records in a related table
- Enforcing Referential Integrity with Access
 - double-click the join line in Relationships window, dialog box
 - if business rules don't allow null foreign keys, set the foreign key's Validation Rule to IS NOT NULL so a user can't leave the foreign key null and gets a helpful error message
- Revisit the Tutorial 2 slides

13

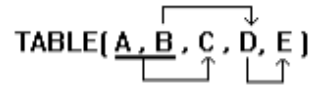


Dependencies and Determinants

- Functional dependency
 - general:
 - $X \rightarrow Y$ "attribute X functionally determines attribute Y"
- Examples:
 - ArticleNum \rightarrow Title ArticleNum \rightarrow Length
 - WriterID \rightarrow LastName WriterID \rightarrow Phone
- Whenever 2 of a relation's rows have the same X value, they also have the same Y value
- Determinant (X)
 - an attribute (or collection of attributes) which determines the value of another
- Dependent (Y)
 - an attribute whose value is determined by another attribute

14

Dependencies and Determinants



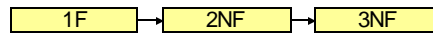
- Good dependency (nib)
 - all attributes of a relation must be functionally dependent only on the relation's primary (and any candidate) keys
 - if a determinant is not the primary/candidate key, the relation will suffer redundancy
- Partial dependency
 - exists when an attribute is dependent on only part of the primary key, instead of the entire primary key
 - can occur only when the relation has a composite primary key
- Transitive dependency
 - a functional dependency between two nonkey attributes
 - exists when an attribute is dependent on another attribute that is neither the primary key nor a candidate key

15

Anomalies

- Problems caused by a flawed design
 - data redundancy created by partial & transitive dependencies
- 3 types:
 - Insertion anomaly
 - when you cannot add a new row because you don't know its entire primary key value (entity integrity)
 - Deletion anomaly
 - when you delete a row from a relation and unintentionally lose other data
 - Update anomaly
 - when you change one field value and either the DBMS must make more than one change to the database or else the database is forced to contain inconsistent data

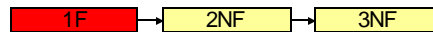
16



Normalization

- Database Design
 - process of determining the relations needed for a given collection of attributes and placing attributes into the correct relations
- Relies upon:
 - understanding the **functional dependencies** of each attribute
 - recognizing **anomalies** caused by data redundancy, partial dependencies and transitive dependencies
- Normalization Process
 - start with an initial set of relations, then apply rules to eliminate anomalies and produce a final set of problem-free relations
 - goal: a relation's attributes should be functionally dependent on the key, the whole key, and nothing but the key...so help me Codd!

17



1NF (First Normal Form)

- A relation that does not contain any **repeating groups**
 - each field must be **atomic** (contain a single item)
- To fix a repeating group:
 - expand the primary key to include the primary key of the repeating group, forming a composite key
- Each repeating group must be processed separately

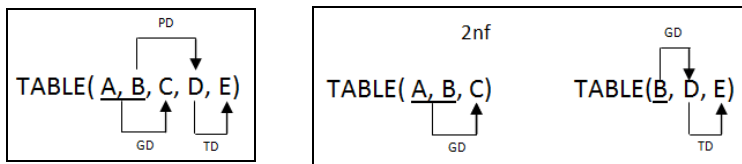
18



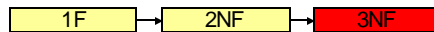
2NF (Second Normal Form)

- A relation in 1NF that contains no **partial** dependencies
 - no non-key attribute is dependent on only part of the primary key
 - a 1NF relation is automatically in 2NF if its primary key is not composite

- To fix partial dependencies:
 - identify the functional dependencies for each attribute
 - create new relations and place each attribute in a relation so the attribute is functionally dependent on the entire primary key

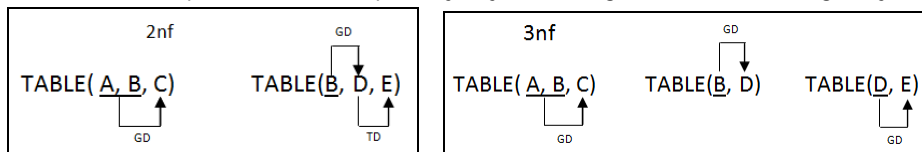


19



3NF (Third Normal Form)

- A relation in 2NF in which every determinant is also a candidate key
 - contains no **transitive** dependencies
- Each nonkey field must be **mutually independent** and dependent only on candidate keys
- To fix transitive dependencies:
 - remove attributes that depend on a non-candidate-key and place them in a new relation with the determinant as the primary key
 - keep the new table's primary key in the original table as a foreign key



20



Design Process

- Preliminaries
 1. Read the case and identify obvious **entities**
 2. Begin an ER diagram
 3. Re-read the case looking for Many-to-Many relationships
 4. Tentatively place each field in a table using shorthand notation
- Normalization
 5. Obtain First Normal Form by eliminating **repeating groups**
 6. Obtain Second Normal Form by eliminating **partial dependencies**
 7. Obtain Third Normal Form by eliminating **transitive dependencies**
- Documentation
 8. Complete shorthand notation
 9. ER Diagram