

Relational Database Design

Overview: Design Process

- A process of determining the tables needed for a given collection of fields and placing each field in the correct table
- Steps
 - Identify all required columns
 - from source documents, reports, ideas from current users
 - Identify entities
 - generally, each entity is placed in a separate table
 - Consider relationships between entities
 - Normalization process
 - Produce an ER diagram
 - Use SQL to create the tables & inter-table relationships (chapter 12)

2

1:Many Relationships

- Each row in Table A can have **many** (zero or more) matching row in Table B but each record in Table B has at most **one** matching row in Table A
- Examples
 - each Course can have many Sections but each Section covers a single Course
 - each ZipCode can have many Students but each Student lives in one ZipCode
 - each Department can have many Employees but each Employee is assigned to one Department
- Implementation
 - include the PK of the one table as a FK in the many table

3

1:1 Relationships

- Each row in one table has **at most** one matching row in the other table
- A relationship between the primary keys of two tables
- Commonly used to
 - control field access
 - avoid nulls

4

Many:Many Relationships

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

5

Functional Dependency

- **$X \rightarrow Y$**
 - "the value in column X functionally determines the value for column Y"
- Examples:
 - ArticleNum \rightarrow Title ArticleNum \rightarrow Length
 - ZipCode \rightarrow City ZipCode \rightarrow State
 - ??????? \rightarrow Final_Grade
- Whenever 2 of a table's rows have the same X value, they will also have the same Y value
 - i.e., data redundancy will occur
- Determinant (X)
 - a column (or collection of columns) which determines value of another
- Dependent (Y)
 - a column whose value is determined by another column

6

Functional Dependency

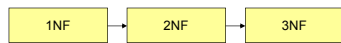
TABLE{ A, B, C, D, E }

- Good Dependency
 - when a column is dependent only on the table's primary key (and on any candidate keys)
- Partial Dependency
 - when a column is dependent on only part of the table's primary key
 - can only occur when a table has a composite primary key
 - if allowed to remain, will cause data redundancy
- Transitive Dependency
 - when one column is dependent on another column that is not the table's primary key
 - if allowed to remain, will cause data redundancy

7

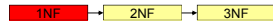
Normalization Process

- Start with an initial set of tables, then apply rules to produce a final set of problem-free tables
- A process that eliminates data redundancy
- Relies on identifying functional dependencies of each column



8

First Normal Form



- A table that does not contain **repeating groups**
 - each field must be **atomic** (contain a single item)
- Fix repeating groups by
 - removing each repeating group column and place them in a table that includes the original table's primary key
- Figure 1.8, pg 14
 - this design can handle up to 3 locations per book
 - what if a book exists in 1 location?
 - what if a book exists in 4 locations?
 - what if a book exists in 400 locations?
 - a table design change is required and potential for waste caused by Nulls
- Figure 1.9, pg 14
 - what if a book exists in 1 location?
 - what if a book exists in 4 locations?
 - what if a book exists in 400 locations?
 - this design can handle any number of locations per book without a design change and without space waste caused by Nulls

9

1NF → 2NF → 3NF

Second Normal Form

- A table in 1NF where every nonkey column is dependent on the entire primary key, not part of it
 - no **partial dependencies**
 - a table in 1NF is automatically in 2NF if its primary key is not a composite
- Fix partial dependencies by
 - identify the functional dependencies for each column
 - place each column in a table where it is functionally dependent on the entire primary key
- Figure 1.10, pg 15
 - what is PHONE_NUMBER dependent on?
 - why is PHONE_NUMBER a problem in this table?

10

1F → 2NF → 3NF

Third Normal Form

- A table in 2NF in which every determinant is also a candidate key
 - no **transitive dependencies**
- Each non-key column must be mutually independent and be dependent only on candidate keys
- Fix transitive dependencies by
 - removing any column that depends on a non-candidate-key
 - place it in a table with the determinant as the primary key and keeping the new table's primary key in the original table as a foreign key
- Figure 1.11, pg 15
 - what is PUBLISHER_PHONE_NUMBER dependent on?

11

Design Process

- Preliminaries
 - Read the case and identify obvious **entities**
 - Begin an ER diagram
 - Re-read the case looking for Many-to-Many relationships
 - Tentatively place each field in a table on the ER diagram
- Normalization
 - Obtain First Normal Form by eliminating **repeating groups**
 - Obtain Second Normal Form by eliminating **partial dependencies**
 - Obtain Third Normal Form by eliminating **transitive dependencies**
- Documentation
 - Produce an ER Diagram

12
