Database Design for the Uninitiated

Paul Litwin
FHCRC Collaborative Data Services

1

CDS Brownbag Series

- This is the ninth in a series of seminars
- Materials for the series can be downloaded from www.deeptraining.com/fhcrc
- Sign up to be on our email list
- Upcoming
  - SQL Server for Administrators
    - March 7 at 12:30 in M1-A303
  - Developing SQL Server Stored Procedures
    - April 4 at 12:30 in TBD

2

CDS

- Collaborative Data Services
  - Providing "data services" through FHCRC (and beyond)
  - Our services include...
    - Telephone interviewing of subjects
    - Data entry & scanning
    - Programming
    - Web and database hosting
- More info at http://cds.fhcrc.org

3

Seminar Materials

- Can be downloaded from
  - www.deeptraining.com/fhcrc
- Include
  - This slide presentation
  - A 21 page white paper on database design

4

Database Design

- Why Bother With Design?
- Relational Model Basics
- Normalization
- Practical Recommendations

5

Database Design

- Why Bother With Design?
- Relational Model Basics
- Normalization
- Practical Recommendations

6

Two Basic Approaches to Design

- Jump in and design as you go
- Design the database on paper first and then carefully apply the principles of relational database design

You almost always spend more time using the first approach!

7

Benefits of Good Design

- You'll be able to easily input and update data
- You'll be able to easily query, summarize and create reports
- It will be easy to modify the design
- The database will be easy to document and maintain

8

What's Involved?

- Making decisions regarding how best to take some system in the real world and *model* it in a database
- Deciding which tables to create, what fields they will contain, as well as the relationships between the tables

9

To Design a Database Properly, You Need...

- Knowledge of the relational model
- Knowledge of your domain

Domain knowledge is essential to good database design!

10

Database Design

- Why bother With Design?
- Relational Model Basics
- Normalization
- Practical Recommendations

11

Origins of the Relational Model

- Created by Dr. E.F. Codd in 1969 while he was at IBM
- Based on Set Theory and Predicate Logic
- So What?

12

Tables

- Tables in the relational model are used to represent "things" in the real world
- Each table should represent only one thing
- These things (or entities) can be real-world *objects* or *events*

13

Tables

- Tables have rows and columns
- Also referred to as records and fields
- (Codd referred to tables as relations, rows as tuples, and columns as attributes)

14

Table Examples

- Objects
  - Customers
  - Patients
  - Specimens
  - Principal Investigators
- Events
  - Blood Draw
  - Follow-up Visit
  - Telephone Call

15

Uniqueness and Keys

- Tables shouldn't have duplicate rows
- You guarantee uniqueness for a table by designating a primary key – one or more columns that contain unique values for a table
- Each table can have only one primary key (PK)

16

Choosing a Primary Key— Simple vs. Composite Keys

- Simple key is made up of one column
  - E.g., for the Patient table
    - RegNo
- Composite key is made up of more than one column
  - E.g., for the Specimen table
    - RegNo
    - SpecimenNo

17

Choosing a Primary Key— Candidate & Alternate Keys

- Candidate key is a set of columns that could be the primary key
- Alternate key is a candidate key that was not chosen as the primary key

18

Pascal's Rules for Deciding on the Best Primary key

- Minimality – less columns are better
- Stability – columns that don't change are better
- Simplicity – keep it simple
- Familiarity – keep it familiar to the user

19

Choosing a Primary Key

- If no existing field makes a suitable PK, you might consider a surrogate key
- AutoNumber/Identity fields make good surrogate keys
- Best PK will *often* have a data type of whole number *(Integer or Long Integer)*

20

Example Choosing the Best Primary Key

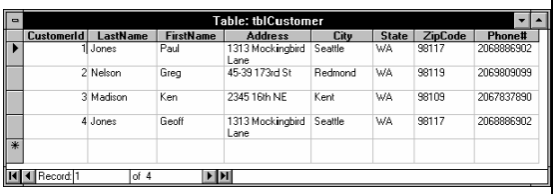


Table: tblCustomer

CustomerId	LastName	FirstName	Address	City	State	ZipCode	Phone#
1	Jones	Paul	1313 Mockingbird Lane	Seattle	WA	98117	2068886902
2	Nelson	Greg	45-39 173rd St	Redmond	WA	98119	2069809099
3	Madison	Ken	2345 16th NE	Kent	WA	98109	2067837890
4	Jones	Geoff	1313 Mockingbird Lane	Seattle	WA	98117	2068886902

Record: 1 of 4

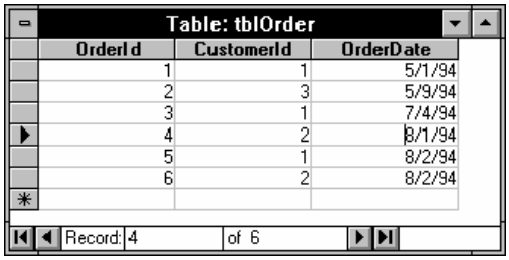
21

Foreign Keys

- While primary keys (PK) are important for maintaining a single table's integrity, they are *essential* when you create relationships joining together multiple tables
- Foreign key (FK): a column in a table used to reference a PK in another table

22

Example Foreign Key



23

Relationships

- Relationships in a relational database are considered as links between pairs of tables:
  - One-to-one
  - One-to-many
  - Many-to-many

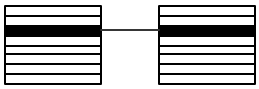
24

One-to-One

1 1

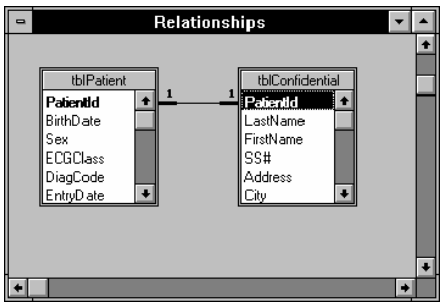
One record in the first table is related to only one record in the second table

One record in the second table is related to only one record in the first table



25

Example
One to One Relationship



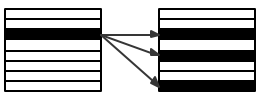
26

One-to-Many

1 Many

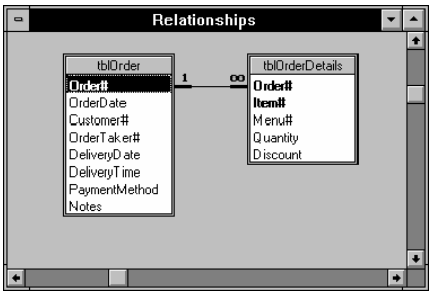
One record in the first table is related to many records in the second table

One record in the second table is related to *only one* record in the first table



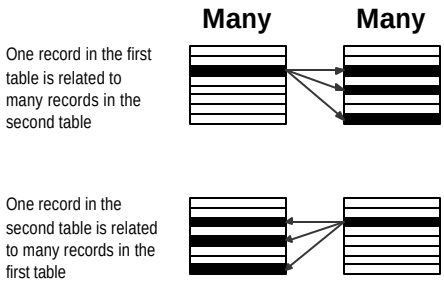
27

Example
One-to-Many Relationship



28

Many-to-Many



29

Many-to-Many (1 of 2)

- A many-to-many relationship is a sign that you are missing intermediary (junction) table
- Replace many-to-many relationships with two one-to-many relationships between each of the original tables and the junction table

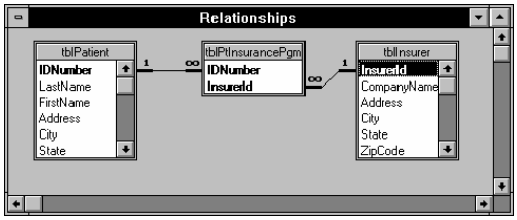
30

Many-to-Many (2 of 2)

- Junction tables may occur naturally
  - For example, the relationship between Orders and Products is M->M
  - What this tells you is that you are not considering the OrderDetails table
- Often you will need to create a junction table that does not already exist
  - It will need to include the PKs of both original tables

31

Example
Many to Many Relationship



32

Referential Integrity

- Foreign keys are more than concepts
- You must create them within your database to ensure *referential integrity*
- RI ensures that when you create a row containing a foreign key reference, that it always points to a valid primary key record in the referenced table
- RI is one type of a *constraint*
  - Constraints can also be used to enforce domain integrity (e.g., that sex can only be "M" or "F", PtId is greater than 0, etc.)

33

Database Design

- Why bother With Design?
- Relational Model Basics
- Normalization
- Practical Recommendations

34

Normalization

- Normalization is the process of simplifying the design of a database so that it achieves optimum structure
- Normal forms are a linear progression of rules you apply to a database, with each higher normal form achieving a better, more efficient design

35

Before We Begin

- No duplicate rows are allowed (i.e., there must be PK for each table)

36

First Normal Form

- First Normal Form (1NF) says that all column values must be atomic
- One column, one value
- Also, no repeating groups

37

Is this Table in 1NF?

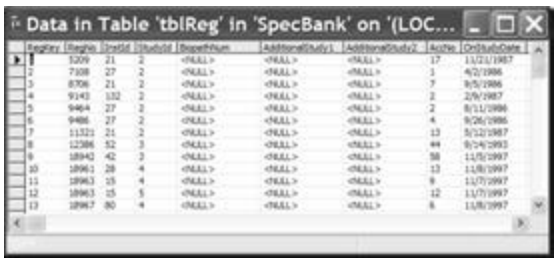
Table: tblOrder1

OrderId	CustomerId	Items
1	4	5 hammer, 3 screwdriver, 6 monkey wrench
2	23	1 hammer
3	15	2 deluxe garden hose, 2 economy nozzle
4	2	15 10' 2x4 untreated pine board
5	23	1 screwdriver
6	2	5 key
*		

Record: 1 of 6

38

How about this one?



39

Order Table in 1NF

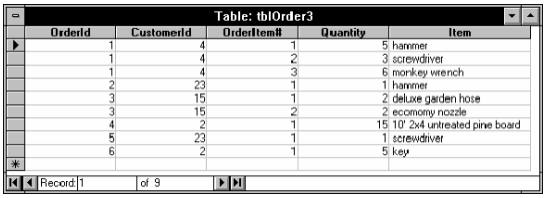


Table: tblOrder3

OrderId	CustomerId	OrderItem#	Quantity	Item
1	4	1	5	hammer
1	4	2	3	screwdriver
1	4	3	6	monkey wrench
2	23	1	1	hammer
3	15	1	2	deluxe garden hose
3	15	2	2	ecomomy nozzle
4	2	1	15	10' 2x4 untreated pine board
5	23	1	1	screwdriver
6	2	1	5	key

Record: 1 of 9

40

Second Normal Form

- A table is said to be in Second Normal Form (2NF), if it is in 1NF and every non-key column is fully dependent on the (entire) primary key
- "The key, the whole key and nothing but the key so help me Codd."
- Tables should only store data relating to one "thing" (or entity) that's described by the PK

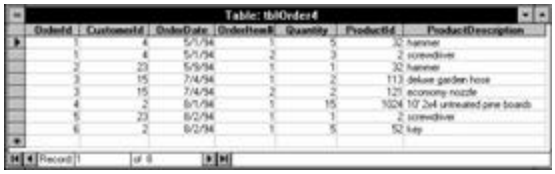
41

Second Normal Form

- 2NF achieved by breaking tables into normalized parts that describe a single entity
- This is called (non-loss) decomposition

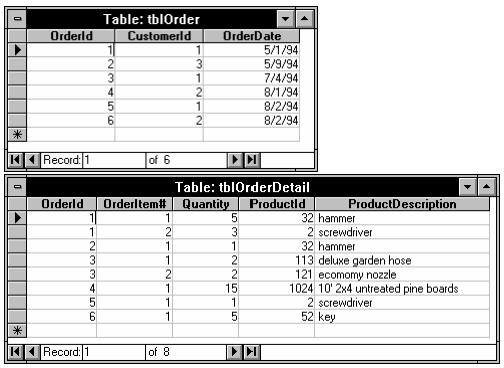
42

Is this Table in 2NF?



43

Now in 2NF



Third Normal Form

- A table is said to be in Third Normal Form (3NF), if it is in 2NF and if all non-key columns are mutually independent
- To achieve 3NF, you need to break out lookup tables and eliminate calculated fields

45

Is this Table in 3NF?

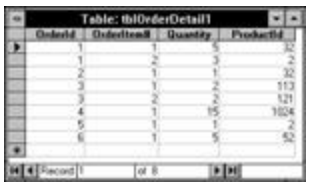
Table: tblOrderDetail

OrderId	OrderItem#	Quantity	ProductId	ProductDescription
1	1	5	32	hammer
1	2	3	2	screwdriver
2	1	1	32	hammer
3	1	2	113	deluxe garden hose
3	2	2	121	ecomomy nozzle
4	1	15	1024	10' 2x4 untreated pine boards
5	1	1	2	screwdriver
6	1	5	52	key

Record: 1 of 8

46

Now in 3NF



47

Review: Getting Normal

- 1NF: One column, one value. Remove repeating groups
- 2NF: Break into tables that describe one entity.
- 3NF: Remove calculations and break out lookup tables

48

When to Denormalize

- Performance
  - Prove it!
- Because the users of the database demand it
  - You may need to educate them!
- Example: storing a calculation in a table
  - Calculation is stored in table (violating 3NF rules) to improve reporting speed

49

How to Denormalize

- Start with a normalized design
- Do so deliberately!
- Have a good reason for denormalizing
- Be fully aware of the tradeoffs this decision entails
- Thoroughly document this decision
- Create the necessary application adjustments to avoid anomalies

50

Database Design

- Why bother With Design?
- Relational Model Basics
- Normalization
- Practical Recommendations

51

Naming Recommendations General Rule

- DON'T USE ALL CAPS FOR TABLE AND FIELD NAMES!!!
- Isn't the above annoying?

52

Table Names

- Spaces are a Bad Idea
- Can be plural (tblCustomers) or singular (tblCustomer)
  - Choose one and be consistent
- Many people prefix table names with tbl, e.g. tblCustomer and view names with either qry or vw
- Recommendation: use singular names and the tbl prefix and use camel case instead of underscores

53

Field Names (1 of 2)

- Some people love to give a table prefix to all field names
  - E.g., for tblCustomers you have CS_Id, CS_FirstName, CS_LastName, CS_Address, etc.
- Others prefer normal proper-cased field names.
  - E.g., CustomerId, FirstName, LastName, Address, etc.

54

Example Using Table Prefix of Field Names (and ALL CAPS)

55

Field Names (2 of 2)

- If you use the prefix technique then foreign keys will have a different name than the primary keys
  - In general, this is a bad idea
- Recommendations:
  - Don't use table prefixes for fields
  - Name fields using ProperCase names w/o underscores
  - Name foreign keys using the exact same names as the primary keys to which they point

56

Relationships & Indexes

- Recommendations
  - All tables should have primary keys
  - Relationships should be created and enforced at the database level
    - Foreign-key based relationships will enforce referential integrity *and* improve performance of joins
  - Indexes should be created to improve performance of queries

57

Optimizing Indexes

- Recommendations
  - Create indexes on all fields used in query where clauses
  - If you are using Access,
    - Turn off the auto-index feature
    - Use performance analyzer to make index suggestions
  - If using SQL Server
    - Run Index Tuning Wizard

58

Summary

- Good database design is critical
- Tables should represent one "thing" and contain primary keys
- Create relationships
- Databases should be normalized to third normal form
- Created indexes on fields used in query where clauses

59

Further Reading

- My white paper (see next slide)
- There are tons of books on database design
  - One book that is written more for the novice is
    Database Design for Mere Mortals
    By Michael J Hernandez

60

Thank You!

- Materials can be downloaded from
  - www.deeptraining.com/fhcrc
- Include
  - This slide presentation
  - A white paper on database design

61