Chapter Three

The Relational Model and Normalization

Chapter Objectives

1. To be able to identify possible insertion, deletion, and update **anomalies** in a relation
2. To understand basic relational terminology
3. To understand the characteristics of ***relations***
4. To understand alternative terminology used in describing *the* ***relational model***
5. To be able to identify ***functional dependencies***, ***determinants***, and dependent attributes
6. To be able to identify ***multivalued dependencies***

Chapter Objectives

1. To identify primary, candidate, and composite ***keys***
2. To be able to place a relation in fourth normal form
3. To be able to place a relation into ***BCNF normal form***

Main Ideas

- **Database design theory** that guides designers in designing a '*normal*' database schema which
  - Stores the required domain data (*requirements analysis*): business rules (constraints)
  - Guarantees certain desirable *performance*.
  - Express *constraints* (*dependencies*) among discrete data elements;
  - Use dependencies to *decompose* the data into *relations*.

- The process of converting a schema to a normal form is referred to as **normalization**.

Chapter Keywords

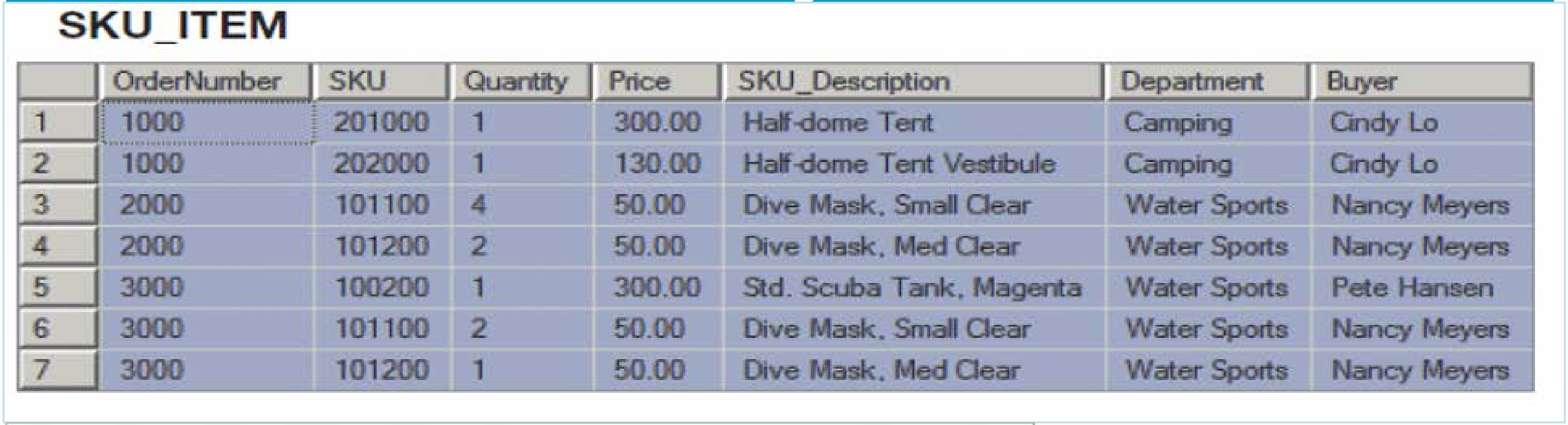
- Purpose: Minimum data redundancy
- Theory: Functional Dependency
- Main tool: Normalization
- Certification: Normal Forms
- Technique: Decompositions

Chapter Premise

- We have received one or more tables (or spreadsheets) of existing data.
- The data is to be stored in a new database.

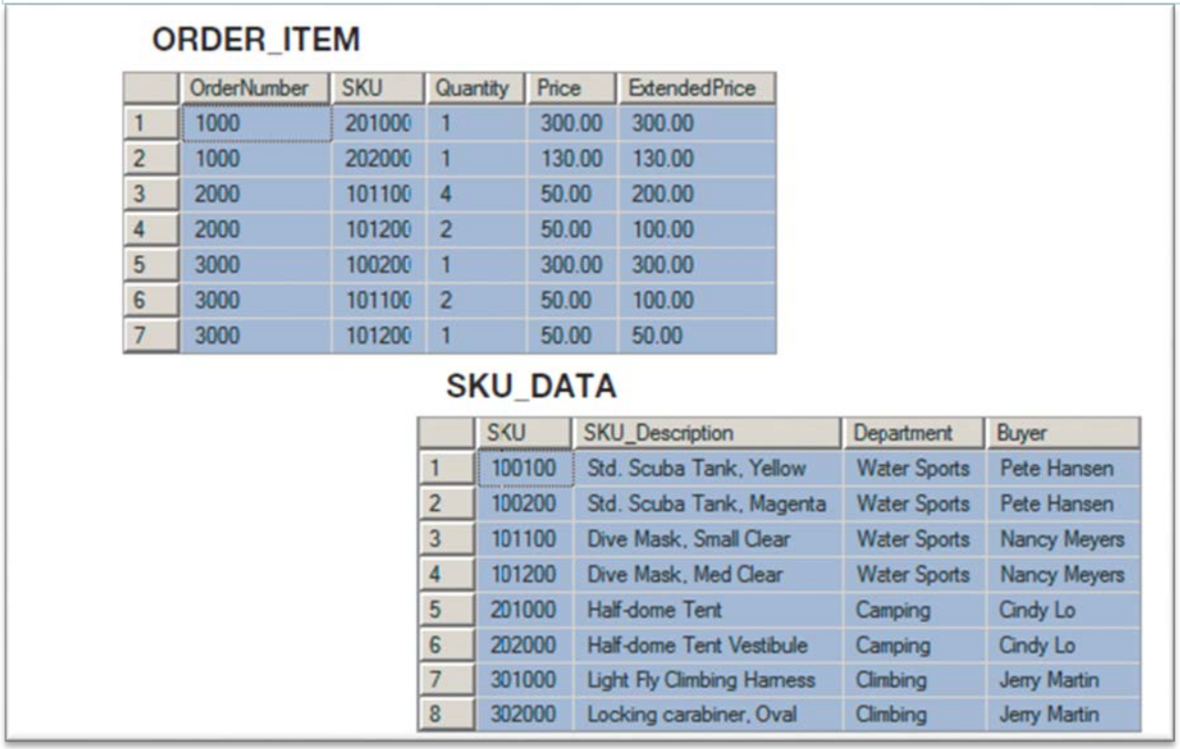
- Evaluate and improve Relation Models for design quality
- <u>QUESTION</u>: *Should the data be stored as received, or should it be transformed for storage?*

How many tables?



SKU_ITEM

	OrderNumber	SKU	Quantity	Price	SKU_Description	Department	Buyer
1	1000	201000	1	300.00	Half-dome Tent	Camping	Cindy Lo
2	1000	202000	1	130.00	Half-dome Tent Vestibule	Camping	Cindy Lo
3	2000	101100	4	50.00	Dive Mask, Small Clear	Water Sports	Nancy Meyers
4	2000	101200	2	50.00	Dive Mask, Med Clear	Water Sports	Nancy Meyers
5	3000	100200	1	300.00	Std. Scuba Tank, Magenta	Water Sports	Pete Hansen
6	3000	101100	2	50.00	Dive Mask, Small Clear	Water Sports	Nancy Meyers
7	3000	101200	1	50.00	Dive Mask, Med Clear	Water Sports	Nancy Meyers



ORDER_ITEM

	OrderNumber	SKU	Quantity	Price	ExtendedPrice
1	1000	201000	1	300.00	300.00
2	1000	202000	1	130.00	130.00
3	2000	101100	4	50.00	200.00
4	2000	101200	2	50.00	100.00
5	3000	100200	1	300.00	300.00
6	3000	101100	2	50.00	100.00
7	3000	101200	1	50.00	50.00

SKU_DATA

	SKU	SKU_Description	Department	Buyer
1	100100	Std. Scuba Tank, Yellow	Water Sports	Pete Hansen
2	100200	Std. Scuba Tank, Magenta	Water Sports	Pete Hansen
3	101100	Dive Mask, Small Clear	Water Sports	Nancy Meyers
4	101200	Dive Mask, Med Clear	Water Sports	Nancy Meyers
5	201000	Half-dome Tent	Camping	Cindy Lo
6	202000	Half-dome Tent Vestibule	Camping	Cindy Lo
7	301000	Light Fly Climbing Harness	Climbing	Jerry Martin
8	302000	Locking carabiner, Oval	Climbing	Jerry Martin

We normalize a relation with the purpose of removing data inconsistency occurring in common database operations.

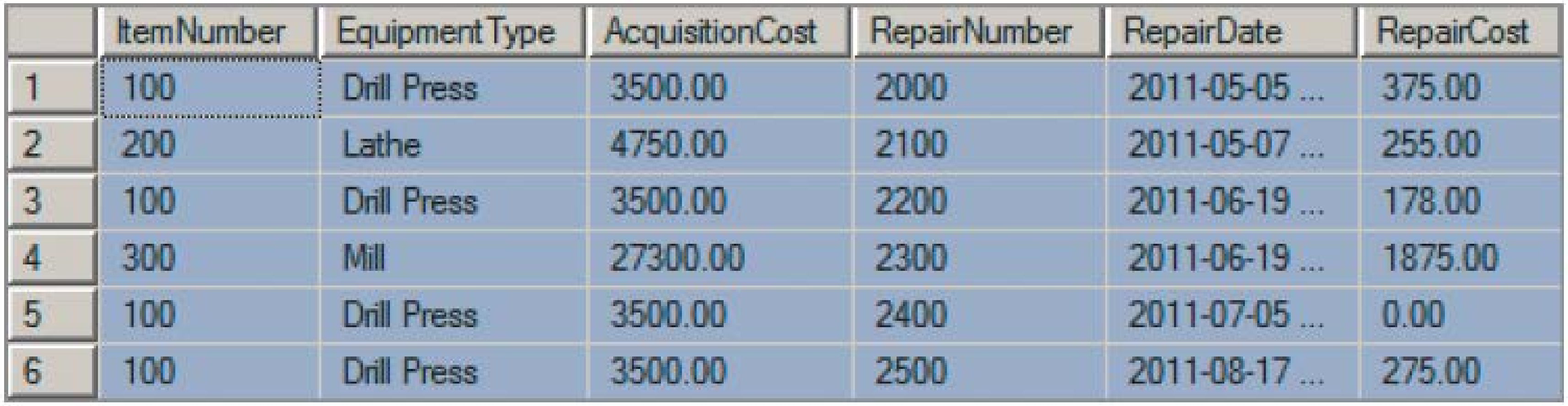
1. MODIFICATION ANOMALIES

Modification Anomalies

- Deletion anomaly
- Insertion anomaly
- Update anomaly

Deletion Anomaly

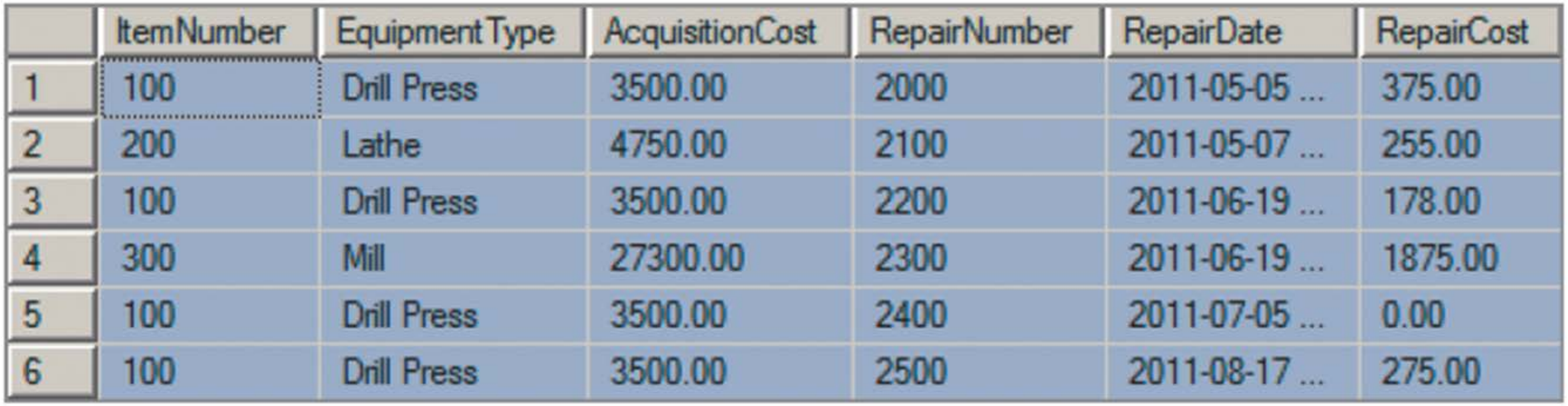
Equipment_Repair



	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2011-05-05 ...	375.00
2	200	Lathe	4750.00	2100	2011-05-07 ...	255.00
3	100	Drill Press	3500.00	2200	2011-06-19 ...	178.00
4	300	Mill	27300.00	2300	2011-06-19 ...	1875.00
5	100	Drill Press	3500.00	2400	2011-07-05 ...	0.00
6	100	Drill Press	3500.00	2500	2011-08-17 ...	275.00

- Delete the row for repair number 2100
- Delete the row for repair number 2300

Insertion Anomaly



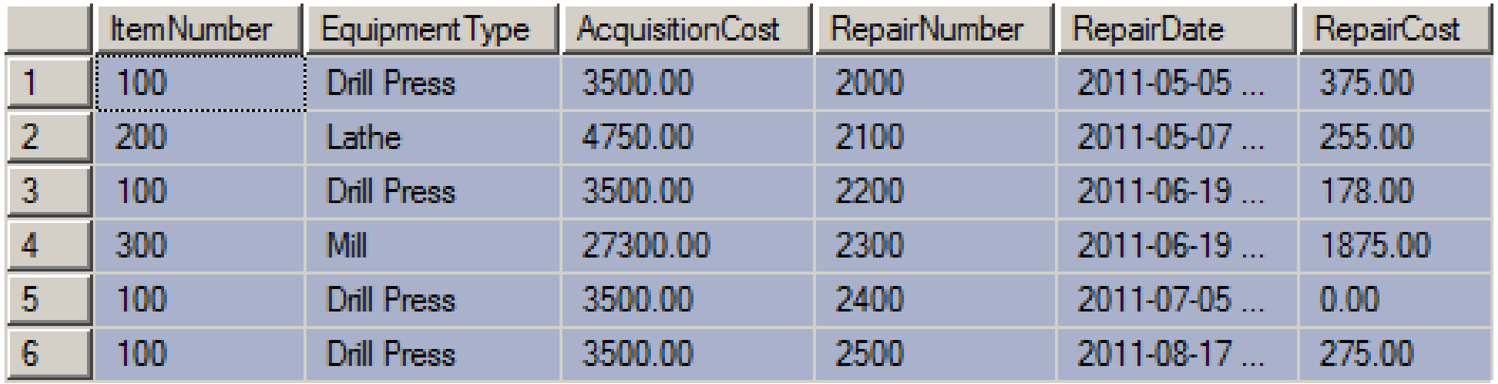
	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2011-05-05 ...	375.00
2	200	Lathe	4750.00	2100	2011-05-07 ...	255.00
3	100	Drill Press	3500.00	2200	2011-06-19 ...	178.00
4	300	Mill	27300.00	2300	2011-06-19 ...	1875.00
5	100	Drill Press	3500.00	2400	2011-07-05 ...	0.00
6	100	Drill Press	3500.00	2500	2011-08-17 ...	275.00

- Insert a row for the first repair of a piece of equipment…

Update Anomaly

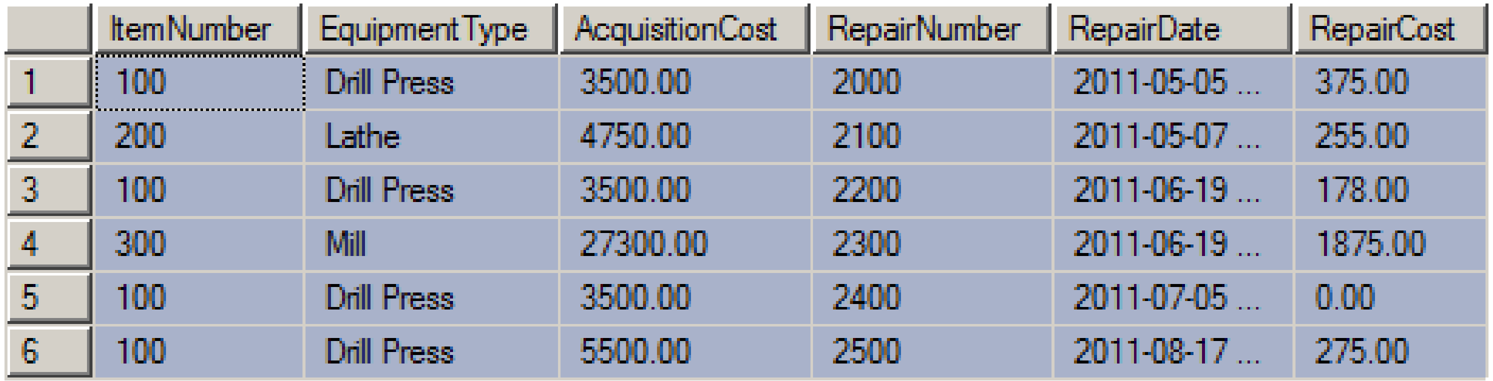
Update the *acquisition cost* for *Drill Press*

Before



	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2011-05-05 ...	375.00
2	200	Lathe	4750.00	2100	2011-05-07 ...	255.00
3	100	Drill Press	3500.00	2200	2011-06-19 ...	178.00
4	300	Mill	27300.00	2300	2011-06-19 ...	1875.00
5	100	Drill Press	3500.00	2400	2011-07-05 ...	0.00
6	100	Drill Press	3500.00	2500	2011-08-17 ...	275.00

After



	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2011-05-05 ...	375.00
2	200	Lathe	4750.00	2100	2011-05-07 ...	255.00
3	100	Drill Press	3500.00	2200	2011-06-19 ...	178.00
4	300	Mill	27300.00	2300	2011-06-19 ...	1875.00
5	100	Drill Press	3500.00	2400	2011-07-05 ...	0.00
6	100	Drill Press	5500.00	2500	2011-08-17 ...	275.00

Data Redundancy

- The design has *redundancy*, because the name of an equipment may be recorded multiple times, once for each new repair.

Another Example

EID	PID	Ename	Pname	Hours
1234	10	John Smith	B2B platform	10
1123	9	Tom Hanks	CRM	40
1234	9	John Smith	CRM	30
1023	10	Susan Mark	B2B platform	40
1033	8	Edward Codd	Database design	25

- Update the name of "`John Smith`" to "`John Gurke`"
- Delete `the last employee` who works for a certain project
- Insert `a new project that no employee` has been assigned to it yet
- Data redundancy

Where anomalies arise from

1. Functional dependencies
2. Multivalued dependencies
3. Data constraints

```
Normalize the original data
```

But First —

- We need to understand:
  - The relational model
  - Relational model terminology

Database schema

2. RELATIONAL MODEL

The Relational Model

- Introduced in a paper published in 1970.
- Created by E.F. Codd
  - He was an IBM engineer
  - The model used mathematics known as "relational algebra"
- It is the standard model for commercial DBMS products.

Important Relational Model Terms

1. Entity
2. Relation
3. Candidate key
4. Primary key
5. Surrogate key
6. Foreign key
7. Composite key
8. Referential integrity constraint
9. Functional dependency
10. Multivalued dependency
11. Determinant
12. Normal form

Entity

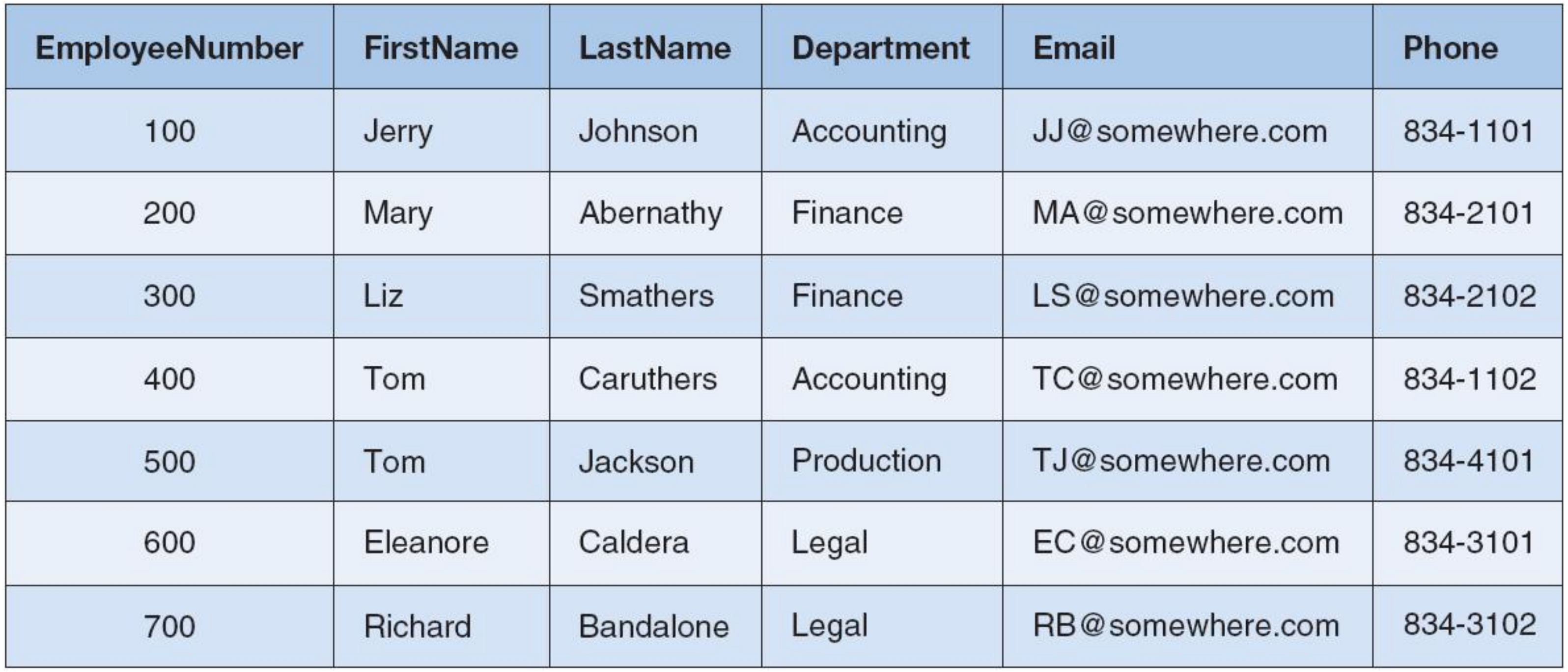
- An **entity** is some identifiable thing that users want to track:
  - Customers
  - Computers
  - Orders
  - Friendships

Relation

- Entities are stored in *relations*.

- A **relation** is a two-dimensional table that has the following eight characteristics:
  1) Each row contains an entity instance.
  2) The order of the rows is unimportant.
  3) Each column contains values for an entity property.
  4) Each column has a unique name.
  5) The order of the columns is unimportant.
  6) **All values in a column should be of the same data type.**
  7) ***Cells of the table hold a single value*.**
  8) *No two rows may be identical.*

A Relation: Employee

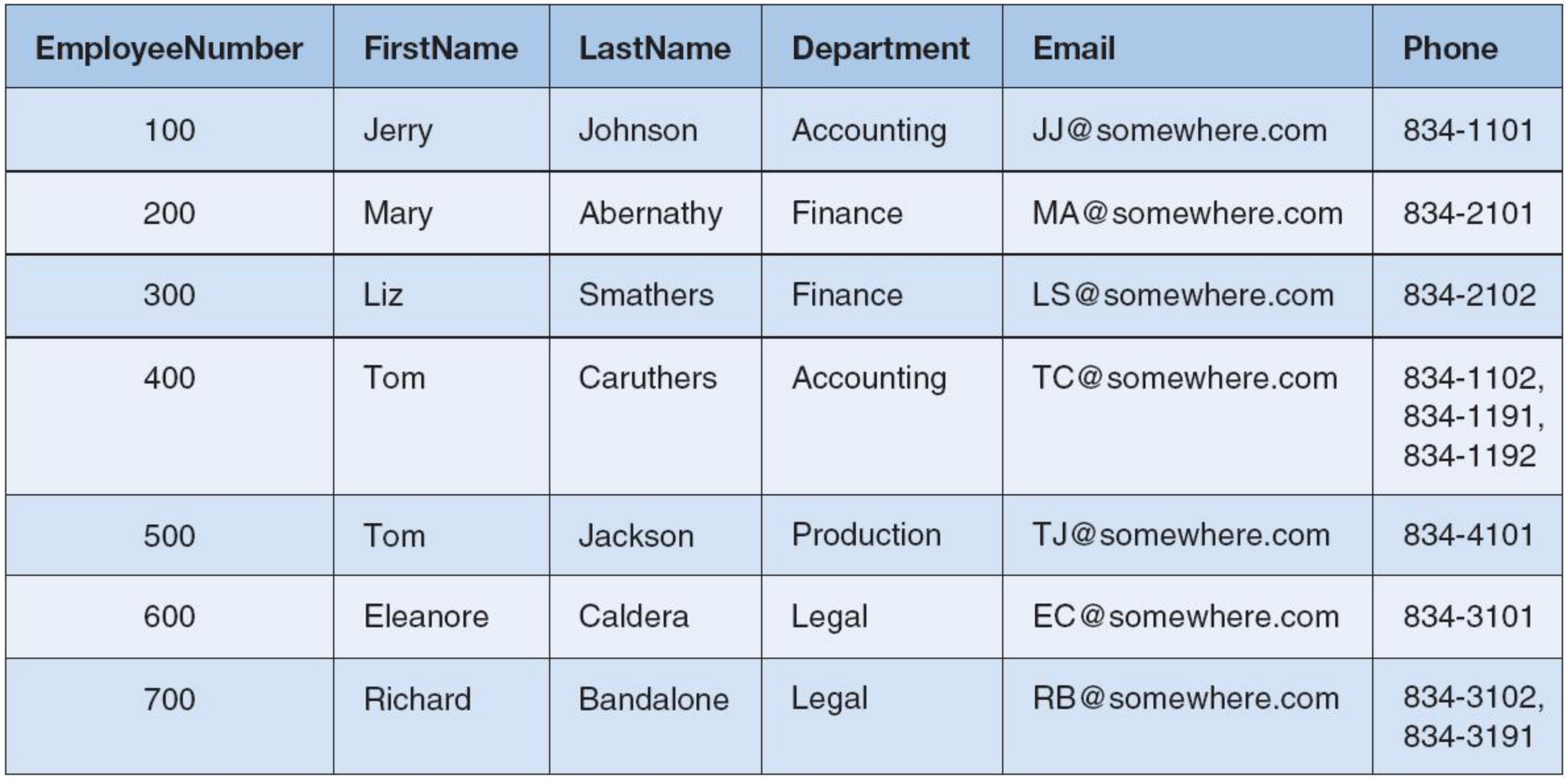


EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

A relation can be written by:

Employee (EmployeeNumber, FirstName, LastName, Department, Email, Phone)

Tables That Are Not Relations

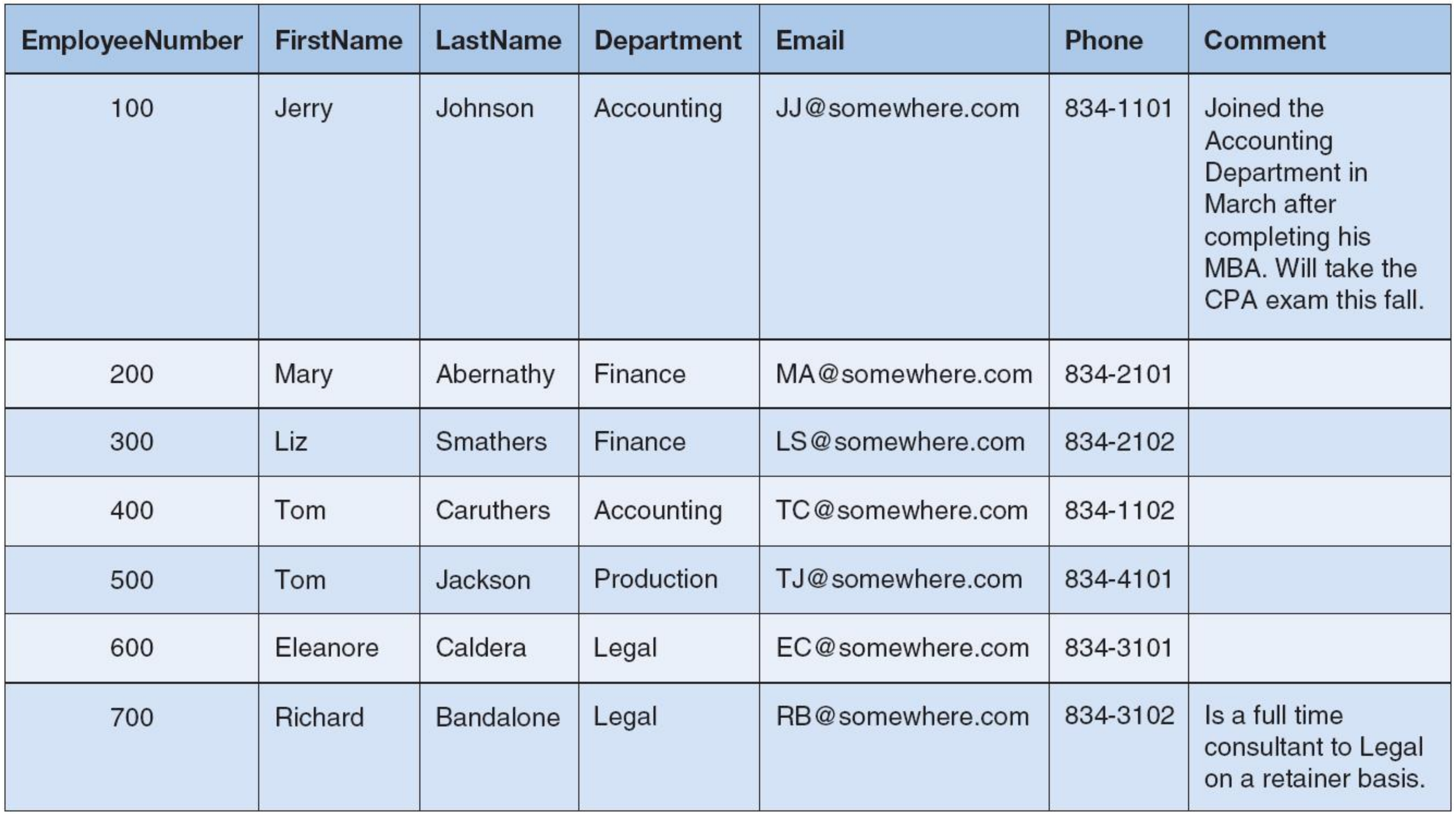


EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102, 834-1191, 834-1192
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102, 834-3191

Tables That Are Not Relations:

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
				Fax:	834-9911
				Home:	723-8795
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
				Fax:	834-9912
				Home:	723-7654
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

A Relation ?



EmployeeNumber	FirstName	LastName	Department	Email	Phone	Comment
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101	Joined the Accounting Department in March after completing his MBA. Will take the CPA exam this fall.
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101	
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102	
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102	
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101	
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101	
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102	Is a full time consultant to Legal on a retainer basis.

Alternative Terminology

- Although not all tables are relations, the terms *table* and *relation* are normally used interchangeably.
- The following three sets of terms are equivalent:

Table	Column	Row
Relation	Attribute	Tuple
File	Field	Record

Constraints between two sets of attributes

3. FUNCTIONAL DEPENDENCY

Functional Dependency

- A **functional dependency (FD)** occurs when the value of one set of attributes, A, determines the value of a second set of attributes, B.
- Notation: A → B
- A is the determinant.
- Statements:
  - B is functionally dependent on A.
  - A determines B.
- *FD is domain specific.*

FD Examples

StudentID → StudentName

StudentID → (DormName, DormRoom, Fee)

ExtendedPrice = Quantity X UnitPrice

(Quantity, UnitPrice) → ExtendedPrice

- Functional dependencies are *not* equations!

Composite Determinants

- **Composite determinant**

  A determinant of a functional dependency that consists of more than one attribute

StudentGrade(StudentID, ClassNumber, Grade)

Functional Dependency Rules

Assuming there are three attributes, A, B and C,

- *Reflexivity*:

  IF B IS SUBSET OF A, A → B.
- *Augmentation:*

  IF A → B, THEN AC → BC.
- *Decomposition*:

  IF A → (B,C), THEN A → B AND A→ C.
- *Transitivity:*

  IF A → B *AND* B → C, THEN A → C.
- *If (A,B) → C, then neither A nor B determines C by itself.*

Closure of a Set of Attributes

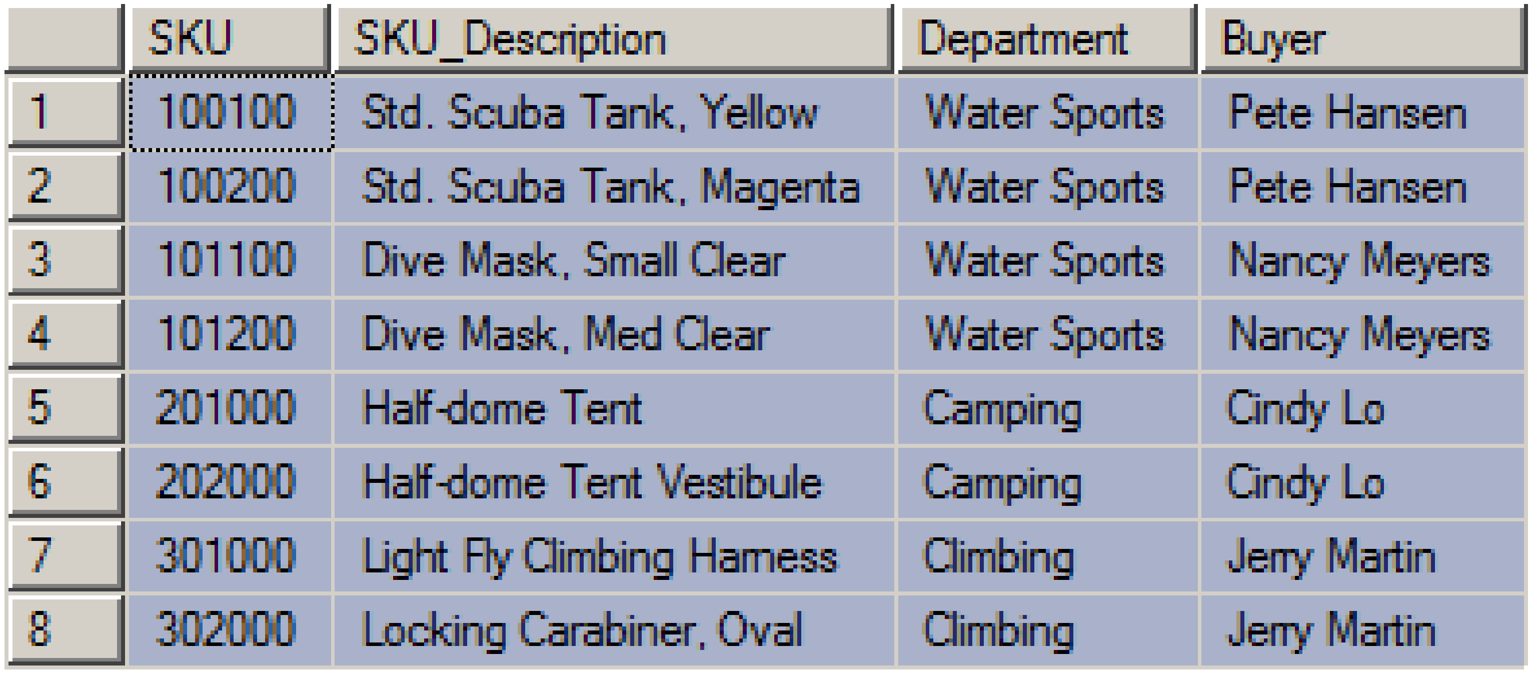
- *Math: The closure of a set A is the smallest closed set containing A*
- Find a set of attributes which can determine every attribute in the relation
- **Such a set makes a candidate key**.

Given a set of five attributes R = {ABCDE}

With two FD rules: F(R) = { A→B, AC→D }

Question: Find a closure that determines {ABCDE}

Functional Dependencies in the SKU_DATA Table



	SKU	SKU_Description	Department	Buyer
1	100100	Std. Scuba Tank, Yellow	Water Sports	Pete Hansen
2	100200	Std. Scuba Tank, Magenta	Water Sports	Pete Hansen
3	101100	Dive Mask, Small Clear	Water Sports	Nancy Meyers
4	101200	Dive Mask, Med Clear	Water Sports	Nancy Meyers
5	201000	Half-dome Tent	Camping	Cindy Lo
6	202000	Half-dome Tent Vestibule	Camping	Cindy Lo
7	301000	Light Fly Climbing Harness	Climbing	Jerry Martin
8	302000	Locking Carabiner, Oval	Climbing	Jerry Martin

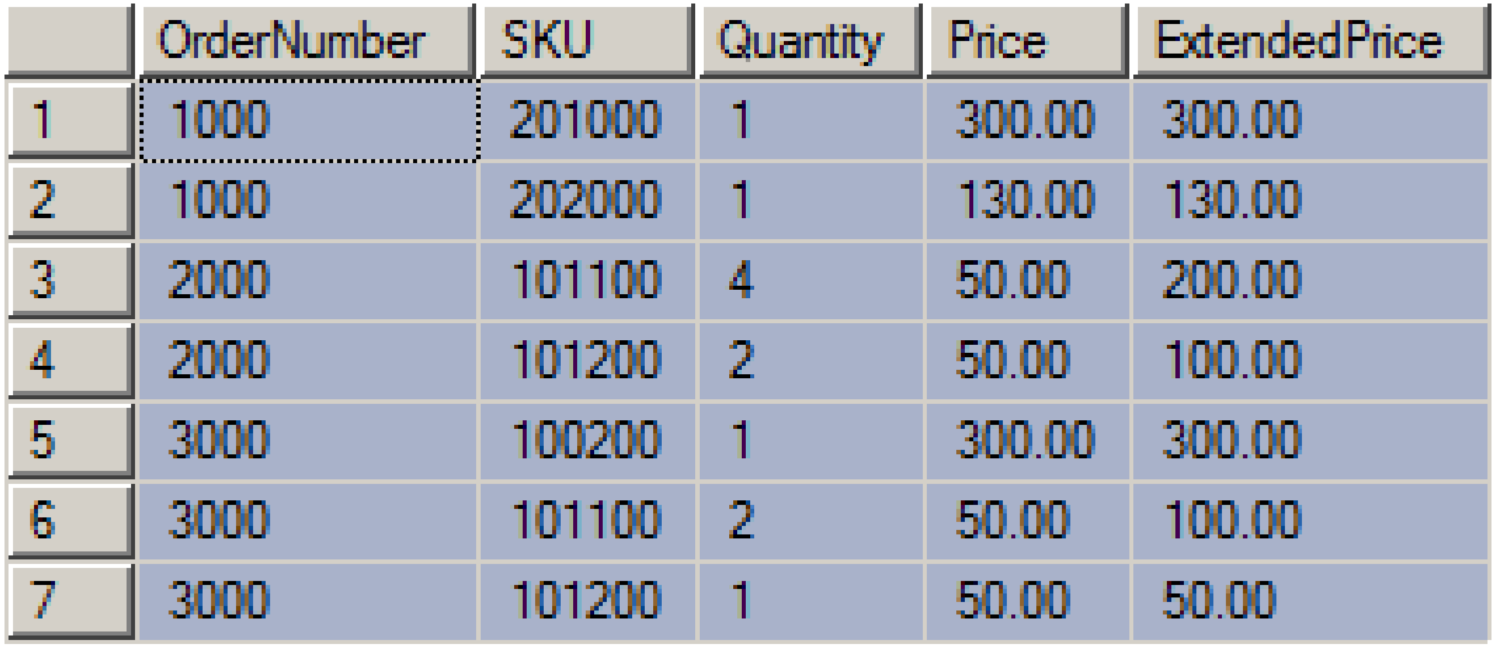
Functional Dependencies in the SKU_DATA Table

SKU → (SKU_Description, Department, Buyer)

SKU_Description → (SKU, Department, Buyer)

Buyer → Department

Functional Dependencies in the ORDER_ITEM Table



	OrderNumber	SKU	Quantity	Price	ExtendedPrice
1	1000	201000	1	300.00	300.00
2	1000	202000	1	130.00	130.00
3	2000	101100	4	50.00	200.00
4	2000	101200	2	50.00	100.00
5	3000	100200	1	300.00	300.00
6	3000	101100	2	50.00	100.00
7	3000	101200	1	50.00	50.00

Functional Dependencies in the ORDER_ITEM Table

(OrderNumber, SKU) → (Quantity, Price, ExtendedPrice)

(Quantity, Price) → (ExtendedPrice)

Exercise

Book (authorID, authorName, email, authorAddress, bookTitle, bookYear, bookEdition, pages)

1. Discover all *FD*s in the relation ‘Book’. Represent them in the standard notation.
2. Which attribute or attribute set can determine every other attribute?
3. Each *FD* should be given with its domain assumption(s).

Multivalued Dependency

- Occurs when a determinant is matched with a particular set of values.

Employee → → Degree

Student → → Club

How many keys do you have?

4. KEYS

Keys

- A **key** is a combination of one or more columns that is used to identify rows in a relation.
  - Can determine every other column

- A **composite key** is a key that consists of two or more columns.

Candidate and Primary Keys

- A **candidate key** is a key that determines all the columns in a relation.

- A **primary key** is a candidate key selected as the primary means of identifying rows in a relation.
  - There is only one primary key per relation.
  - The primary key may be a composite key.
  - *The ideal primary key is short, numeric, and never changes.*

Surrogate Keys

- A **surrogate key** is an artificial column added to a relation to serve as a primary key.
  - Short, numeric, and never changes—an ideal primary key
  - Has artificial values that are meaningless to users
  - Normally hidden in forms and reports

Surrogate Keys

NOTE: The primary key of the relation is <u>underlined</u> below:

- RENTAL_PROPERTY without surrogate key:

  RENTAL_PROPERTY (<u>Street</u>, <u>City</u>, <u>State/Province</u>, <u>Zip/PostalCode</u>, <u>Country</u>, Rental_Rate)

- RENTAL_PROPERTY with surrogate key:

  RENTAL_PROPERTY (<u>PropertyID</u>, Street, City, State/Province, Zip/PostalCode, Country, Rental_Rate)

A link between two relations

5. FOREIGN KEY & REFERENTIAL INTEGRITY

Foreign Keys

- A common key between two relations A and B

- A **foreign key in relation** B (child relation) is **the primary key in relation** A (parent relation) to form ***a link between two relations, A and*** B.
  - A foreign key can contain one or more columns.

Foreign Keys

<u>Standard notation</u>

The primary key of a relation is <u>underlined</u> and a foreign key is in *italics*.

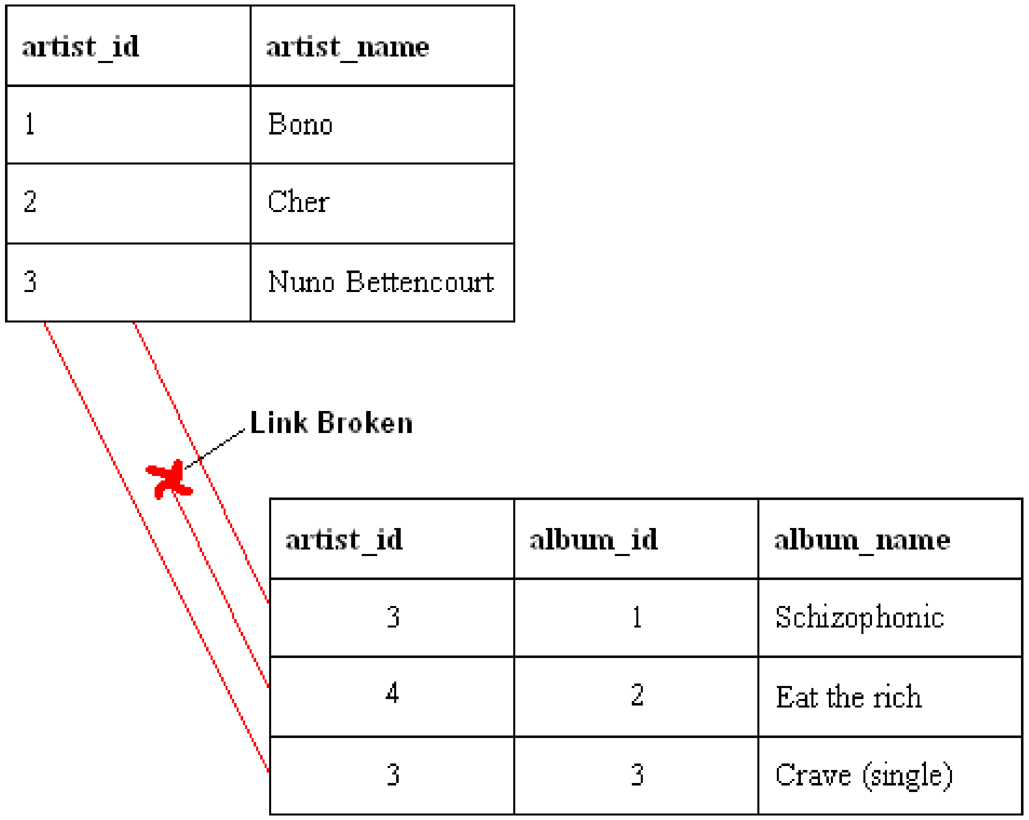
DEPARTMENT (<u>DepartmentName</u>, BudgetCode, ManagerName)

EMPLOYEE (<u>EmployeeNumber</u>, EmployeeLastName, EmployeeFirstName, *DepartmentName*)

The Referential Integrity Constraint

- A **referential integrity constraint** is a restriction that *limits* the foreign key values *in a child relation* to the primary key values that exist in its *parent* relation.

If there is NO foreign key



artist_id	artist_name
1	Bono
2	Cher
3	Nuno Bettencourt

artist_id	album_id	album_name
3	1	Schizophonic
4	2	Eat the rich
3	3	Crave (single)

Schema Example

SKU_DATA (<u>SKU</u>, SKU_Description, Department, Buyer)

ORDER_ITEM (<u>OrderNumber</u>, <u>*SKU*</u>, Quantity, Price, ExtendedPrice)

Where ORDER_ITEM.SKU must exist in SKU_DATA.SKU

Normalize a relation to a normal norm……..

“I swear to construct my tables so that all nonkey columns are dependent on the key, the whole key and nothing but the key, so help me Codd.”

7. NORMAL FORMS

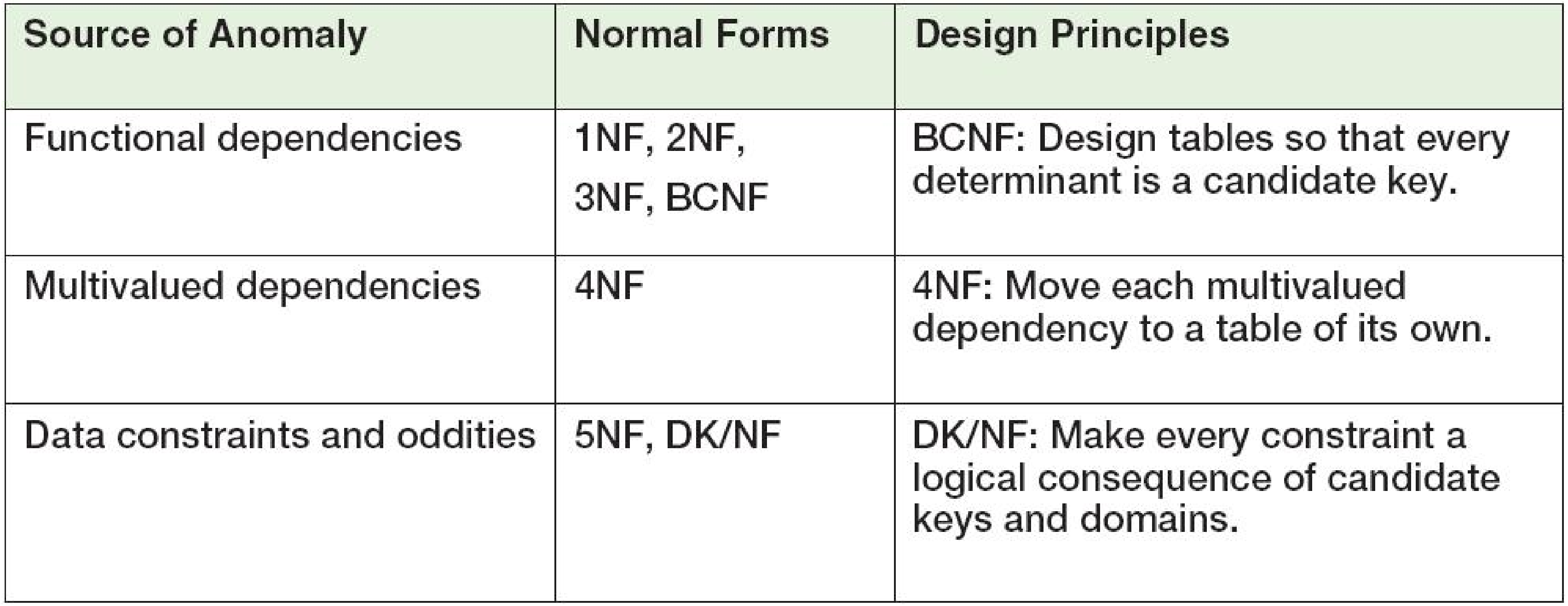
Normalization

- Codd (1972)
- A *normalization* is the process of decomposing unsatisfactory "bad" relations by breaking up their attributes into smaller relations

- A *normal form* is a certification that tells whether a relation schema is in a particular state
- 1NF, 2NF, 3NF, BCNF, 4NF, …..

Normal Forms

- Relations are categorized as a **normal form** based on which modification anomalies or other problems they are subject to:

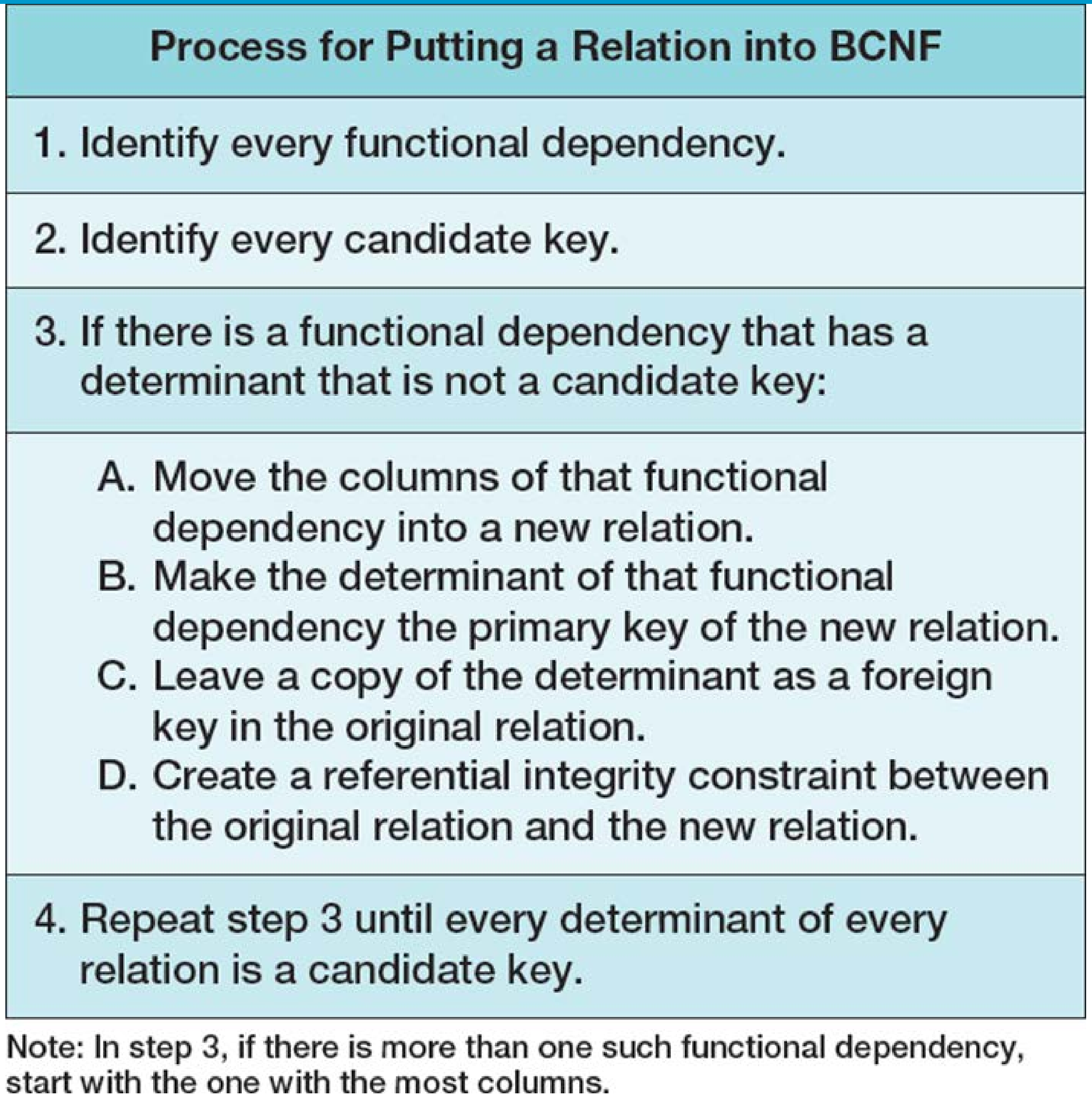


Source of Anomaly	Normal Forms	Design Principles
Functional dependencies	1NF, 2NF, 3NF, BCNF	BCNF: Design tables so that every determinant is a candidate key.
Multivalued dependencies	4NF	4NF: Move each multivalued dependency to a table of its own.
Data constraints and oddities	5NF, DK/NF	DK/NF: Make every constraint a logical consequence of candidate keys and domains.

Normalization Theory

- By *normalization* theory based on *functional dependencies*, we transform a relational model with modification anomalies, into an appropriate *normal form* using *decomposition* to produce multiple smaller relations, mostly to reduce ***data redundancy*** and improve ***data integrity***.
- The transformation should be lossless
- One Fat Relation → Multiple Smaller Relations

Put All Relations into BCNF



Process for Putting a Relation into BCNF
1. Identify every functional dependency.
2. Identify every candidate key.
3. If there is a functional dependency that has a determinant that is not a candidate key:
A. Move the columns of that functional dependency into a new relation. B. Make the determinant of that functional dependency the primary key of the new relation. C. Leave a copy of the determinant as a foreign key in the original relation. D. Create a referential integrity constraint between the original relation and the new relation.
4. Repeat step 3 until every determinant of every relation is a candidate key.

Note: In step 3, if there is more than one such functional dependency, start with the one with the most columns.

(1) 1st Normal Form

- A domain is *atomic* if its elements are considered to be indivisible units.
  - Examples of non-atomic domains
    - Set of names, **composite attributes**
    - Identification numbers like INFO421, INFO400 that can be separated into parts
    - **Multivalued attributes**: Address, phone number
- A relation is in *1NF* if all attribute domains are atomic.

- *Non-atomic values* complicate storage and encourage redundant (repeated) storage of data
  - E.g. Set of accounts stored with each customer, and set of owners stored with each account

A Table → 1NF

- Make it a relation!
- If there is composite attribute,
  Store each component as an attribute/column.
- (If there is multivalued attributes, Move it to a new relation)

- Example:
  R (EID, Ename, PhoneNum, Email)
  PhoneNum is multivalued.

Place PhoneNum into a new relation and define an *FK* in the relation

- R1 (<u>EID</u>, Ename, Email)
- R2 (<u>*EID*</u>, <u>phoneType</u>, <u>phoneNumber</u>)

9/30/2018

2NF

- An attribute A of a relation R is a ***nonprimary(nonkey) attribute*** if it is not part of any key in R, otherwise, A is a ***primary(key) attribute***.

- A table/relation is in 2NF <u>if all nonkey attributes are functionally dependent on the *entire* primary key.</u>
- **No partial dependency**

Example: WorkOn(<u>*EID*</u>, *Ename, Email,* <u>*PID*</u>, *Pname, Hours*)

1NF → 2NF

Given a relation R

1. Discover all FDs
2. Any partial dependencies? (If any determinant is not the entire PK)

3. <u>Take each partial dependency out of R into a new relation RN.</u>
   - Keep the origin copy of the partial key in R
   - Each new relation RN contains a partial key copy with its dependent attribute(s).
   - Define the partial key in RN as FK referencing to its origin in R.

Notations

R (<u>A</u>, <u>B</u>, C, D)

FD {AB→CD, A→C}

R is in 1NF. Transform R into 2NF

Write the new relational model in 2NF

- Keep the original copy of the partial key in R
- Move into a new relation RN, the partial key with its dependent attribute(s)
- Define the partial key in RN as FK referencing to its origin in R.

Notations

R (<u>A</u>, <u>B</u>, C, D)

FD {AB→CD, A→C}

R is in 1NF. Transform R into 2NF

Write the new relational model in 2NF

R (<u>A</u>, <u>B</u>, D)

RN (<u>*A*</u>, C)

Example

EID	PID	Ename	email	Pname	Hours
1234	10	John Smith	jsmith@ac.com	B2B platform	10
1123	9	Ben Liu	bliu@ac.com	CRM	40
1234	9	John Smith	jsmith@ac.com	CRM	30
1023	10	Susan Sidhuk	ssidhuk@ac.com	B2B platform	40

Decomposition

Foreign key

EID	Ename	email
1234	John Smith	jsmith@ac.com
1123	Ben Liu	bliu@ac.com
1023	Susan Sidhuk	ssidhuk@ac.com

EID	**PID**	Pname	Hours
1234	10	B2B platform	10
1123	9	CRM	40
1234	9	CRM	30
1023	10	B2B platform	40

- Decomposition eliminates redundancy
- To get back to the original relation: Join

Decomposition

- Decomposition may be applied recursively

EID	PID	Pname	Hours
1234	10	B2B platform	10
1123	9	CRM	40
1234	9	CRM	30
1023	10	B2B platform	40

PID	Pname
10	B2B platform
9	CRM

EID	***PID***	Hours
1234	10	10
1123	9	40
1234	9	30
1023	10	40

1NF vs. 2NF

EID	PID	Ename	email	Pname	Hours
1234	10	John Smith	jsmith@ac.com	B2B platform	10
1123	9	Ben Liu	bliu@ac.com	CRM	40
1234	9	John Smith	jsmith@ac.com	CRM	30
1023	10	Susan Sidhuk	ssidhuk@ac.com	B2B platform	40

EID	Ename	email
1234	John Smith	jsmith@ac.com
1123	Ben Liu	bliu@ac.com
1023	Susan Sidhuk	ssidhuk@ac.com

PID	Pname
10	B2B platform
9	CRM

EID	***PID***	Hours
1234	10	10
1123	9	40
1234	9	30
1023	10	40

Exercise

Normalize the following relation to 2NF

Customer(cid, cname, address, salesRepNum, salesRepName)

3NF & BCNF

- Every nonkey attribute depends on the key, the whole key, and nothing but the key. (No partial dependency, No nonkey dependency)
- **Boyce-Codd Normal Form (BCNF)**—a relation is in BCNF if every determinant is a candidate key. (No transitive dependency)
- R(<u>Book</u>, Genre, Author, AuthorNationality)

Journey to the Center of the Earth	Science Fiction	Jules Verne	French

→ BCNF

In R, any FD whose determinant is not entire primary key?

<u>Take the FD out of R into a new relation RN</u>

- Keep the origin copy of the determinant in R
- Each new relation RN contains the determinant with its dependent attribute(s).
- Define a PK for RN
- Define the determinant in RN as FK referencing to its origin in R

4NF

- No multivalued dependency!

- Example:
  R (EID, Ename, PhoneNum, email)
  PhoneNum is multivalued.
- EID → PhoneNum

1. Place multivalued FD into a new relation RN
2. Define PK and *FK*

   - R(<u>EID</u>, Ename, email)
   - RN(*<u>EID</u>*, <u>phoneType</u>, <u>phoneNumber</u>)

Exercise

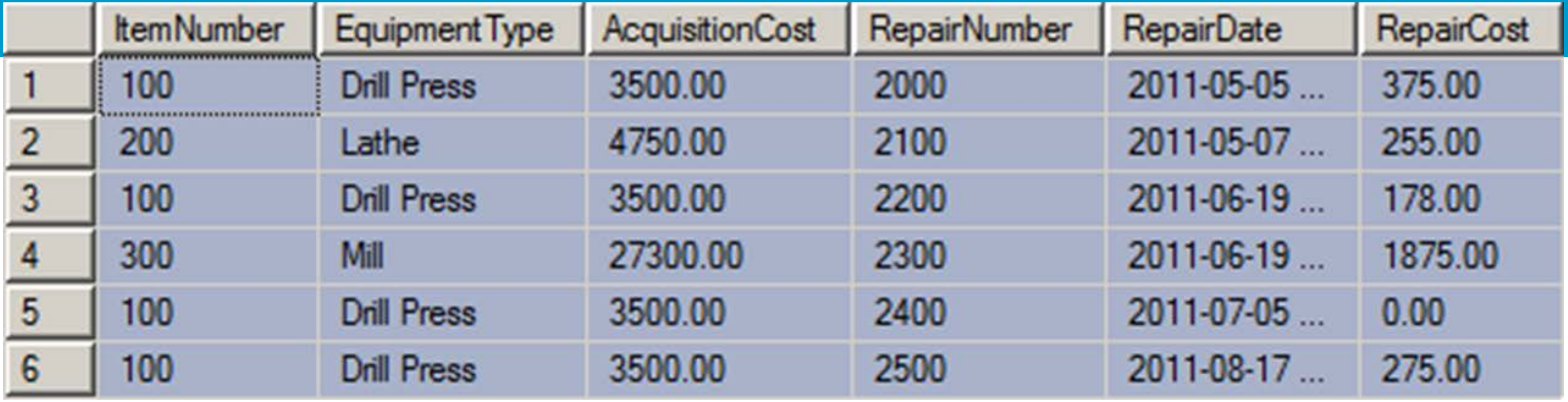
Initial design:

Student(<u>sid</u>, sname, schours, gpa, advisorid, advisorname, <u>cid</u>, cname, ccredit, grade)

- 1NF
- 2NF
- 3NF – BCNF
- 4NF

- The following slides are from the textbook Chapter 3.
- Read the book.

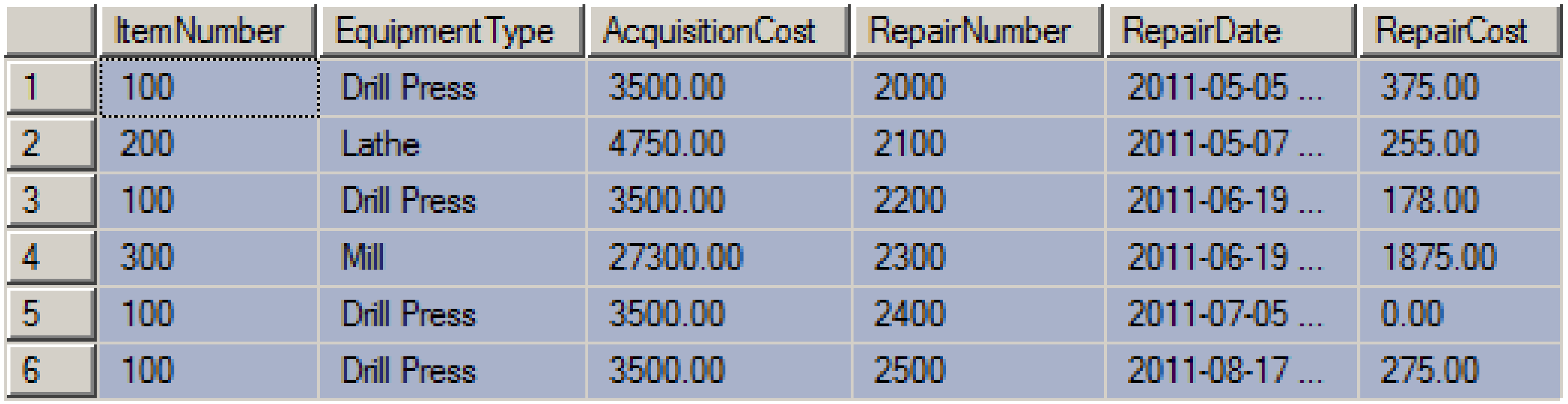
Exercise



	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2011-05-05 ...	375.00
2	200	Lathe	4750.00	2100	2011-05-07 ...	255.00
3	100	Drill Press	3500.00	2200	2011-06-19 ...	178.00
4	300	Mill	27300.00	2300	2011-06-19 ...	1875.00
5	100	Drill Press	3500.00	2400	2011-07-05 ...	0.00
6	100	Drill Press	3500.00	2500	2011-08-17 ...	275.00

- EQUIPMENT_REPAIR (ItemNumber, Type, AcquisitionCost,
  RepairNumber, RepairDate, RepairAmount)

Putting a Relation into BCNF: EQUIPMENT_REPAIR



	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2011-05-05 ...	375.00
2	200	Lathe	4750.00	2100	2011-05-07 ...	255.00
3	100	Drill Press	3500.00	2200	2011-06-19 ...	178.00
4	300	Mill	27300.00	2300	2011-06-19 ...	1875.00
5	100	Drill Press	3500.00	2400	2011-07-05 ...	0.00
6	100	Drill Press	3500.00	2500	2011-08-17 ...	275.00

EQUIPMENT_REPAIR (ItemNumber, Type, AcquisitionCost,
RepairNumber, RepairDate, RepairAmount)

Do It Yourself

Putting a Relation into BCNF: EQUIPMENT_REPAIR

EQUIPMENT_REPAIR (ItemNumber, Type, AcquisitionCost, <u>RepairNumber</u>, RepairDate, RepairAmount)

List all of FDs:

ItemNumber → (Type, AcquisitionCost)

RepairNumber → (ItemNumber, Type, AcquisitionCost,
RepairDate, RepairAmount)

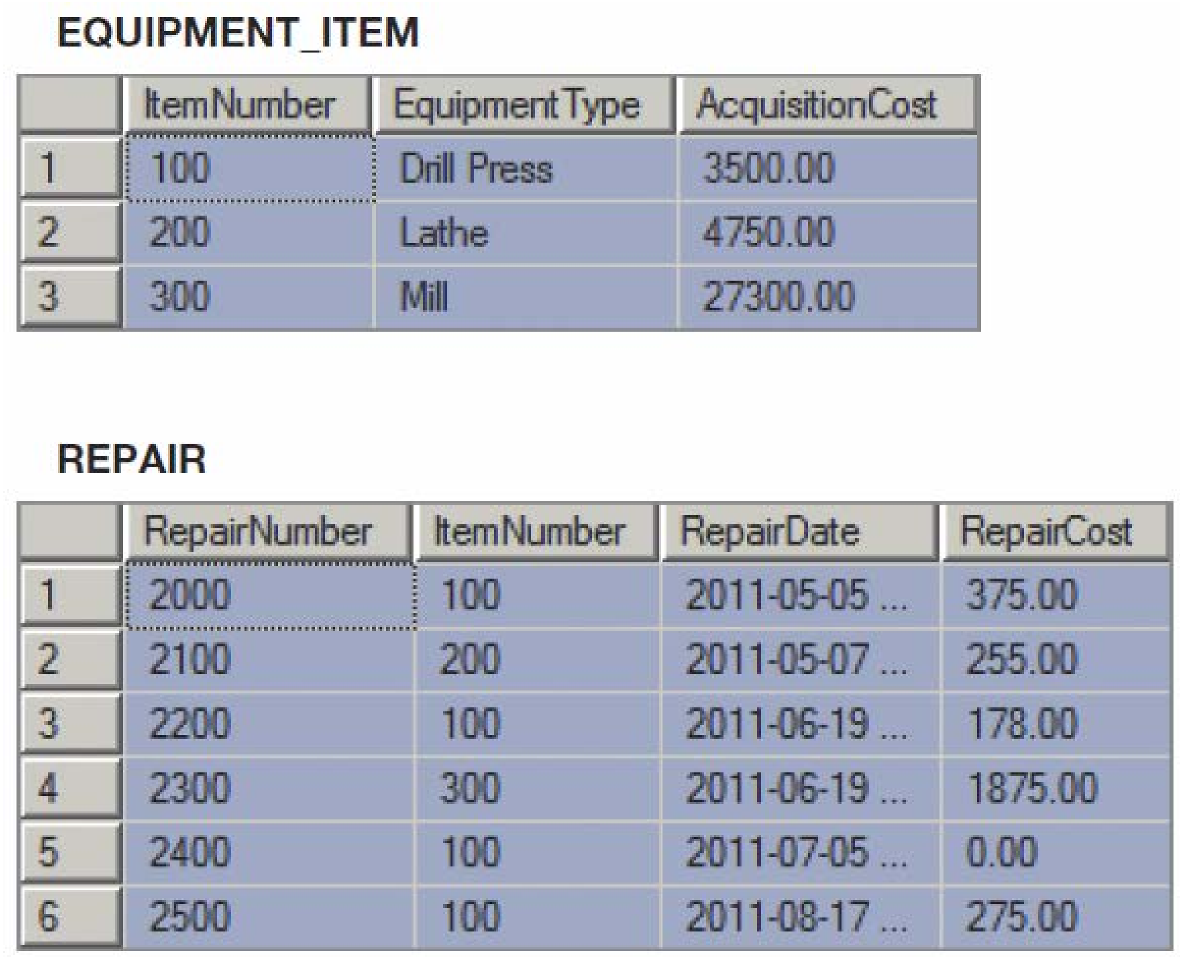
The tables after normalization are as follows:

ITEM (**<u>ItemNumber</u>**, Type, AcquisitionCost)

REPAIR (**<u>RepairNumber</u>**, *ItemNumber*, RepairDate, RepairAmount)

Where REPAIR.*ItemNumber* must exist in ITEM.ItemNumber

Putting a Relation into BCNF: New Relations



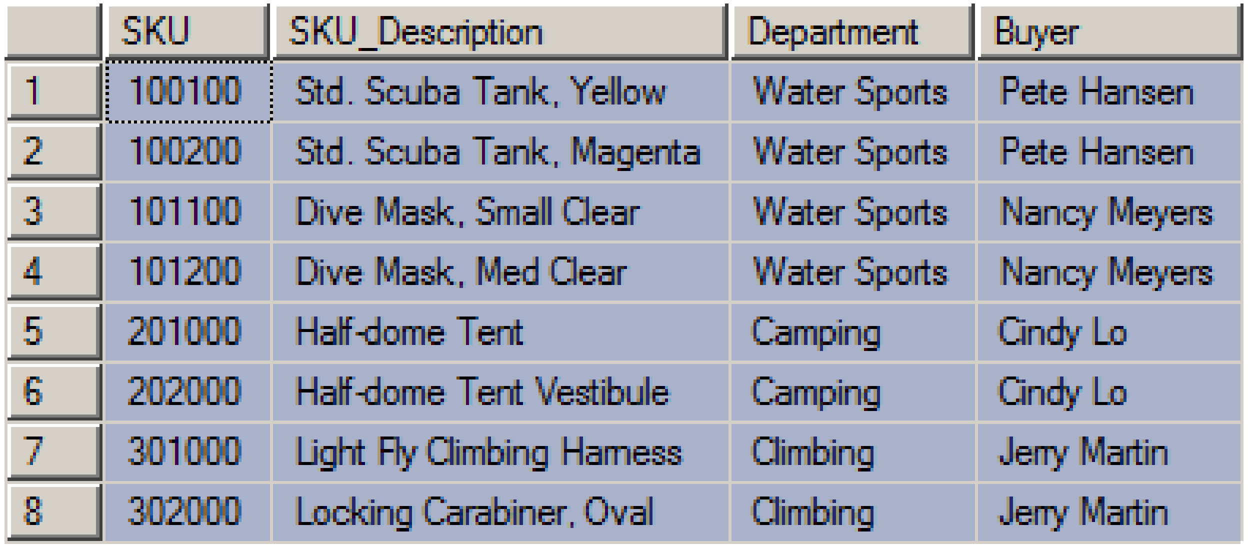
EQUIPMENT_ITEM

	ItemNumber	EquipmentType	AcquisitionCost
1	100	Drill Press	3500.00
2	200	Lathe	4750.00
3	300	Mill	27300.00

REPAIR

	RepairNumber	ItemNumber	RepairDate	RepairCost
1	2000	100	2011-05-05 ...	375.00
2	2100	200	2011-05-07 ...	255.00
3	2200	100	2011-06-19 ...	178.00
4	2300	300	2011-06-19 ...	1875.00
5	2400	100	2011-07-05 ...	0.00
6	2500	100	2011-08-17 ...	275.00

Putting a Relation into BCNF: SKU_DATA Step-by-Step – 1NF



	SKU	SKU_Description	Department	Buyer
1	100100	Std. Scuba Tank, Yellow	Water Sports	Pete Hansen
2	100200	Std. Scuba Tank, Magenta	Water Sports	Pete Hansen
3	101100	Dive Mask, Small Clear	Water Sports	Nancy Meyers
4	101200	Dive Mask, Med Clear	Water Sports	Nancy Meyers
5	201000	Half-dome Tent	Camping	Cindy Lo
6	202000	Half-dome Tent Vestibule	Camping	Cindy Lo
7	301000	Light Fly Climbing Harness	Climbing	Jerry Martin
8	302000	Locking Carabiner, Oval	Climbing	Jerry Martin

SKU_DATA (<u>SKU</u>, SKU_Description, Department, Buyer)

1NF - Checking against the definition of 1NF, this relation is in 1NF.

Putting a Relation into BCNF: SKU_DATA Step-by-Step – 2NF

SKU_DATA (<u>SKU</u>, SKU_Description, Department, Buyer)

SKU → (SKU_Description, Department, Buyer)
SKU_Description → (SKU, Department, Buyer)
Buyer → Department

— SKU and SKU_Description are candidate keys.

— A relation is in 2NF if and only if *it is in 1NF* and *all non-key attributes are determined by the primary key*.
— Since SKU is a single column primary key, all non-key attributes are determined by SKU, and the relation is in 2NF.

Putting a Relation into BCNF: SKU_DATA Step-by-Step – 3NF

SKU_DATA (<u>SKU</u>, SKU_Description, Department, Buyer)

SKU → (SKU_Description, Department, Buyer)
SKU_Description → (SKU, Department, Buyer)
Buyer → Department

— SKU and SKU_Description are candidate keys.

— A relation is in 3NF if and only if *it is in 2NF* and *there are no non-key attributes determined by another non-key attribute*.
— However, the term non-key attribute means an attribute that is neither (1) a candidate key itself, nor (2) part of a composite candidate key.
— Therefore, the only non key attribute is Buyer, and it is a determinant.
— Therefore, not in 3NF.

Putting a Relation into BCNF: SKU_DATA Step-by-Step – 3NF

— Therefore, break out the Buyer → Department functional dependency.

SKU_DATA_2 (<u>SKU</u>, SKU_Description, *Buyer*)
BUYER (<u>Buyer</u>, Department)

Where SKU_DATA_2.Buyer must exist in BUYER.Buyer

— SKU_DATA_2 is in 3NF
— BUYER is in 3NF

Putting a Relation into BCNF: SKU_DATA Step-by-Step – BCNF

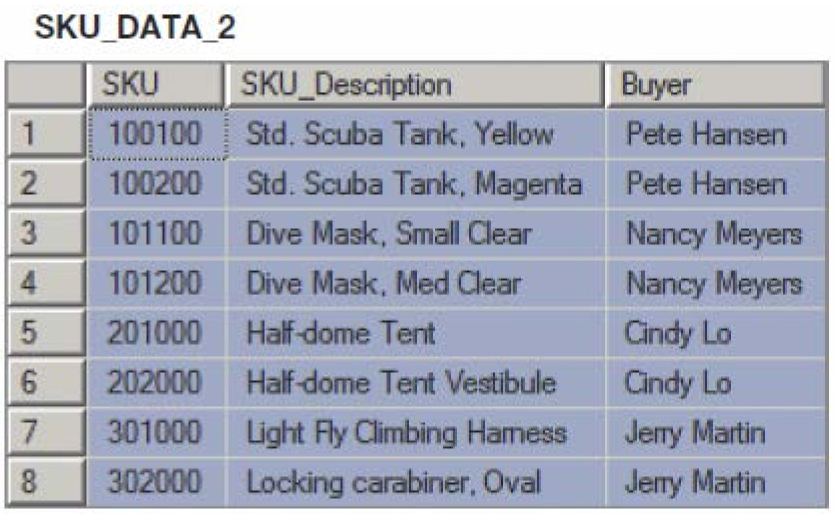
SKU_DATA_2 (<u>SKU</u>, SKU_Description, *Buyer*)
BUYER (<u>Buyer</u>, Department)

Where SKU_DATA_2.Buyer must exist in BUYER.Buyer

SKU → (SKU_Description, Department, Buyer)
SKU_Description → (SKU, Department, Buyer)
Buyer → Department

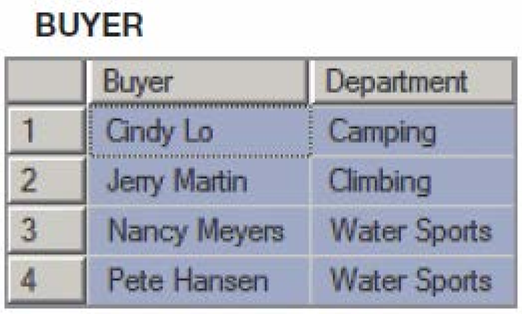
— A relation is in BCNF if and only if *it is in 3NF* and *every determinant is a candidate-key*.
— In SKU_DATA_2, both determinants are determinant keys, so SKU_DATA_2 is in BCNF.
— In BUYER, the determinant is a determinant key, so BUYER is in BCNF.

Putting a Relation into BCNF: SKU_DATA Step-by-Step – New Relations



SKU_DATA_2

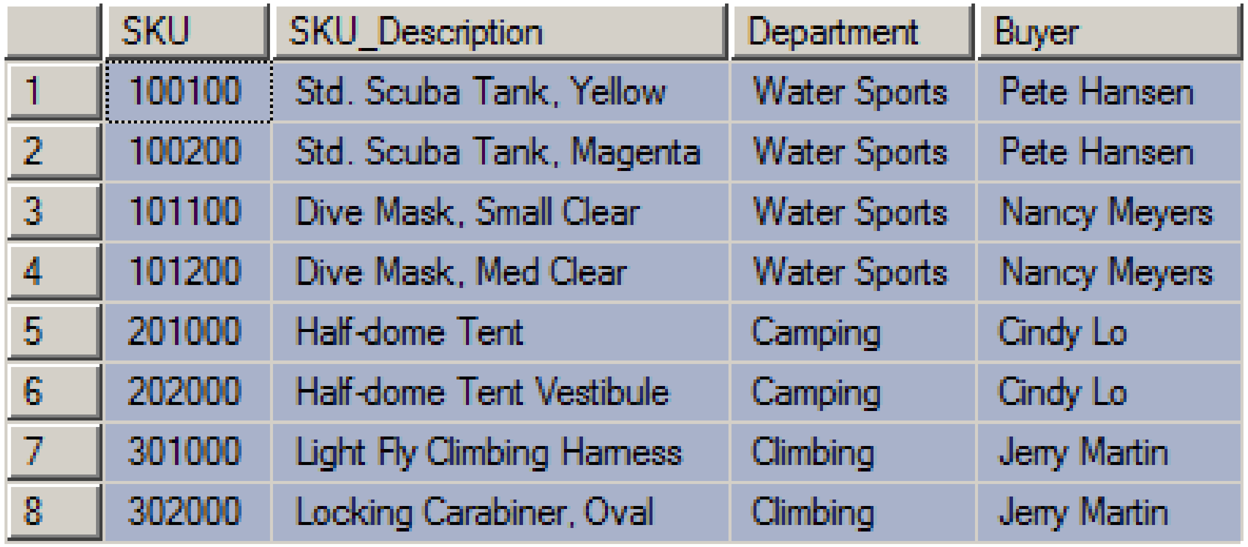
	SKU	SKU_Description	Buyer
1	100100	Std. Scuba Tank, Yellow	Pete Hansen
2	100200	Std. Scuba Tank, Magenta	Pete Hansen
3	101100	Dive Mask, Small Clear	Nancy Meyers
4	101200	Dive Mask, Med Clear	Nancy Meyers
5	201000	Half-dome Tent	Cindy Lo
6	202000	Half-dome Tent Vestibule	Cindy Lo
7	301000	Light Fly Climbing Harness	Jerry Martin
8	302000	Locking carabiner, Oval	Jerry Martin



BUYER

	Buyer	Department
1	Cindy Lo	Camping
2	Jerry Martin	Climbing
3	Nancy Meyers	Water Sports
4	Pete Hansen	Water Sports

Putting a Relation into BCNF: SKU_DATA Straight-to-BCNF



	SKU	SKU_Description	Department	Buyer
1	100100	Std. Scuba Tank, Yellow	Water Sports	Pete Hansen
2	100200	Std. Scuba Tank, Magenta	Water Sports	Pete Hansen
3	101100	Dive Mask, Small Clear	Water Sports	Nancy Meyers
4	101200	Dive Mask, Med Clear	Water Sports	Nancy Meyers
5	201000	Half-dome Tent	Camping	Cindy Lo
6	202000	Half-dome Tent Vestibule	Camping	Cindy Lo
7	301000	Light Fly Climbing Harness	Climbing	Jerry Martin
8	302000	Locking Carabiner, Oval	Climbing	Jerry Martin

Putting a Relation into BCNF: SKU_DATA Straight-to-BCNF

SKU_DATA (<u>SKU</u>, SKU_Description, Department, Buyer)

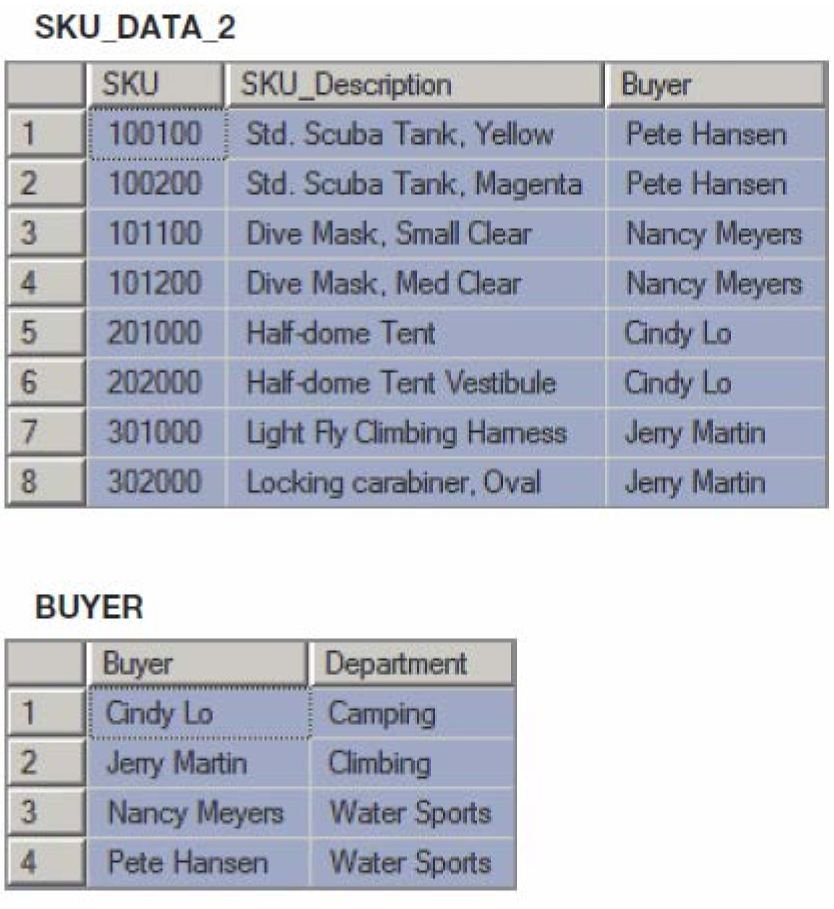
SKU → (SKU_Description, Department, Buyer)
SKU_Description → (SKU, Department, Buyer)
Buyer → Department

— Therefore, break out the Buyer → Department functional dependency.

SKU_DATA	**(<u>SKU</u>, SKU_Description, *Buyer*)**
BUYER	**(<u>Buyer</u>, Department)**

Where BUYER.Buyer must exist in SKU_DATA.Buyer

Putting a Relation into BCNF:
SKU_DATA Straight-to-BCNF New Relations



SKU_DATA_2

	SKU	SKU_Description	Buyer
1	100100	Std. Scuba Tank, Yellow	Pete Hansen
2	100200	Std. Scuba Tank, Magenta	Pete Hansen
3	101100	Dive Mask, Small Clear	Nancy Meyers
4	101200	Dive Mask, Med Clear	Nancy Meyers
5	201000	Half-dome Tent	Cindy Lo
6	202000	Half-dome Tent Vestibule	Cindy Lo
7	301000	Light Fly Climbing Harness	Jerry Martin
8	302000	Locking carabiner, Oval	Jerry Martin

BUYER

	Buyer	Department
1	Cindy Lo	Camping
2	Jerry Martin	Climbing
3	Nancy Meyers	Water Sports
4	Pete Hansen	Water Sports

Multivalued Dependencies

- A **multivalued dependency** occurs when a determinant is matched with a particular *set* of values:

  Employee →→ Degree

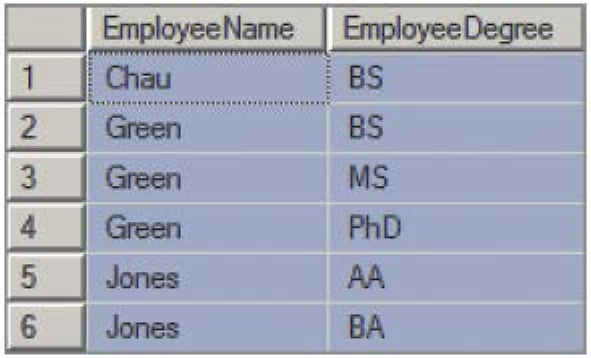
  Employee →→ Sibling

  PartKit →→ Part

- The determinant of a multivalued dependency can *never* be a primary key.

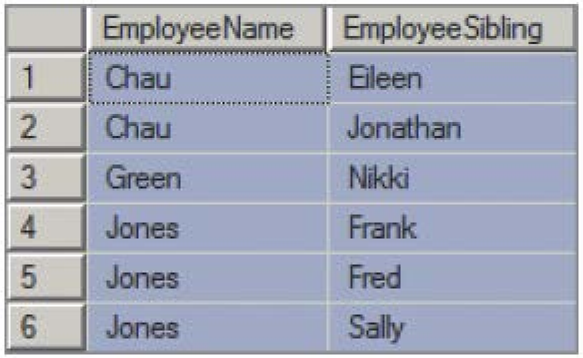
Multivalued Dependencies

EMPLOYEE_DEGREE



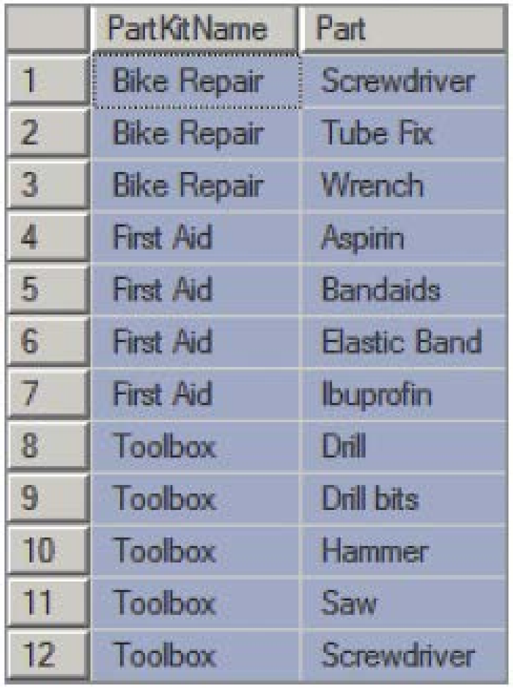
	EmployeeName	EmployeeDegree
1	Chau	BS
2	Green	BS
3	Green	MS
4	Green	PhD
5	Jones	AA
6	Jones	BA

EMPLOYEE_SIBLING



	EmployeeName	EmployeeSibling
1	Chau	Eileen
2	Chau	Jonathan
3	Green	Nikki
4	Jones	Frank
5	Jones	Fred
6	Jones	Sally

PARTKIT_PART



	PartKitName	Part
1	Bike Repair	Screwdriver
2	Bike Repair	Tube Fix
3	Bike Repair	Wrench
4	First Aid	Aspirin
5	First Aid	Bandaids
6	First Aid	Elastic Band
7	First Aid	Ibuprofin
8	Toolbox	Drill
9	Toolbox	Drill bits
10	Toolbox	Hammer
11	Toolbox	Saw
12	Toolbox	Screwdriver

Eliminating Anomalies from Multivalued Dependencies

- Multivalued dependencies are not a problem if they are in a separate relation, so:
  - Always put multivalued dependencies into their own relation.
  - This is known as **Fourth Normal Form (4NF)**.

Two advantages of normalizations

- Reduce data duplication
- Eliminate modification anomalies

Two disadvantages of normalizations

- Increase the complexity of SQL queries
- Slow down the application processing