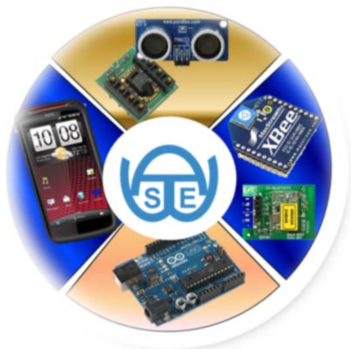
David M. Kroenke and David J. Auer

Database Processing:

Fundamentals, Design, and Implementation



Chapter Three:

The Relational Model and Normalization

Chapter Objectives

- To be able to identify possible insertion, deletion, and update anomalies in a relation
- To be able to place a relation into BCNF normal form
- To understand the special importance of domain/key normal form
- To be able to identify multivalued dependencies
- To be able to place a relation in fourth normal form

Chapter Premise

- We have received one or more tables of existing data.
- The data is to be stored in a new database.
- QUESTION: Should the data be stored as received, or should it be transformed for storage?

How Many Tables?

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

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

Should we store these two tables as they are, or should we combine them into one table in our new database?

A Very Strange Table!

PRODUCT_BUYER

	BuyerName	SKU_Managed	CollegeMajor
1	Pete Hansen	100100	Business Administration
2	Pete Hansen	100200	Business Administration
3	Nancy Meyers	101100	Art
4	Nancy Meyers	101100	Info Systems
5	Nancy Meyers	101200	Art
6	Nancy Meyers	101200	Info Systems
7	Cindy Lo	201000	History
8	Cindy Lo	202000	History
9	Jenny Martin	301000	Business Administration
10	Jenny Martin	301000	English Literature
11	Jenny Martin	302000	Business Administration
12	Jenny Martin	302000	English Literature

To understand why this is a very strange table, consider how you would add the fact that **Nancy Meyers** is now managing **SKU 101300**!

Entity

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

Alternative Terminology

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

Table	Column	Row
Relation	Attribute	Tuple
File	Field	Record

Database Integrity

- We have defined three constraints so far in our discussion:
  - The **domain integrity constraint**
  - The **entity integrity constraint**
  - The **referential integrity constraint**
- The purpose of these three constraints, taken as a whole, is to create **database integrity**, which means that the data in our database will be useful, meaningful data.

The Domain Integrity Constraint

- The requirement that all of the values in a column are of the same kind is know as the **domain integrity constraint**.
- The term domain means a grouping of data that meets a specific type definition.
  - **FirstName** could have a domain of names such as *Albert, Bruce, Cathy, David, Edith*, and so forth.
  - All values of **FirstName** *must* come from the names in that domain.
- Columns in different relations may have the same name.

Functional Dependency Rules

- If A → (B, C), then A → B and A→ C.
  - This is the **decomposition rule**.
- If A → B and A→ C, then A → (B, C).
  - This is the **union rule**.
- However, if (A,B) → C, then *neither A nor B determines C by itself*.

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

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

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

What Makes Determinant Values Unique?

- A determinant is unique in a relation if and only if, it determines every other column in the relation.
- You cannot find the determinants of all functional dependencies simply by looking for unique values in one column:
  - Data set limitations
  - Must be logically a determinant

Candidate and Primary Keys

- A **candidate key** is a key that determines all of the other 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.

The Entity Integrity Constraint

- The requirement that, in order to function properly, the primary key must have unique data values for every row in the table is know as the **entity integrity constraint**.
- The phrase *unique data values* implies that this column is NOT NULL, and does not allow a NULL value in any row.

Foreign Keys

- A **foreign key** is the primary key of one relation that is placed in another relation to form a link between the relations.
  - A foreign key can be a single column or a composite key.
  - The term refers to the fact that key values are *foreign* to the relation in which they appear as foreign key values.

Foreign Keys

NOTE: The primary keys of the relations are underlined and any foreign keys are in *italics* in the relations below:

DEPARTMENT (DepartmentName, BudgetCode, ManagerName)
EMPLOYEE (EmployeeNumber, EmployeeLastName, EmployeeFirstName, *DepartmentName*)

The Referential Integrity Constraint

- A **referential integrity constraint** is a statement that limits the values of the foreign key to those already existing as primary key values in the corresponding relation:

  SKU in ORDER_ITEM must exist in SKU in SKU_DATA

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)

Foreign Key with a Referential Integrity Constraint

NOTE: The primary key of the relation is <u>underlined</u> and any foreign keys are in *italics* in the relations below:

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

Modification Anomalies

- The EQUIPMENT_REPAIR table before and after an incorrect update operation on AcquisitionCost for EquipmentType = Drill Press:

	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2015-05-05	375.00
2	200	Lathe	4750.00	2100	2015-05-07	255.00
3	100	Drill Press	3500.00	2200	2015-06-19	178.00
4	300	Mill	27300.00	2300	2015-06-19	1875.00
5	100	Drill Press	3500.00	2400	2015-07-05	0.00
6	100	Drill Press	3500.00	2500	2015-08-17	275.00

	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2015-05-05	375.00
2	200	Lathe	4750.00	2100	2015-05-07	255.00
3	100	Drill Press	3500.00	2200	2015-06-19	178.00
4	300	Mill	27300.00	2300	2015-06-19	1875.00
5	100	Drill Press	3500.00	2400	2015-07-05	0.00
6	100	Drill Press	5500.00	2500	2015-08-17	275.00

- Deletion
- Insertion
- Update
- Page 180

Data integrity

3-21

Types of Modification Anomalies

- Deletion anomaly
- Insertion anomaly
- Update anomaly
  - Notice that the EQUIPMENT_REPAIR table duplicates data. For example, the AcquisitionCost of the same item of equipment appears several times. Any table that duplicates data is susceptible to update anomalies. A table that has such inconsistencies is said to have **data integrity problems**.

3-22

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.

3-23

To Key, or Not to Key Here is the Answer!

- There are various opinions about whether or not a relation has to have a designated primary key to be in 1NF.
- In this book, we will define **1NF** as a table that:
  - Meet the set of conditions for a relation, and
  - Has a *defined primary key*

3-24

Normal Forms

- **1NF**—a table that qualifies as a relation is in 1NF.(p.182)
- **2NF**—a relation is in 2NF if all of its nonkey attributes are dependent on *all* of the primary keys. (p.183)
- **3NF**—a relation is in 3NF if it is in 2NF and has no determinants except the primary key. (pp.183-185)
- **Boyce-Codd Normal Form (BCNF)**—a relation is in BCNF if every determinant is a candidate key.(pp.185-186)

"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."

First Normal Form : 1NF

Characteristics of Relations
Rows contain data about an entity.
Columns contain data about attributes of the entities.
All entries in a column are of the same kind.
Each column has a unique name.
Cells of the table hold a single value.
The order of the columns is unimportant.
The order of the rows is unimportant.
No two rows may be identical.

1. satisfy the def. of relation

2. Has a primary key

Second Normal Form : 2NF

1. satisfy 1NF

2. all non-key attributes are determined by the entire primary key.

➔ partial dependency ✖

STUDENT_ACTIVITY

	StudentID	Activity	ActivityFee
1	100	Golf	65.00
2	100	Skiing	200.00
3	200	Skiing	200.00
4	200	Swimming	50.00
5	300	Skiing	200.00
6	300	Swimming	50.00
7	400	Golf	65.00
8	400	Swimming	50.00

(StudentID, Activity) → (ActivityFee)

(Activity) → (ActivityFee) ← partial dependency

STUDENT_ACTIVITY

	StudentID	Activity
1	100	Golf
2	100	Skiing
3	200	Skiing
4	200	Swimming
5	300	Skiing
6	300	Swimming
7	400	Golf
8	400	Swimming

ACTIVITY_FEE

	Activity	ActivityFee
1	Golf	65.00
2	Skiing	200.00
3	Swimming	50.00

- STUDENT_ACTIVITY(StudentID, *Activity*)
- ACTIVITY_FEE(Activity, ActivityFee)

Third Normal Form : 3NF

1. satisfy 2NF

2. no non-key attributes determined by another non-key attribute

➔ transitive dependency

STUDENT_HOUSING

	StudentID	Building	BuildingFee
1	100	Randolph	3200.00
2	200	Ingersoll	3400.00
3	300	Randolph	3200.00
4	400	Randolph	3200.00
5	500	Pitkin	3500.00
6	600	Ingersoll	3400.00
7	700	Ingersoll	3400.00
8	800	Pitkin	3500.00

(StudentID) → (Building)

(Building) → (HouseFee)

STUDENT_HOUSING

	StudentID	Building
1	100	Randolph
2	200	Ingersoll
3	300	Randolph
4	400	Randolph
5	500	Pitkin
6	600	Ingersoll
7	700	Ingersoll
8	800	Pitkin

HOUSING_FEE

	Building	BuildingFee
1	Ingersoll	3400.00
2	Pitkin	3500.00
3	Randolph	3200.00

STUDENT_HOUSING(<u>StudentID</u>,*Building*)

HOUSING_FEE(<u>Building</u>, BuildingFee)

Boyce-Codd Normal Form : BCNF

1. satisfy 3NF

2. every determinant is a candidate key

→overlapping or not a candidate key
Modification anomalies

STUDENT_ADVISOR

	StudentID	Major	AdvisorName
1	100	Math	Cauchy
2	200	Psychology	Jung
3	300	Math	Riemann
4	400	Math	Cauchy
5	500	Psychology	Perls
6	600	English	Austin
7	700	Psychology	Perls
8	700	Math	Riemann
9	800	Math	Cauchy
10	800	Psychology	Jung

STUDENT_ADVISOR(<u>StudentID</u>, <u>Major</u>, AdvisorName)
?
STUDENT_ADVISOR(<u>StudentID</u>, Major, <u>AdvisorName</u>)

STUDENT_ADVISOR

	StudentID	AdvisorName
1	100	Cauchy
2	200	Jung
3	300	Riemann
4	400	Cauchy
5	500	Perls
6	600	Austin
7	700	Perls
8	700	Riemann
9	800	Cauchy
10	800	Jung

ADVISOR_MAJOR

	AdvisorName	Major
1	Austin	English
2	Cauchy	Math
3	Jung	Psychology
4	Perls	Psychology
5	Riemann	Math

STUDENT_ADVISOR(<u>StudentID</u>, <u>*AdvisorName*</u>)

SDVISOR_MAJOR(<u>AdvisorName</u>, Major)

Eliminating Modification Anomalies from Functional Dependencies in Relations: <u>Put All Relations into BCNF</u>

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.

P.187

Putting a Relation into BCNF: CASE1 EQUIPMENT_REPAIR

	ItemNumber	EquipmentType	AcquisitionCost	RepairNumber	RepairDate	RepairCost
1	100	Drill Press	3500.00	2000	2015-05-05	375.00
2	200	Lathe	4750.00	2100	2015-05-07	255.00
3	100	Drill Press	3500.00	2200	2015-06-19	178.00
4	300	Mill	27300.00	2300	2015-06-19	1875.00
5	100	Drill Press	3500.00	2400	2015-07-05	0.00
6	100	Drill Press	3500.00	2500	2015-08-17	275.00

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

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

Where REPAIR.ItemNumber must exist in ITEM.ItemNumber

ItemNumber → (Type, AcquisitionCost)

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

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	2015-05-05	375.00
2	2100	200	2015-05-07	255.00
3	2200	100	2015-06-19	178.00
4	2300	300	2015-06-19	1875.00
5	2400	100	2015-07-05	0.00
6	2500	100	2015-08-17	275.00

Putting a Relation into BCNF:CASE2 SKU_DATA Step-by-Step – 1NF

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

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 (SKU, SKU_Description, *Buyer*)
BUYER (Buyer, 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:CASE2-2 SKU_DATA Straight-to-BCNF

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

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

Example 3

	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

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

(Quantity, Price) → ExtendedPrice

Putting a Relation into BCNF: New Relations

ORDER_ITEM_2

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

Example 4

STUDENT_ACTIVITY

	StudentID	StudentName	Activity	ActivityFee	AmountPaid
1	100	Jones	Golf	65.00	65.00
2	100	Jones	Skiing	200.00	0.00
3	200	Davis	Skiing	200.00	0.00
4	200	Davis	Swimming	50.00	50.00
5	300	Garrett	Skiing	200.00	100.00
6	300	Garrett	Swimming	50.00	50.00
7	400	Jones	Golf	65.00	65.00
8	400	Jones	Swimming	50.00	50.00

StudentID → StudentName **Activity → ActivityFee**

(StudentID, Activity) → AmountPaid

Putting a Relation into BCNF: New Relations

STUDENT

	StudentID	StudentName
1	100	Jones
2	200	Davis
3	300	Garrett
4	400	Jones

ACTIVITY

	Activity	ActivityFee
1	Golf	65.00
2	Skiing	200.00
3	Swimming	50.00

PAYMENT

	StudentID	Activity	ActivityFee
1	100	Golf	65.00
2	100	Skiing	200.00
3	200	Skiing	200.00
4	200	Swimming	50.00
5	300	Skiing	200.00
6	300	Swimming	50.00
7	400	Golf	65.00
8	400	Swimming	50.00

Amount Paid

Example 5

SKU_DATA_3

	SKU	SKU_Description	Department	DeptBudgetCode	Buyer
1	100100	Std. Scuba Tank, Yellow	Water Sports	BC-100	Pete Hansen
2	100200	Std. Scuba Tank, Magenta	Water Sports	BC-100	Pete Hansen
3	101100	Dive Mask, Small Clear	Water Sports	BC-100	Nancy Meyers
4	101200	Dive Mask, Med Clear	Water Sports	BC-100	Nancy Meyers
5	201000	Half-dome Tent	Camping	BC-200	Cindy Lo
6	202000	Half-dome Tent Vestibule	Camping	BC-200	Cindy Lo
7	301000	Light Fly Climbing Harness	Climbing	BC-300	Jerry Martin
8	302000	Locking Carabiner, Oval	Climbing	BC-300	Jerry Martin

DEPARTMENT(<u>Department</u>, DeptBudgetCode)
BUYER2(<u>Buyer</u>, *Department*)
SKU_DATA4(<u>SKU</u>,SKU_Description, *Buyer*)
where SKU_DATA4.Buyer must exist in BUYER2.Buyer
BUYER2.Department must exist in DEPARTMENT.Edpartment

Putting a Relation into BCNF: New Relations

DEPARTMENT

	Department	DeptBudgetCode
1	Camping	BC-200
2	Climbing	BC-300
3	Water Sports	BC-100

BUYER_2

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

SKU_DATA_4

	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

Multivalued Dep

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

Sibling

- A **multivalued depende** n a determinant is matche *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(EnployeeName, EmployeeDegree)

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

EMPLOYEE_SIBLING(EmployeeName, EmployeeSibling)

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

PARTKIT_PART(PartKitName, Part)

Two Multivalued Dependencies

EMPLOYEE_DEGREE_SIBLING

	EmployeeName	EmployeeDegree	EmployeeSibling
1	Chau	BS	Eileen
2	Chau	BS	Jonathan
3	Green	BS	Nikki
4	Green	MS	Nikki
5	Green	PhD	Nikki
6	Jones	AA	Frank
7	Jones	AA	Fred
8	Jones	AA	Sally
9	Jones	BA	Frank
10	Jones	BA	Fred
11	Jones	BA	Sally

?

1. Green earns an MBA?
2. Jones earns an MBA?
3. Chau earns an MBA?

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)**.

That Very Strange Table Again!

PRODUCT_BUYER

	BuyerName	SKU_Managed	CollegeMajor
1	Pete Hansen	100100	Business Administration
2	Pete Hansen	100200	Business Administration
3	Nancy Meyers	101100	Art
4	Nancy Meyers	101100	Info Systems
5	Nancy Meyers	101200	Art
6	Nancy Meyers	101200	Info Systems
7	Cindy Lo	201000	History
8	Cindy Lo	202000	History
9	Jenny Martin	301000	Business Administration
10	Jenny Martin	301000	English Literature
11	Jenny Martin	302000	Business Administration
12	Jenny Martin	302000	English Literature

Now we understand why this is a very strange table.

It has **multivalued dependencies**!

4NF

Use **4NF** to resolve the **multivalued dependencies**!

PRODUCT_BUYER_SKU

	BuyerName	SKU_Managed
1	Cindy Lo	201000
2	Cindy Lo	202000
3	Jenny Martin	301000
4	Jenny Martin	302000
5	Nancy Meyers	101100
6	Nancy Meyers	101200
7	Pete Hansen	100100
8	Pete Hansen	100200

PRODUCT_BUYER_MAJOR

	BuyerName	CollegeMajor
1	Cindy Lo	History
2	Jenny Martin	Business Administration
3	Jenny Martin	English Literature
4	Nancy Meyers	Art
5	Nancy Meyers	Info Systems
6	Pete Hansen	Business Administration

David Kroenke and David Auer

Database Processing

Fundamentals, Design, and Implementation

(14th Edition, Global Edition)

End of Presentation:

Chapter Three