

## Chapter 4 The Relational Model and Normalization



**DATABASE PROCESSING**  
Fundamentals, Design,  
and Implementation, 9/e

## Relations

- Relational DBMS products store data in the form of relations, a special type of table
- A relation is a two-dimensional table that has the following characteristics
  - Rows contain data about an entity
  - Columns contain data about attributes of the entity
  - Cells of the table hold a single value
  - All entries in a column are of the same kind
  - Each column has a unique name
  - The order of the columns is unimportant
  - The order of the rows is unimportant
  - No two rows may be identical
- Although not all tables are relations, the terms table and relation are normally used interchangeably
  - Table/row/column = file/record/field = relation/tuple/attribute

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e  
by David M. Kroenke

Chapter 4/2

## Example: Relation

Figure 4.2 Sample Relation

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	236-9987
200	Mary	Abernathy	Finance	MA@somewhere.com	444-8898
300	Liz	Smathers	Finance	LS@somewhere.com	777-0098
400	Tom	Caruthers	Accounting	TC@somewhere.com	236-9987
500	Tom	Jackson	Production	TJ@somewhere.com	444-9980
600	Eleanore	Calders	Legal	EC@somewhere.com	767-0900
700	Richard	Bandstone	Legal	RB@somewhere.com	767-0011

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e  
by David M. Kroenke

Chapter 4/3

## Example: Tables Not Relations

Figure 4.3a Tables Not Relations — Order of Rows is Important

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	236-9987
200	Mary	Abernathy	Finance	MA@somewhere.com	444-8898
300	Liz	Smathers	Finance	LS@somewhere.com	777-0098
400	Tom	Caruthers	Accounting	TC@somewhere.com	236-9987
					236-9987
					555-7171
500	Tom	Jackson	Production	TJ@somewhere.com	444-9980
600	Eleanore	Calders	Legal	EC@somewhere.com	767-0900
700	Richard	Bandstone	Legal	RB@somewhere.com	767-0900
					767-0011

Figure 4.3b Tables Not Relations — Multiple Entries per Cell

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	236-9987
200	Mary	Abernathy	Finance	MA@somewhere.com	444-8898
300	Liz	Smathers	Finance	LS@somewhere.com	777-0098
400	Tom	Caruthers	Accounting	TC@somewhere.com	236-9987
					Fax: 236-9987
					Home: 555-7171
500	Tom	Jackson	Production	TJ@somewhere.com	444-9980
600	Eleanore	Calders	Legal	EC@somewhere.com	767-0900
					Fax: 236-9987
					Home: 555-7171
700	Richard	Bandstone	Legal	RB@somewhere.com	767-0900

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e  
by David M. Kroenke

Chapter 4/4

## Types of Keys

- A key is one or more columns of a relation that identifies a row
- A unique key identifies a single row; a non-unique key identifies several rows
- Composite key is a key that contains two or more attributes
- A relation has one unique primary key and may also have additional unique keys called candidate keys
- Primary key is used to
  - Represent the table in relationships
  - Organize table storage
  - Generate indexes

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e  
by David M. Kroenke

Chapter 4/5

## Functional Dependencies

- A functional dependency occurs when the value of one (set of) attribute(s) determines the value of a second (set of) attribute(s)
- The attribute on the left side of the functional dependency is called the determinant
  - SID → DormName, Fee
  - (CustomerNumber, ItemNumber, Quantity) → Price
- While a primary key is always a determinant, a determinant is not necessarily a primary key

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e  
by David M. Kroenke

Chapter 4/6

## Modification Anomalies

Anomalies are basically "mistakes"

- If we do not define our relations well, they may be subject to anomalies
- Two common types of anomalies
  - Deletion anomaly
    - Removing one record causes two or more facts to be deleted from the database
  - Insertion anomaly
    - Inserting a new fact may be impossible without a new instance first being added to the database

## Deletion Anomalies

- Consider the following relation:
- If we remove Student 100 we lose the fact that Skiing costs \$200

Figure 4.5 Activity Relation  
ACTIVITY (SID, Activity, Fee)  
Sample Data

SID	Activity	Fee
100	Skiing	200
150	Swimming	50
175	Squash	50
200	Swimming	50

## Insertion Anomalies

If we want to set up a new Activity (Scuba Diving costs \$275) we have to wait until someone signs up for it before we can enter it in the DB

Figure 4.5 Activity Relation  
ACTIVITY (SID, Activity, Fee)  
Sample Data

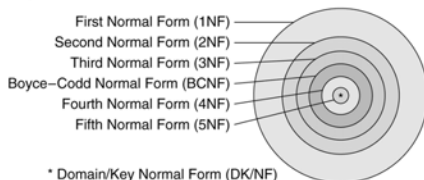
SID	Activity	Fee
100	Skiing	200
150	Swimming	50
175	Squash	50
200	Swimming	50

## Normalization

- Normalization eliminates modification anomalies
  - Deletion anomaly: deletion of a row loses information about two or more entities
  - Insertion anomaly: insertion of a fact in one entity cannot be done until a fact about another entity is added
- Anomalies can be removed by splitting the relation into two or more relations; each with a different, single theme
- However, breaking up a relation may create referential integrity constraints
- Normalization works through classes of relations called normal forms

## Relationship of Normal Forms

Figure 4.7 Relationship of Normal Forms



## Normal Forms

- Any table of data is in 1NF if it meets the definition of a relation
- A relation is in 2NF if all its non-key attributes are dependent on all of the key (no partial dependencies)
  - If a relation has a single attribute key, it is automatically in 2NF
- A relation is in 3NF if it is in 2NF and has no transitive dependencies
- A relation is in BCNF if it is in 3NF and every determinant is a candidate key
- A relation is in fourth normal form if it is in BCNF and has no multi-value dependencies

## 2<sup>nd</sup> Normal Form

- Consider an Activity relation with the key = (SID, Activity)
- But there's a dependency Activity → Fee which is only a partial dependency

**Figure 4.8** ACTIVITIES Relation

ACTIVITIES (SID, Activity, Fee)

SID	Activity	Fee
100	Skiing	200
100	Golf	65
150	Swimming	50
175	Squash	50
175	Swimming	50
200	Swimming	50
200	Golf	65

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/13

## 2<sup>nd</sup> Normal Form (cont.)

- The presence of the partial dependency means that the relation as we defined it is NOT in 2<sup>nd</sup> Normal Form
- If Fee were dependent on ALL of the key fields then there would be no modification anomalies

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/14

## 3<sup>rd</sup> Normal Form

- 2<sup>nd</sup> Normal Form can have other anomalies
- Consider this Housing relation:  
SID → Dorm, and  
Dorm → Fee, thus  
SID → Dorm → Fee  
(transitive dependency)

**Figure 4.9a** Elimination of Transitive Dependency — Relation with Transitive Dependency

HOUSING (SID, Dorm, Fee)  
Key: SID  
Functional dependencies: Dorm → Fee  
SID → Dorm → Fee

SID	Dorm	Fee
100	Randolph	3200
150	Ingersoll	3100
200	Randolph	3200
250	Pitkin	3100
300	Randolph	3200

(a)

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/15

## 3<sup>rd</sup> Normal Form (cont.)

- A relation is in 3<sup>rd</sup> Normal Form if it is in 2<sup>nd</sup> Normal Form and there are no transitive dependencies in the relation
- Eliminating the transitive dependencies can often be done by splitting the relation (X → Y → Z) into two relations (X → Y and Y → Z)
- For our example:

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/16

## 3<sup>rd</sup> Normal Form (cont.)

**Figure 4.9b** Elimination of Transitive Dependency — Relations Eliminating the Transitive Dependency

STU-HOUSING (SID, Dorm)

SID	Dorm
100	Randolph
150	Ingersoll
200	Randolph
250	Pitkin
300	Randolph

BLDG-FEE (Dorm, Fee)

Dorm	Fee
Randolph	3200
Ingersoll	3100
Pitkin	3100

(b)

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/17

## Boyce-Codd Normal Form

- Alas, there are more anomalies possible with relations in 3NF
- Suppose Students can have more than 1 Major; a Major can have more than 1 Adviser; and a Faculty member (Fname) advises in only one Major
  - SID does not determine Major or Adviser
  - But (SID, Fname) → Major

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/18

## But Wait! There's More!!

- Fname → Major, thus Fname is also a determinant (so what, you ask!)
- (Fname) is NOT a candidate key!
  - So there are anomalies:  
Deleting Student 300 removes the fact that Perls advises in Psychology!
  - We cannot add a new Economics adviser!
- The problem is: Not every determinant is a candidate key

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/19

## Example: BCNF

**Figure 4.10a**  
Boyce-Codd Normal Form —  
Relation in Third Normal Form,  
but Not in Boyce-Codd Normal Form

ADVISER (SID, Major, Fname)

Key (candidate): (SID, Fname)

Functional dependencies: Fname → Major

The determinant Fname is not a candidate key!

SID	Major	Fname
100	Math	Cauchy
150	Psychology	Jung
200	Math	Riemann
250	Math	Cauchy
300	Psychology	Perls
300	Math	Riemann

(a)

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/20

## Boyce-Codd Normal Form

- If a relation is in 3<sup>rd</sup> Normal Form and every determinant is a candidate key then it is also in Boyce-Codd Normal Form!
- Relations in BCNF have no anomalies w.r.t. functional dependencies.
- So we're done, right?!

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/21

## Wrong!!

- Still other anomalies are possible
- But let's fix our Adviser relation first by breaking it up into 2 relations
  - But how?
  - (SID, Major) → Fname,
  - ~~(SID, Fname) → Major,~~ Subsumed
  - Fname → Major

Solution: SID → Fname and Fname → Major

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/22

## Example: BCNF

**Figure 4.10b** Boyce-Codd Normal Form —  
Relations in Boyce-Codd Normal Form

STU-ADV (SID, Fname)      ADV-SUBJ (Fname, Subject)

SID	Fname
100	Cauchy
150	Jung
200	Riemann
250	Cauchy
300	Perls
300	Riemann

Fname	Subject
Cauchy	Math
Jung	Psychology
Riemann	Math
Perls	Psychology

(b)

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/23

## 4<sup>th</sup> Normal Form

- Relations in BCNF are still vulnerable to anomalies caused by “multi-valued dependencies”
- To show that Student 100 Swims and plays Tennis while being both a Music and an Accounting major, the following 4 rows must exist in the relation:

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e by David M. Kroenke Chapter 4/24

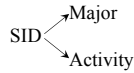
## Example: 4NF

**Figure 4.11** Relation with Multi-Value Dependencies

STUDENT (SID, Major, Activity)

Multi-value dependencies:  $SID \twoheadrightarrow Major$   
 $SID \twoheadrightarrow Activity$

SID	Major	Activity
100	Music	Swimming
100	Accounting	Swimming
100	Music	Tennis
100	Accounting	Tennis
150	Math	Jogging



## Example: 4NF

**Figure 4.13** Elimination of Multi-Value Dependency

STU-MAJOR (SID, Major)

STU-ACT (SID, Activity)

SID	Major
100	Music
100	Accounting
150	Math

SID	Activity
100	Skiing
100	Swimming
100	Tennis
150	Jogging

Now if Student 100 changes a Major or changes an Activity, there's only 1 row to add/delete.

## 5<sup>th</sup> Normal Form

- There are still anomalies possible!
- 5<sup>th</sup> Normal Form has to do with relations that can be divided into subrelations, but cannot be reconstructed into the original form.
- Not important enough to show by example (or easy enough!).

## Domain/Key Normal Form

- First published in 1981 by Fagin
- DK/NF has no modification anomalies; so no higher normal form is needed
- A relation is in DK/NF if every constraint on the relation is a logical consequence of the definition of keys and domains

## Domain/Key Normal Form

- Constraint
  - Broadly defined as any rule governing static values that is precise enough to ascertain whether it is true or false.
    - Edit rules, constraints within relations, constraints between relations, functional dependencies are all Constraints
    - Time dependent constraints are about the only thing that can be ruled out.

## Domain/Key Normal Form

- Key
  - A unique identifier of a row
- Domain
  - A named set of possible attribute values
- A relation is in DK/NF if enforcing key and domain restrictions causes all of the constraints to be satisfied

## Domain/Key Normal Form

- Significance
  - Since modification anomalies cannot occur, the DBMS can prevent them just by enforcing key and domain constraints
- Difficulty
  - No algorithm is known for putting a relation into DK/NF.

## Example: DK/NF (pg. 134)

Figure 4.14 Example 1 of DK/NF

```
STUDENT (SID, GradeLevel, Dorm, Fee)
Key: SID
Constraints: Dorm → Fee
             SID must not begin with digit 1
```

## Example: DK/NF

Figure 4.15 Domain Key Definition for Example 1

Domain Definitions:

Attribute	Domain Name	Values
SID	StudentID	4 decimal digits, first digit not 1
GradeLevel	StudentYear	{'FR', 'SO', 'JR', 'SN', 'GR'}
Dorm	BuildingNames	Char(4)
Fee	StudentFees	Any currency value

Relation and Key Definitions:

STUDENT (SID, GradeLevel, Dorm)

BLDG-FEE (Dorm, Fee)

## Example: DK/NF (pg. 135)

Figure 4.16 Example 2 of DK/NF

```
PROFESSOR (FID, Fname, Class, SID, Sname)
Constraints: FID → Fname
            Fname → FID
            FID → → Class | SID
            Fname → → Class | SID
            SID → FID
            SID → Fname
            SID → Sname
            FID must start with 1; SID must not start with 1
```

## Example: DK/NF

Figure 4.17 Domain Key Definition for Example 2

Domain Definitions:

Attribute	Domain Name	Values
FID	FacultyID	4 decimal digits, first digit is 1
Fname	PersonNames	Char(50)
Class	ClassNames	Char(10); values (list of valid course names)
SID	StudentID	4 decimal digits, first digit is not 1
Sname	PersonNames	Char(50)

Relation and Key Definitions:

FACULTY (FID, Fname)

Candidate key: Fname

PREPARATION (Fname, Class)

STUDENT (SID, Sname, Fname)

## Example: DK/NF (pg. 136)

Figure 4.18 Example 3 of DK/NF

```
STU-ADVISER (SID, Sname, FID, Fname, GradFacultyStatus)
Key: SID
Constraints: FID → Fname
            Fname → FID
            FID and Fname → GradFacultyStatus
            Only graduate faculty can advise graduate students
            FID begins with 1
            SID must not begin with 1
            SID of graduate student begins with 9
            GradFacultyStatus = { 0 for undergraduate faculty
                                1 for graduate faculty
```

## Example: DK/NF

Figure 4.19 Domain Key Definition for Example 3

Domain Definitions:

Attribute	Domain Name	Values
FID	FacultyID	4 decimal digits, first digit is 1
Fname	PersonNames	Char(50)
GradFacultyStatus	FacultyStatus	Values {0, 1}
GSID	GradStudentID	4 decimal digits, first digit is 9
UGSIDID	UnderGradStudentID	4 decimal digits, first digit is not 1 and not 9
Sname	PersonNames	Char(50)
Gname	PersonNames	Values {FACULTY.Fname where GradFacultyStatus = 1 }

Relation and Key Definitions:

FACULTY (FID, Fname, GradFacultyStatus)

Candidate key: Fname

G\_ADV (GSID, Sname, Gname)

UG\_ADV (UGSID, Sname, Fname)

## The Synthesis of Relations

- Given a set of attributes with certain functional dependencies, what relations should we form?
- Example: A and B are two attributes
  - If  $A \rightarrow B$  and  $B \rightarrow A$ 
    - A and B have a one-to-one attribute relationship
  - If  $A \rightarrow B$ , but  $B \not\rightarrow A$ 
    - A and B have a many-to-one attribute relationship
  - If  $A \not\rightarrow B$  and  $B \not\rightarrow A$ 
    - A and B have a many-to-many attribute relationship

## Types of Attribute Relationship

Figure 4.21 Summary of Three Types of Attribute Relationships

Relation Definition*	Type of Attribute Relationship		
	One to One	Many to One	Many to Many
R(A,B)		S(C,D)	T(E,F)
Dependencies	$A \rightarrow B$ $B \rightarrow A$	$C \rightarrow D$ $D \rightarrow C$	$E \rightarrow F$ $F \rightarrow E$
Key	Either A or B	C	(E,F)
Rule for Adding Another Attribute	Either $A \rightarrow C$ or $B \rightarrow C$	$C \rightarrow E$	$(E,F) \rightarrow G$

\* The letters used in these relation definitions match those used in Figure 4-22.

## One-to-One Attribute Relationships

- Attributes that have a one-to-one relationship must occur together in at least one relation
- Call the relation R and the attributes A and B:
  - Either A or B must be the key of R
  - An attribute can be added to R if it is functionally determined by A or B
  - An attribute that is not functionally determined by A or B cannot be added to R
  - A and B must occur together in R, but should not occur together in other relations
  - Either A or B should be consistently used to represent the pair in relations other than R

## Many-to-One Attribute Relationships

- Attributes that have a many-to-one relationship can exist in a relation together
- Assume C determines D in relation S
  - C must be the key of S
  - An attribute can be added to S if it is determined by C
  - An attribute that is not determined by C cannot be added to S

## Many-to-Many Attribute Relationships

- Attributes that have a many-to-many relationship can exist in a relation together
- Assume attributes E and F reside together in relation T
  - The key of T must be (E, F)
  - An attribute can be added to T if it is determined by the combination (E, F)
  - An attribute may not be added to T if it is not determined by the combination (E, F)
  - If adding a new attribute, G, expands the key to (E, F, G), then the theme of the relation has been changed
    - Either G does not belong in T or the name of T must be changed to reflect the new theme



## De-normalized Designs

- When a normalized design is unnatural, awkward, or results in unacceptable performance, a de-normalized design is preferred
- Example
  - Normalized relation
    - CUSTOMER (CustNumber, CustName, Zip)
    - CODES (Zip, City, State)
  - De-Normalized relations
    - CUSTOMER (CustNumber, CustName, City, State, Zip)

Copyright © 2004 Database Processing: Fundamentals, Design and Implementation, 9/e  
by David M. Kroenke

Chapter 4/43

## Chapter 4 The Relational Model and Normalization

---



**DATABASE PROCESSING**  
Fundamentals, Design,  
and Implementation, 9/e