

NET3000

Database Concepts and SQL

Material prepared by: Kambiz Ghazinour

Week 4

Normalization

September 28, 2014

What we are going to cover

- Data Normalization
- Data Decomposition
- Primary and Foreign Keys
- Normal Forms
- Normalization versus Denormalization

Normalization

- Is the process of organizing data into related tables
- A set of rules is applied to the data model to achieve a normal form
- *Data decomposition* is often used to reduce an attribute to an atomic (scalar) level
 - E.g. name = first name AND last name

Goals of Normalization

- Eliminate redundant data
- Keep entities as independent of one another as possible
- Eliminating data redundancy is based upon the theory of *functional dependencies*

Functional Dependency

- A functional dependency means using the known value of one column, the corresponding value of another column can always be uniquely determined.
- Examples,
 - $au_id \twoheadrightarrow au_lname$
 - (read as “au_lname is functionally dependent on au_id”)
 - $ISBN \twoheadrightarrow book_title$
 - $MAC_address \twoheadrightarrow network_appliance$

A Typical Spreadsheet File

Emp No	Employee Name	Time Card No	Time Card Date	Dept No	Dept Name
10	Thomas Arquette	106	11/02/2002	20	Marketing
10	Thomas Arquette	106	11/02/2002	20	Marketing
10	Thomas Arquette	106	11/02/2002	20	Marketing
10	Thomas Arquette	115	11/09/2002	20	Marketing
99	Janice Smitty			10	Accounting
500	Alan Cook	107	11/02/2002	50	Shipping
500	Alan Cook	107	11/02/2002	50	Shipping
700	Ernest Gold	108	11/02/2002	50	Shipping
700	Ernest Gold	116	11/09/2002	50	Shipping
700	Ernest Gold	116	11/09/2002	50	Shipping

Employee, Department, and Time Card Data in Three Tables

Table: Employees

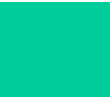
EmpNo	EmpFirstName	EmpLastName	DeptNo
10	Thomas	Arquette	20
500	Alan	Cook	50
700	Ernest	Gold	50
99	Janice	Smitty	10

Table: Departments

DeptNo	DeptName
10	Accounting
20	Marketing
50	Shipping

Table: Time Card Data

TimeCardNo	EmpNo	TimeCardDate
106	10	11/02/2002
107	500	11/02/2002
108	700	11/02/2002
115	10	11/09/2002
116	700	11/09/2002

 Primary Key

Normal Forms (NFs)

- Measures the degree of normalization
- 5 normal forms
- Each level of normalization addresses a specific problem
- Expectation for this course: 3NF

Remember: Primary Key (PK)

- A *primary key* is a special attribute(s) that uniquely defines one and only one tuple within an entity.
- In other words, no 2 tuples can have the same value for a given PK.
- An entity can have only one PK.

Remember: Composite Primary Key

- True, an entity can have only one PK.
- HOWEVER, the PK can include more than one attribute.
- This is called a *composite PK*

1NF

Eliminate Repeating Groups

- Make a separate entity for each set of related attributes, with a unique identifier for each entity.
- Table format.
- No repeating groups.
- PK identified.

Conversion to 1NF

1. Eliminate the Repeating Groups
2. Identify the Primary Key
3. Identify All Dependencies

1NF

Satisfied Conditions

1. Every unit of data is represented within scalar (atomic) attributes.
2. All data must be represented in unique attributes.
3. All data must be represented within unique tuples.

1NF

Summary

1. Uniquely identify each tuple.
2. Eliminate repeating groups of data
 - no array attributes

The following example is based on “Relational Databases – a simplified account” by Robert Timmer-Arends and some slides from Mark Kelly

Take the following table.

StudentID is the primary key.

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
					Maths	\$50	A
					Info Tech	\$100	B+

Is it 1NF?

No. There are repeating groups (subject, subjectcost, grade)

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
					Maths	\$50	A
					Info Tech	\$100	B+

How can you make it 1NF?

Create new rows so each cell contains only one value

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
					Maths	\$50	A
					Info Tech	\$100	B+




StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

But now look – is the *studentID* primary key still valid?

No – the studentID no longer uniquely identifies each row

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+



You now need to declare *studentID* and *subject* **together** to uniquely identify each row.

So the new **key** is StudentID *and* Subject.

So. We now have 1NF.

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

Is it 2NF?

2NF

- usually used in tables with a multiple-field primary key (composite key)
- each non-key field relates to the entire primary key
- any field that does not relate to the primary key is placed in a separate table
- MAIN POINT –
 - eliminate redundant data in a table
 - Create separate tables for sets of values that apply to multiple records

2NF

Eliminate Redundant Data

- << pre-condition: must be in 1NF >>
- If an attribute depends on only part of a multi-valued key, remove it to a separate entity.
- No Partial dependencies.

Studentname and **address** are dependent on
studentID (which is part of the key)
This is good.

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

But they are **not**
dependent on *Subject*
(the *other* part of the key)

And 2NF requires...

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

All non-key fields are
dependent on the
ENTIRE key (studentID
+ subject)

So it's not 2NF

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

How can we fix it?

Make new tables

- Make a new table for each primary key field
- Give each new table its own primary key
- Move columns from the original table to the new table that matches their primary key...

Step 1

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

STUDENT TABLE (key = StudentID)

Step 2

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Sub

Step 3

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

RESULTS TABLE (key = StudentID+Subject)

Step 3

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
19594332X	Mary Watson	10 Charles Street	Bob	Red	Maths	\$50	A
19594332X	Mary Watson	10 Charles Street	Bob	Red	Info Tech	\$100	B+

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

RESULTS TABLE (key = StudentID + Subject)

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

Step 4 - relationships

STUDENT TABLE (key = StudentID)

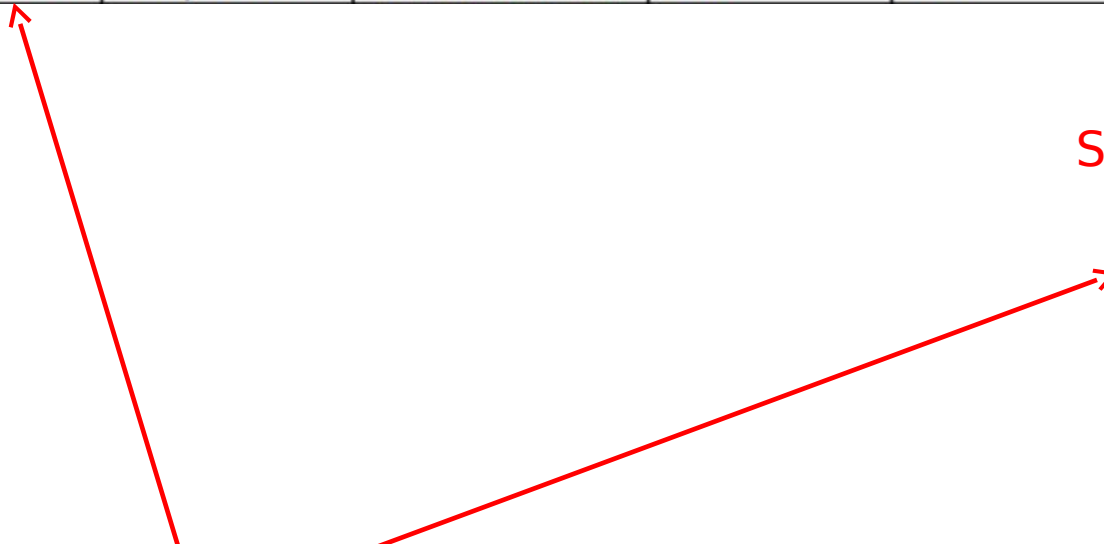
StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)



Step 4 - cardinality

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

Each student can only appear ONCE in the student table

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

Step 4 - cardinality

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

1

Each subject can only appear ONCE in the subjects table

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

Step 4 - cardinality

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

1

A subject can be listed
MANY times in the
results table (for
different students)

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)

Step 4 - cardinality

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

A student can be listed
MANY times in the
results table (for
different subjects)

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)

1

1

∞

A 2NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

SubjectCost is only dependent on the primary key, *Subject* 😊

A 2NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

Grade is only dependent on the primary key (*studentID* + *subject*)



A 2NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

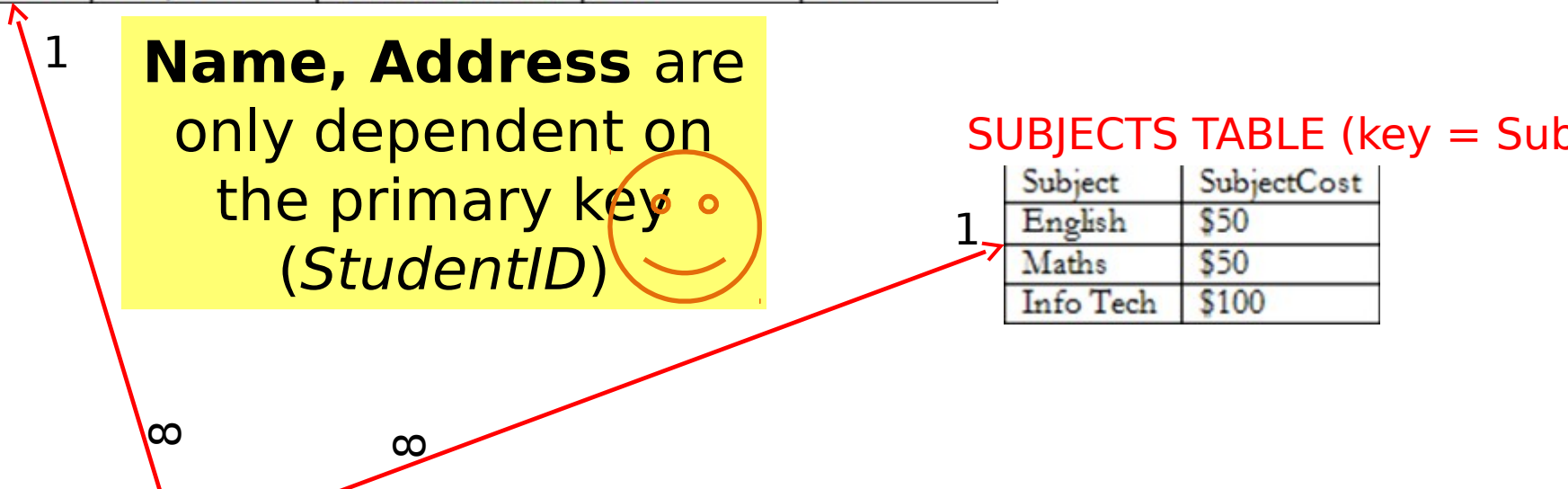
Name, Address are
only dependent on
the primary key
(*StudentID*)

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)



STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

**So it is
2NF!**

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

1

∞

∞

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)

But is it
3NF?

3NF

- usually used in tables with a single-field primary key
- records do not depend on anything other than a table's primary key
- each non-key field is a fact about the key
- Values in a record that are not part of that record's key do not belong in the table. In general, any time the contents of a group of fields may apply to more than a single record in the table, consider placing those fields in a separate table.

3NF

Eliminate Attributes Not Dependent on the PK

- << pre-condition :: must be in 2NF >>
- If attributes do not add to the description of the identifier, remove them to another entity.
- No transitive dependencies.

Definition: A transitive dependency occurs when there is an indirect relationship that causes a functional dependency.

Example- $A \rightarrow C$ is a transitive dependency only when $A \rightarrow B$ and $B \rightarrow C$ are also true.

3NF

Satisfied Conditions

- All non-key attributes must not depend on any other non-key attributes.
- Also expressed as:
 - All transitive dependencies must be removed
- A transitive dependency exists when:
 $\text{attribute_B} \twoheadrightarrow \text{attribute_A}$
 $\text{PK} \twoheadrightarrow \text{attribute_B}$
- Solution: move non-dependent attribute to a new entity.
- Example,
 - ZIP code

A 3NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

Oh
oh...
What?

1

∞

∞

1

A 3NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

HouseName is
dependent on
both
***StudentID +
HouseColour***

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

1

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)

A 3NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

Or
HouseColour
is dependent
on both
StudentID +
HouseName

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

1

A 3NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

*But either way,
non-key fields are
dependent on
MORE THAN THE
PRIMARY KEY
(studentID)*

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

1

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)

A 3NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

*And 3NF says
that non-key
fields must
depend on
**nothing but the
key***

∞

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key =
StudentID+Subject)

SUBJECTS TABLE (key = Subject)

1

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

A 3NF check

STUDENT TABLE (key = StudentID)

StudentID	StudentName	Address	HouseName	HouseColor
19594332X	Mary Watson	10 Charles Street	Bob	Red

1

WHAT DO WE DO?

SUBJECTS TABLE (key = Sub

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

1



oo

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

Again, carve off the offending fields

StudentTable

StudentID	StudentName	Address	HouseName
19594332X	Mary Watson	10 Charles Street	Bob

Primary key: StudentID

1

∞

∞

1

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

SUBJECTS TABLE (key = Subject)

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

RESULTS TABLE (key = StudentID+Subject)

A 3NF fix

StudentTable

StudentID	StudentName	Address
19594332X	Mary Watson	10 Charles Street

Primary key: StudentID

HouseTable

HouseName	HouseColor
Bob	Red

Primary key: HouseName

1

∞

∞

1

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

SUBJECTS TABLE (key = Su

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

RESULTS TABLE (key =
StudentID+Subject)

A 3NF fix

StudentTable

StudentID	StudentName	Address	HouseName
19594332X	Mary Watson	10 Charles Street	Bob

Primary key: StudentID

∞

1

HouseTable

HouseName	HouseColor
Bob	Red

Primary key: HouseName

1

∞

∞

1

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

SUBJECTS TABLE (key = Su

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

RESULTS TABLE (key =
StudentID+Subject)

A 3NF win!

StudentTable

StudentID	StudentName	Address	HouseName
19594332X	Mary Watson	10 Charles Street	Bob

Primary key: StudentID

∞

HouseTable

HouseName	HouseColor
Bob	Red

Primary key: HouseName

1

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

∞

∞

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

SUBJECTS TABLE (key = Subject)

RESULTS TABLE (key = StudentID+Subject)

Or...

The Reveal

Before...

StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English	\$50	B
					Maths	\$50	A
					Info Tech	\$100	B+

After...

StudentTable

StudentID	StudentName	Address	HouseName
19594332X	Mary Watson	10 Charles Street	Bob

Primary key: StudentID

1

HouseTable	
HouseName	HouseColor
Bob	Red

Primary key: HouseName

1
∞

StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

∞

1

Subject	SubjectCost
English	\$50
Maths	\$50
Info Tech	\$100

SUBJECTS TABLE (key = Subject)

Why Normalize Your Data?

1. Redundant data is eliminated
 - less storage is required
 - may result in faster table scans and searches (since less physical data has to be processed)
 - easier to maintain referential integrity
2. Prevention of data anomalies
 - An anomaly occurs when two different pieces of data can be found describing the same attribute

Disadvantages to Normalization

- Normalization usually results in data being stored in multiple tables.
- Therefore, a join operation is required.
- Joins are very expensive from an I/O point of view.

Summary

Normal Form	Characteristic
1NF	Table format; no repeating groups and PK identified
2NF	1NF and no partial dependencies
3NF	2NF and no transitive dependencies