Normalization

Lecture Overview

- **What is Normalisation**
  - Purpose
  - Insert – Update – Delete Anomalies
  - Functional Dependency
  - Transitive Dependency
- **The Process of Normalisation**
  - 1NF
  - 2NF
  - 3NF
- **Normalisation Example**

Normalisation
An Introduction

What is Normalisation?

- E-R is a TOP-DOWN methodology
- Normalisation is effectively a BOTTOM-UP methodology
  - Highly formal
  - Looks at relationships in data
  - Tries to eliminate "undesirable dependencies" in data
  - Leads to high degrees of "data independence"

Purpose of Normalisation

- Normalisation is a technique for producing a set of suitable relations
- Characteristics of relations include:
  - the minimal number of attributes necessary to support the data requirements
  - minimal redundancy with each attribute represented
  - exception of attributes that form all or part of foreign keys.

Data Redundancy

- Major aim of relational database design is to group attributes into relations to minimize data redundancy.
- Potential benefits for implemented database include:
  - Updates to the data stored in the database are achieved with a minimal number of operations thus reducing the opportunities for data inconsistencies.
  - Reduction in the file storage space required by the base relations thus minimizing costs

Update Anomalies

- Relations that contain redundant information may potentially suffer from update anomalies.
- Types of update anomalies include
  - Insertion
  - Deletion
  - Modification

Update Anomaly

- An Update Anomaly exists when one or more instances of duplicated data is updated, but not all.
  - For example, consider Jones moving address - you need to update all instances of Jones's address.

StudentNum	CourseNum	Name	Address	Course
S21	9201	Jones	Edinburgh	Accounts
S21	9267	Jones	Edinburgh	Accounts
S24	9267	Smith	Glasgow	Physics
S30	9201	Richards	Manchester	Computing
S30	9322	Richards	Manchester	Maths

Delete Anomaly

- A Delete Anomaly exists when certain attributes are lost because of the deletion of other attributes.
  - For example, consider what happens if Student S30 is the last student to leave the course - All information about the course is lost.

StudentNum	CourseNum	Name	Address	Course
S21	9201	Jones	Edinburgh	Accounts
S21	9267	Jones	Edinburgh	Accounts
S24	9267	Smith	Glasgow	Physics
S30	9201	Richards	Manchester	Computing
S30	9322	Richards	Manchester	Maths

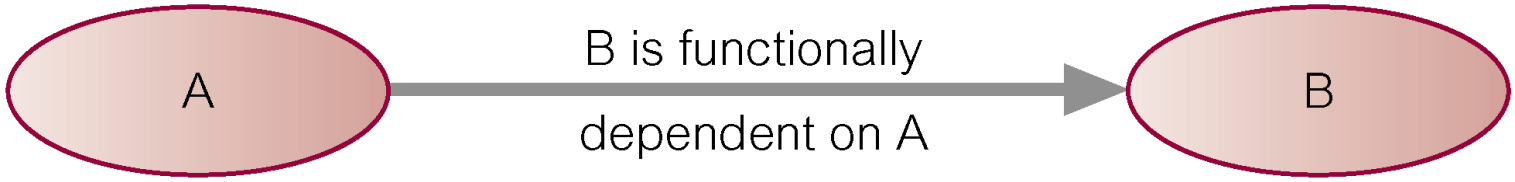
Insert Anomaly

- An Insert Anomaly occurs when certain attributes cannot be inserted into the database without the presence of other attributes.
  - For example this is the converse of delete anomaly - we can't add a new course unless we have at least one student enrolled on the course and vise-versa

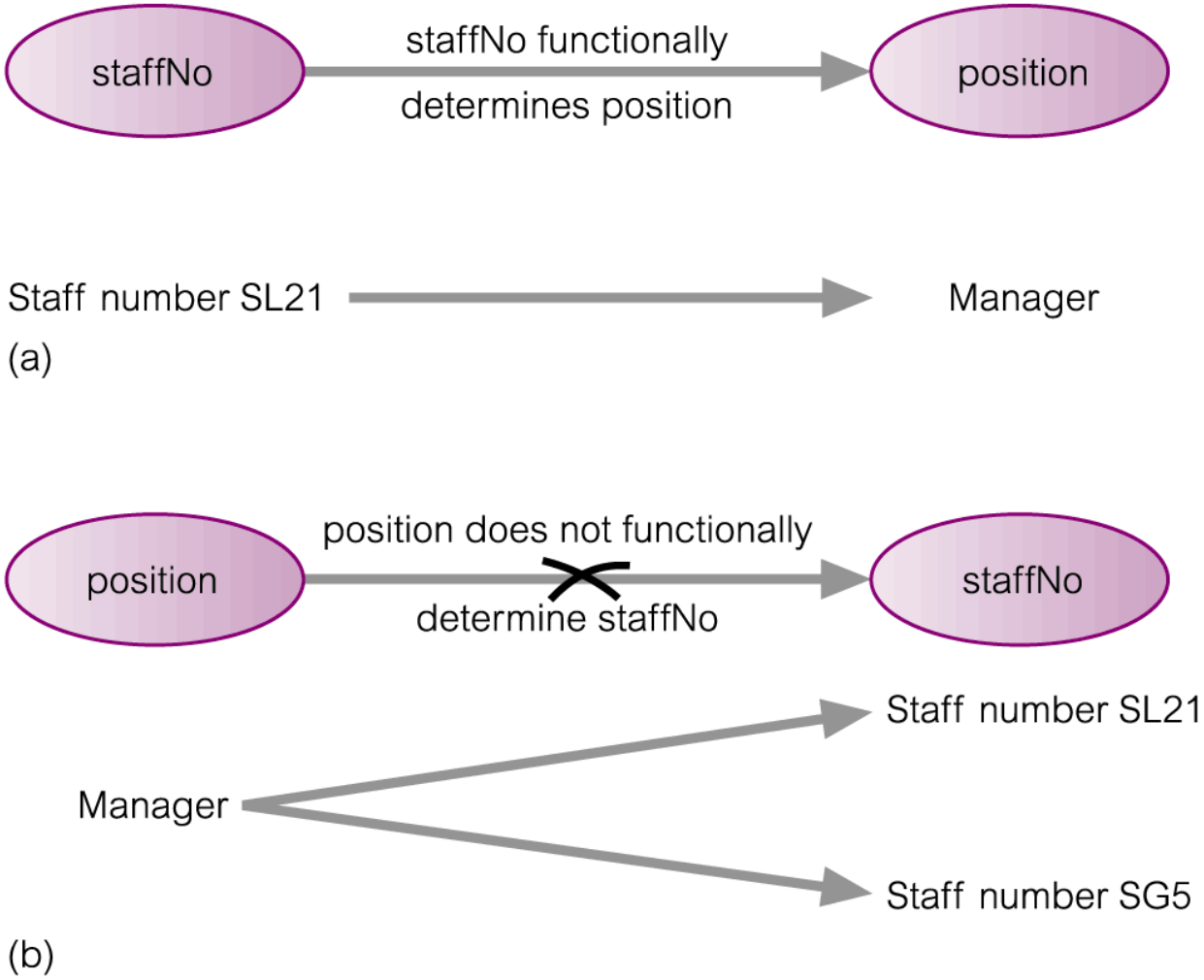
StudentNum	CourseNum	Name	Address	Course
S21	9201	Jones	Edinburgh	Accounts
S21	9267	Jones	Edinburgh	Accounts
S24	9267	Smith	Glasgow	Physics
S30	9201	Richards	Manchester	Computing
S30	9322	Richards	Manchester	Maths
S31	--------	Hobbs	London	--------

Functional Dependencies

- Important concept associated with normalization.
- Functional dependency describes relationship between attributes.
- For example, if A and B are attributes of relation R, B is functionally dependent on A (denoted A -> B), if each value of A in R is associated with exactly one value of B in R.
- The **determinant** of a functional dependency refers to the attribute or group of attributes on the left-hand side of the arrow.



Functional Dependency Example



Transitive Dependency

- Important to recognize a transitive dependency because its existence in a relation can potentially cause update anomalies.
- Transitive dependency describes a condition where A, B, and C are attributes of a relation such that if A → B and B → C, then C is transitively dependent on A via B (provided that A is not functionally dependent on B or C).
- A functional dependency is said to be transitive if it is indirectly formed by two functional dependencies

Transitive Dependency Example

- {Book} → {Author}
  (if we know the book, we know the author name)

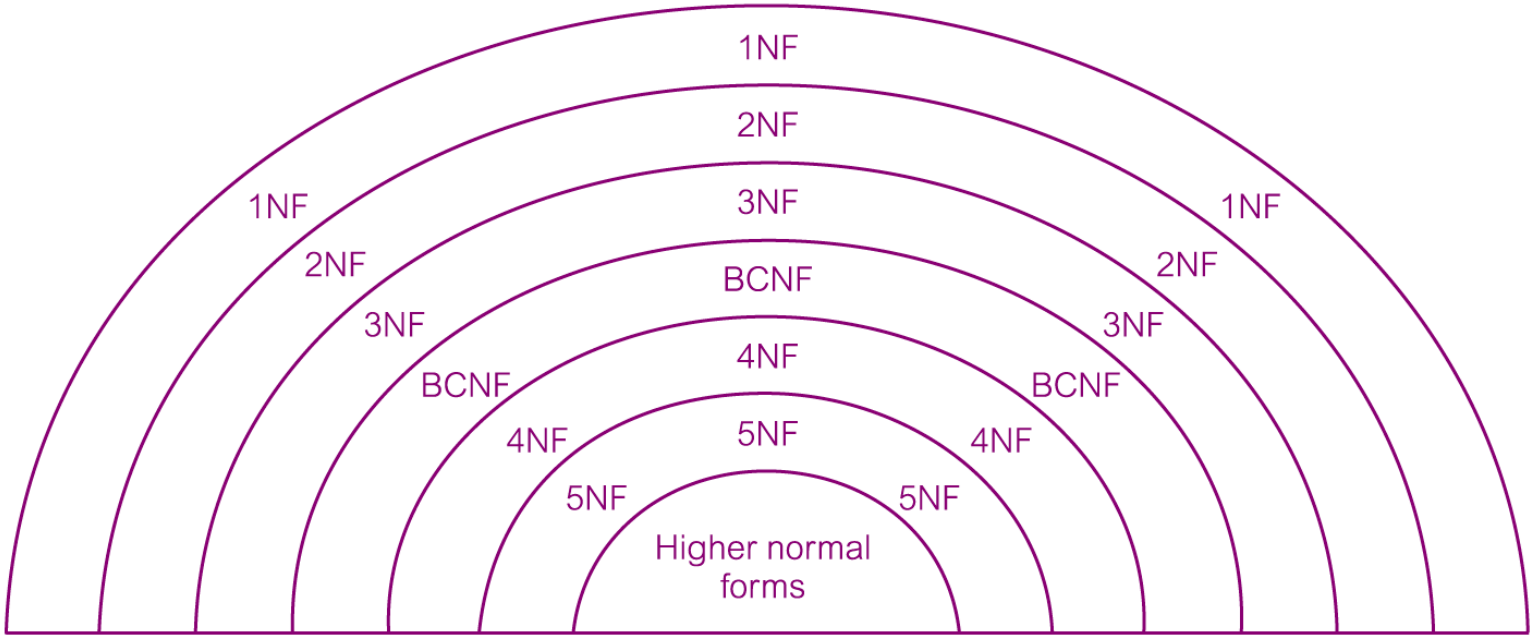
  But in the following table
  {Author} → {Author_age}
  Therefore as per the rule of transitive dependency:
- {Book} → {Author_age} should hold, that makes sense because if we know the book name we can know the author's age

Book	Author	Author_age
Game of Thrones	George R. R. Martin	66
Harry Potter	J. K. Rowling	49
Dying of the Light	George R. R. Martin	66

The Process of Normalisation

The Process of Normalisation

- As normalisation proceeds, the relations become progressively more restricted (stronger) in format and also less vulnerable to update anomalies.



Basic Definitions

- Student(SSN, STNO, Name, Address, Salary)
  - Superkeys
    - {SSN,Name}/{SSN,STNO,Name,Address,Salary}
  - Candidate keys
    - {SSN, STNO};
  - Prime Attribute:
    - SSN and STNO
  - Nonprime Attributes:
    - {Name, Address, Salary}

Unnormalised Form (UNF)

- A table that contains one or more repeating groups.
- To create an unnormalised table
  - Transform the data from the information source (e.g. form) into table format with columns and rows.

First Normal Form (1NF)

- A relation in which the intersection of each row and column contains one and only one value.
- To transform the unnormalised table to First Normal Form, we identify and remove repeating groups within the table.
- A repeating group is an attribute, or group of attributes, within a table that occurs with multiple values for a single occurrence of the nominated key attribute(s) for that table.

UNF to 1NF

- Remove the repeating group by
  - Entering appropriate data into the empty columns of rows containing the repeating data („flattening“ the table).Or by
  - Placing the repeating data along with a copy of the original key attribute(s) into a separate relation.

UNF to 1NF Example

ClientRental

clientNo	cName	propertyNo	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	John Kay	PG4	6 Lawrence St, Glasgow	1-Jul-07	31-Aug-08	350	CO40	Tina Murphy
		PG16	5 Novar Dr, Glasgow	1-Sep-08	1-Sep-09	450	CO93	Tony Shaw
CR56	Aline Stewart	PG4	6 Lawrence St, Glasgow	1-Sep-06	10-June-07	350	CO40	Tina Murphy
		PG36	2 Manor Rd, Glasgow	10-Oct-07	1-Dec-08	375	CO93	Tony Shaw
		PG16	5 Novar Dr, Glasgow	1-Nov-09	10-Aug-10	450	CO93	Tony Shaw

- The structure of the repeating group is:
  Repeating Group = (propertyNo, pAddress, rentStart, rentFinish, rent, ownerNo, oName)
- Multiple values at the intersection of certain rows and columns. For example, there are two values for propertyNo(PG4 and PG16) for the client named John Kay.

Example 1NF

ClientRental

clientNo	propertyNo	cName	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	PG4	John Kay	6 Lawrence St, Glasgow	1-Jul-07	31-Aug-08	350	CO40	Tina Murphy
CR76	PG16	John Kay	5 Novar Dr, Glasgow	1-Sep-08	1-Sep-09	450	CO93	Tony Shaw
CR56	PG4	Aline Stewart	6 Lawrence St, Glasgow	1-Sep-06	10-Jun-07	350	CO40	Tina Murphy
CR56	PG36	Aline Stewart	2 Manor Rd, Glasgow	10-Oct-07	1-Dec-08	375	CO93	Tony Shaw
CR56	PG16	Aline Stewart	5 Novar Dr, Glasgow	1-Nov-09	10-Aug-10	450	CO93	Tony Shaw

1NF (First Normal Form)

- A relation schema R is in 1NF if every attribute of R takes only single and atomic values.

Empno	Ename	Job	Educ	Deptno
33	Jones	Pres	BS EE, MS EE, PhD Comp Sci	3
324	Chu	VP	BS EE, MBA	3
88	Kumar	Sales	BS EE, MA Comm	4
65	Yu	Quality Contr.	BS CS, MS CS, PhD CS	5

Second Normal Form (2NF)

- Based on the concept of full functional dependency.
- Full functional dependency indicates that if
  - A and B are attributes of a relation,
  - B is fully dependent on A if B is functionally dependent on A but not on any proper subset of A.
- A relation in 2NF **must be** in 1NF and every non-primary-key attribute is fully functionally dependent on the primary key.

1NF to 2NF

- Identify the primary key for the 1NF relation.
- Identify the functional dependencies in the relation.
- If partial dependencies exist on the primary key remove them by placing then in a new relation along with a copy of their determinant.

2nd Normal Form

- Satisfies 1st Normal Form conditions
- Has full functional dependency (i.e whole key essential for determining all non-key attributes - automatic if single-attribute primary key)
- Elements which are not fully dependent are separated off into another table with the relevant part of the key
- **NB only necessary to check if you have a multi-part key!**

ID	First Name	Last Name	Marital Status	Course Title	Fee	Qualification	Grade
1	Kevin	Drumm	Single	Computer Science	£2000	Advanced Level	A
1	Kevin	Drumm	Single	Mathematics	£2500	Advanced Level	B
1	Kevin	Drumm	Single	Physics	£1800	Advanced Level	C
2	Murvin	Drake	Single	Physics	£1800	Advanced Level	B
2	Murvin	Drake	Single	Chemistry	£1800	Advanced Level	C
3	John	Jones	Single	Music	£1200	Diploma	C
4	Sally-Jane	Jones	Single	Biology	£1000	Certificate	A
4	Sally-Jane	Jones	Single	Economics	£1500	Diploma	B
5	David	Smith	Married	Mathematics	£2500	Advanced Level	C
5	David	Smith	Married	Physics	£1800	Advanced Level	D

Students

ID	First Name	Last Name	Marital Status
1	Kevin	Drumm	Single
2	Murvin	Drake	Single
3	John	Jones	Single
4	Sally-Jane	Jones	Single
5	David	Smith	Married

StudentCourse

ID	Course Title	Grade
1	Computer Science	A
1	Mathematics	B
1	Physics	C
2	Physics	B
2	Chemistry	C
3	Music	C
4	Biology	A
4	Economics	B
5	Mathematics	C
5	Physics	D

Courses

Course Title	Fee	Qualification
Computer Science	£2000	Advanced Level
Mathematics	£2500	Advanced Level
Physics	£1800	Advanced Level
Chemistry	£1800	Advanced Level
Music	£1200	Diploma
Biology	£1000	Certificate
Economics	£1500	Diploma

2nd Normal Form Example

ClientRental

clientNo	propertyNo	cName	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	PG4	John Kay	6 Lawrence St, Glasgow	1-Jul-07	31-Aug-08	350	CO40	Tina Murphy
CR76	PG16	John Kay	5 Novar Dr, Glasgow	1-Sep-08	1-Sep-09	450	CO93	Tony Shaw
CR56	PG4	Aline Stewart	6 Lawrence St, Glasgow	1-Sep-06	10-Jun-07	350	CO40	Tina Murphy
CR56	PG36	Aline Stewart	2 Manor Rd, Glasgow	10-Oct-07	1-Dec-08	375	CO93	Tony Shaw
CR56	PG16	Aline Stewart	5 Novar Dr, Glasgow	1-Nov-09	10-Aug-10	450	CO93	Tony Shaw

fd1	clientNo, propertyNo → rentStart, rentFinish	(Primary key)
fd2	clientNo → cName	(Partial dependency)
fd3	propertyNo → pAddress, rent, ownerNo, oName	(Partial dependency)
fd4	ownerNo → oName	(Transitive dependency)
fd5	clientNo, rentStart → propertyNo, pAddress, rentFinish, rent, ownerNo, oName	(Candidate key)
fd6	propertyNo, rentStart → clientNo, cName, rentFinish	(Candidate key)

1NF to 2NF

- This results in the creation of three new relations called Client, Rental, and PropertyOwner,
- These three relations are in second normal form, as every non-primary-key attribute is fully functionally dependent on the primary key of the relation.

Client

clientNo	cName
CR76	John Kay
CR56	Aline Stewart

Rental

clientNo	propertyNo	rentStart	rentFinish
CR76	PG4	1-Jul-07	31-Aug-08
CR76	PG16	1-Sep-08	1-Sep-07
CR56	PG4	1-Sep-06	10-Jun-07
CR56	PG36	10-Oct-07	1-Dec-08
CR56	PG16	1-Nov-09	10-Aug-10

PropertyOwner

propertyNo	pAddress	rent	ownerNo	oName
PG4	6 Lawrence St, Glasgow	350	CO40	Tina Murphy
PG16	5 Novar Dr, Glasgow	450	CO93	Tony Shaw
PG36	2 Manor Rd, Glasgow	375	CO93	Tony Shaw

Second Normal Form (2NF)

- Emp-Proj(SSN, Pnumber, Hours, Ename, Pname, Plocation)
  - FD1: {SSN, Pnumber} → Hours (FD)
  - FD2: SSN → Ename (PD)
  - FD3: Pnumber → {Pname, Plocation} (PD)
  - Because of FD2 and FD3 Emp-Proj is not in 2NF
- 2NF Normalization
  - Emp(SSN, Ename)
  - Proj(Pnumber, Pname, Plocation)
  - Work(SSN, Pnumber, Hours)

Second Normal Form (2NF)

- Consider
  - Bank-Loans (Bank_name,Loan_no, Assets, Headquarter, Customer_name, Amount)
  - FD1: Bank_name → {Assets, Headquarter}
  - FD2: {Bank_name, Loan_no} → {Customer_name, Amount}
  - Because if FD1, Bank-Loans is not in 2NF.
- 2NF Normalization
  - Banks(Bank_name, Assets, Headquarter)
  - Loans(Bank_name, Loan_no, Customer_name, Amount)

Second Normal Form (2NF)

- 2NF is not good enough:
  - A relation schema in 2NF can still have serious redundancy problem as well as insertion and deletion anomalies.
- Consider Parts(Part_no, Name, Location, Unit_price, Manu_id, Manu_name, Manu_Address)
  - It is obvious that Parts is in 2NF
  - Redundancy and various anomalies are introduced by
    - Manu_id → {Manu_name, Manu_Address}

Third Normal Form (3NF)

- Based on the concept of transitive dependency.
- Transitive Dependency is a condition where
  - A, B and C are attributes of a relation such that if A ->B and B ->C,
  - then C is transitively dependent on A through B. (Provided that A is not functionally dependent on B or C).

3rd Normal Form

- Satisfies 2nd Normal Form conditions
- Has non-transitive dependency (i.e. non-key attributes are mutually independent)
- Elements which are mutually dependent are separated off into another table with the independent part as the key
- A relation that is in 1NF and 2NF and in which no non-primary-key attribute is transitively dependent on the primary key.

2NF to 3NF

- Identify the primary key in the 2NF relation.
- Identify functional dependencies in the relation.
- If transitive dependencies exist on the primary key remove them by placing them in a new relation along with a copy of their dominant.

2NF to 3NF Example

- All the non-primary-key attributes within the PropertyOwner relation are functionally dependent on the primary key, with the exception of oName, which is transitively dependent on **ownerNo**

Client

clientNo	cName
CR76	John Kay
CR56	Aline Stewart

Rental

clientNo	propertyNo	rentStart	rentFinish
CR76	PG4	1-Jul-07	31-Aug-08
CR76	PG16	1-Sep-08	1-Sep-07
CR56	PG4	1-Sep-06	10-Jun-07
CR56	PG36	10-Oct-07	1-Dec-08
CR56	PG16	1-Nov-09	10-Aug-10

PropertyOwner

propertyNo	pAddress	rent	ownerNo	oName
PG4	6 Lawrence St, Glasgow	350	CO40	Tina Murphy
PG16	5 Novar Dr, Glasgow	450	CO93	Tony Shaw
PG36	2 Manor Rd, Glasgow	375	CO93	Tony Shaw

2NF to 3NF Example

- To transform the PropertyOwner relation into 3NF, we must first remove this transitive dependency by creating two new relations called PropertyForRent and Owner,

PropertyForRent

propertyNo	pAddress	rent	ownerNo
PG4	6 Lawrence St, Glasgow	350	CO40
PG16	5 Novar Dr, Glasgow	450	CO93
PG36	2 Manor Rd, Glasgow	375	CO93

Owner

ownerNo	oName
CO40	Tina Murphy
CO93	Tony Shaw

Tournament Name	Year	Winner	Winner's DOB
Indiana Invitational	1998	Al Fredrickson	21 July 1975
Cleveland Open	1999	Bob Albertson	28 September 1968
Des Moines Masters	1999	Al Fredrickson	21 July 1975
Indiana Invitational	1999	Chip Masterson	14 March 1977

Tournament	Year	Winner
Indiana Invitational	1998	Al Fredrickson
Cleveland Open	1999	Bob Albertson
Des Moines Masters	1999	Al Fredrickson
Indiana Invitational	1999	Chip Masterson

Winner	Date of birth
Chip Masterson	14 March 1977
Al Fredrickson	21 July 1975
Bob Albertson	28 September 1968

Normalisation
Complete Example

Based on Robert Timmer-Arends and Mark Kelly

- Vceit.com

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 Maths Info Tech	$50 $50 $100	B A 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 Maths Info Tech	$50 $50 $100	B A B+

How can you make it 1NF?

Create new rows so each cell contains only one value

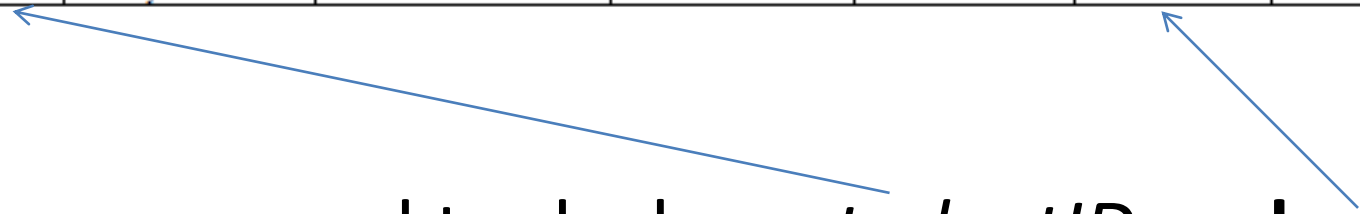
StudentID	StudentName	Address	HouseName	HouseColor	Subject	SubjectCost	Grade
19594332X	Mary Watson	10 Charles Street	Bob	Red	English Maths Info Tech	$50 $50 $100	B A B+

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?

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 = 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)

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)

1

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

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)

1

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

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

1

SUBJECTS TABLE (key = Subject)

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

1

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)

A 2NF check

STUDENT TABLE (key = StudentID)

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

1

∞

SUBJECTS TABLE (key = Subject)

1

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

1

∞

SUBJECTS TABLE (key = Subject)

1

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

1

∞

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

SUBJECTS TABLE (key = Subject)

1

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)

1

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)

But is it 3NF?

A 3NF check

STUDENT TABLE (key = StudentID)

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

1

Oh oh...

What?

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

∞

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

SUBJECTS TABLE (key = Subject)

1

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)

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**

SUBJECTS TABLE (key = Subject)

1

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)

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)

1

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)

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***

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

∞

WHAT DO WE DO?

SUBJECTS TABLE (key = Subject)

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

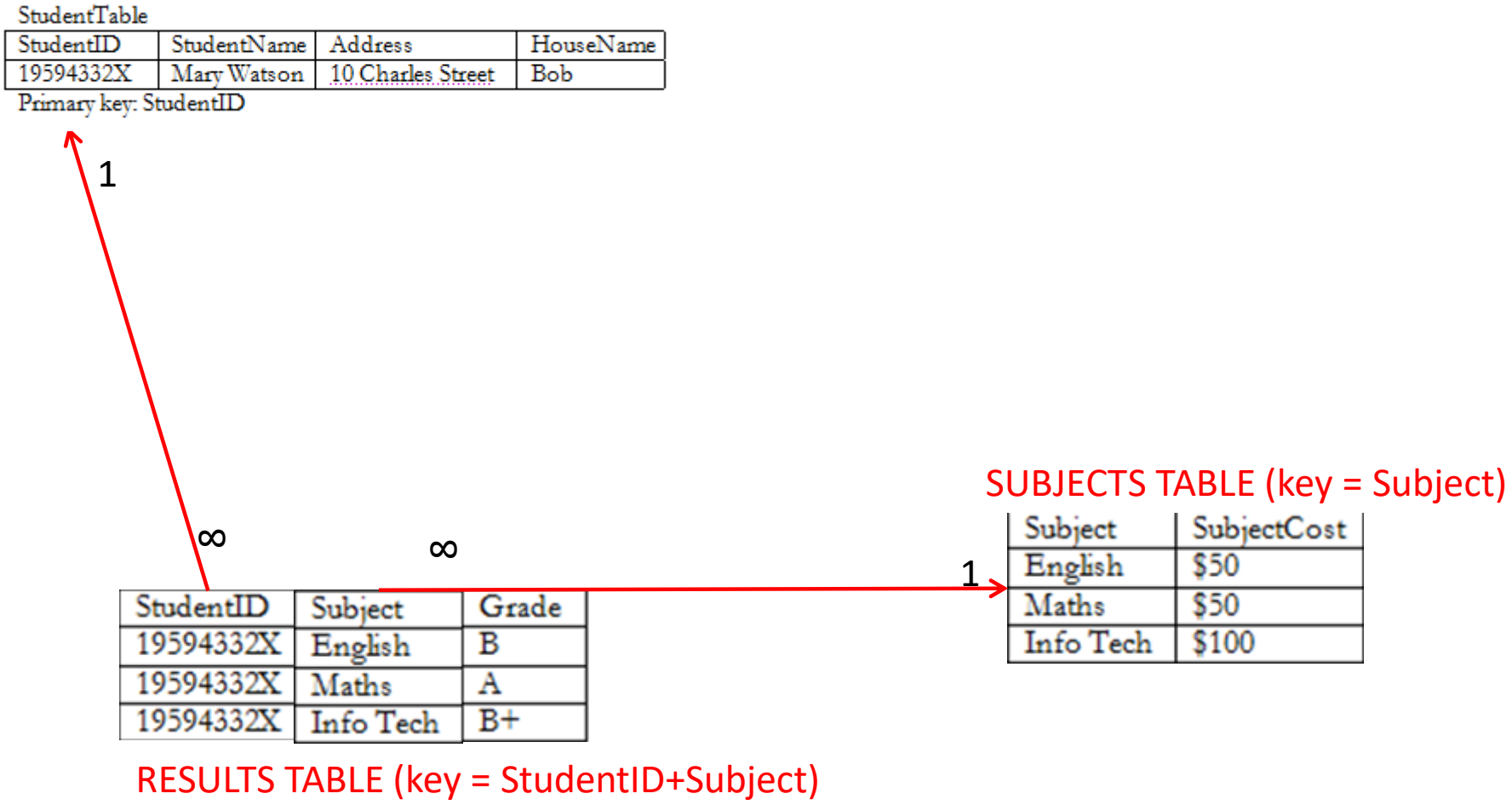
1

∞

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

RESULTS TABLE (key = StudentID+Subject)

Again, carve off the offending fields



A 3NF fix

StudentTable

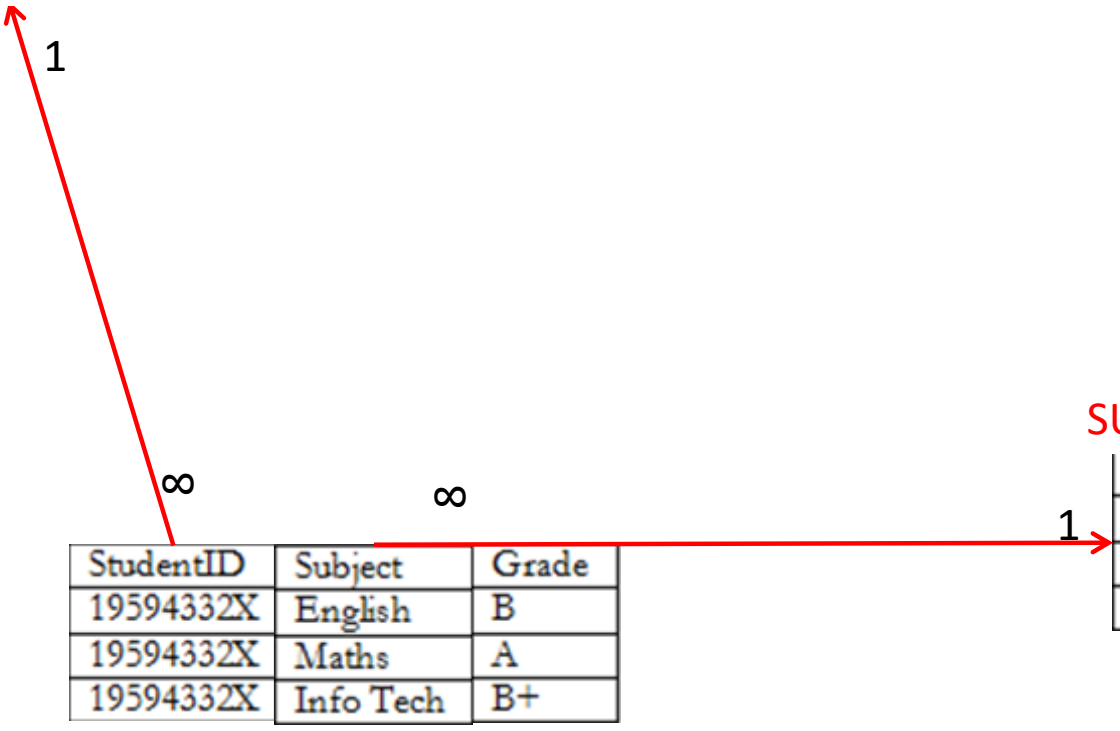
StudentID	StudentName	Address
19594332X	Mary Watson	10 Charles Street

Primary key: StudentID

HouseTable

HouseName	HouseColor
Bob	Red

Primary key: HouseName



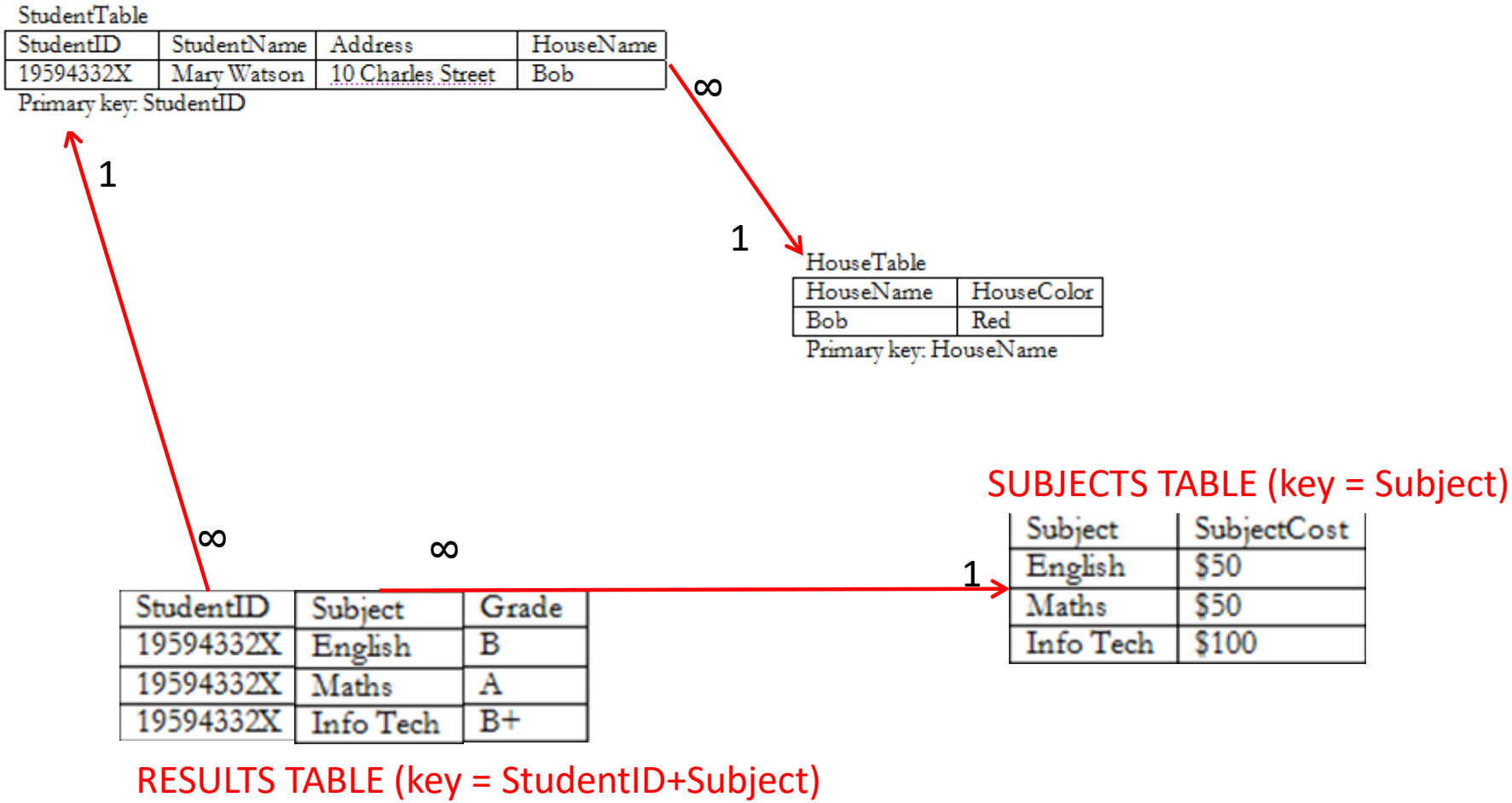
StudentID	Subject	Grade
19594332X	English	B
19594332X	Maths	A
19594332X	Info Tech	B+

RESULTS TABLE (key = StudentID+Subject)

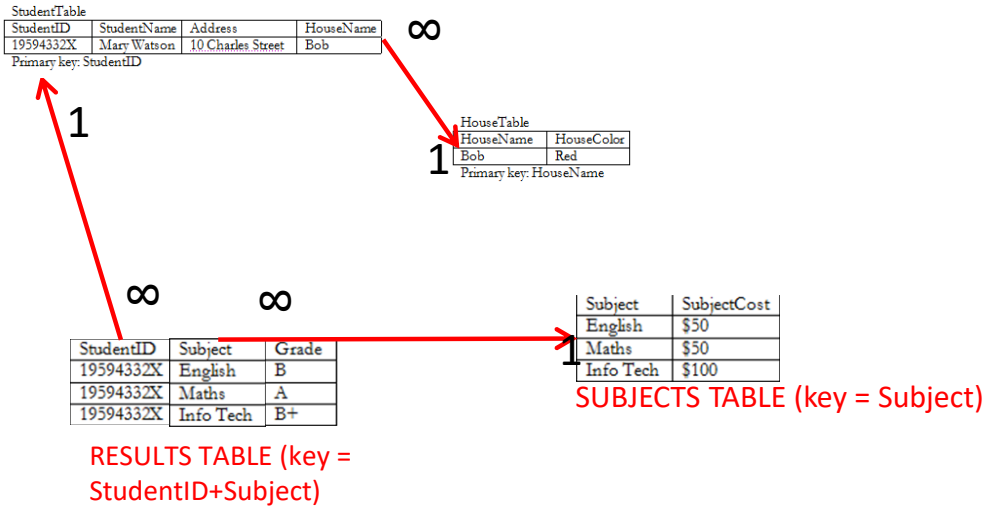
SUBJECTS TABLE (key = Subject)

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

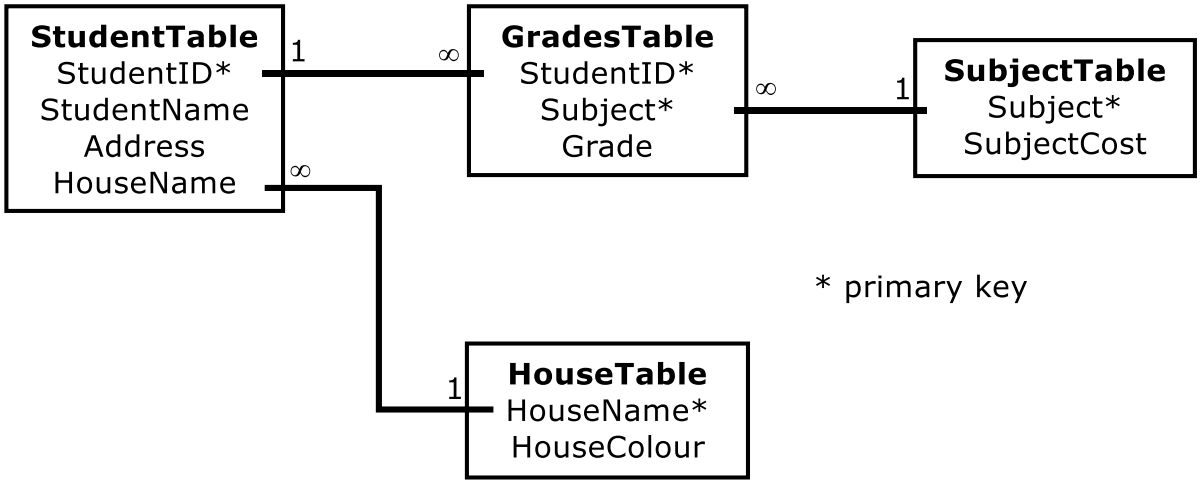
A 3NF fix


StudentTable
StudentID | StudentName | Address | HouseName
19594332X | Mary Watson | 10 Charles Street | Bob
Primary key: StudentID
∞
1
1
HouseTable
HouseName | HouseColor
Bob | Red
Primary key: HouseName
∞
∞
1
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)

A 3NF win!


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)

Or...


StudentTable
StudentID*
StudentName
Address
HouseName
1
∞
∞
GradesTable
StudentID*
Subject*
Grade
∞
1
SubjectTable
Subject*
SubjectCost
∞
1
HouseTable
HouseName*
HouseColour
* primary key

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…