NORMALIZATION

- This is a formal method to check tables for potential data storage problems termed anomalies.
- In order to clarify the discussion, we will formalize the definition of several terms.
  - **KEYS:** The term KEY is often confusing because it has different meanings during design and implementation of a system.
  - **DESIGN:** During design, **KEY** means a combination of one or more attributes (columns) of a relational table that uniquely identify rows in the table.
  - **KEY** guarantees uniqueness; no two rows can be identical.
  - **IMPLEMENTATION:** During implementation,
  - the term **KEY** is a column on which the DBMS builds an index or other data structure, to allow quick access to rows. Such keys need not be unique - they may be secondary keys enabling access to a SET of rows.

.

NORMALIZATION

- Sometimes the terms **Logical Key** and **Physical Key** are used to distinguish between these two meanings.
- **INDEXES:** Since a physical key is usually an index, we often use the term **Index** for a physical key.
- Indexes are created to:
  - Allow quick access.
  - Facilitate sorting or sorted order access.
  - Insure no duplicates if the keyword **UNIQUE** is used when defining an index.

Functional Dependency

- **. FUNCTIONAL DEPENDENCY:**
- A **Functional Dependency** (FD) is a relationship between or among attributes,
- i.e. given a unique value for one or more attributes, such as the **CustomerAccountNumber**, we can find a corresponding value for another attribute such as the **CustomerBalance** attribute.
- Equations represent functional dependencies. Consider the equation:
- **TotalPrice = ItemPrice x Quantity**

Functional Dependency

- Unlike equations, however, **FDs** cannot be worked out using arithmetic; instead, they are listed in the database.
- **FDs** are written following standard notation. For example, if the attribute **A** determines the attribute **B**, we write the notation:
- **A -> B (read A functionally determines B)**

- The attribute "**A**" is called a **determinant**, here it is a determinant of **B**.

Armstrong Axioms
Rules of Functional Dependencies

- If **W**, **X**, **Y**, and **Z** are attributes of a table, then:
- **Reflexive rule**
  - X -> X
    - (- simply means that if we know X, then we know X).
- **Augmentation rule**
  - If X-> Y, then XZ ->Y
    - (**augmentation rule** –
    - Note Y is not really dependent on Z, but if we know X and can determine a value for Y from X, then knowing Z has no effect on our ability to determine a value for Y).
  - .

Rules of Functional Dependencies

- If **W**, **X**, **Y**, and **Z** are attributes of a table, then:
- **UNION RULE**
  - If X -> Y and X ->Z, then X -> YZ
    - (union rule - useful for combining tables).
- **DECOMPOSITION RULE**
  - If X -> Y then X -> Z if Z is a subset of Y
    - (decomposition rule).
- **TRANSITIVITY RULE**
  - If X -> Y and Y ->Z, then X -> Z
    - (transitivity rule - useful for avoiding transitive dependencies).
- **PSEUDOTRANSITIVITY RULE**
  - If X -> Y and YZ -> W, then XZ -> W
    - (pseudotransitivity rule - useful in understanding multivalued dependencies.

Employee

EmpId	Name	Dept	Salary	Course	DateTook	Fee
130	Margaret	Math	45,000	Calculus	01/15	150
130	Margaret	Math	45,000	Biology	02/15	200
200	Susan	Sci	38,000	Biology	01/15	200
250	Chris	Math	52,000	Calculus	03/15	150
250	Chris	Math	52,000	Biology	03/15	200
425	Bill	Math	48,000	Algebra	03/15	200
425	Bill	Math	48,000	Calculus	04/15	

- **Problems With This Table:**
  - Redundancy of data storage.
  - Potential inconsistencies on updating data.

What is the Primary Key of this table?

- To determine an appropriate key, you should first examine the **FDs**.
- **EmpId -> Name, Dept, Salary**
  Course -> Fee
  EmpId, Course -> DateTook

- Assuming employees only take a course once (no time dependencies), then uniqueness for the rows in the table is ensured by a composite key of **EmpId + Course**.

Data Anomalies

- **Data Anomalies** are problems with data storage caused by poorly structured tables.

- **Insertion Anomaly**.
- If the primary key is **EmpId + Course**, to add a new employee, the employee must first be enrolled in a course.

- If an employee is not enrolled in a course, then the **COURSE** column that is part of the composite primary key will be **null**, and null key values are **not** allowed.

Data Anomalies

- **Deletion Anomaly**.
- Deleting data for Employee #425 (Bill) causes us to lose data about Algebra and the course fee for Algebra because Bill is the only employee who has enrolled in Algebra.

- **Modification Anomaly**.
- If the fee for Calculus is increased, the data must be updated for more than one row.
- Note there is also a time-sensitivity between **EmpId** and **Course** since an employee could take a course many times, but the table does not track this fact

First Normal Form (1NF).

- **Remove Repeating Groups**

EmpId	**Name**	**Dept**	**Salary**	**Course**	**DateTook**	**Fee**
130	Margaret	Math	45,000	Calculus	01/15	150
				Biology	02/15	200
200	Susan	Sci	38,000	Biology	01/15	200
250	Chris	Math	52,000	Calculus	03/15	150
				Biology	03/15	200
425	Bill	Math	48,000	Algebra	03/15	200
				Calculus	04/15	150

- In order to store information for employees who take more than one course, a possible table structure is:
- **EMPLOYEE (EmpId, Name, Dept, Salary, Course1, DateTook1, Fee1, Course2, DateTook2, Fee2, …)**

- Obviously there is a problem anticipating the number of times the group (Course, DateTook, and Fee will repeat).
- A better table structure (allowing for the fact that there is significant data redundancy) and a composite primary key of **EmpId + Course**, is:

- **EMPLOYEE (EmpId, Name, Dept, Salary, Course, DateTook, Fee)**

EmpId	**Name**	**Dept**	**Salary**	**Course**	**DateTook**	**Fee**
130	Margaret	Math	45,000	Calculus	01/15	150
130	Margaret	Math	45,000	Biology	02/15	200
200	Susan	Sci	38,000	Biology	01/15	200
250	Chris	Math	52,000	Calculus	03/15	150
250	Chris	Math	52,000	Biology	03/15	200
425	Bill	Math	48,000	Algebra	03/15	200
425	Bill	Math	48,000	Calculus	04/15	150

- An obvious problem associated with the above solution is the storage of a lot of redundant data.
- We can eliminate this problem by further normalizing the table.

First Normal Form (1 NF)

TABLE_PRODUCT

Product ID	Color	Price
1	red, green	15.99
2	yellow	23.99
3	green	17.50
4	yellow, blue	9.99
5	red	29.99

First Normal Form (1 NF)

- A database is in first normal form if it satisfies the following conditions:
  - **Contains only atomic values**
  - **There are no repeating groups**
- An atomic value is a value that cannot be divided.
- For example, in the table shown below, the values in the [Color] column in the first row can be divided into "red" and "green", hence [TABLE_PRODUCT] is not in 1NF.

First Normal Form (1 NF)

- A **repeating group** means that a table contains two or more columns that are closely related.
- For example, a table that records data on a book and its author(s) with the following columns:

- [Book ID], [Author 1], [Author 2], [Author 3] is not in 1NF
- because [Author 1], [Author 2], and [Author 3] are all repeating the same attribute.

First Normal Form (1 NF)

TABLE_PRODUCT

Product ID	Color	Price
1	red, green	15.99
2	yellow	23.99
3	green	17.50
4	yellow, blue	9.99
5	red	29.99

First Normal Form (1 NF)

- This table is not in first normal form because the [Color] column can contain multiple values. For example, the first row includes values "red" and "green."
- To bring this table to first normal form, we split the table into two tables and now we have the resulting tables:

First Normal Form (1 NF)

TABLE_PRODUCT_PRICE

Product ID	Price
1	15.99
2	23.99
3	17.50
4	9.99
5	29.99

TABLE_PRODUCT_COLOR

Product ID	Color
1	red
1	green
2	yellow
3	green
4	yellow
4	blue
5	red

Second Normal Form (2NF).

- **Remove Partial Dependencies.**

Second Normal Form (2NF).

- **Remove Partial Dependencies.**

- A database is in second normal form if it satisfies the following conditions:
  - **It is in first normal form**
  - **All non-key attributes are fully functional dependent on the primary key**

Second Normal Form (2NF).

- In a table, if attribute B is functionally dependent on A,
  - but is not functionally dependent on a proper subset of A, then B is considered fully functional dependent on A.
- Hence, in a 2NF table, all non-key attributes cannot be dependent on a subset of the primary key.
- Note that if the primary key is not a composite key, all non-key attributes are always fully functional dependent on the primary key.
- A table that is in **1st normal form** and contains only a **single key as the primary key** is automatically in **2nd normal form.**

Second Normal Form (2NF).

- **2nd Normal Form Example**

Consider the following example:

TABLE_PURCHASE_DETAIL

Customer ID	Store ID	Purchase Location
1	1	Los Angeles
1	3	San Francisco
2	1	Los Angeles
3	2	New York
4	3	San Francisco

Second Normal Form (2NF).

- **2nd Normal Form Example**
- Consider the following example:
  - This table has a composite primary key [Customer ID, Store ID].
  - The non-key attribute is [Purchase Location].
  - In this case, [Purchase Location] only depends on [Store ID], which is only part of the primary key.
  - Therefore, this table does not satisfy second normal form.
  - To bring this table to second normal form, we break the table into two tables, and now we have the following:
-

Second Normal Form (2NF).

- **2nd Normal Form Example**
-

TABLE_PURCHASE

Customer ID	Store ID
1	1
1	3
2	1
3	2
4	3

TABLE_STORE

Store ID	Purchase Location
1	Los Angeles
2	New York
3	San Francisco

Second Normal Form (2NF).

- **2nd Normal Form Example**
- What we have done is to remove
- the partial functional dependency that we initially had.
- Now, in the table [TABLE_STORE], the column [Purchase Location] is fully dependent on the primary key of that table, which is [Store ID].

First normal form (1NF)

- First normal form: **A relation is in first normal form if every attribute in every row can contain only one single (atomic) value.**
- A university uses the following relation:

Student(Surname, Name, Skills)

The attribute Skills can contain multiple values and therefore the relation is not in the first normal form.

But the attributes Name and Surname are atomic attributes that can contain only one value.

First Normal Form(1NF)

Students

FirstName	LastName	Knowledge
Thomas	Mueller	Java, C++, PHP
Ursula	Meier	PHP, Java
Igor	Mueller	C++, Java

Startsituation

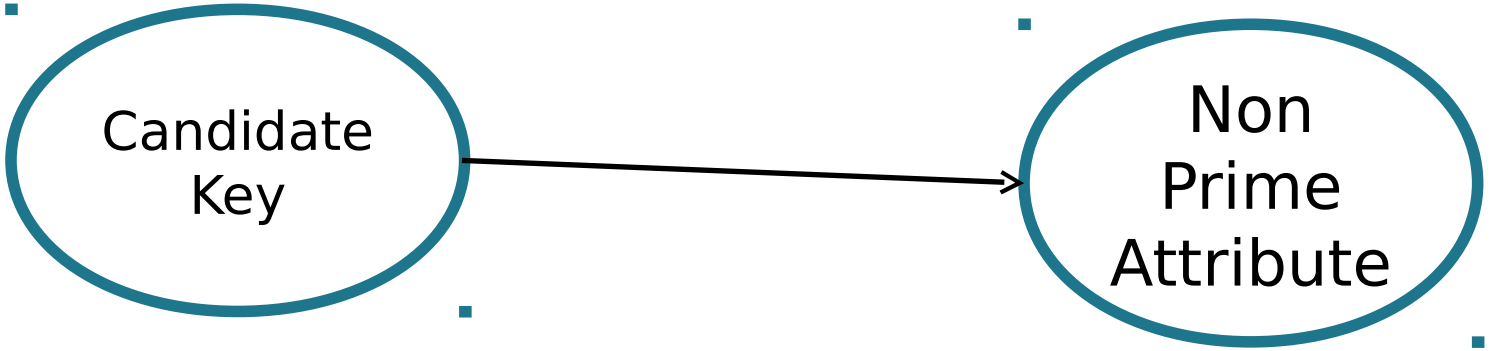
Result after Normalisation

Students

FirstName	LastName	Knowledge
Thomas	Mueller	C++
Thomas	Mueller	PHP
Thomas	Mueller	Java
Ursula	Meier	Java
Ursula	Meier	PHP
Igor	Mueller	Java
Igor	Mueller	C++

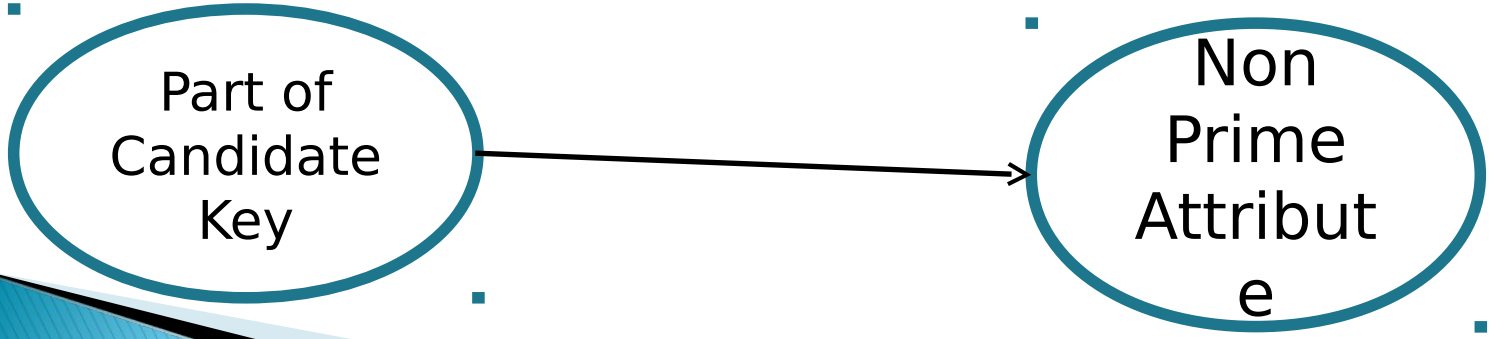
Full Dependency

- R(S,T,U,V)
- S→T T→U U→V V→S
- S,T,U ,V are candidate keys



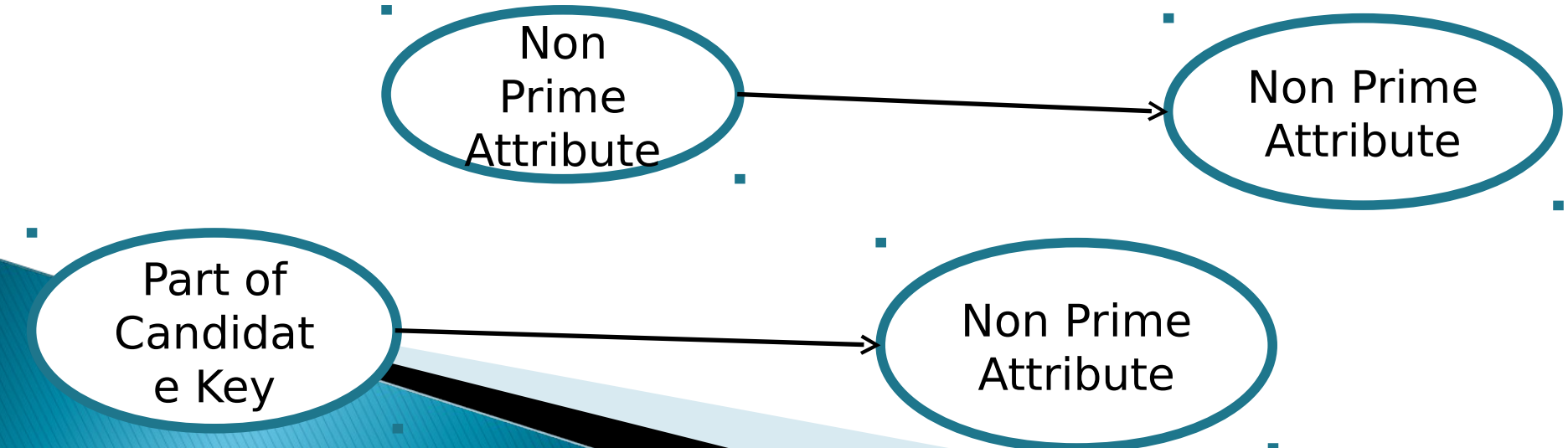
Partial Dependency

- Determinant FD is part of Candidate key
- Dependent FD is a Non Prime Attribute
- **R(A,B,C,D)**
- A-> C, B→D
- AB is candidate key
- A->C   A is part of Candidate key,
- C is Non Prime Attribute



Transitive Dependency

- Determinant FD contains
- Part of Candidate key +Non Prime Attribute
- Dependent FD is a Non Prime Attribute
- R(A,B,C,D)
- A-> C, B→D BC->D C->D
- AB is candidate key



Second normal form (2NF)

- **Second normal form**: A relation is in second normal form if it is in 1NF and every non key attribute is fully functionally dependent on the primary key.
- A university uses the following relation:

  Student(IDSt, StudentName, IDProf, ProfessorName, Grade)

  The attributes IDSt and IDProf are the identification keys.
  All attributes a single valued (1NF).

Second normal form (2NF)

The following functional dependencies exist:

1. The attribute ProfessorName is functionally dependent on attribute IDProf

(IDProf --> ProfessorName)

2. The attribute StudentName is functionally dependent on IDSt

(IDSt --> StudentName)

3. The attribute Grade is fully functional dependent on IDSt and IDProf

(IDSt, IDProf --> Grade)

Second normal form (2NF)

Students

IDSt	LastName	IDProf	Prof	Grade
1	Mueller	3	Schmid	5
2	Meier	2	Borner	4
3	Tobler	1	Bernasconi	6

Startsituation

Result after normalisation

Students

ID	LastName
1	Mueller
2	Meier
3	Tobler

Professors

IDProf	Professor
1	Bernasconi
2	Borner
3	Schmid

Grades

IDStIDProf	Grade	
1	3	5
2	2	4
3	1	6

- The table in this example is in first normal form (1NF) since all attributes are single valued. But it is not yet in 2NF.
- If student 1 leaves university and the tuple is deleted, then we loose all information about professor Schmid, since this attribute is fully functional dependent on the primary key IDSt.
- To solve this problem, we must create a new table Professor with the attribute Professor (the name) and the key IDProf.
- The third table Grade is necessary for combining the two relations Student and Professor and to manage the grades.
- Besides the grade it contains only the two IDs of the student and the professor. If now a student is deleted, we do not loose the information about the professor.

Third normal form (3NF)

- Third normal form**: A relation is in third normal form if it is in 2NF and no non key attribute is transitively dependent on the primary key**.
- A bank uses the following relation:

  Vendor(ID, Name, Account_No, Bank_Code_No, Bank)

  The attribute ID is the identification key. All attributes are single valued (1NF). The table is also in 2NF.

Third normal form (3NF)

The following dependencies exist:

1. Name, Account_No, Bank_Code_No are functionally dependent on ID

(ID --> Name, Account_No, Bank_Code_No)

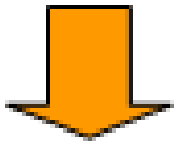
2. Bank is functionally dependent on Bank_Code_No (Bank_Code_No --> Bank)

Third normal form (3NF)

Vendor

ID	Name	Account_No	Bank_Code_No	Bank

Startsituation



Result after normalisation

Vendor

ID	Name	Account_No	Bank_Code_No

Bank

Bank_Code_No	Bank

Boyce and Codd Normal Form (BCNF)

- **“A relation is in BCNF, if and only if, every determinant is a candidate key.”.**

Boyce and Codd Normal Form (BCNF)

- **Boyce and Codd Normal Form** is a higher version of the Third Normal form.
- This form deals with certain type of anamoly that is not handled by 3NF.
- **A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF.**
- For a table to be in BCNF, following conditions must be satisfied:
- **R** must be in **3rd Normal Form**
- and, **for each functional dependency (X -> Y), X should be a super Key.**

Consider the following relationship : **R (A,B,C,D)**

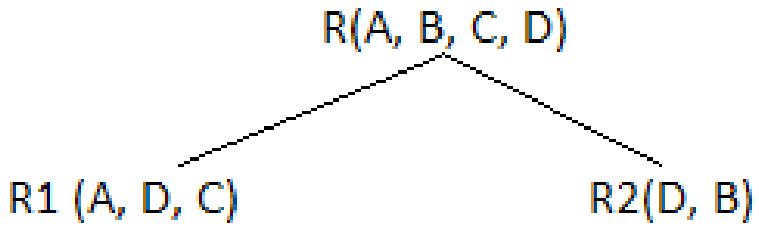
and following dependencies :

A -> BCD
BC -> AD
D -> B

Above relationship is already in 3rd NF. Keys are **A** and **BC**.

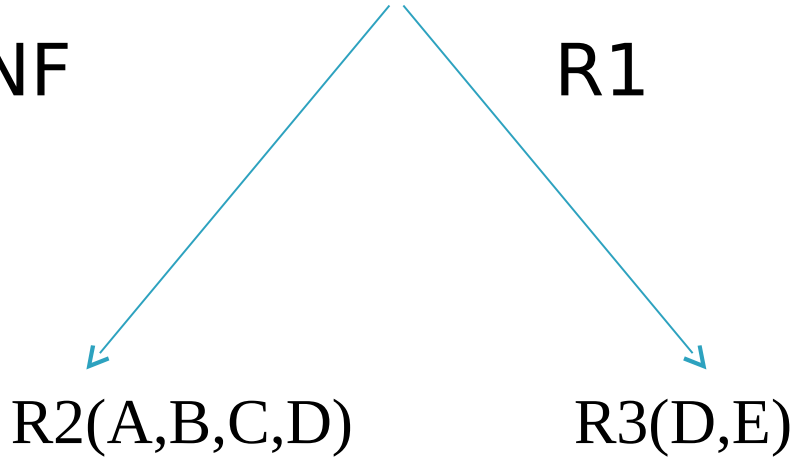
Hence, in the functional dependency, **A -> BCD**, A is the super key.
in second relation, **BC -> AD**, BC is also a key.
but in, **D -> B**, D is not a key.

Hence we can break our relationship R into two relationships **R1** and **R2**.



Breaking, table into two tables, one with A, D and C while the other with D and B.

- R1(A,B,C,D,E) R2(B,F) CANDIDATE KEY IS B
- A→BCDE
- BC→ADE
- D→E
- 2NF



Normalisation

- – Steps
- 1NF
- – **Removing repeating groups**
- 2NF
- – **Remove partial dependencies**
- • 3NF
- – **Remove transitive dependencies**
- BCNF
- – **Remove non-candidate key dependencies**

- **1NF Example**
- Reminder: Rules for first normal form
- There are no columns with repeated or similar data
- Each data item cannot be broken down any further.
- Each row is unique i.e. it has a primary key
- Each field has an unique name
-

Which of these tables are NOT in first normal form?

- 1.

Title Firstname Surname Full name Address City Postcode

Mr Tom Smith Tom Smith 42 Mill Street London WE13GW

- 2.

ID IP Address username last accessed Activity Result active

1003 198.168.1.5 Smith 20081021:14.10 Save file success y

- 3.

ItemID Product Description Size Colour Colour Colour

234 Shoe High Heel 6 red blue brown

- 4.

StudentID Firstname Surname SchoolID* ClassID*

354 Tom Smith 6 5F Comments:

Which of these tables are NOT in first normal form?

Title Firstname Surname Full name Address City Postcode

Mr Tom Smith Tom Smith 42 Mill Street London WE13GW

- **Table 1.** This is not in 1NF. There is no primary key defined and so this record cannot be guaranteed to be unique. Also Full name is redundant - data is not atomic - as it is simply a combination of Firstname and Surname.

Which of these tables are NOT in first normal form?

ID IP Address username last accessed Activity Result active

1003 198.168.1.5 Smith 20081021:14.10 Save file success y

- **Table 2.** This is in at least 1NF. It has a primary key. The data is atomic. Each field has an unique name. There are no repeat data.

Which of these tables are NOT in first normal form?

ItemID Product Description Size Colour Colour Colour

234 Shoe High Heel 6 red blue brown

- **Table 3.** This is not in 1NF. It has a primary key, so it passes that test, data is atomic - tick in the box, but the colour the shoe can come in is being repeated - and furthermore the fields have the same name - so not in 1NF

StudentID Firstname Surname SchoolID* ClassID*

354 Tom Smith 6 5F Comments:

- **Table 4.** This is in 1NF as it meets all the rules for the first normal form.

Project Code	Project Title	Project Manager	Project Budget	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	Pensions System	M Phillips	24500	S10001	A Smith	L004	IT	22.00
PC010	Pensions System	M Phillips	24500	S10030	L Jones	L023	Pensions	18.50
PC010	Pensions System	M Phillips	24500	S21010	P Lewis	L004	IT	21.00
PC045	Salaries System	H Martin	17400	S10010	B Jones	L004	IT	21.75
PC045	Salaries System	H Martin	17400	S10001	A Smith	L004	IT	18.00
PC045	Salaries System	H Martin	17400	S31002	T Gilbert	L028	Database	25.50
PC045	Salaries System	H Martin	17400	S13210	W Richards	L008	Salary	17.00
PC064	HR System	K Lewis	12250	S31002	T Gilbert	L028	Database	23.25
PC064	HR System	K Lewis	12250	S21010	P Lewis	L004	IT	17.50
PC064	HR System	K Lewis	12250	S10034	B James	L009	HR	16.50

Project Tables: Repeating Attributes Removed

Project Code	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	S10001	A Smith	L004	IT	22.00
PC010	S10030	L Jones	L023	Pensions	18.50
PC010	S21010	P Lewis	L004	IT	21.00
PC045	S10010	B Jones	L004	IT	21.75
PC045	S10001	A Smith	L004	IT	18.00
PC045	S31002	T Gilbert	L028	Database	25.50
PC045	S13210	W Richards	L008	Salary	17.00
PC064	S31002	T Gilbert	L028	Database	23.25
PC064	S21010	P Lewis	L004	IT	17.50
PC064	S10034	B James	L009	HR	16.50

Project Code	Project Title	Project Manager	Project Budget
PC010	Pensions System	M Phillips	24500
PC045	Salaries System	H Martin	17400

2NF Tables: Partial Key Dependencies Removed

Project Code	Project Title	Project Manager	Project Budget
PC010	Pensions System	M Phillips	24500
PC045	Salaries System	H Martin	17400
PC064	HR System	K Lewis	12250

Project Code	Employee No.	Hourly Rate
PC010	S10001	22.00
PC010	S10030	18.50
PC010	S21010	21.00
PC045	S10010	21.75
PC045	S10001	18.00
PC045	S31002	25.50
PC045	S13210	17.00
PC064	S31002	23.25
PC064	S21010	17.50
PC064	S10034	16.50

Employee No.	Employee Name	Department No.	Department Name
S10001	A Smith	L004	IT
S10030	L Jones	L023	Pensions
S21010	P Lewis	L004	IT
S10010	B Jones	L004	IT
S31002	T Gilbert	L028	Database
S13210	W Richards	L008	Salary
S10034	B James	L009	HR