DATABASE NORMALIZATION

Prepared & Presented by Neldi Grace Y. Canlas, Sgt PA (RES)

WHAT IS NORMALIZATION ?

- **NORMALIZATION** is a database design technique that organizes tables in a manner that reduces redundancy and dependency of data.
- Normalization divides larger tables into smaller tables and links them using relationships.
- The purpose of Normalization is to eliminate redundant (useless) data and ensure data is stored logically.
- The inventor of the relational model E.F.Codd proposed the theory of normalization.

REDUNDANCY

- R

SID	SName	Age
1	Jojo	20
2	Kit	25
1	Jojo	20

- If the SID is primary key to each row, you can use it to remove the duplicates as shown below:

SID	SName	Age
1	Jojo	20
2	Kit	25

REDUNDANCY (CONT..)

- Column Level Redundancy:
- Now Rows are same but in column level because of Sid is primary key but columns are same.

Sid	Sname	Cid	Cname	Fid	Fname	Salary
1	AA	C1	DBMS	F1	Jojo	30000
2	BB	C2	JAVA	F2	KK	50000
3	CC	C1	DBMS	F1	Jojo	30000
4	DD	C1	DBMS	F1	Jojo	30000

Redundant Column Values

WHAT IS AN ANOMALY?

- Problems that can occur in poorly planned, unnormalized databases where all the data is stored in one table (a flat-file database).
- Types of Anomalies:
  - Insert
  - Delete
  - Update

ANOMALIES IN DBMS

- **Insert Anomaly :** An Insert Anomaly occurs when certain attributes cannot be inserted into the database without the presence of other attributes.
- **Delete Anomaly:** A Delete Anomaly exists when certain attributes are lost because of the deletion of other attributes.
- **Update Anomaly:** An Update Anomaly exists when one or more instances of duplicated data is updated, but not all.

ANOMALY EXAMPLE

- Below table U … Salary. And the Sid acts as

Table: University

Sid	Sname	Cid	Cname	Fid	Fname	Salary
1	Ram	C1	DBMS	F1	Sachin	30000
2	Shyam	C2	Java	F2	Boby	28000
3	Ankit	C1	DBMS	F1	Sachin	30000
4	saurabh	C1	DBMS	F1	Sachin	30000

INSERTION ANOMALY

- Suppo the above table. aly is know

Table: University

Sid	Sname	Cid	Cname	Fid	Fname	Salary
1	Ram	C1	DBMS	F1	Sachin	30000
2	Shyam	C2	Java	F2	Boby	28000
3	Ankit	C1	DBMS	F1	Sachin	30000
4	saurabh	C1	DBMS	F1	Sachin	30000
				F3	Arun	29000

Insertion Anomaly

DELETE ANOMALY

SQL:
DELETE FROM *University* WHERE *Sid=2*;

- When th … e above table, then it will delete th …

Sid	Sname	Cid	Cname	Fid	Fname	Salary
1	Ram	C1	DBMS	F1	Sachin	30000
~~2~~	~~Shyam~~	Deletion anomaly				
3	Ankit	C1	DBMS	F1	Sachin	30000
4	Saurabh	C1	DBMS	F1	Sachin	30000

UPDATE ANOMALY

Sid	Sname	Cid	Cname	Fid	Fname	Salary
1	Ram	C1	DBMS	F1	Sachin	30000
2	Shyam	C2	Java	F2	Boby	28000
3	Ankit	C1	DBMS	F1	Sachin	30000
4	saurabh	C1	DBMS	F1	Sachin	30000

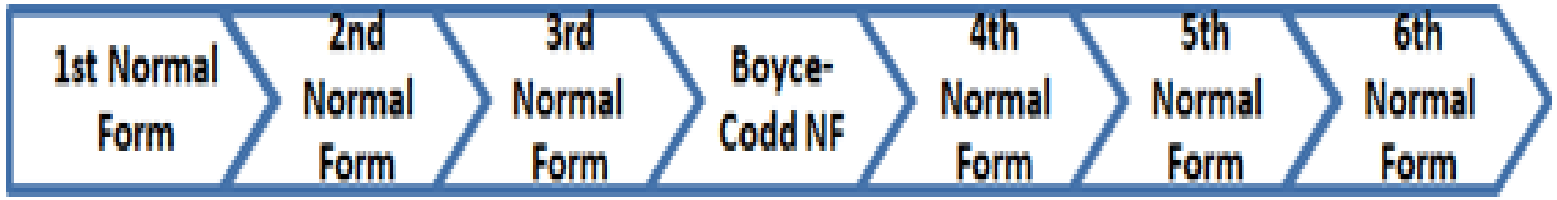
SQL:
UPDATE *University*
SET *Salary= 40000*
WHERE *Fid="F1"*;

'om 30000 to 40000 in above table
data redundancy. So, this is an

To remove all these anomalies, we need to normalize the data in the database.

NORMAL FORMS

- The Theory of Data Normalization in SQL is still being developed further. For example, there are discussions even on 6th Normal Form. **However, in most practical applications, normalization achieves its best in 3rd Normal Form**. The evolution of Normalization theories is illustrated below-



FIRST NORMAL FORM (1NF)

- According to the E.F. Codd, a relation will be in 1NF, if each cell of a relation contains only an atomic value.

1NF EXAMPLE

- Example:

Course	Content
Programming	Java, c++
Web	HTML, PHP, ASP

The following Course_Content relation is not in 1NF because the Content attribute contains multiple values.

1NF EXAMPLE (CONT..)

- The below relation student is in 1NF:

Course	Content
Programming	Java
Programming	c++
Web	HTML
Web	PHP
Web	ASP

RULES OF 1NF

The official qualifications for 1NF are:

1. Each **attribute name** must be unique.
2. Each **attribute value** must be single.
3. Each **row** must be unique.

- Additional:
  - Choose a primary key.
- Reminder:

A primary key is ***unique***, ***not null***, ***unchanged***. A primary key can be either an attribute or combined attributes.

SECOND NORMAL FORM (2NF)

- According to the E.F. Codd, a relation is in 2NF, if it satisfies the following conditions:
  - The table should be in the First Normal Form.
  - There should be no Partial Dependency.

PRIME AND NON PRIME ATTRIBUTES

Prime attributes: The attributes which are used to form a candidate key are called prime attributes.

Non-Prime attributes: The attributes which do not form a candidate key are called non-prime attributes.

Roll. No.	First Name of Student	Last Name of Student	Course code
01.	Adam	Gilchrist	A100
02	Adam	Peter	B50
03	John	Gilchrist	C80

- Prime Attribute: Roll No., Course Code
- Non-Prime Attribute: First Name of Student, Last Name of Student

FUNCTIONAL DEPENDENCY

- A dependency FD: X → Y means that the values of Y are determined by the values of X. Two tuples sharing the same values of X will necessarily have the same values of Y.
- We illustrate this as:
  - X→ Y *(read as: X determines Y or Y depends on X)*

FUNCTIONAL DEPENDENCY

Student ID	Semester	Lecture	TA
1234	6	Numerical Methods	John
1221	4	Numerical Methods	Smith
1234	6	Visual Computing	Bob
1201	2	Numerical Methods	Peter
1201	2	Physics II	Simon

- Whenever two rows in this table feature the same StudentID, they also necessarily have the same Semester values. This basic fact can be expressed by a functional dependency:

  StudentID → Semester.

PARTIAL DEPENDENCY

- If a non-prime a known as a partial dependency.

Example of partial Dependency: Suppose there is a relation **R** with attributes **A, B,** and **C.**

R(A,B,C)

Where,

{AB} is a candidate key.

{C} is a non-prime attribute.

Then,

{A, B} are the prime attributes.

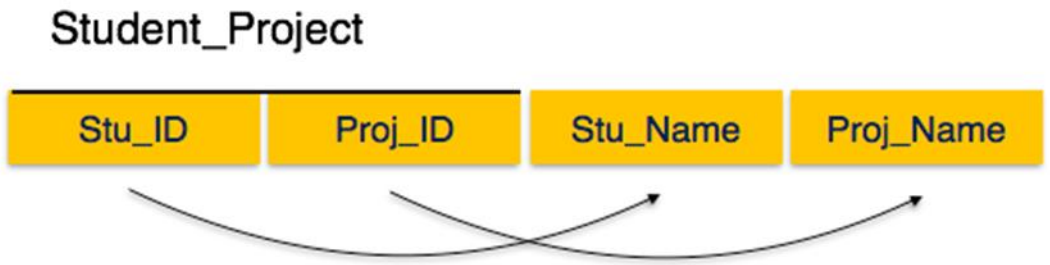
A → C is a partial dependency.

Part of a candidate key

Non- prime attribute

2NF EXAMPLE

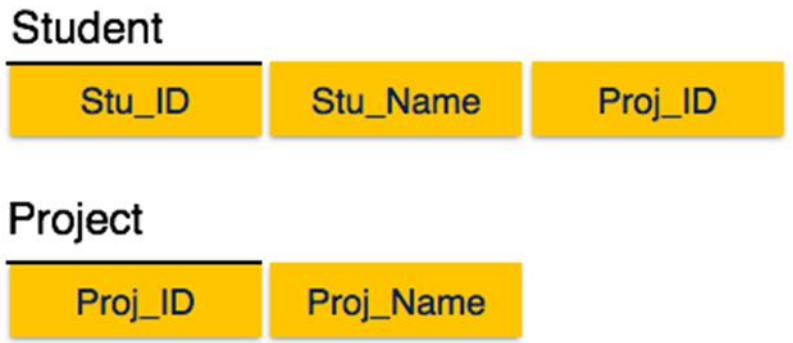
- In Student_Project relation that the prime key attributes are Stu_ID and Proj_ID.
- According to the rule, non-key attributes, i.e. Stu_Name and Proj_Name must be dependent upon both and not on any of the prime key attribute individually.
- But we find that Stu_Name can be identified by Stu_ID and Proj_Name can be identified by Proj_ID independently. This is called partial dependency, which is not allowed in Second Normal Form.



- **Candidate Keys:** {Stu_ID, Proj_I
- **Non-prime attribute:** Stu_Name, Proj_Name

2NF EXAMPLE (CONT..)

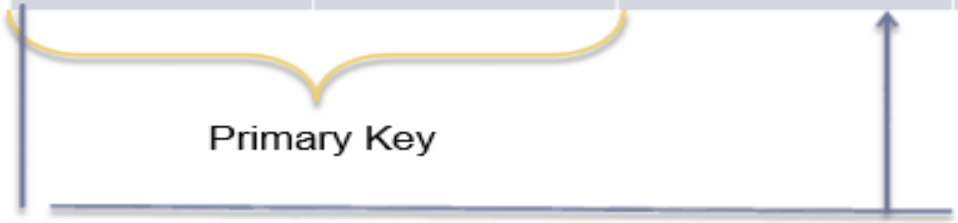
- We broke the relation in two as



ts no partial dependency.

EXAMPLE 2NF

CourseID	SemesterID	Num Student	Course Name
IT101	201301	25	Database
IT101	201302	25	Database
IT102	201301	30	Web Prog
IT102	201302	35	Web Prog
IT103	201401	20	Networking



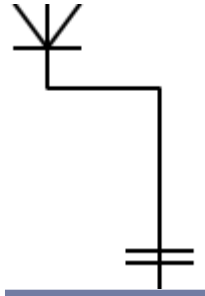
- The Course Name depends on only CourseID, a part of the primary key not the whole primary {CourseID, SemesterID}.It's called partial dependency.
- **Solution:**
- *Remove **CourseID** and **Course Name** together to create a new table.*

EXAMPLE 2NF (CONT..)

CourseID	Course Name
IT101	Database
IT101	Database
IT102	Web Prog
IT102	Web Prog
IT103	Networking

Done? Oh no, it is still not in 1NF yet.
Remove the repeating groups too.
Finally, connect the relationship.

CourseID	SemesterID	Num Student
IT101	201301	25
IT101	201302	25
IT102	201301	30
IT102	201302	35
IT103	201401	20



CourseID	Course Name
IT101	Database
IT102	Web Prog
IT103	Networking

THIRD NORMAL FORM (3NF)

- According to the E.F. Codd, a relation is in third normal form (3NF) if it satisfies the following conditions:
  - ✓ It should be in the Second Normal form.
  - ✓ It should not have Transitive Dependency.
  - ✓ All transitive dependencies are removed to place in another table.

TRANSITIVE DEPENDENCY

- A functional dependency is said to be transitive if it is indirectly formed by two functional dependencies. For e.g.
- X -> Z is a transitive dependency if the following three functional dependencies hold true:

  X->Y

  Y does not ->X

  Y->Z

TRANSITIVE DEPENDENCY(CONT..)

- Let's take an example to understand it better:

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

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

{Author} does not ->{Book}

{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.

3NF EXAMPLE

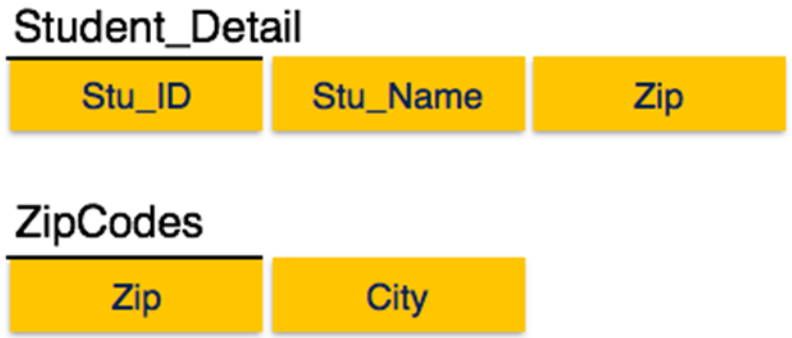
- We find that in the above Student_detail relation, Stu_ID is the key and only prime key attribute.
- We find that City can be identified by Stu_ID as well as Zip itself.
- Neither Zip is a superkey nor is City a prime attribute. Additionally, Stu_ID → Zip → City, so there exists transitive dependency.

Prime attribute: Stu_ID

Non-prime attribute: {Stu_Name, City, Zip}

3NF EXAMPLE (CONT..)

- To bring this relation into third n



elations as follows −

EXAMPLE 3NF

StudyID	Course Name	Teacher Name	Teacher Tel
1	Database	Sok Piseth	012 123 456
2	Database	Sao Kanha	0977 322 111
3	Web Prog	Chan Veasna	012 412 333
4	Web Prog	Chan Veasna	012 412 333
5	Networking	Pou Sambath	077 545 221

Primary Key

Solution:

*Remove **Teacher Name** and **Teacher Tel** together to create a new table.*

The Teacher Tel is a nonkey attribute, and the Teacher Name is also a nonkey atttribute. But Teacher Tel depends on Teacher Name. It is called **transitive dependency**.

EXAMPLE 3NF

Teacher Name	Teacher Tel
Sok Piseth	012 123 456
Sao Kanha	0977 322 111
Chan Veasna	012 412 333
Chan Veasna	012 412 333
Pou Sambath	077 545 221

StudyID	Course Name	T.ID
1	Database	T1
2	Database	T2
3	Web Prog	T3
4	Web Prog	T3
5	Networking	T4

Done?
Oh no, it is still not in 1NF yet.
Remove Repeating row.

Note about primary key:

- In theory, you can choose Teacher Name to be a primary key.
- But in practice, you should add Teacher ID as the primary key.

ID	Teacher Name	Teacher Tel
T1	Sok Piseth	012 123 456
T2	Sao Kanha	0977 322 111
T3	Chan Veasna	012 412 333
T4	Pou Sambath	077 545 221

EXAMPLE 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?

How can you make it 1NF?

EXAMPLE 1 (CONT..)

- 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
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 the studentID no longer uniquely identifies each row. You now need to declare *studentID* **and** *subject* **together** to uniquely identify each row. So the new **key** is StudentID *and* Subject.

Is it 2NF?

EXAMPLE 1 (CONT..)

- **Studentname** and **address** are dependent on studentID (which is part of the key) This is good. But they are **not** dependent on *Subject* (the *other* part of the key)
- **And 2NF requires…**

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

EXAMPLE 1 (CONT..)

- 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…

EXAMPLE (CONT..)

- STUDENT TABLE (key = StudentID)

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

- RESULTS TABLE (key = StudentID+Subject)

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

But is it 3NF?

EXAMPLE 1 (CONT..)

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

Or

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

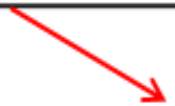
- But either way, non-key fields are dependent on MORE THAN THE PRIMARY KEY (studentID). And 3NF says that non-key fields must depend on **nothing but the key**

EXAMPLE 1 (CONT..)

StudentTable

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

Primary key: StudentID



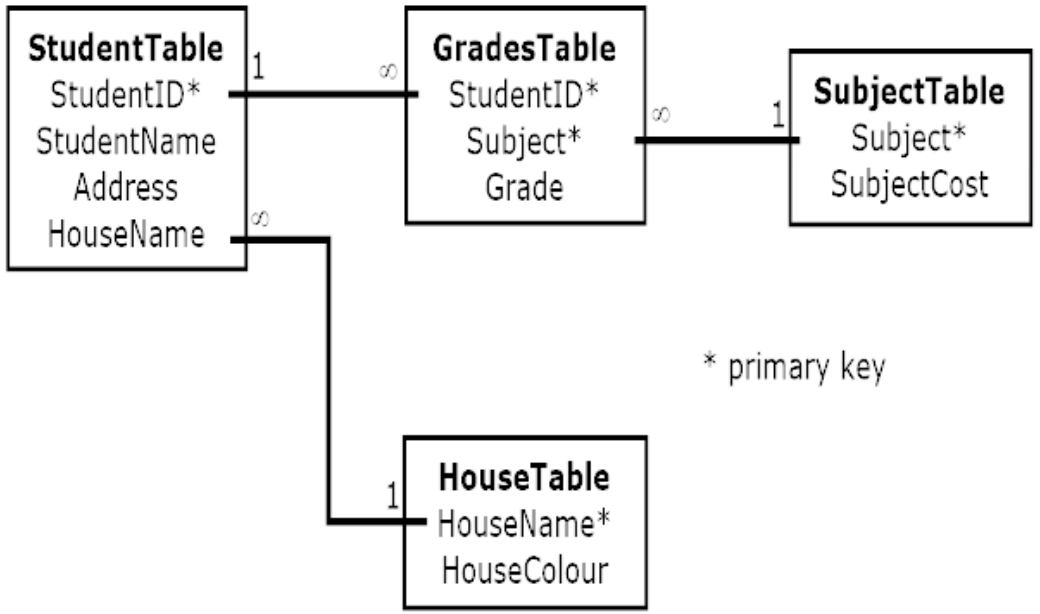
useTable

HouseName	HouseColor
Bob	Red

Primary key: HouseName

EXAMPLE 1 (CONT..)

- The Final Scheme



EXAMPLE 2

- We will use the Student_Grade_Report table below, from a School database, as our example to explain the process for 1NF.

Student_Grade_Report (StudentNo, StudentName, Major, CourseNo, CourseName, InstructorNo, InstructorName, InstructorLocation, Grade)

PROCESS FOR 1NF

- In the Student Grade Report table, the repeating group is the course information. A student can take many courses.
- Remove the repeating group. In this case, it's the course information for each student.
- Identify the PK for your new table.
- The PK must uniquely identify the attribute value (StudentNo and CourseNo).
- After removing all the attributes related to the course and student, you are left with the student course table (StudentCourse).
- The Student table (Student) is now in first normal form with the repeating group removed.
- The two new tables are shown below:

Student (StudentNo, StudentName, Major)

StudentCourse (StudentNo, CourseNo, CourseName, InstructorNo, InstructorName, InstructorLocation, Grade)

EXAMPLE 2 (CONT..)

Student (StudentNo, StudentName, Major)

StudentCourse (StudentNo, CourseNo, CourseName, InstructorNo, InstructorName, InstructorLocation, Grade)

- To move to 2NF, a table must first be in 1NF.
- The Student table is already in 2NF because it has a single-column PK.
- When examining the Student Course table, we see that not all the attributes are fully dependent on the PK; specifically, all course information. The only attribute that is fully dependent is grade.
- Identify the new table that contains the course information.
- Identify the PK for the new table.
- The three new tables are shown below.

EXAMPLE 2 (CONT..)

Student (StudentNo, StudentName, Major)

CourseGrade (StudentNo, CourseNo, Grade)

CourseInstructor (CourseNo, CourseName, InstructorNo, InstructorName, InstructorLocation)

PROCESS FOR 3NF

- Eliminate all dependent attributes in transitive relationship(s) from each of the tables that have a transitive relationship.
- Create new table(s) with removed dependency.
- Check new table(s) as well as table(s) modified to make sure that each table has a determinant and that no table contains inappropriate dependencies.
- See the four new tables below.

PROCESS FOR 3NF

Student (StudentNo, StudentName, Major)

CourseGrade (StudentNo, CourseNo, Grade)

Course (CourseNo, CourseName, InstructorNo)

Instructor (InstructorNo, InstructorName, InstructorLocation)

PROCESS FOR 3NF

- At this stage, there should be no anomalies in third normal form.

Student (StudentNo, StudentName, Major)

StudentCourse (StudentNo, CourseNo, CourseName, InstructorNo, InstructorName, InstructorLocation, Grade)