15CSE302 Database Management Systems

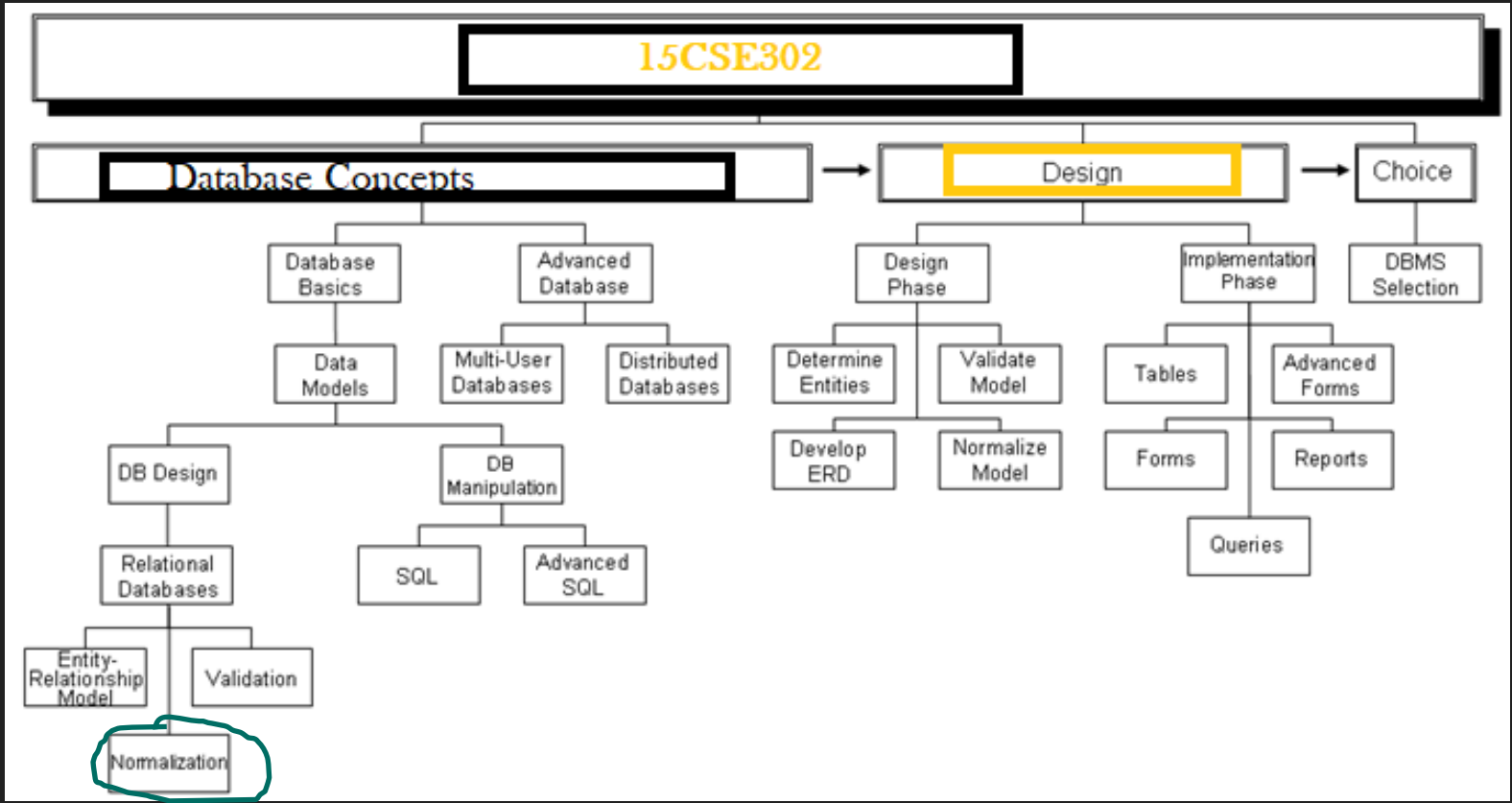
Lecture 19 Third Normal Form

B.Tech /III Year CSE/V Semester **L T P C 2 0 2 3**

DBMS Team
Dr G Jeyakumar
Bindu K R
Dr Priyanka Kumar
R. Manjusha
Department of CSE
Amrita School of Engineering

Slides Courtesy : Carlos Alvarado,
San Jose State University

Syllabus


15CSE302
Database Concepts
Design
Choice
Database Basics
Advanced Database
Design Phase
Implementation Phase
DBMS Selection
Data Models
Multi-User Databases
Distributed Databases
Determine Entities
Validate Model
Tables
Advanced Forms
DB Design
DB Manipulation
Develop ERD
Normalize Model
Forms
Reports
Relational Databases
SQL
Advanced SQL
Queries
Entity-Relationship Model
Validation
Normalization

Brief Recap of Previous Lecture

- Partial Dependency
- Second Normal Form

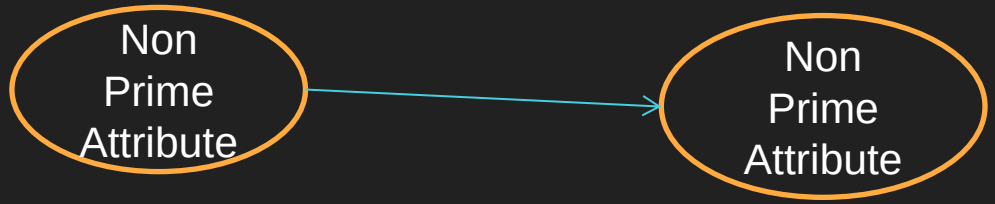
Today we'll discuss

Transitive Dependency

Third Normal Form

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



Example

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).

Example

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

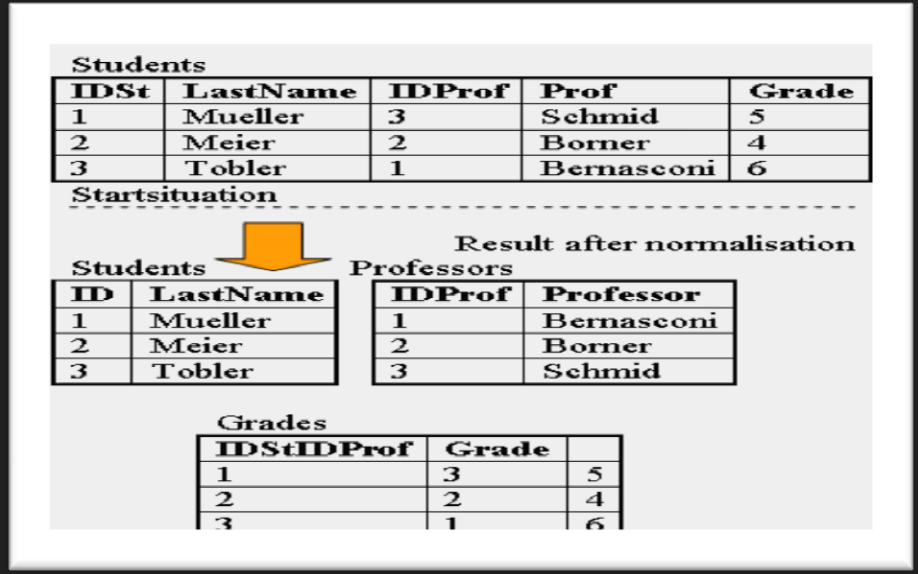
IDProf	Professor
1	Bernasconi
2	Borner
3	Schmid

Grades

IDStIDProf	Grade	
1	3	5
2	2	4
3	1	6

Example

- 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.



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

Example

- 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.

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

Third normal form (3NF)

A relation is in third normal form if

- It is in 2NF and
- No non key attribute is transitively dependent on the primary key.

Third normal form (3NF)

Example

A bank uses the following relation:

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

Third normal form (3NF)

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:

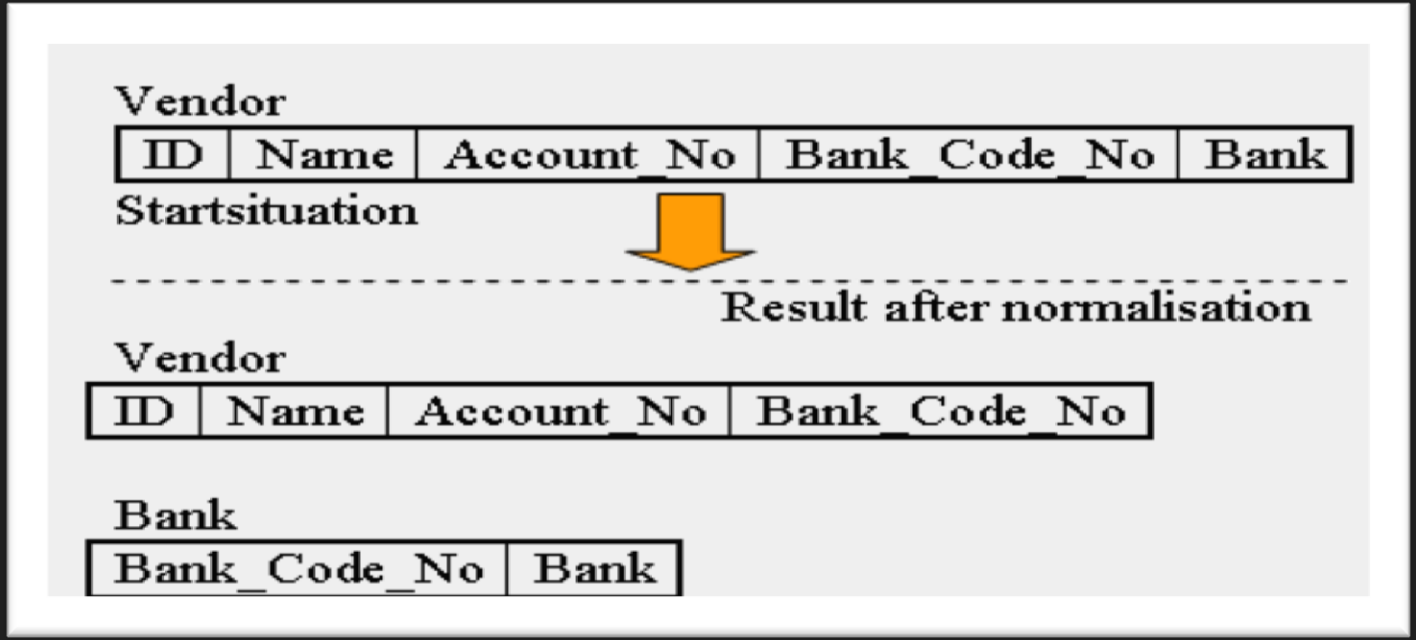
Vendor(ID, Name, Account_No, Bank_Code_No, Bank)

- Name, Account_No, Bank_Code_No are functionally dependent on ID
- ID -> Name, Account_No, Bank_Code_No

Bank is functionally dependent on Bank_Code_No

Bank_Code_No --> Bank

Third normal form (3NF)



Third normal form (3NF)

EmpId	EmployeeName	Gender	Salary	DeptName	DeptHead	DeptLocation
1	Sam	Male	4500	IT	John	London
2	Pam	Female	2300	HR	Mike	Sydney
3	Simon	Male	1345	IT	John	London
4	Mary	Female	2567	HR	Mike	Sydney
5	Todd	Male	6890	IT	John	London

Third normal form (3NF)

EmpId	EmployeeName	Gender	Salary	DeptName	DeptHead	DeptLocation
1	Sam	Male	4500	IT	John	London
2	Pam	Female	2300	HR	Mike	Sydney
3	Simon	Male	1345	IT	John	London
4	Mary	Female	2567	HR	Mike	Sydney
5	Todd	Male	6890	IT	John	London

We need to decompose the table into 2, and move the redundant department data (DeptName, DeptHead and DeptLocation) into it's own table.

To link the tables with each other, we use the DeptId foreign key.

The tables are in 2NF.

Table design in Second Normal Form (2NF)

DeptId	DeptName	DeptHead	DeptLocation
1	IT	John	London
2	HR	Mike	Sydney

EmpId	EmployeeName	Gender	Salary	DeptId
1	Sam	Male	4500	1
2	Pam	Female	2300	2
3	Simon	Male	1345	1
4	Mary	Female	2567	2
5	Todd	Male	6890	1

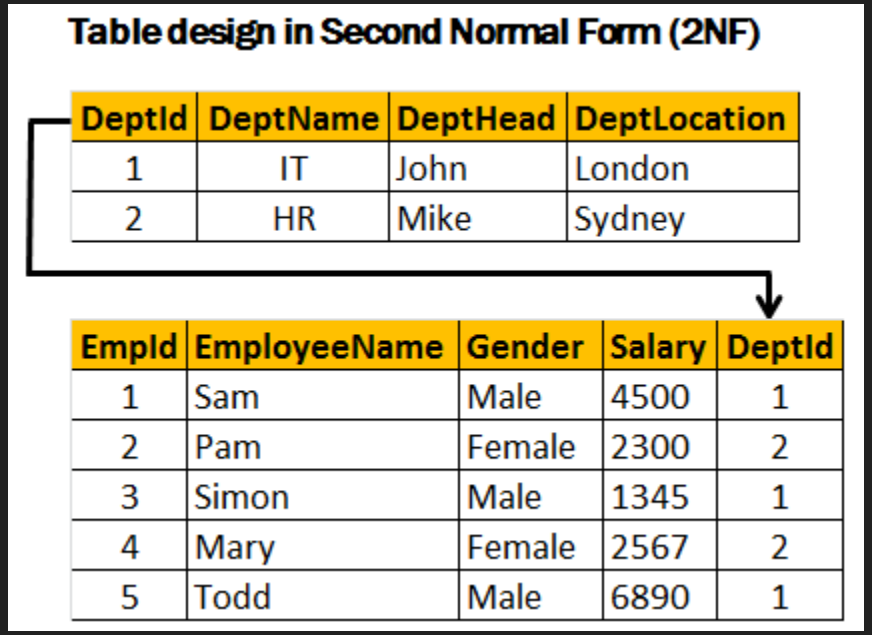
Third normal form (3NF)

EmpId	EmployeeName	Gender	Salary	DeptName	DeptHead	DeptLocation
1	Sam	Male	4500	IT	John	London
2	Pam	Female	2300	HR	Mike	Sydney
3	Simon	Male	1345	IT	John	London
4	Mary	Female	2567	HR	Mike	Sydney
5	Todd	Male	6890	IT	John	London

Tables are in 3NF

- It is in 2NF and
- No non key attribute is transitively dependent on the primary key.

Courtesy: https://csharp-video-tutorials.blogspot.com/2012/09/second-normal-form-and-third-normal.html?m=0



Trivial functional dependency

- The dependency of an attribute on a set of attributes is known as **trivial functional dependency** if the set of attributes includes that attribute.

A ->B is trivial functional dependency if B is a subset of A.

- The following dependencies are also trivial:

A->A & B->B

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	

Employee -Problems With This Table:

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	

- 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

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	

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

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	

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**
  - In order to store information for employees who take more than one course, a possible table structure is:

EMPLOYEE (<u>EmpId</u>, 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).

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	

- A better table structure (allowing for the fact that there is significant data redundancy) and a composite primary key of **EmpId + Course**, is:
- **EMPLOYEE (<u>EmpId</u>, Name, Dept, Salary, <u>Course</u>, DateTook, Fee)**

Normalisation

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
15	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
		101	John G. News	Database Designer	$105.00	19.4
		105	Alice K. Johnson *	Database Designer	$105.00	35.7
		106	William Smithfield	Programmer	$35.75	12.6
		102	David H. Senior	Systems Analyst	$96.75	23.8
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
		118	James J. Frommer	General Support	$18.36	45.3
		104	Anne K. Ramoras *	Systems Analyst	$96.75	32.4
		112	Darlene M. Smithson	DSS Analyst	$45.95	44.0

Normalisation

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
15	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
		101	John G. News	Database Designer	$105.00	19.4
		105	Alice K. Johnson *	Database Designer	$105.00	35.7
		106	William Smithfield	Programmer	$35.75	12.6
		102	David H. Senior	Systems Analyst	$96.75	23.8
18	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
		118	James J. Frommer	General Support	$18.36	45.3
		104	Anne K. Ramoras *	Systems Analyst	$96.75	32.4
		112	Darlene M. Smithson	DSS Analyst	$45.95	44.0

Table entries have data inconsistencies (note blank entries)

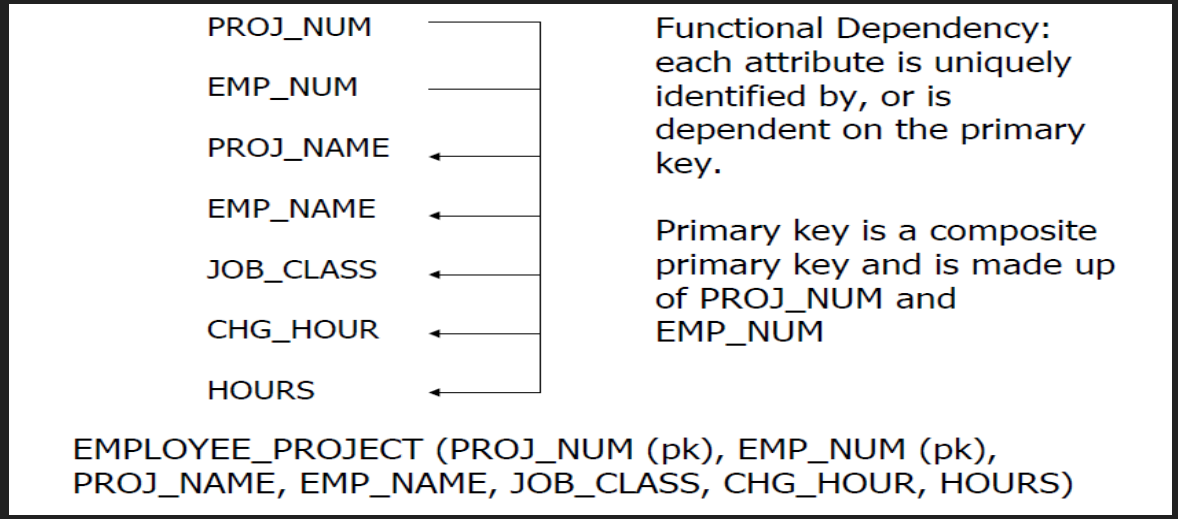
- Table displays data anomalies:
- Update - Modifying JOB_CLASS
- Insertion - New employee must be assigned project
- Deletion – If an employee is deleted, other vital data lost

Normalisation

PROJ_NUM	PROJ_NAME	EMP_NUM	EMP_NAME	JOB_CLASS	CHG_HOUR	HOURS
5	Evergreen	103	June E. Arbough	Elect. Engineer	$84.50	23.8
5	Evergreen	101	John G. News	Database Designer	$105.00	19.4
5	Evergreen	105	Alice K. Johnson *	Database Designer	$105.00	35.7
5	Evergreen	106	William Smithfield	Programmer	$35.75	12.5
5	Evergreen	102	David H. Senior	Systems Analyst	$96.75	23.9
8	Amber Wave	114	Annelise Jones	Applications Designer	$48.10	24.6
8	Amber Wave	118	James J. Frommer	General Support	$18.36	45.3
8	Amber Wave	104	Anne K. Ramoras *	Systems Analyst	$96.75	32.1
8	Amber Wave	112	Darlene M. Smithson	DSS Analyst	$45.95	44.0

Eliminate repeating groups – populate every cell of the ‘table’.

Dependency Diagram



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)

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).

- **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
- Now, in the table [TABLE_STORE], the column [Purchase Location] is fully dependent on the primary key of that table, which is [Store ID].

Conclusion

We have seen how

- Third Normal Form →Transitive Dependency

By implementing three of different levels of normalization called Normal Forms.

- The first three Normal Forms are usually sufficient for most small to medium size applications.

Next Lecture

Functional dependency Closure

References

- Hillyer Mike, MySQL AB. An Introduction to Database Normalization, http://dev.mysql.com/tech-resources/articles/intro-to-normalization.html, accessed October 17, 2006.

- Microsoft. Description of the database normalization basics, http://support.microsoft.com/kb/283878 , accessed October 17, 2006.

- Wikipedia. Database Normalization. http://en.wikipedia.org/wiki/Database_normalization.html , accessed October 17, 2006.
https://www.db-book.com/db6/index.html
- https://www.youtube.com/watch?v=mfVCesoMaGA&list=PLroEs25KGvwzmvIxYHRhoGTz9w8LeXek0&index=22

Thank You

Happy to answer any questions ! ! !