Normalization

20 October 2024 11:39

Normalization in database management system is the process of organizing data in a database to minimize redundancy and improve data integrity. It involves dividing a database into two or more tables and defining relationship between them to reduce duplication and ensure data dependencies make sense.

Problems caused by redundancy.

Storing the same information redundantly, that is, in more than one places within a database can lead to several problems.

- **Redundant storage:** Some information is stored repeatedly.
- **Update anomalies:** If one copy of such repeated data is updated and. inconsistency is created unless all copies are similarly updated.
- **Insertion anomalies:** It may not be possible to store some information unless some other information is stored as well.
- **Deletion anomalies:** It may not be possible to delete some information without losing some other information as well.

Functional Dependencies (FD)
A functional dependency in a database refers. to a relationship between two sets of attributes or columns within a table. Specifically of functional dependency, occurs when one attribute or a group of attributes uniquely determines another attribute. If knowing the values of one attribute allows you to know the value of another, then the 2nd attribute is set to be functionally dependent on the first.

Notation: A->B {Attribute A determines attribute B}

$$\left.\begin{array}{l} \text{Stud\_ID} \rightarrow \text{stud\_Name} \\ \text{Stud\_ID} \rightarrow \text{dep\_ID} \\ \boxed{\text{stud\_ID} \rightarrow (\underline{\text{stud\_Name, dep\_id}})} \end{array}\right\}$$

1. **Determinant:** The attribute or set of attributes on the left. On the left side of the functional dependency is called the determinant. It uniquely determines the value of the other attribute.
2. **Dependent:** The attribute on the right side. is called the dependent attribute. Its value is determined by the determinant.

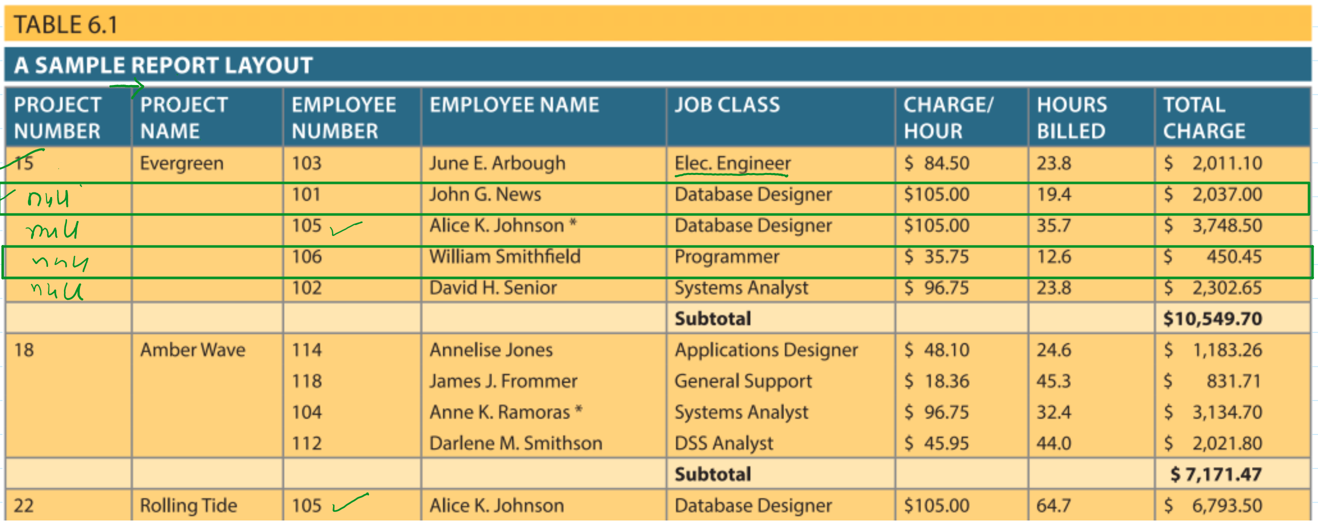


TABLE 6.1

A SAMPLE REPORT LAYOUT

PROJECT NUMBER	PROJECT NAME	EMPLOYEE NUMBER	EMPLOYEE NAME	JOB CLASS	CHARGE/ HOUR	HOURS BILLED	TOTAL CHARGE
15 ✓	Evergreen	103	June E. Arbough	Elec. Engineer	$ 84.50	23.8	$ 2,011.10
✓ null		101	John G. News	Database Designer	$105.00	19.4	$ 2,037.00
null		105 ✓	Alice K. Johnson *	Database Designer	$105.00	35.7	$ 3,748.50
null		106	William Smithfield	Programmer	$ 35.75	12.6	$ 450.45
null		102	David H. Senior	Systems Analyst	$ 96.75	23.8	$ 2,302.65
				Subtotal			**$10,549.70**
18	Amber Wave	114	Annelise Jones	Applications Designer	$ 48.10	24.6	$ 1,183.26
		118	James J. Frommer	General Support	$ 18.36	45.3	$ 831.71
		104	Anne K. Ramoras *	Systems Analyst	$ 96.75	32.4	$ 3,134.70
		112	Darlene M. Smithson	DSS Analyst	$ 45.95	44.0	$ 2,021.80
				Subtotal			**$ 7,171.47**
22	Rolling Tide	105 ✓	Alice K. Johnson	Database Designer	$105.00	64.7	$ 6,793.50

		112	Darlene M. Smithson	DSS Analyst	$ 45.95	44.0	$ 2,021.80
				Subtotal			**$ 7,171.47**
22	Rolling Tide	105 ✓	Alice K. Johnson	Database Designer	$105.00	64.7	$ 6,793.50
		104	Anne K. Ramoras	Systems Analyst	$96.75	48.4	$ 4,682.70
		113	Delbert K. Joenbrood *	Applications Designer	$48.10	23.6	$ 1,135.16
		111	Geoff B. Wabash	Clerical Support	$26.87	22.0	$ 591.14
		106	William Smithfield	Programmer	$35.75	12.8	$ 457.60
				Subtotal			**$13,660.10**
25	Starflight	107	Maria D. Alonzo	Programmer	$ 35.75	24.6	$ 879.45
		115	Travis B. Bawangi	Systems Analyst	$ 96.75	45.8	$ 4,431.15
		101	John G. News *	Database Designer	$105.00	56.3	$ 5,911.50
		114	Annelise Jones	Applications Designer	$ 48.10	33.1	$ 1,592.11
		108	Ralph B. Washington	Systems Analyst	$ 96.75	23.6	$ 2,283.30
		118	James J. Frommer	General Support	$ 18.36	30.5	$ 559.98
		112	Darlene M. Smithson	DSS Analyst	$ 45.95	41.4	$ 1,902.33
				Subtotal			**$17,559.82**
				Total			**$48,941.09**

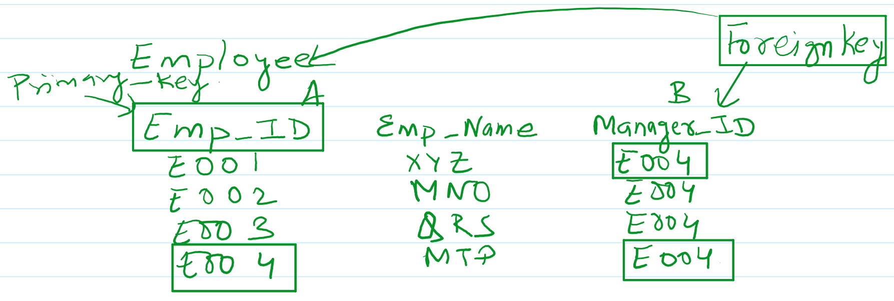
Note: A * indicates the project leader.

FUNCTIONAL DEPENDENCE CONCEPTS

CONCEPT	DEFINITION
Functional dependence	The attribute *B* is fully functionally dependent on the attribute *A* if each value of *A* determines one and only one value of *B*. Example: PROJ_NUM → PROJ_NAME (read as *PROJ_NUM functionally determines PROJ_NAME*) In this case, the attribute PROJ_NUM is known as the determinant attribute, and the attribute PROJ_NAME is known as the dependent attribute.
Functional dependence (generalized definition)	Attribute *A* determines attribute *B* (that is, *B* is functionally dependent on *A*) if all (generalized definition) of the rows in the table that agree in value for attribute *A* also agree in value for attribute *B*.
Fully functional dependence (composite key)	If attribute *B* is functionally dependent on a composite key *A* but not on any subset of that composite key, the attribute *B* is fully functionally dependent on *A*.

Types of functional dependencies:

1. **Trivial Functional dependency:** Of function dependency is trivial if the dependent attribute is a subset of the determinant. For example, {A,B}->A is a trivial functional dependency because knowing both A and B trivially determines A.
2. **Non-trivial functional dependency**: If the dependent attribute is not a subset of a determinant. it is nontrivial, functional dependency. For example, A->B is non-trivial if is not part of A.



Emp_ID	Emp_Name	Manager_ID
E001	XYZ	E004
E002	MNO	E004
E003	QRS	E004
E004	MTP	E004

$$\{A, B\} \rightarrow A$$

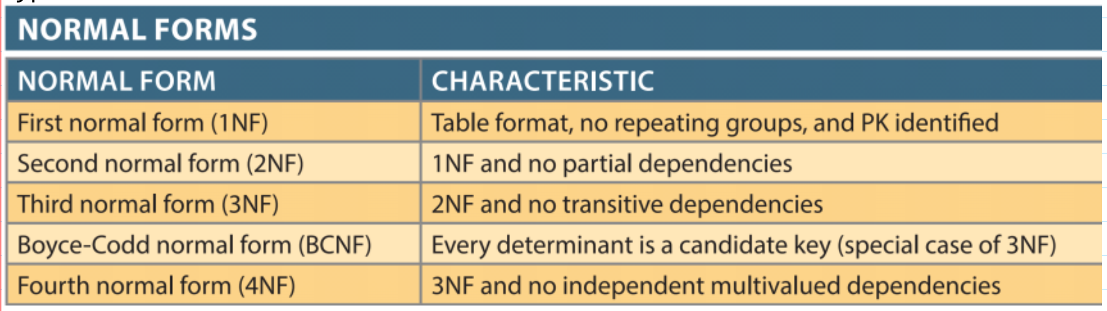
3. **Partial functional dependency:** In the case of composite keys. a partial dependency exist when. an attribute depends on. only part of the composite key rather than the whole key. For example, in a table with a composite key (A,B), if C depends only on A

3. **Partial functional dependency:** In the case of composite keys. a partial dependency exist when. an attribute depends on. only part of the composite key rather than the whole key. For example, in a table with a composite key (A,B), if C depends only on A but not on B this is a partial dependency.

$\text{Primary key} \rightarrow \boxed{\overset{A}{\text{Emp\_id}} \quad \overset{B}{\text{Email}}} \quad \boxed{\overset{C}{\text{EName}}}$

4. **Transitive functional dependency:** A transitive dependency occurs when an attribute depends indirectly on a determinant. For instance, if A->B and B->, then A->C is a transitive functional dependency.

Types of Normal Forms:



NORMAL FORMS	
NORMAL FORM	**CHARACTERISTIC**
First normal form (1NF)	Table format, no repeating groups, and PK identified
Second normal form (2NF)	1NF and no partial dependencies
Third normal form (3NF)	2NF and no transitive dependencies
Boyce-Codd normal form (BCNF)	Every determinant is a candidate key (special case of 3NF)
Fourth normal form (4NF)	3NF and no independent multivalued dependencies

What is Normalization?
It is a process for evaluating and correcting table structures to minimize data redundancies there. reducing the likelihood of data anomalies.

Normalization walks through a series of stages called normal forms. The. first three stages are described as first normal form or (1NF), second normal form (2NF) and third normal form 3NF.
From a structural point of view, 2NF is better than 1NF and 3NF is better than 2NF.

Denormalization:
Produce a lower normal form which is a 3NF will be converted to a 2NF through denormalization. A successful design must also consider end user demand for fast performance. Therefore, you will occasionally be expected to de-normalize some portion of database design in order to meet performance requirement.

The Normalization Process:
We will learn how to use normalization to produce a set of normalized table to store the data that will be used to generate the required information. The objective of normalization is to ensure that each table conforms to the concept of well-formed relation, that is, tables that have the following characteristics.

- Each payable represents a single subject for example. a course table will contain only data that directly pertains. to courses. similarly, a student table will contain only student data
- No data item will be unnecessarily stored in more than one table (In short, tables have minimal. controlled redundancy.) The reason for this requirement is to ensure that the data are updated in one place.
- All non-prime attributes in a table are dependent on the primary key. The reason for this requirement is to ensure that the data are uniquely identifiable by a primary key value.
- Each table is void of insertion update or deletion anomalies. This is to ensure the integrity and consistency of the data.

Conversion to First Normal Form(1NF)
Step-1:: Eliminate the Repeating Groups

Start by presenting the data in tabular format where each cell has a single value and there are no repeating groups. A repeating group derives its name from the fact that a group of multiple entries of the same type can exist for any single key attribute occurrence. To eliminate the repeating groups, eliminate the nulls by making sure that each repeating group attribute contains an appropriate data value.

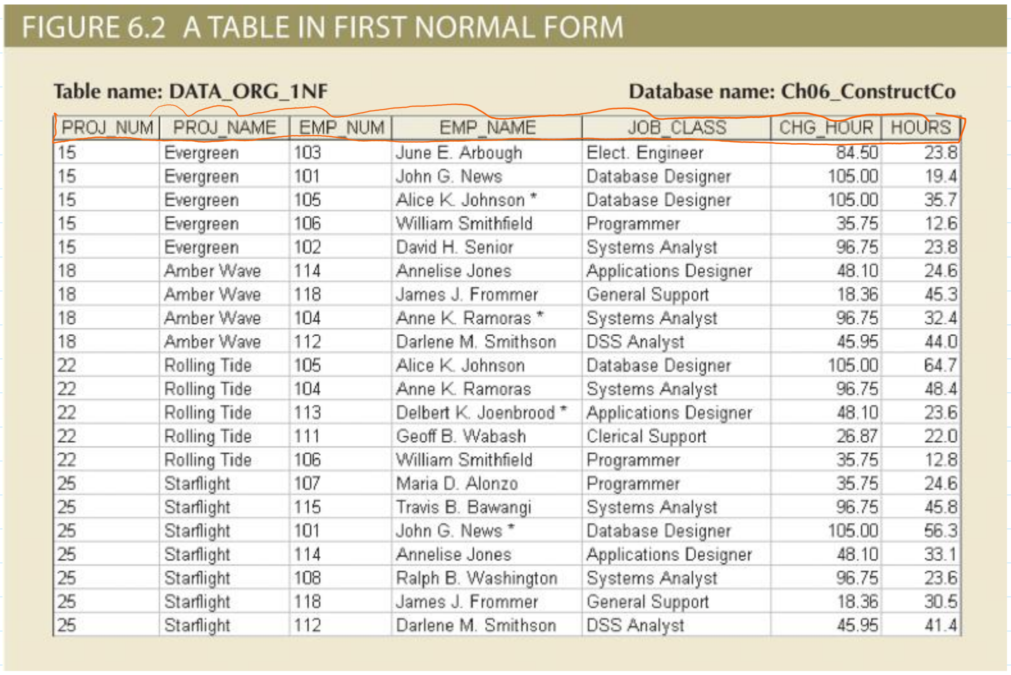


FIGURE 6.2 A TABLE IN FIRST NORMAL FORM

Table name: DATA_ORG_1NF Database name: Ch06_ConstructCo

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
15	Evergreen	101	John G. News	Database Designer	105.00	19.4
15	Evergreen	105	Alice K. Johnson *	Database Designer	105.00	35.7
15	Evergreen	106	William Smithfield	Programmer	35.75	12.6
15	Evergreen	102	David H. Senior	Systems Analyst	96.75	23.8
18	Amber Wave	114	Annelise Jones	Applications Designer	48.10	24.6
18	Amber Wave	118	James J. Frommer	General Support	18.36	45.3
18	Amber Wave	104	Anne K. Ramoras *	Systems Analyst	96.75	32.4
18	Amber Wave	112	Darlene M. Smithson	DSS Analyst	45.95	44.0
22	Rolling Tide	105	Alice K. Johnson	Database Designer	105.00	64.7
22	Rolling Tide	104	Anne K. Ramoras	Systems Analyst	96.75	48.4
22	Rolling Tide	113	Delbert K. Joenbrood *	Applications Designer	48.10	23.6
22	Rolling Tide	111	Geoff B. Wabash	Clerical Support	26.87	22.0
22	Rolling Tide	106	William Smithfield	Programmer	35.75	12.8
25	Starflight	107	Maria D. Alonzo	Programmer	35.75	24.6
25	Starflight	115	Travis B. Bawangi	Systems Analyst	96.75	45.8
25	Starflight	101	John G. News *	Database Designer	105.00	56.3
25	Starflight	114	Annelise Jones	Applications Designer	48.10	33.1
25	Starflight	108	Ralph B. Washington	Systems Analyst	96.75	23.6
25	Starflight	118	James J. Frommer	General Support	18.36	30.5
25	Starflight	112	Darlene M. Smithson	DSS Analyst	45.95	41.4

Step-2::Identify the primary key
Even casual observers will note that PROJ_NUM is not an adequate primary key because the project number does not uniquely identify all of the remaining entity (row) attributes. To maintain a proper primary key that will uniquely identify any attribute value. The new key must be. composed of combination of a PROJ_NUM and EMP_NUM.

Step-3::Identify All Dependencies:
The identification of the primary key in step 2 means that you have already identified the following dependencies:

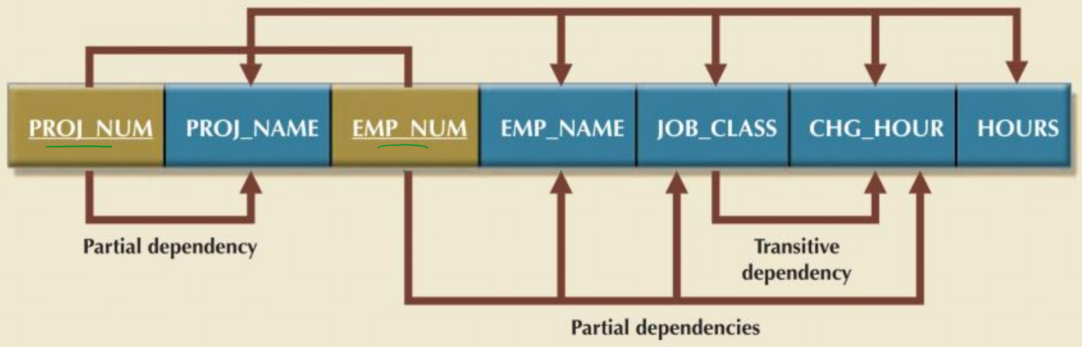
* PROJ_NUM, $\boxed{\text{EMP\_NUM} \rightarrow \text{PROJ\_NAME,}}$ EMP_NAME, JOB_CLASS. CHG_HOUR, HOURS.

* PROJ_NUM → PORJ_NAME

* EMP_NUM → (EMP_NAME, JOB_CLASS, CHG_HOUR)

* JOB_CLASS → CHG_HOUR.

FIGURE 6.3 FIRST NORMAL FORM (1NF) DEPENDENCY DIAGRAM



1NF (PROJ_NUM, EMP_NUM, PROJ_NAME, EMP_NAME, JOB_CLASS, CHG_HOURS, HOURS)

composite key

PARTIAL DEPENDENCIES:

(PROJ_NUM $\rightarrow$ PROJ_NAME)

(EMP_NUM $\rightarrow$ EMP_NAME, JOB_CLASS, CHG_HOUR)

} a dependency based on only a part of a composite Primary key.

TRANSITIVE DEPENDENCY:

(JOB_CLASS $\rightarrow$ CHG_HOUR)

↑Non-prime ↑Non-prime.

Step-1: Eliminate the repeating Groups.

Step-2: Identify the Primary-key

Step-3: Identify all dependencies.

The first normal form describes the tabular format in which:
- All of the key attributes are defined.
- There are no repeating groups in the table. In other words, each row or column intersection contains 1 and only 1 value, not a set of values.
- All attributes are dependent on the primary key.

The problem with the first normal form table structure is that it contains partial dependencies, while, partial dependencies are sometimes used for performance reason, they should be used with caution.

First normal or 1NF

A relation in 1NF if:
- All values in each column are atomic, meaning they are indivisible. (no repeating groups or arrays.)
- Each column contains values of single type.
- Each row uniquely identifies an instance of a data.

↑object.

In 1NF tables might. still contain redundancy and partial dependencies, which is addressed by advancing to higher normal forms.

Second Normal Form (2NF)

A relation is 2NF if:
1. It is already in 1NF.
2. It has no partial dependencies, meaning:
   a. Every non-key attribute must depend on the primary key not just part of it.

A **partial dependency** occurs when a non-key attribute depends on only part of a composite primary key, rather than the entire primary key. This issue is only present in tables where the primary key is composite (that is made up of more than one attribute) if the primary key is simple, or a single attribute, then the table in 1NF is automatically in 2NF.

Steps to convert 1NF to 2NF

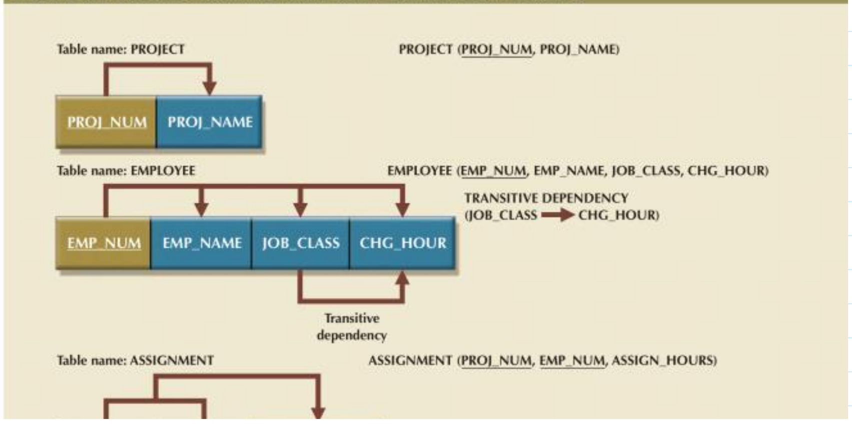
1. **Identify the primary key and non-key attributes:**
   i. **First, determine the primary key for the table, especially. if it is a composite key**
   ii. **Identify which attributes or columns are non-key attributes. Non key attributes are columns that are the part of the primary key.**
2. **Check for partial dependencies:**
   i. **For each non key attribute verify if it depends on the entire primary key.**
   ii. **If a non-key attribute only depends on a part of a composite key, then a partial dependency exists, and the table is not in 2NF.**
3. **Eliminate partial dependencies:**
   i. **If there are partial dependencies, decompose the table into two or more tables to eliminate them.**
   ii. **Each new table should have its own primary key and each nonkey attribute should fully depend on that key.**
4. **Re-establish relationship using foreign keys:**
   i. **If you have decomposed the original table into multiple tables add foreign keys as necessary to maintain relationship between them.**

To construct the revised dependency diagram, write each key component on a separate line and then write the original (composite) key on the last line. For example:
PROJ_NUM
EMP_NUM
PROJ_NUM EMP_NUM
Each component will become the key in a new table. In other words, the original table is now divided into three tables (PROJECT, EMPLOYEE, and ASSIGNMENT).

PROJECT (PROJ_NUM, PROJ_NAME)
EMPLOYEE (EMP_NUM, EMP_NAME, JOB_CLASS, CHG_HOUR)
ASSIGNMENT (PROJ_NUM, EMP_NUM, ASSIGN_HOURS)

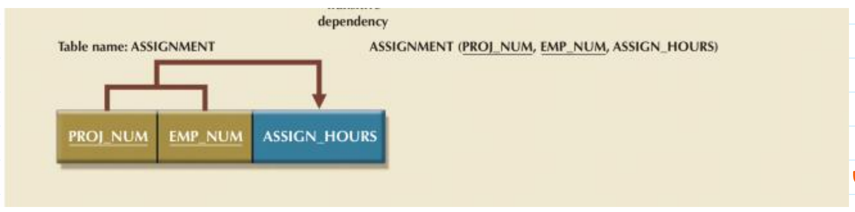
Because the number of hours spent on each project by each employee is dependent on both PROJ_NUM and EMP_NUM in the ASSIGNMENT table, you leave those hours in the ASSIGNMENT table as ASSIGN_HOURS. Notice that the ASSIGNMENT table contains a composite primary key composed of the attributes PROJ_NUM and EMP_NUM. Notice that by leaving the determinants in the original table as well as making them the primary keys of the new tables, primary key/foreign key relationships have been created. For example, in the EMPLOYEE table, EMP_NUM is the primary key. In the ASSIGNMENT table, EMP_NUM is part of the composite primary key (PROJ_NUM, EMP_NUM) and is a foreign key relating the EMPLOYEE table to the ASSIGNMENT table.

FIGURE 6.4 SECOND NORMAL FORM (2NF) CONVERSION RESULTS



No - partial dependencies are present

Hence, this is 2NF.



Second Normal Form (2NF)

A relation is in 2NF if:

1. **It is already in first normal form, or 1NF.**
2. **It has no partial dependencies, which means.:**
   - **Every non key attribute be fully functionally dependent on the entire primary key.**

In simpler terms, there should be no non key attribute depending on a part of a composite key. However, a table in 2NF can still have transitive dependency, which can, lead to redundancy, transitive dependencies are what 3NF aims to eliminate.

Third Normal Form (3NF)

A relation is in 3NF if:

1. **It is already in 2NF.**
2. **It has no transitive dependencies, meaning:**
   - **No non-key attribute should depend on another non-key attribute.**

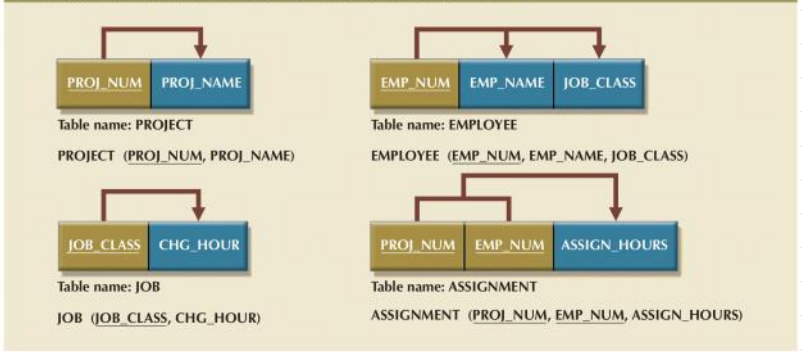
A **transitive dependency** occurs when a non-key attribute depends indirectly on the primary key through another non key attribute. If this happens, it usually means that certain non-key attributes depend on one another, rather than directly on the primary key, **creating potential redundancy** and **update anomalies.**

Steps to Convert 2NF to 3NF

1. **Identify the primary key and non-key attributes:**
   i. **Determine the primary key for the table.**
   ii. **List out the non-key attributes to check for any dependencies between them.**
2. **Identify transitive dependencies:**
   i. **Look for attributes that depend on other non-key attribute, rather than directly on the primary key.**
   ii. **If you find a non-key attribute that is functionally dependent on another non-key attribute, then transitive dependency exist.**
3. **Eliminate transitive dependencies:**
   i. **Decompose the table by creating separate tables for each transitive dependency.**
   ii. **Ensure that in each new table, non-key attributes depend only on the primary key of the table.**
4. **Reestablish relationship using foreign keys:**
   i. **If you decompose the original table into multiple tables, use foreign keys to maintain the relationship between them as needed.**

Draw a new dependency diagram to show all of the tables you have defined in Steps 1 and 2. Name the table to reflect its contents and function. In this case, JOB seems appropriate. Check all of the tables to make sure that each table has a determinant and that no table contains inappropriate dependencies. When you have completed these steps, you will see the results in Figure 6.5

FIGURE 6.5 THIRD NORMAL FORM (3NF) CONVERSION RESULTS



In other words, after the 3NF conversion has been completed, your database will contain four tables:

PROJECT (PROJ_NUM, PROJ_NAME)
EMPLOYEE (EMP_NUM, EMP_NAME, JOB_CLASS)
JOB (JOB_CLASS, CHG_HOUR)
ASSIGNMENT (PROJ_NUM, EMP_NUM, ASSIGN_HOURS)

Note that this conversion has eliminated the original EMPLOYEE table’s transitive dependency. The tables are now said to be in third normal form (3NF)