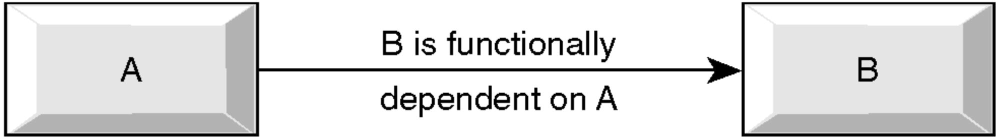
Functional dependency

Functional Dependency



- An attribute B is FUNCTIONALLY DEPENDENT on another attribute A, if a value of A determines a single value of B at any one time.

A → B

Examples:

- Emp_Num → Emp_Name
- Proj_Num → Proj_Name
  - Proj_Num is an independent variable, also know as the DETERMINANT
  - Proj_Name is a dependant variable

Types of Functional Dependency

- **Total Dependency**

  Attribute A determines B and attribute B determines A

  Example: Emp_Num ←→ Emp_Tax_File_Num

- **Full Dependency**

  An attribute Y is considered to have a full dependency to a SET of ATTRIBUTES X when:

  - Y is functionally dependent on SET X
  - Y is not functionally dependent on a proper SUBSET of X.

  X → Y

  Example: {Proj_Num, Emp_Num} → Hours

  X = {Proj_Num, Emp_Num} and

  Y = Hours.

Types of Functional Dependency

- **Partial Dependency**

  An attribute Y is considered to have a partial dependency to SET X when Y is functionally dependent on a proper subset of X.

  Example: {Proj_Num, Emp_Num} → Emp_Name

  X = {Proj_Num, Emp_Num} and

  Y = Emp_Name.

  - Emp_Name is said to be ***partially dependent*** on {Proj_Num, Emp_Num} because Emp_Name can be determined by only Emp_Num.
  - Emp_Name is determined by a subset of X, {Emp_Num}

Notes:

- Partial dependency exists only if the primary key is a composite key and the determinant is a part of the Primary Key.
- If primary key is NOT a composite key, then no partial dependency

Types of Functional Dependency

- **Transitive Dependency**
  - Y depends on X, and Z depends on Y, thus Z also depends on X:

$$X \rightarrow Y \rightarrow Z$$

Example: Emp_Num → Job_Class and Job_Class → Chg_Hour

This means, Emp_Num → Job_Class → Chg_Hour

X = Emp_Num,

Y = Job_Class and

Z = Chg_Hour.

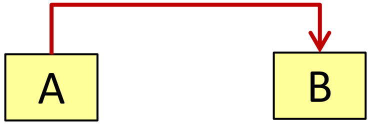
- Chg_Hour shows a ***transitive dependency*** to Emp_Num through Job_Class

Notes:

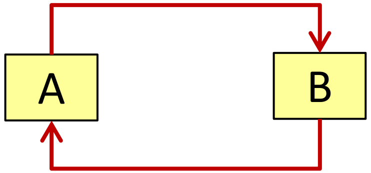
- Harder to determine than partial dependency
- Only occur if there is a functional dependency among non-keys

Dependency Summary

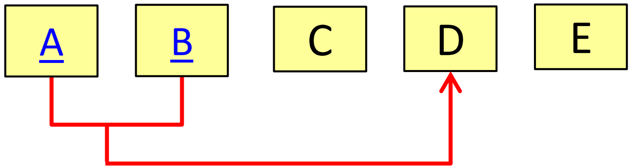
- **Functional**



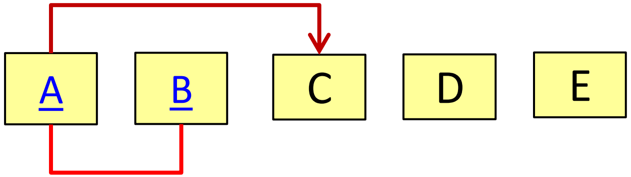
- **Total**



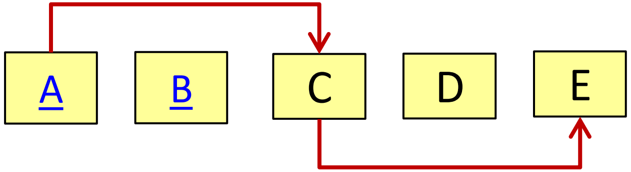
- **Full**



- **Partial**



- **Transitive**



Normalisation: UNF to 3NF

UNF = Un-Normalised From

- A relation (table) that contains repeating groups (or multiple entries for a single or part of a record)
    - Redundant (repeating) data is prone to the introduction of anomalies during update/delete and insert
- Raw data obtained from table/form/grid

For example, from the table shown before

ProjectAssign (Proj_Num, Proj_Name, Emp_Num, Emp_Name, Job_Class, Chg_Hour, Hours)

ProjectAssign	Proj_Num	Proj_Name	Emp_Num	Emp_Name	Job_Class	Charge_hour	Hours

1NF

Conditions of 1NF:

- It is a valid relation (in particular, no null entries)
- A unique key (primary key) has been identified for each row
- All attributes are functionally dependent on all or part of the key

1NF

Convert UNF to 1NF conversion

- **Step 1:** Identify the repeating group(s), if any, in the un-normalised relation.
  - Put valid values in all table entries

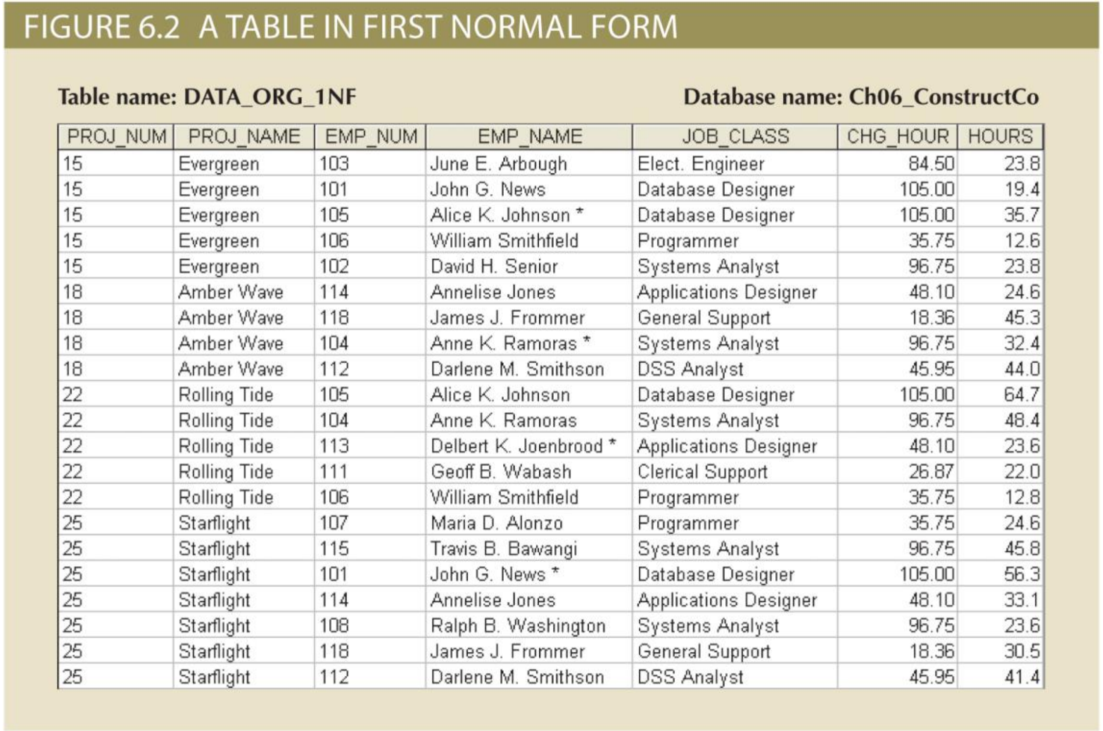


FIGURE 6.2 A TABLE IN FIRST NORMAL FORM

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

Convert UNF to 1NF conversion

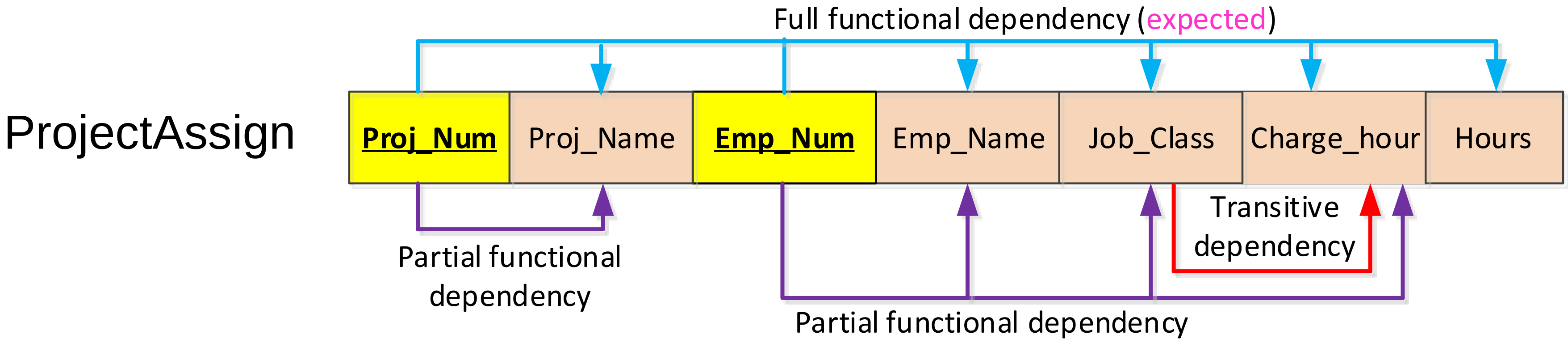
- **Step 2:** Determine the Primary Key in the table
  - If you cannot find a PK, insert a new column that can be a primary key

Proj_Num	Proj_Name	**Emp_Num**	Emp_Name	Job_Class	Charge_hour	Hours

Table: ProjectAssign

Convert UNF to 1NF conversion

- **Step 3:** Find all dependencies


Full functional dependency (expected)
ProjectAssign
Proj_Num
Proj_Name
Emp_Num
Emp_Name
Job_Class
Charge_hour
Hours
Partial functional dependency
Transitive dependency
Partial functional dependency

- Dependencies above the diagram are expected, and
- Those below are unexpected, so should be eliminated!

2NF

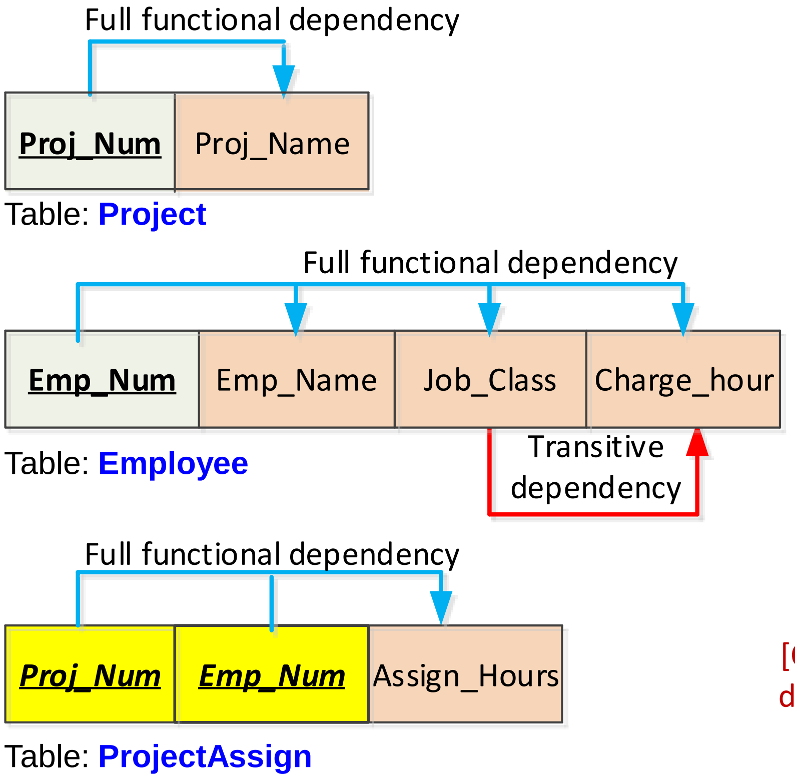
Conditions of 2NF:

- Relation is in 1NF and
- No partial dependency exists
  - So, if the primary key is a composite key, partial dependency may exist and we need to remove partial dependency to convert to 2NF
  - If the primary key is NOT a composite key, the table is already in 2NF!

  - Note: If right hand side is a key attribute, then above 2NF conditions are not violated. For example, if $\{\underline{X},\underline{Y}\} \rightarrow Z$ and $\underline{Y} \rightarrow Z$, but if Z is a part of a candidate key, then partial dependency does not occur.

Convert 1NF to 2NF conversion

- **Step 4:** Make new tables to eliminate partial dependencies
  - Make a new table for each part of the primary key that shows a partial dependency
  - Move all attributes which have partial dependencies into a new relation. The attribute(s) which these attributes are partially dependent on will be the PK in the new relation.
  - Leave that each part of the primary key as a foreign key in the original table to maintain relationships.



Full functional dependency

Proj_Num	Proj_Name

Table: **Project**

[New table!]

Full functional dependency

Emp_Num	Emp_Name	Job_Class	Charge_hour

Table: **Employee**

[New table!]

Full functional dependency

Proj_Num	***Emp_Num***	Assign_Hours

Table: **ProjectAssign**

[Original table decomposed!]

3NF

Conditions of 3NF:

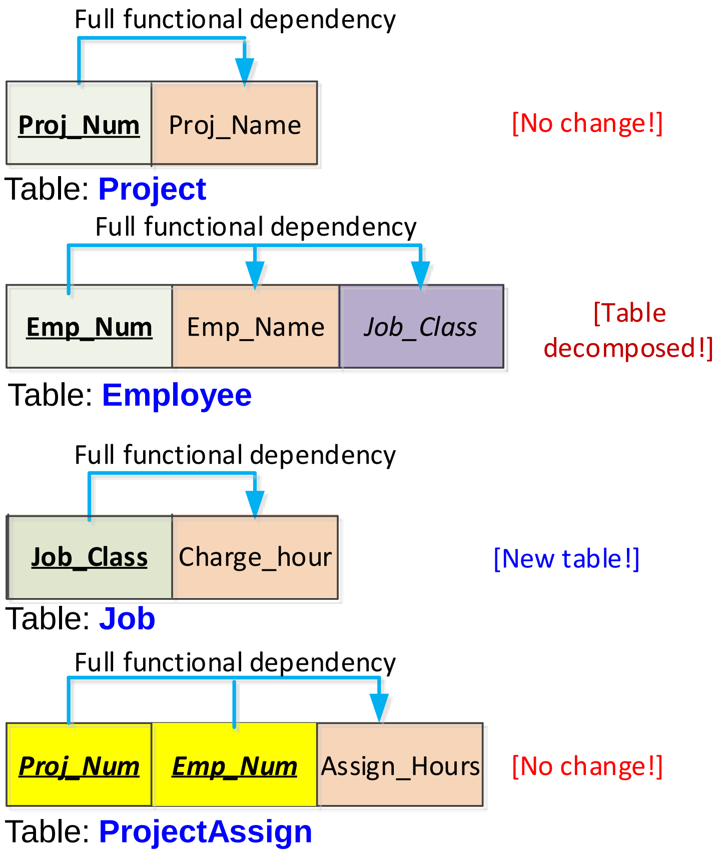
- Relation is in 2NF and
- No transitive dependency exists
  - So, if NO transitive dependency exists in a table, the table is already in 3NF!

  - Note: If right hand side is a key attribute, then above 3NF condition is not violated. For example, if X → Y, Y → Z and Z → X, so since Y & Z are candidate keys, then transitive dependency does not occur.

3NF

Convert 2NF to 3NF conversion

- **Step 5:** Make new tables to eliminate transitive dependencies
  - Make a new table for each non key attribute set that is a determinant for a transitive dependency
  - Move attributes which have transitive dependencies into a new relation. The attribute(s) which these attributes are transitively dependent on will be the PK in the new relation
  - Leave that non key attribute set as a foreign key in the original table to maintain relationships.



Full functional dependency

Proj_Num	Proj_Name

Table: **Project**

[No change!]

Full functional dependency

Emp_Num	Emp_Name	*Job_Class*

Table: **Employee**

[Table decomposed!]

Full functional dependency

Job_Class	Charge_hour

Table: **Job**

[New table!]

Full functional dependency

Proj_Num	***Emp_Num***	Assign_Hours

Table: **ProjectAssign**

[No change!]

Thank you