DATA REDUNDANCY

Redundancy exists when the same data are stored unnecessarily at different places.

Data anomaly develops when all of the required changes in the redundant data are not made successfully.

Table name: CUSTOMER
Primary key: CUS_CODE
Foreign key: none
Database name: Ch03_SaleCo

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE
10010	Ramas	Alfred	A	615	844-2573
10011	Dunne	Leona	K	713	894-1238
10012	Smith	Kathy	W	615	894-2285
10013	Olowski	Paul	F	615	894-2180
10014	Orlando	Myron		615	222-1672
10015	O'Brian	Amy	B	713	442-3381
10016	Brown	James	G	615	297-1228
10017	Williams	George		615	290-2556
10018	Farriss	Anne	G	713	382-7185
10019	Smith	Olette	K	615	297-3809

Table name: INVOICE
Primary key: INV_NUMBER
Foreign key: CUS_CODE

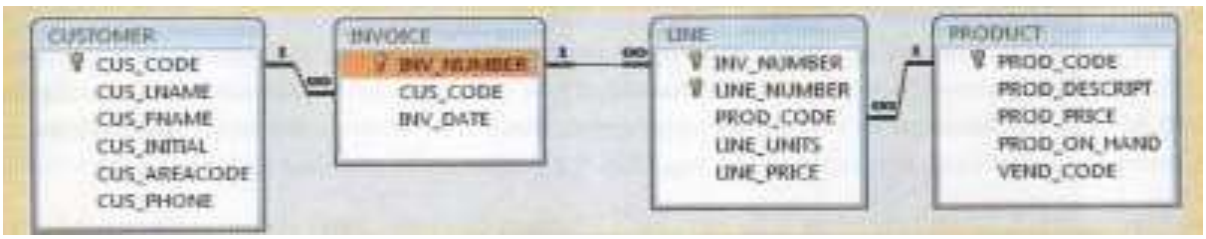
INV_NUMBER	CUS_CODE	INV_DATE
1001	10014	08-Mar-06
1002	10011	08-Mar-06
1003	10012	08-Mar-06
1004	10011	09-Mar-06

Table name: LINE
Primary key: INV_NUMBER + LINE_NUMBER
Foreign keys: INV_NUMBER, PROD_CODE

INV_NUMBER	LINE_NUMBER	PROD_CODE	LINE_UNITS	LINE_PRICE
1001	1	123-21UUY	1	189.99
1001	2	SRE-657UG	3	2.99
1002	1	GER-34256	2	18.63
1003	1	ZZX/3245Q	1	6.79
1003	2	SRE-657UG	1	2.99
1003	3	001278-AB	1	12.95
1004	1	001278-AB	1	12.95
1004	2	SRE-657UG	2	2.99

Table name: PRODUCT
Primary key: PROD_CODE
Foreign key: none

PROD_CODE	PROD_DESCRIPT	PROD_PRICE	PROD_ON_HAND	VEND_CODE
001278-AB	Claw hammer	12.95	23	232
123-21UUY	Houselite chain saw, 16-in. bar	189.99	4	235
GER-34256	Sledge hammer, 16-lb. head	18.63	6	231
SRE-657UG	Rat-tail file	2.99	15	232
ZZX/3245Q	Steel tape, 12-ft. length	6.79	8	235

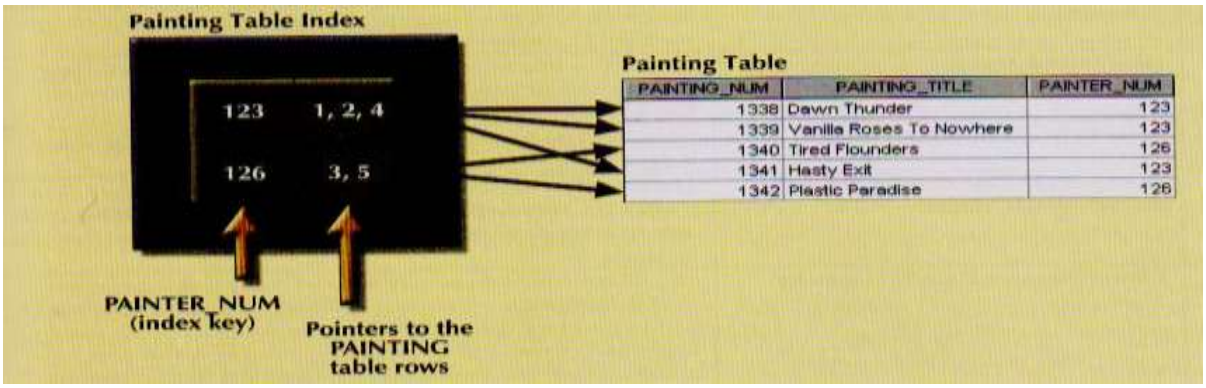


LINE_PRICE and PROD_PRICE are redundant and it is used to maintain the historical accuracy of the transactions.

LINE_NUMBER in LINE table is redundant .But given its automatic generation ,the redundancy is not a source of anomalies, and the order of the retrieved invoicing data will always match the order in which the data were entered. if product codes are used as part of the primary key, indexing will arrange those product codes as soon as the voice is completed And the data are stored.

INDEXES

Index is an orderly arrangement used to logically access rows in a table and it is composed of an index key and a set of pointers. Each key points to the location of the data identified by the key.



DBMS use indexes for many different purposes:

- an index can be used to retrieve data more efficiently .
- indexes can also be used by a DBMS to retrieve data ordered by a specific attribute or attributes.
- An index key can be composed of one or more attributes.
- Indexes play an important role in DBMSs for the implementation of primary keys. when you define a table's primary key, the DBMS automatically creates a unique index on the primary key column you declared. A unique index, is an index in which the index key can have only one pointer value(row) associated with it.
- A table can have many indexes, but each index is associated with only one table.
- The index key can have multiple attributes (composite index).

NORMALIZATION

Is a process for evaluating and correcting table structures to minimize data redundancies, thereby reducing the likelihood of data anomalies.

Normalization works through a series of stages called normal forms. The first three stages are described as first normal form (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

Produces a lower normal form, that 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 denormalize some portions of database design in order to meet performance requirements

THE NEED FOR NORMALIZATION

In following example

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

We see in that example the structure of data set does not conform to the requirements of table nor does it handle data very well.
Consider the following deficiencies:

1. The project number (PROJ_NUM) is apparently intended to be primary key or at least a part of a PK, but it contains nulls.
2. The table entries invite data inconsistencies. For example the JOB_CLASS value "Elect. Engineer" might be entered as "Elect. Eng."
3. The table displays data redundancies. Those data redundancies yield the following anomalies:

   a. Update anomalies. Modifying the JOB_CLASS for employee number 105 requires (potentially) many alterations, one for each EMP_NUM=105.
   b. Insertion anomalies. Just to complete a row definition, a new employee must be assigned to a project. If the employee is not assigned, a phantom project must be created to complete the employee data entry
   c. Deletion anomalies. Suppose that only one employee is associated with a given project, if that employee leaves the company and the employee data are deleted , the project information will also be deleted .to prevent the loss of the project information ,a fictitious employee must be created just to save the project information.

THE NORMALIZATION PROCESS

We will learn how to use normalization to produce a set of normalized tables 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 relations, that is, tables that have the following characteristics:

_ Each table 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 minimum controlled redundancy).The reason for this requirement is to ensure that the data are update in only one place.

_ All nonprime 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 attributes occurrence .To eliminate the repeating groups ,eliminate the nulls by making sure that each repeating group attribute contains an appropriate data value.

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

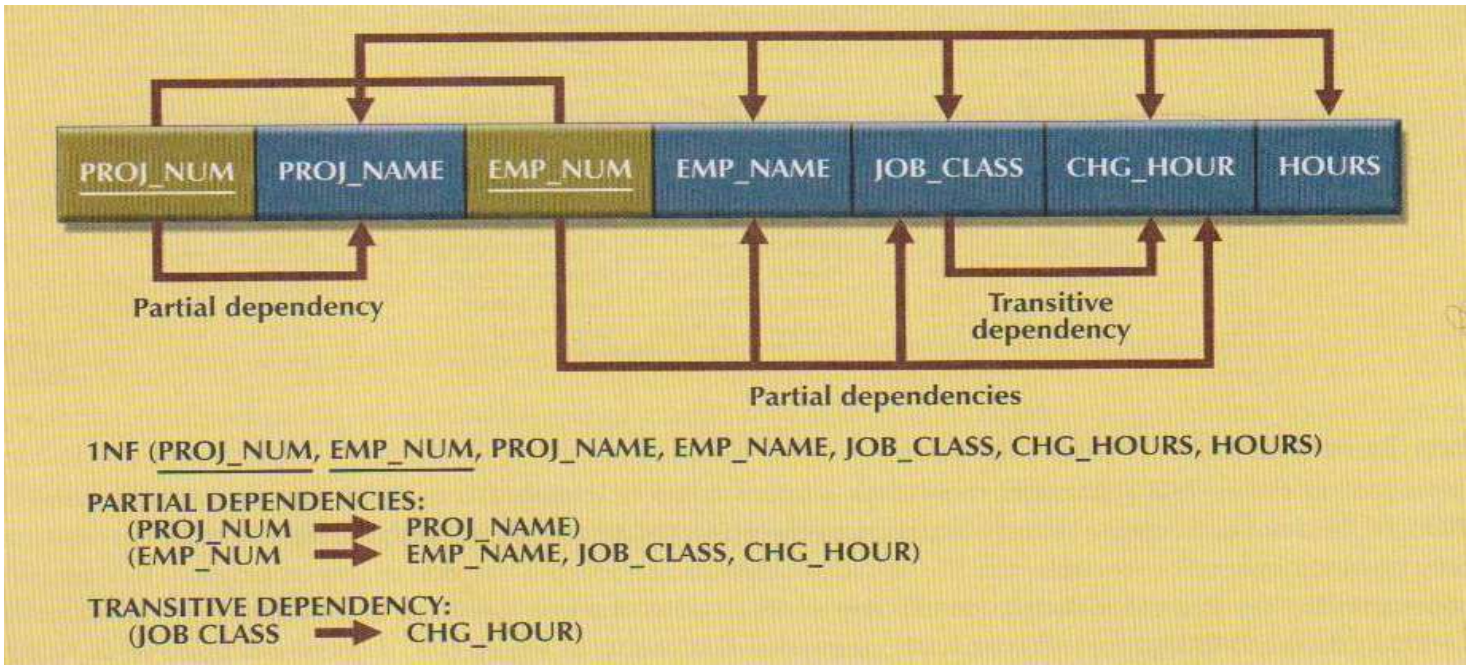
Step 2: Identify the primary key

Even a causal observer will not 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 compost of a combination of a PROJ_NUM and EMP_NUM

Step 3:Identify All Dependencies

The identification of the PK in Step 2 means that you have already identified the following dependency:

PROJ_NUM,EMP_NUM→PROJ_NAME,EMP_NAME,JOB_CLASS,CHG_HOUR,HOURS
PROJ_NUM →PROJ_NAME
EMP_NUM →EMP-NAME, JOB-CLASS,CHG-HOUR
JOB_CLASS→CHG_HOUR



The dependencies you have just examined can also be depicted with the help of the diagram known as dependency diagram

Partial dependency a dependency based only a part of a composite primary key

Transitive dependency is a dependency of one nonprime attribute on another nonprime attribute

The term first **normal form (1NF)** 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/column intersection contains one and only one value, not a set of values.
- All attributes are dependent on the primary key.

The problem with the 1NF table structure is that it contains partial dependencies.
While partial dependencies are sometimes used for performance reasons, they should be used with caution

Conversion to Second Normal Form(2NF)

Converting to 2NF is done only when the 1NF has a composite primary key. if the 1NF has a single attribute primary key, then the table is automatically in 2NF.The 1NF-to-2NF conversion is simple starting with

Step 1:Write Each Key Component on a Separate Line

Write each key component on a separate line; then write the original (composite) key on the last line.

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 in to three tables
(PROJECT ,EMPLOYEE, and ASSIGNMENT).

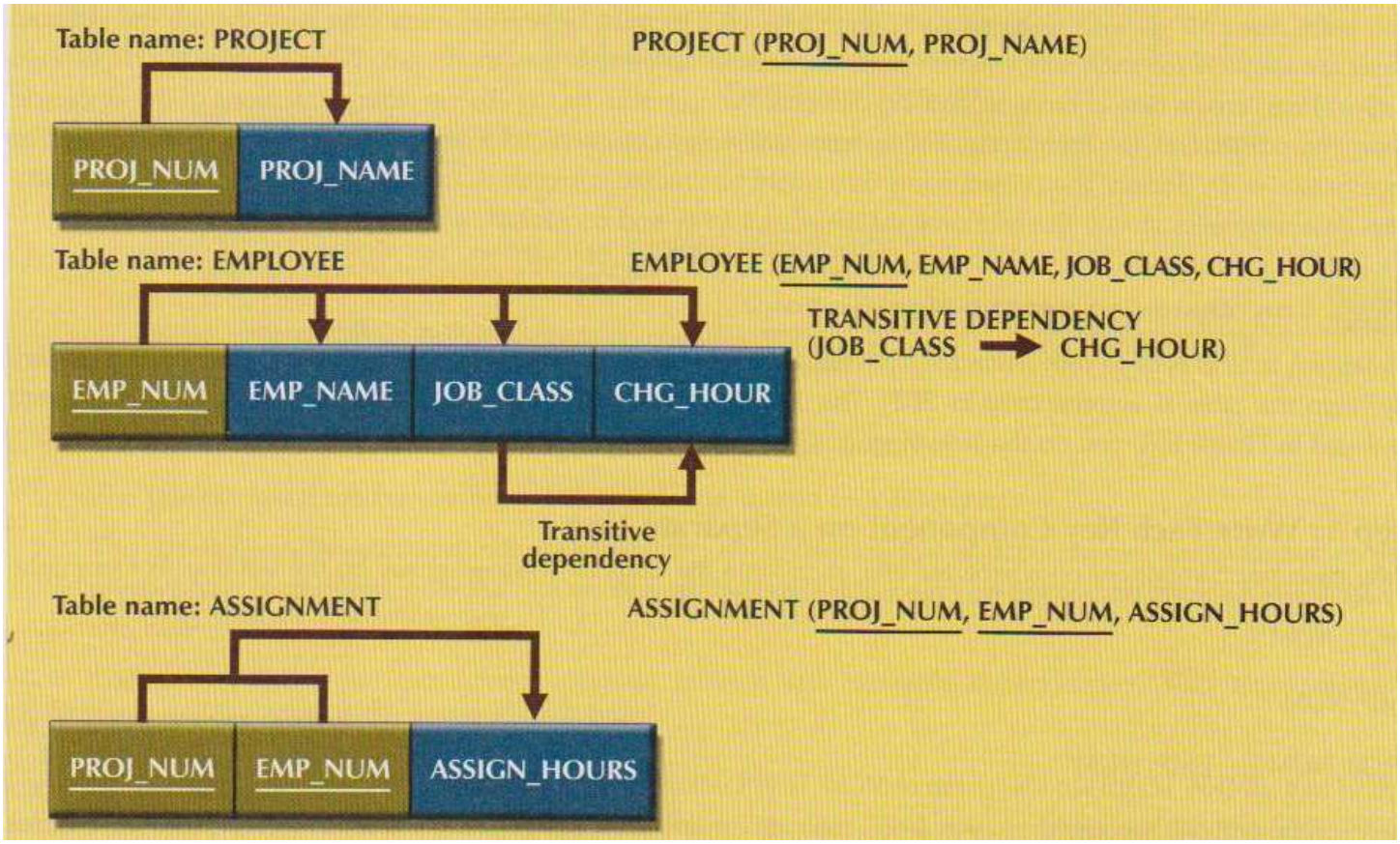
Step 2: Assign Corresponding Dependent Attributes

Use dependency diagram to determine those attributes that are dependent on other attributes

PROJECT(**PROJ_NUM,**PROJ_NAME)

EMPLOYEE**(EMP_NUM**,EMP_NAME,JOB_CLASS,CHG_HOUR)

ASSIGNMENT(**PROJ_NUM**,**EMP_NUM**, ASSIGN_HOURS)



A table is in second normal form (2NF)when

- it is in 1NF

And

- it includes no partial dependencies ;that is, no attribute is dependent on only portion of the primary key. Note that is still possible for a table in 2NF to exhibit transitive dependency; that is, one or more attributes may be functionally dependent on non key attributes.

Conversion to Third Normal

Step 1: Identify the Dependent Attributes

For every transitive dependency, write its determinant as PK for a new table.
JOB_CLASS

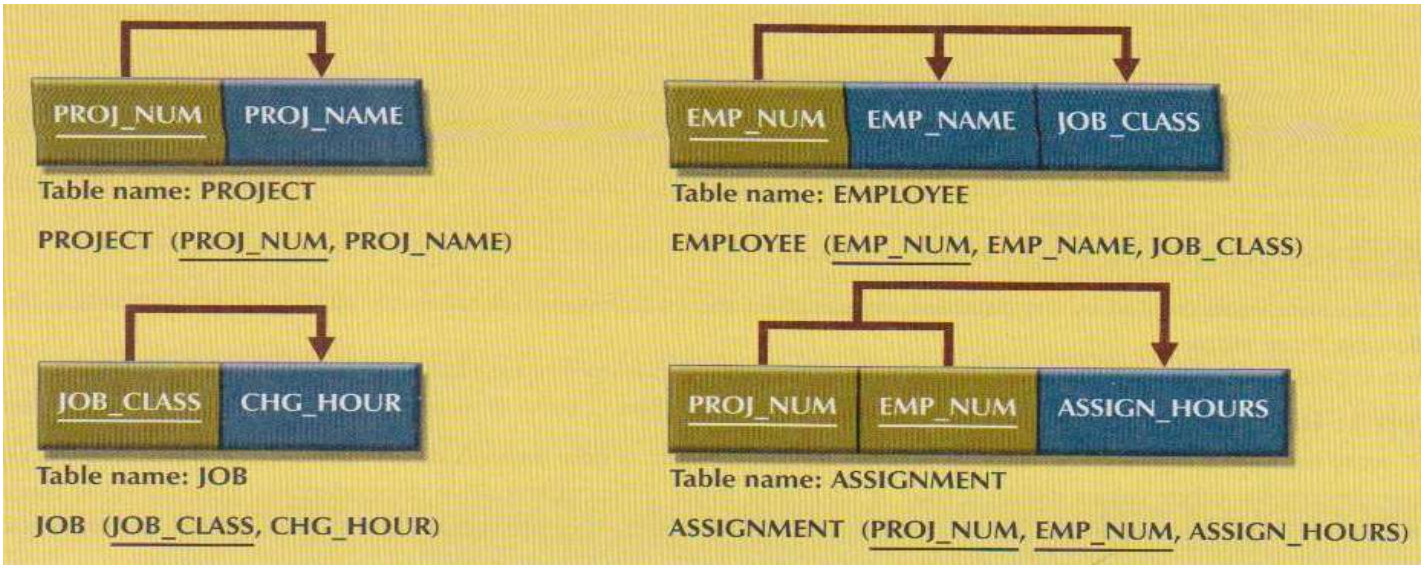
Step 2: Identify the Dependent Attributes

Identify the attributes that are dependent on each determinant identified in Step 1 and identify the dependency.
JOB_CLASS →CHG_HOUR
Name the table to reflect its contents and function. In this case, JOB seems appropriate.

Step 3: Remove the Dependent Attributes from Transitive Dependencies

Eliminate all dependent attributes in the transitive relationship(s) from each of the tables that have such a transitive relationship.
EMP_NUM → EMP_NAME,JOB_CLASS
Note that the JOB_CLASS remains in the EMPLOYEE table to save as FK.

After the 3NF conversion has been completed, your database contains four tables:



A table is in 3NF when

- It is in 2NF
- It contains no transitive dependencies