Chapter 3

Question 1:

Using the STUDENT and PROFESSOR tables shown in Figure Q3.8 to illustrate the difference between a natural join, an equijoin, and an outer join.

FIGURE Q3.8 The Ch03_CollegeQue Database Tables

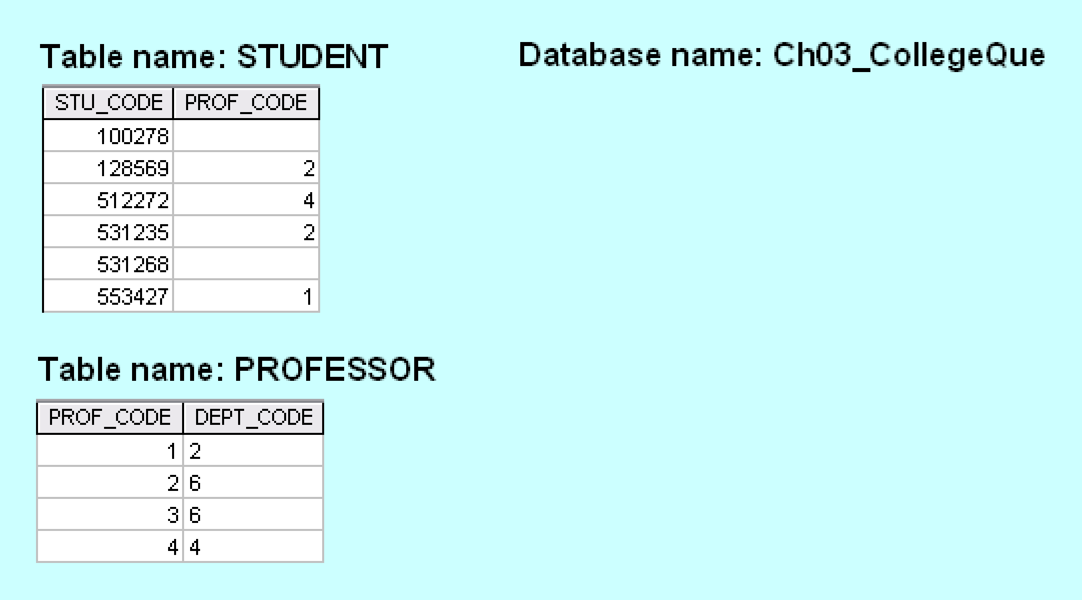


Table name: STUDENT

Database name: Ch03_CollegeQue

STU_CODE	PROF_CODE
100278	
128569	2
512272	4
531235	2
531268	
553427	1

Table name: PROFESSOR

PROF_CODE	DEPT_CODE
1	2
2	6
3	6
4	4

The natural JOIN process begins with the PRODUCT of the two tables. Next, a SELECT (or RESTRICT) is performed on the PRODUCT generated in the first step to yield only the rows for which the PROF_CODE values in the STUDENT table are matched in the PROF table. Finally, a PROJECT is performed to produce the natural JOIN output by listing only a single copy of each attribute. The order in which the query output rows are shown is not relevant.

STU_CODE	PROF_CODE	DEPT_CODE
128569	2	6
512272	4	4
531235	2	6
553427	1	2

The equiJOIN's results depend on the specified condition. At this stage of the students' understanding, it may be best to focus on equijoins that retrieve all matching values in the common attribute. In such a case, the output will be:

STU_CODE	STUDENT. PROF_CODE	PROFESSOR. PROF_CODE	DEPT_CODE
128569	2	2	6
512272	4	4	4

531235	2	2	6
553427	1	1	2

Notice that in equijoins, the common attribute appears from both tables. It is normal to prefix the attribute name with the table name when an attribute appears more than once in a table. This maintains the requirement that attribute names be unique within a relational table.

In the Outer JOIN, the unmatched pairs would be retained and the values that do not have a match in the other table would be left null. It should be made clear to the students that Outer Joins are not the opposite of Inner Joins (like Natural Joins and Equijoins). Rather, they are "Inner Join Plus" – they include all of the matched records found by the Inner Join **plus** the unmatched records. Outer JOINs are normally performed as either a Left Outer Join or a Right Outer Join so that the operator specifies which table's unmatched rows should be included in the output. Full Outer Joins depict the matched records plus the unmatched records from both tables. Also, like Equijoins, Outer Joins do not drop a copy of the common attribute. Therefore, a Full Outer Join will yield these results:

STU_CODE	**STUDENT. PROF_CODE**	**PROFESSOR. PROF_CODE**	**DEPT_CODE**
128569	2	2	6
512272	4	4	4
531235	2	2	6
553427	1	1	2
100278			
531268			
		3	6

A Left Outer Join of STUDENT to PROFESSOR would include the matched rows plus the unmatched STUDENT rows:

STU_CODE	**STUDENT. PROF_CODE**	**PROFESSOR. PROF_CODE**	**DEPT_CODE**
128569	2	2	6
512272	4	4	4
531235	2	2	6
553427	1	1	2
100278			
531268			

A Right Outer Join of STUDENT to PROFESSOR would include the matched rows plus the unmatched PROFESSOR row.

STU_CODE	**STUDENT. PROF_CODE**	**PROFESSOR. PROF_CODE**	**DEPT_CODE**
128569	2	2	6
512272	4	4	4
531235	2	2	6

553427	1	1	2
		3	6

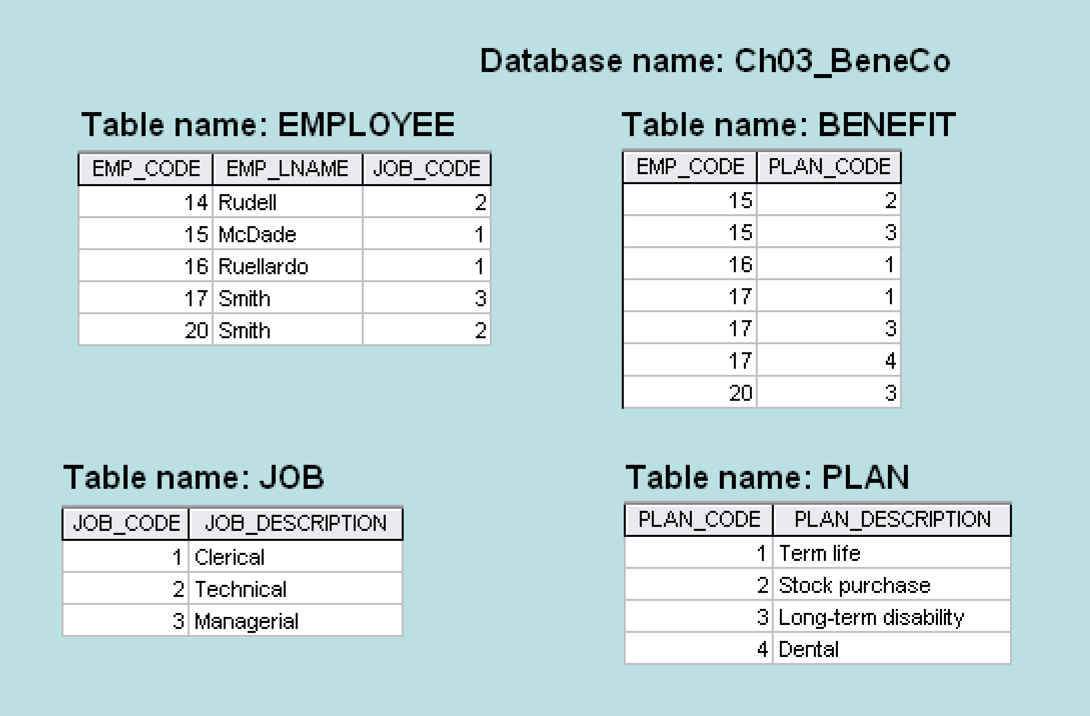
Question 2

Use the database shown in Figure P3.10 to work Problems 10–16. Note that the database is composed of four tables that reflect these relationships:

- **An EMPLOYEE has only one JOB_CODE, but a JOB_CODE can be held by many EMPLOYEEs.**
- **An EMPLOYEE can participate in many PLANs, and any PLAN can be assigned to many EMPLOYEEs.**

Note also that the M:N relationship has been broken down into two 1:M relationships for which the BENEFIT table serves as the composite or bridge entity.

FIGURE P3.10 The Ch03_BeneCo Database Tables



Database name: Ch03_BeneCo

Table name: EMPLOYEE

EMP_CODE	EMP_LNAME	JOB_CODE
14	Rudell	2
15	McDade	1
16	Ruellardo	1
17	Smith	3
20	Smith	2

Table name: BENEFIT

EMP_CODE	PLAN_CODE
15	2
15	3
16	1
17	1
17	3
17	4
20	3

Table name: JOB

JOB_CODE	JOB_DESCRIPTION
1	Clerical
2	Technical
3	Managerial

Table name: PLAN

PLAN_CODE	PLAN_DESCRIPTION
1	Term life
2	Stock purchase
3	Long-term disability
4	Dental

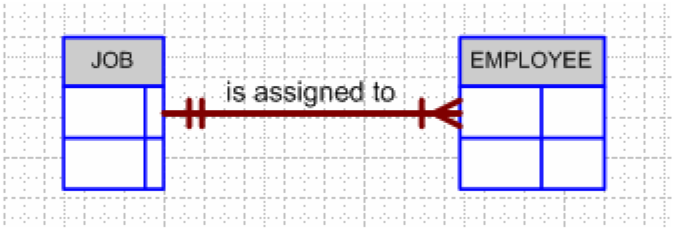
1. **For each table in the database, identify the primary key and the foreign key(s). If a table does not have a foreign key, write *None* in the assigned space provided.**

TABLE	PRIMARY KEY	FOREIGN KEY(S)
EMPLOYEE	EMP_CODE	JOB_CODE
BENEFIT	EMP_CODE + PLAN_CODE	EMP_CODE, PLAN_CODE
JOB	JOB-CODE	None
PLAN	PLAN_CODE	None

2. **Create the ERD to show the relationship between EMPLOYEE and JOB.**

The ERD is shown in Figure P3.11. Note that the JOB_CODE = 1 occurs twice in the EMPLOYEE table, as does the JOB_CODE = 2, thus providing evidence that a JOB can be assigned to many EMPLOYEEs. But each EMPLOYEE has only one JOB_CODE, so there exists a 1:M relationship between JOB and EMPLOYEE.

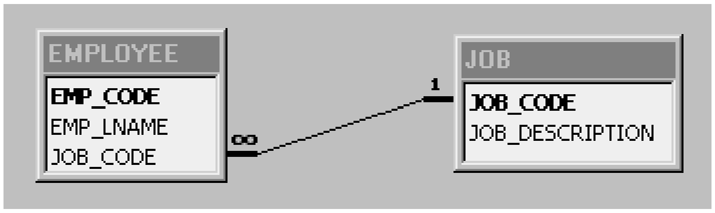
Figure P3.11 The ERD for the EMPLOYEE-JOB Relationship



3. **Create the relational diagram to show the relationship between EMPLOYEE and JOB.**

The relational schema is shown in Figure P3.12.

Figure P3.12 The Relational Diagram



4. **Do the tables exhibit entity integrity? Answer yes or no and then explain your answer.**

TABLE	ENTITY INTEGRITY	EXPLANATION
EMPLOYEE	Yes	Each EMP_CODE value is unique and there are no nulls.
BENEFIT	Yes	Each ***combination*** of EMP_CODE and PLAN_CODE values is unique and there are no nulls.
JOB	Yes	Each JOB_CODE value is unique and there are no nulls.
PLAN	Yes	Each PLAN_CODE value is unique and there are no nulls.

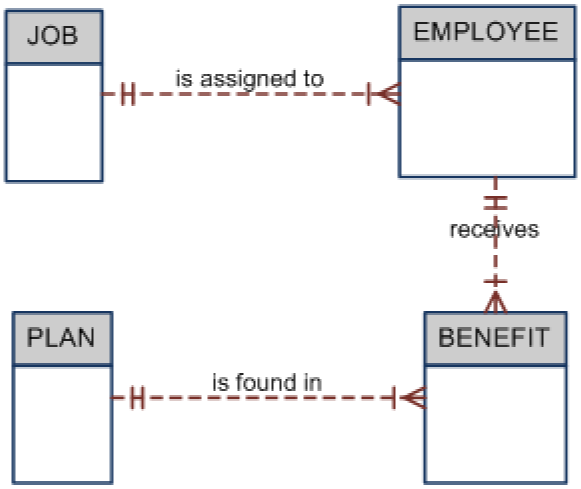
5. **Do the tables exhibit referential integrity? Answer yes or no and then explain your answer. Write *NA* (Not Applicable) if the table does not have a foreign key.**

TABLE	REFERENTIAL INTEGRITY	EXPLANATION
EMPLOYEE	Yes	Each JOB_CODE value in EMPLOYEE points to an existing JOB_CODE value in JOB.
BENEFIT	Yes	Each EMP_CODE value in BENEFIT points to an existing EMP_CODE value in EMPLOYEE and each PLAN_CODE value in BENEFIT points to an existing PLAN_CODE value in PLAN.
JOB	NA	
PLAN	NA	

6. **Create the ERD to show the relationships among EMPLOYEE, BENEFIT, JOB, and PLAN.**

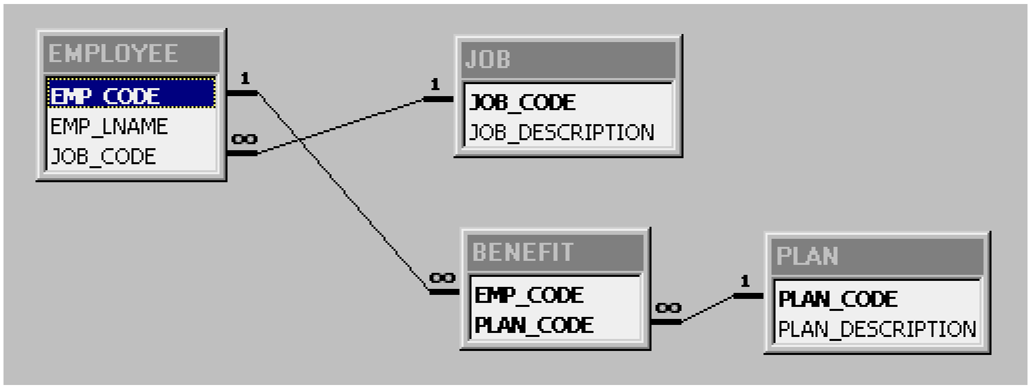
The Crow's Foot ERD is shown in Figure P3.15.

Figure P3.15 BeneCo Crow's Foot ERD



7. **Create the relational diagram to show the relationships among EMPLOYEE, BENEFIT, JOB, and PLAN.**
   The relational diagram is shown in Figure P3.16. Note that the location of the entities is immaterial – the relationships move with the entities.

Figure P3.16 The Relational Diagram


EMPLOYEE
EMP_CODE
EMP_LNAME
JOB_CODE
1
∞
JOB
1
JOB_CODE
JOB_DESCRIPTION
BENEFIT
∞
EMP_CODE
PLAN_CODE
∞
PLAN
1
PLAN_CODE
PLAN_DESCRIPTION