**Generic Tech Company**

Water Bottling Company database

database systems management report

Nawfel Belgasium

Daniel hayek

rayan abusrewil

Zyad al-azazi

dr. ramzi Haraty

CSC375

December 13, 2019

**I. System Description and Constraints:**

First off, the company that this database is built for has a main headquarters (HQ) branch that it is based out off. This was the first branch to be founded and holds importance in the company’s history. Throughout the years, multiple other branches have been opened all throughout the country, and the company now operates out of multiple locations. Each branch includes multiple departments which it oversees its function and work, with each department serving its own use as with any other company of such magnitude. Each branch also oversees its respective factory(s). Each branch is managed by one employee of the company (or in this case manager). When an employee works for a department or factory he/she by default works for Branch.

All departments have more than one phone number as to reach the different personnel in charge of that department.

Factories on the other hand are identified by its factory number. It is assumed that all factories use machinery. All machinery that is registered in the database is used only by the factories. Furthermore, each machine is identified by its serial number. The factory’s main function is to produce product and consume supplies, hence it is connected to both entities respectively.

The main base the company’s function revolves around is the signing of contracts.it is assumed in this database that each type of item is supplied or sold through the signing of a contract by a supplier or customer respectively. Hence the contract entity includes the Quantity attribute because one contract is designed to deal with one type of item no matter the quantity. Every contract is identified by its unique contract ID that is generated when the contract is made.

As soon as a supplier engages in dealings with the company, he is given a unique ID to represent the supplying company in the database. Only one contact person deals with the communication between the supplying company and our company.

The same goes for the customer, but in this case, we also included an attribute called loyalty points. As a gesture of appreciation, the company likes to reward its customers who stick with it throughout time. These loyalty points stack up with each purchase and result in gifts or discounts on products when an invoice is issued in the contract entity (represented as the rewards attribute in contract). If, however a specific customer is deleted from the database and then rejoins, his loyalty points are zeroed and start again.

In the case a customer does not purchase products from the company itself, he/she can buy the company’s products from local shops that belong to the company. These shops sell all the company products that can be bought from the company itself, however a customer is not contracted nor is he/she included in the database when such a purchase is made, hence it can be considered an “over the counter” purchase. Each shop is also managed by one employee of the company. These shops not only sell products made by the company, but also offer services such as refilling water or delivering water gallons to home.

**II. Entities:**

**1. Employee:**

Gender

Day

EMPLOYEE

Age

Month

Birthdate

Year

Phone\_number

Salary

Address

Employees in this water bottling company may have different roles within the company. Employee is a strong entity that has eight attributes, starting with the ID which is also the primary key for the entity. Each employee that works for this company has a unique ID that is given to her or him by the company and is used as a unique identification for each employee. Then comes the name which is a composite attribute made up of first, middle and last name. The gender is also another attributes that identifies the gender of the employee. Age is a derived attribute from the birthdate attribute which is in of itself a composite attribute that is composed of the day, month and year a specific employee was born on. Moreover, each employee may have a phone number or even more than one phone number, therefore the attribute phone number is multi valued. An employee also should have an address that is specified in the address attribute. Lastly, there is the salary attribute, which contains the amount of money that an employee makes in each month.

**2. Supply:**

SUPPLY

As with any company that has means of mass production, a steady flow of supplies is required. This entity exists to fulfill that purpose. It is here to keep track of supplied material independently of the supplier. This helps as the same supplier could supply different items, or multiple suppliers could be supplying the same item, and as such it is important to keep track of them separately. This entity has supply\_ID as its primary key, a unique serial number given to each piece of stock in the company warehouses. This entity also stores information on the type of supply or what material it’s made of, as well as having a name for each supply. Supply is also usually attached to contract as that is how the company and supplier decide on the price, amount etc.

**3. Supplier:**

Contact name

Name

Supplier\_ID

Contact\_Title

SUPPLIER

A supplier is another company or individual that supplies this water bottle company with the supplies. This supplier entity has a key consisting of supplier\_ID. The supplier also has a name to indicate the company being dealt with (if applicable), as well as a contact name and title to identify specifically the person being interacted with and their position within that company (if applicable). Location is a composite attribute that keeps track of where the supplies will be coming from.

**4. Customer:**

C\_ID

Name

Type

Loyalty points

CUSTOMER

For each customer entity, there are 4 attributes the C\_ID (customer ID), name, type and loyalty points for each Customer entity. The primary key is the C\_ID that is used to identify each customer. The customer ID is given to each customer by the company once they make a purchase. The name attribute contains the name of customer. The type attribute refers to whether the customer is a single customer or is a company. The loyalty points are used to give the customer some offers and discounts, considering the number of point they have. Each customer also has a location associated with them as it is essential to know where the product must be delivered. This attribute is composite, made up of country, city and street.

**5. Product:**

Month

Year

ProductionStart\_Date

Volume

Type

Production\_time

P\_name

Product\_ID

PRODUCT

The products that are produced by the company will all be included in the product entity, which hassix attributes these include the Product ID, P (product) Name, production start date, size, type and production time. The product ID is the primary key for the entity and product name is the name assigned by the company to the product. The other attribute is the production start date which is a composite attribute consisting of the month and year the product was first produced by the company. Then comes the size of the product in other words the length, width and height of the product which might be useful for transportation purposes. The type of the product is extremely important since the company is producing variety of products such as plastic water bottles, glass water bottles, metallic bottles, costumed bottles and etc. The production time is a derived attribute from the production start date. The purpose for this attribute is to provide exact time that was taken to produce a specific product.

**6. Factory:**

Factory\_nmb

NmbOfEmployees

FACTORY

  The previously mentioned supplies must be put to good use, and this is where that happens. The factories are the place where all the “dirty work” of the company is done. As this company is supposed to produce water on a large scale, it needs to have multiple factories in different locations to keep up with demand. Each factory has a unique number given to it, serving as its primary key (factory\_number), as well as each factory having a certain number of employees that work there. These factories are also the primary place where any machinery the company owns is used.

**7. Branch:**

BRANCH

B\_number

NoOfEmployees

Phone\_nmb

The company has different branches operating form different cities and countries around the world. Thus, each branch entity has 3 attributes. The B\_Number (branch number) is the branch number that is the primary key, the number of employees that work for each branch and the phone number of the branch. Naturally, knowing the location of each branch matters as well, and as such each branch has exact information regarding country, city and street.

**8. Machinery:**

SerialNumber

Type

Service

MACHINERY

In the modern day, physical hand labor has been found to be less efficient than outsourcing production to machines. A large-scale company that produces as much product a water-bottling company does needs a lot of machinery to automate production and increase productivity. But with the large number of machineries comes a need to keep track of all of them, including whether they are in working order or not, what type of machine it is, and they are all identified by a unique SerialNumber (primary key) to help differentiate similar machinery.

**9. Shops:**

SH\_ID

NoOfEmployees

Phone\_no

SHOP

Having multiple business ventures is essential for a company to adequately grow. Aside from simply shipping water bottles to stores and supermarkets, the water-bottling company that this database is built for also owns its own shops that it sells products out of. These shops not only sell bottled water, but also provide refilling services (water cooler gallons, bottles etc.). These shops each have a manager and a location obviously, however they also keep track of the employees working there and each shop has its own phone number. The attribute SH\_ID serves as the primary key for this attribute, as each shop will be assigned its own number. As usual, location is also stored for each shop with country, city and street being the attributes that are part of the composite attribute location.

**10. Contract:**

CONTRACT

Invoice

reward

Payment

Contract\_ID

StartDate

EndDate

Quantity

Item\_ID

This entity is what relates the company to its suppliers as well as its customers. Contracts are an essential part of any large corporation as they ensure that all the terms of a deal are met. It dictates the terms of any agreements that the company undertakes and as such must include a lot of details and technicalities like the start date and the end dates of the agreement (if they exist) as well as the agreed amount of product/supplies that are being provided. The item in question is stored as Item\_ID. The unique identifier used as a primary key is Contract\_ID to help identify the terms of the contract and who the contract was made with.

**11. Department:**

DEPARTMENT

No\_Employees

D\_Number

In each branch, there exist multiple departments, each with its own unique function. Each department has a certain number of employees dedicated to it, as well as possibly multiple phone numbers with different extensions, as such requiring a multivalued attribute to store that information. Each department is uniquely identified by D\_Number, and also has a name.

**12. Schedule:**

SCHEDULE

This weak entity is used to keep track of employee schedules. Since two employees could have similar start and end times, it is difficult to find a specific attribute or combination of attributes to use a primary key and as such it is left as a weak entity. It has attribute EndTime to show at what time the employee ended their shift, as well as an attribute to store the number of days worked besides any extra hours that they may have clocked in. The attribute StartTime serves as the key that combines with the foreign key coming from entity EMPLOYEE.

**III. Relationships*:***

**Employee -> \*has\* <- Schedule:**

An employee must operate under a specific schedule, and a schedule cannot exist on its own without an employee, as such the relation is “total to total”. All employees have one specific schedule made for them, and each schedule can only belong to one employee, hence the one to one aspect of the relation.

HAS

EMPLOYEE

SCHEDULE

1

1

SCHEDULE

**Employee -> \*manages 3\* <- Shop:**

One employee manages one shop in a certain location. Thus, the “manages” relation connects the “Employee” entity with the “Shop” entity and the participation constraint is “partial to total” from employee to shop respectively. Representing the fact that all shops are managed by an employee but not all employees are shop managers.

EMPLOYEE

SHOP

MANAGES3

1

1

**Employee -> \*manages 1\* <- Branch:**

One employee manages one branch. The “manages” relation connects the “Employee” entity to the “Branch” entity. The participation constraint is “partial to total,” as it is known that not all employees manage branches.

EMPLOYEE

BRANCH

MANAGES1

1

1

**Employee -> \*works for 1\* <- Department:**

Multiple employees may work for one department of any branch. Hence, the “works for” relationship connects the employee entity to the department they work at. Many employees can work in the same department; however, an employee cannot work for more than one department thus the “many to one” cardinality ratio. All departments have employees working for them but not all employees work for departments and thus the participation constraint is “partial to total.”

EMPLOYEE

DEPARTMENT

WORKSFOR1

1

N

**Employee -> \*works for 3\* <- Shop:**

Many employees work in one shop. As many employees work in one shop, all the shops have employees but not all employees work for shops. Thus, the cardinality of the relationship is “many to one” and the participation cardinality is “partial to total.”

EMPLOYEE

SHOP

WORKSFOR3

1

N

**Employee -> \*works for 2\* <- Factory:**

An employee works in a factory. Many employees may work for one factory, representing the “many to one” cardinality ratio. All factories have employees working for them, but not all employees necessarily work for factories. Hence, the participation constraint is “partial to total.”

EMPLOYEE

FACTORY

WORKSFOR2

1

N

**Employee -> \*manages 2\* <- Department**

An employee can serve as a manager for a certain department. Only one employee can serve as a manager for one specific department; hence, the cardinality ratio is “one to one.” All departments are managed by employees, whereas not all employees serve as managers for a department, thus the “partial to total” participation cardinality.

EMPLOYEE

DEPARTMENT

MANAGES2

1

1

**Customer -> \*signs 1\* <- Contract:**

A customer signs a contract with the company. All customers have a contract with the company and the company has only one contract with each supplier; thus, the cardinality ratio is “one to one.” The participation cardinality is “total to partial.”

CUSTOMER

CONTRACT

SIGNS1

1

1

**Supplier -> \*signs 2\* <- Contract:**

 A supplier signs a contract with the company. All suppliers have a contract with the company and the company has only one contract for each supplier, thus the cardinality ratio is “one- to- one.” The participation constraint is “total to partial” because not all contracts are for suppliers.

SUPPLIER

CONTRACT

SIGNS2

1

1

**Contract -> \*includes 1\* <- Supply:**

A contract includes the supply to be purchased by the company from a certain supplier. One contract can purchase one type of supplies but not every contract purchases a supply. Thus, the cardinality ratio is “one to one” and the participation cardinality is “partial to total.”

CONTRACT

1

1

SUPPLY

SUPPLY

**Contract -> \*includes 2\* <- Product:**

A contract includes the product to be sold by the company to the customer. One contract can sell one type of products but not every contract sells a product. Hence, the cardinality ratio is “one to one” and the participation cardinality is “partial to total.”

CONTRACT

PRODUCT

INCLUDES2

1

1

**Branch -> \*oversees 1\* <- Department:**

Each branch administers a number of different departments. One branch oversees multiple departments; thus, the cardinality ratio is “one to many” and the participation cardinality is “total to total” since all branches and all departments are involved in this relationship.

BRANCH

DEPARTMENT

OVERSEES1

N

1

**Branch -> \*oversees 2\* <- Factory:**

Each branch may administer one or more different factories. One branch oversees multiple factories; thus, the cardinality ratio is “one to many” and the participation cardinality is “total to total.”

BRANCH

FACTORY

OVERSEES2

N

1

**Factory-> \*utilizes\* <-Machinery:**

Each factory has and uses machines to carry out its daily and needed functions. Each factory can use many pieces of machinery, but the opposite does not apply, hence the cardinality ratio is “one to many”. Every factory uses machinery and every machinery is used by factories, which explains the “total to total” participation.

FACTORY

MACHINERY

UTILIZES

N

1

**Factory-> \*produces\* <-Product:**

A function of each factory is to produce products. Many factories produce many products. Every product is produced by factories, and all factories produce products, hence the participation cardinality is “total to total” and the cardinality ratio is “many to many.”

FACTORY

PRODUCT

PRODUCES

M

N

Quantity

**Factory-> \*consumes\* <-Supply:**

A function of each factory is to consume supplies to produce. Many factories consume many products. Every supply is consumed by factories, and all factories consume supplies, hence the cardinality ratio is “one to one” and the participation constraint is “total to total.”

FACTORY

SUPPLY

CONSUMES

M

N

**Shop -> \*sells\* <- product:**

A function of each shop is to sell products. Many shops sell many products. Every product is sold by shops and all shops sell products. Hence, the cardinality ratio is “many to many” and the participation cardinality is “total to total.”

SHOP

PRODUCT

SELLS

N

1

**Phase2:**

**Step1: Mapping Regular Entities**

The first step is creating the relation schemas for the regular entities; entities with primary keys. We map regular entities by including all the attributes except for the multivalued ones and disassembling composite attributes into their simpler components.

**Employee:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | FirstName | MiddleName | LastName | Gender | B.Day | B.Month | B.Year | Salary | Address | Age |

The employee entity has seven attributes - four of which are simple, two which are composite, one that is multivalued which is not presented in this step and one that is derived. The attribute “ID” is the primary key. The employee’s name is displayed in the form of its simpler attributes: “FirstName”, “LastName” and “MiddleName.” The employee’s birthdate is also displayed in its three simpler attributes: “B.Day”, “B.Month” and “B.Year”.

**Product:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| P\_ID | ProductName | ProductionMonth | ProductionYear | Volume | Type | ProductionTime |

The product entity has six attributes; one of these attributes is composite, which is the “ProductionStartDate”. This attribute’s simpler components are “ProductionMonth” and “ProductionYear.” The “ProductionTime” attribute is a derived attribute from the “ProductionStartDate” and it shows the number of years or months since the start of the production of a specific product.

**Customer:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| C\_ID | Name | Type | Loyalty points | Country | City | Street |

The customer entity has four simple attributes and one composite. The primary key is the “C\_ID.” The composite attribute “Location” is decomposed to its simpler attributes: “Country”, “City” and “Street.”

**Factory:**

|  |  |
| --- | --- |
| FactoryNumber | NoOfEmployees |

The factory entity has 2 simple attributes. The first one is the “FactoryNumber”, which is the primary key. The second attribute is the “NoOfEmployees” in the factory.

**Shop:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SH\_ID | No\_OfEmployees | Phone\_No | Country | City | Street |

The shop entity includes one composite attribute and three simple attributes. “Sh\_ID”, “No\_OfEmployees” and “Phone\_No”. The attribute “SH\_ID” is the primary key of this entity. The composite attribute “Location” is decomposed to its simpler attributes: “Country”, “City” and “Street.”

**Department:**

|  |  |  |
| --- | --- | --- |
| D\_number | D\_name | NoOfEmployees |

The Department entity is identified by three simple attributes and one multivalued which will not be mentioned here. “D\_number”, which serves as the primary key in this relation, “D\_name” and “NoOfEmployees”.

**Branch:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| B\_number | NoOfEmployees | Phone\_number | Country | City | Street |

The branch entity is characterized by three simple attributes and one composite. “B\_number” serving as the primary key, in addition to “NoOfEmployees” and “Phone\_number” of the branch. The composite attribute “Location” is decomposed to its simpler attributes: “Country”, “City” and “Street.”

**Supply:**

|  |  |  |  |
| --- | --- | --- | --- |
| Supply\_ID | SupplyName | Material | Type |

The entity supply consists of four simple keys. The primary key is the “Supply\_ID” attribute and the other keys are used in identifying the properties of the supply.

**Supplier:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SupplierID | Name | Contact\_name | Contact\_Title | Country | City | Street |

The Supplier entity contains four simple attributes and one composite. The primary key which is the “SupplierID”. The other three attributes are “Name”, “Contact\_name” and “Contact\_Title.” The composite attribute “Location” is decomposed to its simpler attributes: “Country”, “City” and “Street.”

**Contract:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ContractID | StartDate | EndDate | Quantity | Payment | Reward |

The contract entity contains five simple attributes and one composite, which is the “Invoice” attribute. The primary key for this entity is “ContractID.” The composite attribute “Invoice” is mapped as its basic attributes: “Payment” and “Reward.”

**Machinery**

|  |  |  |
| --- | --- | --- |
| SerialNumber | Type | Service |

All factories use a variety of machinery. The entity “Machinery” has three simple attributes. It has the “SerialNumber” as its primary key; in addition to, “Type” and “Service” as its two other attributes.

**Step 2: mapping of weak entity**

This section will concentrate on mapping the weak entity types. For each entity type in the ER schema, a relationship will be created that will contain all the simple attributes in the entity type excluding the multivalued attributes. For the composite attributes, all their constituent simple attributes will be included.

**SCHEDULE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EID | StartTime | EndTime | NumberOfDays | ExtraHours |

Since the schedule entity is a weak one, we add to it the primary key of the owner entity type (Employee) as a foreign key called “EID”. The primary key for schedule is the combination of “EID” and “StartTime,” which is the partial key of the weak entity type.

**Step 3: Mapping of 1:1 Binary relationship Type**

This step involves mapping of the 1:1 Binary relationship types using the forein key approach and merge relation approach. The foreign key approach chooses one of the relations, for example A, and include the primary key of another relation (B) as a foreign key in relation A.  All the attributes of the relation entity will be added to the relation A. The entity type with total participation will play the role of relation A. The merge relation approach will also be used when there is total participation by the two participating entity type. One of the relations will be chosen and it will contain all of the attributes of the other entity type and the relation entity.

**Manages1**

Branch

|  |  |  |  |
| --- | --- | --- | --- |
| B\_number | NoOfEmployees | Phone\_number | EID |

Because manages1 is a 1 to 1 relationship between employee and branch, the relation with the total participation which is the branch relation has to be chosen. The other relation “Employee” has only partial participation; therefore, its primary key is represented as foreign key called “EID” in Branch.

**Manages2**

Department

|  |  |  |  |
| --- | --- | --- | --- |
| D\_number | D\_name | NoOfEmployees | EID |

For this 1 to 1 relationship, we choose the “Department” entity type because it exhibits total participation. We include the employee primary as a foreign key in the “Department” relation.

We call this foreign key as “EID”. The phone number attribute is multivalued, therefore we don’t include it here.

**Manages3**

Shop

|  |  |  |  |
| --- | --- | --- | --- |
| SH\_ID | No\_OfEmployees | Phone\_No | EID |

This is a one to one relationship between “Shop” and “Employee” entities. The “Shop” entity has a total participation in the “Maneges3” relationship type, but the “Employee” has partial participation. Therefore, we choose “Shop” entity and then include the primary key of the “Employee” as a foreign key called “EID” in “Shop.”

**Includes1**

Supply

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Supply\_ID | SupplyName | Material | Type | ContractID |

The “Includes1” relationship has two participating entities.The “Supply” entity totally participates and as such we chose it. Afterwards, we add the primary key of “Contract,” which is, as foreign key to the “Supply” relation.

**Includes2**

Product

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P\_ID | ProductName | ProductionMonth | ProductionYear | Volume | Type | ProductionTime | ContractID |

The “Includes2” relationship has two participating entities. The “Product” shows total participation and for this reason we add the primary key of “Contract” to it as a foreign key under the name of “ContractID.”

**Has**

Employee

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First Name | MiddleName | Last Name | Gender | B.Day | B.Month | B.Year | Salary | Address | Age | startTime | End Time | numbOfDays | ExtraHours |

Since the participation constraint of this relationship is total for both sides, we use the merging approach by adding the attributes of the “Schedule” entity to the “Employee” relation schema.

**Step 4: Mapping 1:N Binary Relationship Type:**

This step is concerned with creating the schemas for relationships with cardinality ratio of 1:N. Mapping such relationships is achieved by adding the primary key of the relationship with ratio of (1) as a foreign key to the attributes of relationship with ratio of (N). In addition to any relationship attributes.

**Sells:**

Product

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P\_ID | ProductName | ProductionMonth | ProductionYear | Volume | Type | ProductionTime | contractID | SH\_ID |

In the “sells” relationship, the cardinality ratio of the product to shop is 1:N. This is why when mapping the relationship we add the primary key of the “Shop” relationship to the schema of the “Product” relationship as a foreign key.

**Utilizes:**

Machinery

|  |  |  |  |
| --- | --- | --- | --- |
| SerialNumber | Type | Service | Factory\_number |

The relationship of “utilizes” has a cardinality ratio of 1:N of the number of factories to the number of machineries. When mapping this relationship we include the primary key of the “Factory” entity next to the attributes of the “Machinery” entity.

**Works for 1:**

Employee

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First Name | MiddleName | Last Name | Gender | B. Day | B.  Month | B. Year | Salary | Address | Age |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| startTime | End Time | NumbOfDays | ExtraHours | D\_  Number |

The “Works for 1” relationship has a cardinality ratio of 1:N of the entity types “Department” and “Employee,” respectively. The relation should be mapped by adding the primary key “D\_Number” to the relation schema of “Employee”.

**Works for 2;**

Employee

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First Name | MiddleName | Last Name | Gender | B. Day | B.  Month | B. Year | Salary | Address | Age |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| startTime | End Time | NumbOfDays | ExtraHours | D\_  Number | Factory\_number |

The “Works for 2” relationship relation is created by adding the primary key of “Factory” (Factory\_number) to the “Employee” relation schema as a foreign key. The ratio of factories to employees is 1:N.

**Works for 3:**

Employee

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First Name | MiddleName | Last Name | Gender | B. Day | B.  Month | B. Year | Salary | Address | Age |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| startTime | End Time | NumbOfDays | ExtraHours | D\_  Number | Factory\_number | SH\_ID |

The “Works for 3” relationship has a cardinality ratio of 1:N of the entity types “Shop” and “Employee,” respectively. The relation should be mapped by adding the primary key “SH\_ID” to the relation schema of “Employee”.

**Oversees 1:**

Department

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| D\_number | D\_name | NoOfEmployees | EID | B\_Number |

The relationship “Oversees1” has a cardinality ratio of 1:N of the entity types of “Branch” and “Department,” respectively. The relation is mapped by adding the primary key of “Branch” which is “B\_Number” as a foreign key to the “Department” relationship schema.

**Oversees2:**

Factory

|  |  |  |
| --- | --- | --- |
| FactoryNumber | NoOfEmployees | B\_Number |

The cardinality ratio of the “Oversees2” relationship is 1:N of the entity types of “Branch” and “Factory,” respectively. We add the primary key “B\_Number” of “Branch” as a foreign key to the “Factory” relation schema.

**Signs1**

Contract

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ContractID | StartDate | EndDate | Quantity | Payment | Reward | C\_ID |

The cardinality ratio of the “Signs1” relation is 1:N of the entity types of “Customer” and “Contract”, respectively. We add the primary key “C\_ID” from “Customer” as a foreign key to the “Contract” relation schema.

**Signs2**

Contract

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ContractID | StartDate | EndDate | Quantity | Payment | Reward | C\_ID | Supplier\_ID |

The cardinality ratio of the “Signs2” relation is 1:N of the entity types of “Supplier” and “Contract”, respectively. We add the primary key “Supplier\_ID” as a foreign key to “Contract” relation schema.

**Step 5: Mapping M:N Binary Relationship Type:**

This step is concerned with mapping the relationships with ratios of M:N. The relation schemas of such relationships are obtained by including both primary keys of both entities in addition to any relationship attributes. The super key of this relation is composed of both of the primary keys of both attributes.

**Consumes:**

Consumes

|  |  |
| --- | --- |
| Supply\_ID | FactoryNumber |

The cardinality ratio of the “Consumes” relationship is M:N of the entity types “Supply” and “Factory.” We map the relation schema of the relationship “Consumes” by including both of the primary keys of both of the entities and make the super key of this relation a combination of both of the primary keys “Supply\_ID” and “FactoryNumber.”

**Produces:**

Produces

|  |  |  |
| --- | --- | --- |
| Product\_ID | FactoryNumber | Quantity |

Since the cardinality ratio of the “Produces” relationship is M:N of the entity types “Supply” and “Factory.,” we map the relation schema of the relationship “Consumes” by including both of the primary keys of both of the entities and make the super key of this relation a combination of both of the primary keys “Supply\_ID” and “FactoryNumber.” We also add the relationship attribute “Quantity” to the schema.

**Step 6: Mapping of Multivalued Attributes**

This step is concerned with multivalued attributes. The multivalued attributes are represented as a separate relation whose primary key is a combination of the primary key of entity type that the multivalued attribute belong to and  multivalued attribute itself.

**DEPT\_PHONENUMBER**

|  |  |
| --- | --- |
| Dnumber | DPhoneNumber |

This multivalued attribute belong to “Department” entity type. Therefore, when it is represented as a separate relation, the primary key of “Department” must be included in the relation. The primary key is a combination of “Dnumber” and “DPhoneNumber” which is the multivalued attribute itself.

**EMP\_PHONENUMBER**

|  |  |
| --- | --- |
| ID | PhoneNumber |

This multivalued attribute belong to “Employee” entity type. Therefore, when it is represented as a separate relation, the primary key of “Employee” must be included in the relation. The primary key is a combination of “ID” and “PhoneNumber” which is the multivalued attribute itself.

**Final displays**

EMPLOYEE

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First Name | MiddleName | Last Name | Gender | B\_ Day | B\_ Month | B\_Year | Salary | Address | Age |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| startTime | EndTime | NumbOfDays | ExtraHours | D\_ Number | Factory\_number | SH\_ID |

PRODUCT

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P\_ID | P\_Name | ProductionMonth | ProductionYear | Volume | Type | ProductionTime | ContractID | SH\_ID |

CUSTOMER

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| C\_ID | Name | Type | LoyaltyPoints | Country | City | Street |

FACTORY

|  |  |  |
| --- | --- | --- |
| FactoryNumber | NoOfEmployees | B\_Number |

DEPARTMENT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| D\_number | D\_name | NoOfEmployees | EID | B\_Number |

BRANCH

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| B\_number | NoOfEmployees | Phone\_number | EID | Country | City | Street |

SUPPLY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Supply\_ID | SupplyName | Material | Type | ContractID |

SUPPLIER

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SupplierID | Name | Contact\_name | Contact\_Title | Country | City | Street |

CONTRACT

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ContractID | StartDate | EndDate | Quantity | Payment | Reward | C\_ID | Supplier\_ID |

MACHINERY

|  |  |  |  |
| --- | --- | --- | --- |
| SerialNumber | Type | Service | Factory\_number |

SHOP

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SH\_ID | No\_OfEmployees | Phone\_No | EID | Country | City | Street |

CONSUMES

|  |  |
| --- | --- |
| Supply\_ID | FactoryNumber |

PRODUCES

|  |  |  |
| --- | --- | --- |
| Product\_ID | FactoryNumber | Quantity |

DEPT\_PHONENUMBER

|  |  |
| --- | --- |
| Dnumber | DPhoneNumber |

EMP\_PHONENUMBER

|  |  |
| --- | --- |
| E ID | EPhoneNumber |

**Phase3:**

1. **Table structures for GTC database**

* **Table structure for the EMPLOYEE table**

CREATE TABLE EMPLOYEE

(

ID CHAR(9) NOT NULL,

PRIMARY KEY(ID),

FirstName VARCHAR(20) ,

MiddleName VARCHAR(20) ,

LastName VARCHAR(20),

Gender CHAR(1) CHECK (Gender IN ('F', 'M', 'O')) NOT NULL,

B\_Day INT CHECK (B\_Day > -1 AND B\_Day <32),

B\_Month INT  CHECK (B\_Month > -1 AND B\_Month <13),

B\_Year INT CHECK (B\_Year >1965 AND B\_Year < 2001),

Salary FLOAT(5),

Address VARCHAR(50),

startTime VARCHAR(5) NOT NULL,

endTime VARCHAR(5) NOT NULL,

NumbOfDays INT ,

ExtraHours  INT DEFAULT 0,

D\_Number INT ,

Factory\_number INT ,

SH\_ID INT

);

And the following is added at the end so that employee will be able to relate to relations that are mentioned after it

ALTER TABLE EMPLOYEE ADD FOREIGN KEY (D\_Number) REFERENCES DEPARTMENT(D\_number)ON DELETE SET NULL;

ALTER TABLE EMPLOYEE ADD FOREIGN KEY (Factory\_number) REFERENCES FACTORY(FactoryNumber)ON DELETE SET NULL;

ALTER TABLE EMPLOYEE ADD FOREIGN KEY (SH\_ID) REFERENCES SHOP(SH\_ID) ON DELETE CASCADE;

* **Table structure for the BRANCH table**

CREATE TABLE BRANCH

(

B\_number INT PRIMARY KEY NOT NULL,

NoOFEmployees INT,

Phone\_number CHAR(10),

country VARCHAR(20),

city VARCHAR (20),

street VARCHAR(20),

EID CHAR(9),

FOREIGN KEY (EID) REFERENCES EMPLOYEE(ID)  ON DELETE SET NULL

);

* **Table structure for the DEPARTMENT table**

CREATE TABLE DEPARTMENT

(

D\_number INT PRIMARY KEY NOT NULL CHECK (D\_number > 0 AND D\_number < 30),

D\_name VARCHAR(20) NOT NULL,

NoOfEmployees INT,

EID CHAR(9) ,

B\_Number INT NOT NULL,

FOREIGN KEY (EID) REFERENCES EMPLOYEE(ID) ON DELETE SET NULL,

FOREIGN KEY (B\_Number) REFERENCES BRANCH(B\_number) ON DELETE CASCADE

);

* **Table structure for the SHOP table**

CREATE TABLE SHOP

(

SH\_ID INT PRIMARY KEY NOT NULL,

No\_OfEmployees INT,

country VARCHAR(20),

city VARCHAR (20),

street VARCHAR(20),

Phone\_No CHAR(10),

EID CHAR(9) NOT NULL,

FOREIGN KEY (EID) REFERENCES EMPLOYEE(ID)ON DELETE SET NULL

);

* **Table structure for the FACTORY table**

CREATE TABLE FACTORY

(

FactoryNumber INT PRIMARY KEY NOT NULL,

NoOfEmployees INT,

B\_Number INT DEFAULT '1' NOT NULL ,

FOREIGN KEY (B\_Number) REFERENCES BRANCH(B\_number)

);

* **Table structure for the CUSTOMER table**

CREATE TABLE CUSTOMER

(

C\_ID CHAR(9) PRIMARY KEY NOT NULL,

Name VARCHAR(50) UNIQUE,

Type VARCHAR(7) DEFAULT 'First' CHECK (Type IN ('First', 'Regular', 'Loyal')) NOT NULL,

LoyaltyPoints INT DEFAULT 0 ,

country VARCHAR(20),

city VARCHAR (20),

street VARCHAR(20)

);

* **Table structure for the SUPPLIER table**

CREATE TABLE SUPPLIER

(

Supplier\_ID CHAR(7) PRIMARY KEY NOT NULL,

Name VARCHAR(50) UNIQUE,

Contact\_name VARCHAR(50) ,

Contact\_Title VARCHAR(30),

country VARCHAR(20),

city VARCHAR (20),

street VARCHAR(20)

);

* **Table structure for the CONTRACT table**

CREATE TABLE CONTRACT

(

Contract\_ID CHAR(7) NOT NULL,

PRIMARY KEY(Contract\_ID),

StartDate CHAR(10) NOT NULL,

EndDate CHAR(10) NOT NULL,

Quantity INT,

Payment FLOAT(5) NOT NULL,

Reward FLOAT(5),

C\_ID CHAR(9),

Supplier\_ID CHAR(7),

FOREIGN KEY (C\_ID) REFERENCES CUSTOMER(C\_ID) ON DELETE SET NULL,

FOREIGN KEY (Supplier\_ID) REFERENCES SUPPLIER(Supplier\_ID) ON DELETE SET NULL

);

* **Table structure for the PRODUCT table**

CREATE TABLE PRODUCT

(

P\_ID CHAR(5) PRIMARY KEY NOT NULL,

ProductName VARCHAR(20) UNIQUE,

ProductionMonth INT CHECK (ProductionMonth > 0 AND ProductionMonth< 13),

ProductionYear INT CHECK (ProductionYear > 2000),

Volume FLOAT(5),

Type VARCHAR(20),

contractID CHAR(7) NOT NULL,

SH\_ID INT NOT NULL,

FOREIGN KEY (contractID) REFERENCES CONTRACT(Contract\_ID)ON DELETE SET NULL,

FOREIGN KEY (SH\_ID) REFERENCES SHOP(SH\_ID) ON DELETE SET NULL

);

* **Table structure for the MACHINERY table**

CREATE TABLE MACHINERY

(

SerialNumber CHAR(50) PRIMARY KEY NOT NULL,

Type CHAR (20),

Service CHAR(1) CHECK (Service IN ('W', 'N')) NOT NULL,

Factory\_number INT NOT NULL,

FOREIGN KEY (Factory\_number) REFERENCES FACTORY(FactoryNumber) ON DELETE CASCADE

);

* **Table structure for the SUPPLY table**

CREATE TABLE SUPPLY

(

Supply\_ID CHAR(5) PRIMARY KEY NOT NULL,

SupplyName VARCHAR(20) NOT NULL,

Material VARCHAR(20),

Type VARCHAR(20),

ContractID CHAR(9) NOT NULL,

FOREIGN KEY (ContractID) REFERENCES CONTRACT(Contract\_ID) ON DELETE CASCADE

);

* **Table structure for the CONSUMES table**

CREATE TABLE CONSUMES

(

Supply\_ID CHAR(5) NOT NULL,

FactoryNumber INT NOT NULL,

PRIMARY KEY (Supply\_ID, FactoryNumber),

FOREIGN KEY (FactoryNumber) REFERENCES FACTORY(FactoryNumber) ON DELETE CASCADE,

FOREIGN KEY (Supply\_ID) REFERENCES SUPPLY(Supply\_ID) ON DELETE CASCADE

);

* **Table structure for the PRODUCES table**

CREATE TABLE PRODUCES

(

Product\_ID CHAR(9) NOT NULL,

FactoryNumber INT NOT NULL,

Quantity INT,

PRIMARY KEY (Product\_ID, FactoryNumber),

FOREIGN KEY (FactoryNumber) REFERENCES FACTORY(FactoryNumber) ON DELETE CASCADE,

FOREIGN KEY (Product\_ID) REFERENCES PRODUCT(P\_ID) ON DELETE CASCADE

);

* **Table structure for the DEPT\_PHONENUMBER table**

CREATE TABLE DEPT\_PHONENUMBER

(

Dnumber INT NOT NULL CHECK (Dnumber > 0 AND Dnumber < 30),

DPhoneNumber CHAR(10) NOT NULL,

PRIMARY KEY (Dnumber, DPhoneNumber),

FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT (D\_number) ON DELETE CASCADE

);

* **Table structure for the EMP\_PHONENUMBER table**

CREATE TABLE EMP\_PHONENUMBER

(

EID CHAR(9) NOT NULL,

EPhoneNumber CHAR(10) NOT NULL,

FOREIGN KEY (EID) REFERENCES EMPLOYEE (ID) ON DELETE CASCADE ,

PRIMARY KEY (EID, EPhoneNumber)

);

1. **Table descriptors**

**EMPLOYEE:**

Name Null? Type

-------------- -------- ------------

ID NOT NULL CHAR(9)

FIRSTNAME VARCHAR2(20)

MIDDLENAME VARCHAR2(20)

LASTNAME VARCHAR2(20)

GENDER NOT NULL CHAR(1)

B\_DAY NUMBER(38)

B\_MONTH NUMBER(38)

B\_YEAR NUMBER(38)

SALARY FLOAT(5)

ADDRESS VARCHAR2(50)

STARTTIME NOT NULL VARCHAR2(5)

ENDTIME NOT NULL VARCHAR2(5)

NUMBOFDAYS NUMBER(38)

EXTRAHOURS NUMBER(38)

D\_NUMBER NUMBER(38)

FACTORY\_NUMBER NUMBER(38)

SH\_ID NUMBER(38)

**BRANCH :**

Name Null? Type

------------- -------- ------------

B\_NUMBER NOT NULL NUMBER(38)

NOOFEMPLOYEES NUMBER(38)

PHONE\_NUMBER CHAR(10)

COUNTRY VARCHAR2(20)

CITY VARCHAR2(20)

STREET VARCHAR2(20)

EID CHAR(9)

**DEPARTMENT:**

Name Null? Type

------------- -------- ------------

D\_NUMBER NOT NULL NUMBER(38)

D\_NAME NOT NULL VARCHAR2(20)

NOOFEMPLOYEES NUMBER(38)

EID CHAR(9)

B\_NUMBER NOT NULL NUMBER(38)

**SHOP**

Name Null? Type

-------------- -------- ------------

SH\_ID NOT NULL NUMBER(38)

NO\_OFEMPLOYEES NUMBER(38)

COUNTRY VARCHAR2(20)

CITY VARCHAR2(20)

STREET VARCHAR2(20)

PHONE\_NO CHAR(10)

EID NOT NULL CHAR(9)

**FACTORY:**

Name Null? Type

------------- -------- ----------

FACTORYNUMBER NOT NULL NUMBER(38)

NOOFEMPLOYEES NUMBER(38)

B\_NUMBER NOT NULL NUMBER(38)

**CUSTOMER:**

Name Null? Type

------------- -------- ------------

C\_ID NOT NULL CHAR(9)

NAME VARCHAR2(50)

TYPE NOT NULL VARCHAR2(7)

LOYALTYPOINTS NUMBER(38)

COUNTRY VARCHAR2(20)

CITY VARCHAR2(20)

STREET VARCHAR2(20)

**SUPPLIER:**

Name Null? Type

------------- -------- ------------

SUPPLIER\_ID NOT NULL CHAR(7)

NAME VARCHAR2(50)

CONTACT\_NAME VARCHAR2(50)

CONTACT\_TITLE VARCHAR2(30)

COUNTRY VARCHAR2(20)

CITY VARCHAR2(20)

STREET VARCHAR2(20)

**CONTRACT:**

Name Null? Type

----------- -------- ----------

CONTRACT\_ID NOT NULL CHAR(7)

STARTDATE NOT NULL CHAR(10)

ENDDATE NOT NULL CHAR(10)

QUANTITY NUMBER(38)

PAYMENT NOT NULL FLOAT(5)

REWARD FLOAT(5)

C\_ID CHAR(9)

SUPPLIER\_ID CHAR(7)

**PRODUCT:**

Name Null? Type

-------- --------------- ------------

P\_ID NOT NULL CHAR(5)

PRODUCTNAME VARCHAR2(20)

PRODUCTIONMONTH NUMBER(38)

PRODUCTIONYEAR NUMBER(38)

VOLUME FLOAT(5)

TYPE VARCHAR2(20)

CONTRACTID NOT NULL CHAR(7)

SH\_ID NOT NULL NUMBER(38)

**MACHINERY:**

Name Null? Type

-------------- -------- ----------

SERIALNUMBER NOT NULL CHAR(50)

TYPE CHAR(20)

SERVICE NOT NULL CHAR(1)

FACTORY\_NUMBER NOT NULL NUMBER(38)

**SUPPLY:**

Name Null? Type

---------- -------- ------------

SUPPLY\_ID NOT NULL CHAR(5)

SUPPLYNAME NOT NULL VARCHAR2(20)

MATERIAL VARCHAR2(20)

TYPE VARCHAR2(20)

CONTRACTID NOT NULL CHAR(9)

**CONSUMES:**

Name Null? Type

------------- -------- ----------

SUPPLY\_ID NOT NULL CHAR(5)

FACTORYNUMBER NOT NULL NUMBER(38)

**PRODUCES:**

Name Null? Type

------------- -------- ----------

PRODUCT\_ID NOT NULL CHAR(9)

FACTORYNUMBER NOT NULL NUMBER(38)

QUANTITY NUMBER(38)

**DEPT\_PHONENUMBER:**

Name Null? Type

------------ -------- ----------

DNUMBER NOT NULL NUMBER(38)

DPHONENUMBER NOT NULL CHAR(10)

**EMP\_PHONENUMBER:**

Name Null? Type

------------ -------- --------

EID NOT NULL CHAR(9)

EPHONENUMBER NOT NULL CHAR(10)

**3- Populating the Database**

1.       Inserting data into EMPLOYEE

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, StartTime, EndTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000001’, ‘Alex’, ‘mac’, ‘Soles’, ‘M’, ‘3’, ‘4’, ‘1970’, ‘1000’, ‘Hamra-Lebanon’, ‘9:00’, ‘5:00’, ‘5’, ‘0’, ‘1’, NULL, NULL) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000002’, ‘Isabella’, ‘Blay’, ‘Kiker’, ‘F’, ‘1’, ‘1’, ‘1971’, ‘1500’, ‘Bliss-Lebanon’, ‘9:00’, ‘5:00’, ‘5’, ‘0’, ‘1’, NULL, NULL) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000003’, ‘Adam’ , ‘chris’, ‘George’, ‘M’, ‘2’, ‘1’, ‘1990’, ‘1000’, ‘Hamra-Lebanon’, ‘9:00’, ‘5:00’, ‘5’, ‘0’, NULL, ‘2’, NULL) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000004’, ‘Olena’, ‘Ching’, ‘Kia’, ‘F’, ‘5’, ‘5’, ‘1985’, ‘1500’, ‘NYC-USA’, ‘9:00’, ‘5:00’, ‘5’, ‘0’, ‘2’, NULL, NULL) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000005’, ‘Earth’, ‘Ching’, ‘Kia’, ‘F’, ‘5’, ‘5’, ‘1985’, ‘3000’, ‘NYC-USA’, ‘9:00’, ‘5:00’, ‘5’, ‘0’, ‘2’, NULL, NULL) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000006’, ‘Fallon’, ‘calen’, ‘Adams’, ‘F’, ‘7’, ‘12’, ‘1977’, ‘1500’, ‘Siada-Lebanon’, ‘9:00’, ‘6:00’, ‘5’, ‘1’, NULL , NULL, ‘0000001’) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000007’, ‘Ahmed’ ’, ‘abdullah’, ‘benAshuer’, ‘M’, ‘8’, ‘9’, ‘1999’, ‘800’, ‘Tripoli-Lebanon’, ‘9:00’, ‘7:00’, ‘7’, ‘2’, NULL , NULL, ‘0000002’) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID)

VALUES (‘000000008’, ‘George’ ’, ‘Viki’, ‘far’ ’, ‘M’, ‘30’, ‘10’, ‘1989’, ‘800’, ‘NYC-USA’, ‘9:00’, ‘5:00’, ‘5’, ‘0’, NULL , ‘5’ , NULL) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID);

VALUES (‘000000009’, ‘Azar’ ’, ‘kali’, ‘Jaklin’, ‘M’, ‘8’, ‘9’, ‘1990’, ‘2000’, ‘Tripoli-Libya’, ‘9:00’, ‘7:00’, ‘7’, ‘2’, NULL , NULL, ‘0000003’) ;

INSERT INTO EMPLOYEE (ID,FirstName, MiddleName, LastName, Gender, B\_Day, B\_Month, B\_Year, Salary, Address, startTime, endTime, NumbOfDays, ExtraHours, D\_Number, Factory\_number, SH\_ID);

VALUES (‘000000010’, ‘Risa’ ’, ‘Samuel’, ‘bener’, ‘F’, ‘24’, ‘6’, ‘1991’, ‘3000’, ‘Jersey-USA’, ‘9:00’, ‘7:00’, ‘7’, ‘2’, NULL , NULL, ‘0000004’) ;

2.       Insert data into table PRODUCT

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00001’,‘plastic bottle A’. ‘1’, ‘2’, ‘2012’, ‘1’, ‘plastic’, ‘1’, ‘0000001’, ‘001’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00002’,glass bottle B’. ‘2’, ‘1’, ‘2012’, ‘2’, ‘glass’, ‘2’, ‘0000002’, ‘002’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00003’,‘plastic bottle C’. ‘1’, ‘1’, ‘2013’, ‘1’, ‘plastic’, ‘1’, ‘0000003’, ‘003’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00004’, ‘metallic bottle D’. ‘1’, ‘2’, ‘2013’, ‘2’, ‘metal’, ‘1’, ‘0000004’, ‘004’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00005’,‘glass bottle E’. ‘6’, ‘2’, ‘2014’, ‘3’, ‘glass, ‘3’, ‘0000005’, ‘005’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00006’,‘plastic bottle F’. ‘1’, ‘1’, ‘2014’, ‘10’, ‘plastic’, ‘2’, ‘0000006’, ‘006’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00007’,‘plastic bottle E’. ‘1’, ‘2’, ‘2012’, ‘1’, ‘plastic’, ‘1’, ‘0000007’, ‘007’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00008’,‘metallic bottle H’. ‘3’, ‘10’, ‘2015’, ‘2’, ‘metal’, ‘2’, ‘0000008’, ‘008’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00009’,‘plastic bottle I’. ‘1’, ‘2’, ‘2012’, ‘1’, ‘plastic’, ‘1’, ‘0000009’, ‘009’) ;

INSERT INTO PRODUCT(P\_ID,ProductName, ProductionMonth, ProductionYear, Volume, Type, ProductionTime, ContractID, SH\_ID)

VALUES (‘00010’,‘glass bottle J’. ‘1’, ‘2’, ‘2012’, ‘1’, glass’, ‘3’, ‘0000010’, ‘010’) ;

3. Insert data into Table CUSTOMER

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000001’, ‘Shirlene Desir’, ‘Regular’, ‘5’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000002’, ‘Olene Howley’, ‘First’, ‘1’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000003’, ‘Eartha Bradham’, ‘Loyal’, ‘10’, ‘Lebanon’, ‘Beirut’, ‘Jal El Dib’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000004’, ‘Georgine Campbell’, ‘Loyal’, ‘10’, ‘Lebanon’, ‘Beirut’, ‘Downtown’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000005’, ‘Asley Trumble’, ‘First’, ‘1’, ‘Lebanon’, ‘Beirut’, ‘Bab Idris’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000006’, ‘Viki Legere’, ‘Regular’, ‘5’, ‘Lebanon’, ‘Beirut’, ‘Kantari’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000007’, ‘Hue Allee’, ‘Regular’, ‘5’, ‘Lebanon’, ‘Beirut’, ‘Seraili’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000008’, ‘Cari Juhasz’, ‘Loyal’, ‘10’, ‘Lebanon’, ‘Beirut’, ‘Basta Taht’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000009’, ‘Kiersten Chapa’, ‘Loyal’, ‘10’, ‘Lebanon’, ‘Beirut’, ‘Roucha’);

INSERT INTO CUSTOMER (C\_ID, Name, Type, LoyaltyPoints, Country, City, Street)

VALUES (‘000000010’, ‘Sally Deibler’, ‘Regular’, ‘5’, ‘Lebanon’, ‘Beirut’, ‘Gemmayzee’);

4. Insert data into table FACTORY

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘1’, ‘15’, ‘2’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘2’, ‘20’, ‘3’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘3’, ‘15’, ‘2’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘4’, ‘15’, ‘5’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘5’, ‘50’, ‘5’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘6’, ‘30’, ‘2’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘7’, ‘14’, ‘8’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘8’, ‘44’, ‘8’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘9’, ‘25’, ‘2’);

INSERT INTO FACTORY (FactoryNumber, NoOfEmployees, B\_Number)

VALUES(‘10’, ‘30’, ‘2’);

5. Insert data into table DEPARTMENT

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES ('1', 'Marketing', '50', '000000001', '1');

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘2’, ‘Human resources’, ‘50’, ‘000000002’, ‘1’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘3’, ‘Advertisement’, ‘50’, ‘000000003’, ‘1’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘4’, ‘Human resources’, ‘50’, ‘000000004’, ‘2’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘5’, ‘Operation’, ‘18’, ‘000000005’, ‘2’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘6’, ‘Finance’, ‘21’, ‘000000006’, ‘2’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘7’, ‘Advertisement’, ‘50’, ‘000000007’, ‘3’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘8’, ‘Sales’, ‘50’, ‘000000008’, ‘3’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘9’, ‘Purchase’, ‘50’, ‘000000009’, ‘3’);

INSERT INTO DEPARTMENT (D\_number, D\_name, NoOfEmployees, EID, B\_Number)

VALUES (‘10’, ‘General Management’, ‘50’, ‘000000010’, ‘4’);

6. Insert data into table BRANCH

INSERT INTO Branch(B\_number, NoOfEmployees, Phone\_number,country, city, street, EID)

VALUES (‘1’, ‘100’, ‘8764257026’, 'Lebanon', 'Beirut', 'Bliss', ‘000000001’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street,EID)

VALUES (‘2’, ‘200’, ‘6774257444’, ‘Lebanon’, ‘Beirut’, ‘Hamra’,‘000000002’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street,EID)

VALUES (‘3’, ‘300’, ‘6937544661’, ‘Lebanon’, ‘Beirut’, ‘Hamra’,‘000000003’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street,EID)

VALUES (‘4’, ‘400’, ‘3274284750’, ‘Lebanon’, ‘Beirut’, ‘Hamra’, ‘000000004’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street, EID)

VALUES (‘5’, ‘500’, ‘4506763931’, ‘Lebanon’, ‘Beirut’, ‘Hamraa’, ‘000000005’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street, EID)

VALUES (‘6’, ‘600’, ‘8073511468’, ‘Lebanon’, ‘Beirut’, ‘Hamraa’, ‘000000006’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street, EID)

VALUES (‘7’, ‘700’, ‘9073511468’, ‘USA’, ‘NYC’, ‘xzxxxx’, ‘000000007’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street, EID)

VALUES (‘8’, ‘800’, ‘9123511468’, ‘USA’, ‘chicago’, ‘road island’, ‘000000008’);

INSERT INTO Branch (B\_number, NoOfEmployees, Phone\_number, country, city, street, EID)

VALUES (‘9’, ‘900’, ‘6508815765’, ‘USA’, ‘chicago’ ,‘road island’, ‘000000009’);

INSERT INTO BRANCH (B\_number, NoOfEmployees, Phone\_number,country, city, street, EID)

VALUES ('10', '1000', '1548815765', 'Libya', 'Tripoli', 'Madina', '000000010');

7. Insert data into table SUPPLY

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00001’, ‘Chinese plastic’, ‘plastic’, ‘non-recycled’, ‘ ‘00000001’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00002’, ‘brown paper’, ‘paper’. ‘eco-friendly’, ‘ ‘00000002’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00003’, ‘glassX’, ‘glass’. ‘recycled ’, ‘ ‘00000003’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00004’, ‘staedtler pens’, ‘glass’. ‘recycled ’, ‘ ‘00000004’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00005’, ‘staedtler pens’, ‘plastic’. ‘non-recycled ’, ‘ ‘00000005’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00006’, ‘smiley pencils’, ‘wood’. ‘recycled ’, ‘ ‘00000006’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00007’, ‘staedtler pens’, ‘glass’, ‘non-recycled ’, ‘ ‘00000007’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00008’, ‘Indian glass’, ‘glass’. ‘recycled ’, ‘ ‘00000008’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00009’, ‘molds’, ‘plastic’. ‘non-recycled ’, ‘ ‘00000009’);

INSERT INTO SUPPLY (Supply\_ID, SupplyName, Material, Type, ContractID)

VALUES (‘00010’, ‘petrol 91’, ‘petrol’. ‘non-recycled ’, ‘ ‘00000010’);

8. Insert into table into SUPPLIER

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000001’, ‘companyX’, ‘Alex Costa’, ‘sales manager’, ‘Lebanon’, ‘Beirut’, ‘Hamra’’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title ,country, city, street)

VALUES(‘000002’, ‘companyY’, ‘Flor Loyd’, ‘sales manager’, ‘Lebanon’, ‘Beirut’, ‘Bliss’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000003’, ‘companyZ’, ‘Alex Costa’, ‘company representative', ‘Lebanon’, ‘Beirut’, ‘Bab Idris’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000004’, ‘companyA’, ‘Rayan Abusrewil’, ‘president', ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000005’, ‘companyB’, ‘Adam Andres’, ‘purchase department director', ‘Lebanon’, ‘Beirut’, ‘Kantari’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000006’, ‘companyC’, ‘Calum Jones’, ‘operation director', ‘Lebanon’, ‘Tripoli’, ‘ben Ashur’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000007’, ‘companyD’, ‘Ruba jack’, ‘sales director', ‘Lebanon’, ‘Tripoli’, ‘zzzzz’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000008’, ‘companyE’, ‘Calum Jones’, ‘vice president', ‘USA’, ‘NYC’, ‘Manhattan’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000009’, ‘companyF’, ‘Franco john’, ‘manager', ‘USA’, ‘NYC’, ‘Manhattan’);

INSERT INTO SUPPLIER (Supplier\_ID, Name, Contact\_name, Contact\_Title, country, city, street)

VALUES(‘000010’, ‘companyG’, ‘Tala Abonahla’, ‘president', , ‘USA’, ‘NYC’, ‘Manhattan’);

9. Insert into table into CONTRACT

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000001','1/1/2011', '2/1/2011', '12', '800', '80', '000000001', NULL);

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000002','4/4/2013', '4/5/2013', '20', '1000', '100', NULL, '0000001');

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000003','1/1/2011', '2/1/2014', '12', '8000', '50', '000000002', NULL);

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000004','1/1/2019','2/2/2019', '30', '9000', '90', NULL, '0000002');

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000005','1/2/2012', '3/3/2013', '50', '8000', '70', '000000003', NULL);

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000006','1/2/2013', '3/3/2014', '90', '900', '70', NULL, '0000003');

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000007','1/2/2014', '1/3/2015', '90', '1000', '70', '000000004', NULL);

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000008','1/2/2015', '3/3/2015', '80', '1000', '70', NULL, '0000004');

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000009','1/2/2017', '3/3/2017', '80', '1050','90', '000000005', NULL);

INSERT INTO CONTRACT(Contract\_ID, StartDate, EndDate, Quantity, Payment, Reward, C\_ID, Supplier\_ID)

VALUES('0000010','1/2/2017', '3/3/2017', '990', '10590', '90', NULL, '0000005');

10. MACHINERY

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000001’, ‘bottling’, ‘N’, ‘1’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000002’, ‘plastic creation ’, ‘W’, ‘1’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000003’, ‘paper wrapping’, ‘N’, ‘1’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000004’, ‘bottling’, ‘W’, ‘2’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000005’, ‘bottling’, ‘N’, ‘2’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000006’, ‘bottling’, ‘W’, ‘3’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000007’, ‘moving’, ‘W’, ‘3’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000008’, ‘transport’, ‘N’, ‘3’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000009’, ‘boiling’, ‘N’, ‘4’);

INSERT INTO MACHINERY(SerialNumber, Type, Service, Factory\_number)

VALUES(‘0000010’, ‘cooling’, ‘W’, ‘4’);

11. Insert into table into SHOP

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘001’, ‘5’, ‘6508815765’, ‘000000001’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘002’, ‘6’, ‘9908815765’, ‘000000002’, ‘Lebanon’, ‘Beirut’, ‘Bliss’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘003’, ‘7’, ‘8508815765’, ‘000000003’, ‘USA’, ‘NYC’, ‘Fourth’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘004’, ‘8’, ‘0508815765’, ‘000000004’, ‘Libya’, ‘Tripoli’, ‘Bab eissa’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘005’, ‘9’, ‘2508815765’, ‘000000005’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘006’, ‘10’, ‘5108815765’, ‘000000006’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘007’, ‘11’, ‘9908815765’, ‘000000007’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘008’, ‘5’, ‘3308815765’, ‘000000008’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘009’, ‘5’, ‘3308815765’, ‘000000009’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

INSERT INTO SHOP(SH\_ID, No\_OfEmployees, Phone\_No, EID, country, city, street)

VALUES(‘010’, ‘5’, ‘0008815765’, ‘000000010’, ‘Lebanon’, ‘Beirut’, ‘Hamra’);

12. Insert into table CONSUMES

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00001’, ‘001’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00002’, ‘003’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00003’, ‘003’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00004’, ‘004’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00005’, ‘005’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00006’, ‘006’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00007’, ‘007’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00008’, ‘008’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00009’, ‘009’);

INSERT INTO CONSUMES (Supply\_ID, FactoryNumber)

VALUES(‘00010’, ‘010’);

13. Insert into table PRODUCES

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000001’, ‘1’, ‘10’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000002’, ‘2’, ‘15’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000003’, ‘3’, ‘20’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000004’, ‘4’, ‘25’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000005’, ‘5’, ‘10’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000006’, ‘6’, ‘19’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000007’, ‘7’, ‘17’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000008’, ‘8’, ‘10’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000009’, ‘9’, ‘13’);

INSERT INTO PRODUCES(Product\_ID, FactoryNumber, Quantity)

VALUES(‘000000010’, ‘10’, ‘14’);

14. Insert into table DEPT\_PHONENUMBER

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘1’, ‘0912123203’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘2’, ‘0912123204’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘3’, ‘0912123205’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘4’, ‘0912123206’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘5’, ‘0912123207’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘6’, ‘0912123208’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘7’, ‘0912123209’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘8’, ‘0912123210’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘9’, ‘0912123211’);

INSERT INTO DEPT\_PHONENUMBER(Dnumber, DPhoneNumber)

VALUES(‘10’, ‘0912123210’);

15. Insert into table EMP\_PHONENUMBER

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000001’, ‘0912123330’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000002’, ‘0912123331’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000003’, ‘0912123332’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000004’, ‘0912123333’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000005’, ‘0912123334’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000006’, ‘0912123335’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000007’, ‘0912123336’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000008’, ‘0912123337’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000009’, ‘0912123338’);

INSERT INTO EMP\_PHONENUMBER(EID, EPhoneNumber)

VALUES(‘000000010’, ‘0912123339’);

BRANCH :

**Branch No NoOfEmp. Phone\_Num Country City Street EID**

1 100 8764257026 Lebanon Beirut Bliss 000000001

2 200 6774257444 Lebanon Beirut Hamra 000000002

3 300 6937544661 Lebanon Beirut Hamra 000000003

4 400 3274284750 Lebanon Beirut Hamra 000000004

5 500 4506763931 Lebanon Beirut Hamra 000000005

6 600 8073511468 Lebanon Beirut Jal Al dib 000000006

7 700 9073511468 USA NYC xzxxzx 000000007

8 800 9123511468 USA chicago road Island 000000008

9 900 6508815765 USA New Mexico New England 000000009

10 1000 1548815765 Libya Tripoli Madina 000000010

EMPLOYEE:

**ID FName MName LName Sex B\_Day Month Year Salary Address**

**Start Time End Time NumOfDays Extra Hours D\_Number Factory\_Num SH\_ID**

000000001 Alex mac Soles M 3 4 1970 1000 Hamra-Lebanon 9:00 5:00 5 0 1

000000002 Isabella Blay Kiker F 1 1 1971 1500 Bliss-Lebanon 9:00 5:00 5 0 2

000000003 Addam chris George M 2 1 1990 1000 Hamra-Lebanon 9:00 5:00 5 0 3

000000004 Olena Ching Kia F 5 5 1985 1500 NYC-USA 9:00 5:00 5 0

000000005 Earth Ching Kiaa F 5 5 1985 3000 NYC-USA 9:00 5:00 5 0

000000006 Fallon calen Adams F 7 12 1977 1500 Siada-Lebanon 9:00 6:00 5 1

000000007 Ahmed abdullah benAshuer M 8 9 1999 800 Tripoli-Lebanon 9:00 7:00 7 2

000000008 George Vikki far M 30 10 1989 800 NYC-USA 9:00 5:00 5 0

000000009 Georges Vikki far M 30 10 1989 800 NYC-USA 9:00 5:00 5 0

000000010 Ali Vision kar M 30 10 1989 800 NYC-USA 9:00 5:00 5 0

FACTORY:

**Num NoOfEmp B\_Number**

1 25 2

2 20 3

3 19 2

4 25 5

5 50 5

6 32 2

7 12 8

8 44 8

9 35 2

10 31 2

MACHINERY:

**SerialNumber Type InService**

0000001                                           bottlings           N 1

0000002                                           plastic creating    W 1

0000003                                           Paper wrapping      N 1

0000004                                           Bottling            W 2

0000005                                           Bottling            N 2

0000006                                           Bottling            W 3

0000007                                           Moving              W 3

0000008                                           Transport           N 3

0000009                                           Boiling            N 4

0000010                                           Cooling             W 4

PRODUCES:

**Product\_Id FactoryNumber Quantity**

00001 1 10

00002 2 15

00003 3 20

00004 4 25

00005 5 10

00007 6 2

00009 7 17

00008 8 10

PRODUCT:

Product ID ProductName Production Date Type ContractID SH\_ID

00001 plastic bottle A 1 2012 1 plastic 0000001 1

00002 glass bottle B 2 2012 2 glass 0000002 2

00003 plastic bottle C1 2013 1 plastic 0000003 3

00004 metallic bottle D1 2013 2 metal 0000004 4

00005 glass bottle E 6 2014 3 glass 0000005 5

00006 plastic bottle F 1 2014 10 plastic 0000006 6

00007 plastic bottle E 1 2012 1 plastic 0000007 7

00008 metallic bottle H3 2015 2 metal 0000008 8

00009 plastic bottle I 1 2012 1 plastic 0000009 9

00010 glass bottle J 1 2012 1 glass 0000010 10

SHOP:

**SH\_ID NoOfEmp Country City Street Phone No EID**

1 5 Lebanon Beirut Hamra 6508815755 000000001

2 6 Lebanon Beirut Bliss 9908817765 000000002

3 7 USA NYC Fourth 8558815765 000000003

4 8 Libya Tripoli Bab eissa 5508815765 000000004

5 9 Lebanon Beirut Hamra 2508715765 000000005

6 10 Lebanon Beirut Hamra 5108815765 000000006

7 11 Lebanon Beirut Hamra 9908015765 000000007

8 5 Lebanon Beirut Hamra 3608815765 000000008

9 5 Lebanon Beirut Hamra 3307815765 000000009

10 5 Lebanon Beirut Hamra 1008815765 000000010

SUPPLIER:

**SupplierID Name ContactName ContactPosition Country City Street**

000001 companyX Alex Costa sales manager Lebanon Beirut Hamra

000002 companyY Flor Loyd sales manager Lebanon Beirut Bliss

000003 companyZ Alex Costa company representative Lebanon Beirut Bab Idris

000004 companyA Rayan Abusrewil president Lebanon Beirut Hamra

000005 companyB Adam Andres purchase department director Lebanon Beirut Kantari

000006 companyC Calum Jones operation director Lebanon Tripoli benn Ashur

000007 companyD Ruba jack sales director Lebanon Tripoli zzzzz

000008 companyE Calum Jones vice president USA NYC Manhanttan

000009 companyF Franco john manager USA NYC Manhattan

000010 companyG Tala Abonahla president USA NYCs Manhattan

SUPPLY:

**SupplyID SupplyName Material Type ContractID**

00001 Chinese Plastic Plastic non-recycled 0000001

00002 Brown paper paper eco-friendly 0000002

00003 glassX glass Recycled 0000003

00004 Staedtler pens glass recycled 0000004

00005 Staedtler pens plastic non-recycled 0000005

00006 Smiley pencils wood recycled 0000006

00007 staedtler pens glass non-recycled 0000007

00008 Indian glass recycled 0000008

00009 molds plastic non-recycled 0000009

00010 petrol 91 petrol non-recycled 0000010

Queries:

1. List all the customers (name and ID) that have more than 8 loyalty points or have ordered more than 80 units of any product.

**SELECT** Name, C\_ID

**FROM** CUSTOMER

**WHERE** **EXISTS ( SELECT** \*

**FROM** CUSTOMER **AS** CU, CONTRACT **AS** CO

**WHERE** CU.LoyaltyPoints > 8 **OR** CO.Quantity > 80 **AND** CU.C\_ID = CO.C\_ID; )

1. Retrieve contracts ID’s that have payments due below 5000$.

**SELECT** ContractID

**FROM** CONTRACT

**WHERE** Payment < 5000;

1. Count how many suppliers have signed a contract that started in 2013 or earlier

**SELECT COUNT**(\*)

**FROM** SUPPLIER, CONTRACT

**WHERE** StartDate <= ‘\*\*\*\*2013’ **AND** Supplier\_ID = SupplierID;

1. For each department with at least 40 employees, find the average number of employees

**SELECT AVG**(NoOfEmployees)

**FROM** DEPARTMENT **AS** D

**WHERE** NoOfEmployees >= 40

**GROUP BY** D\_Number;

1. Retrieve all the serial numbers of machinery that is out of service in factory number 02

**SELECT** SerialNumber;

**FROM** MACHINERY, FACTORY;

**WHERE** Service = N **AND** Factory\_Number = ‘002’ **AND** Factory\_Number = FactoryNumber;

1. List the name of all the employees who start work at 8 A.M. for the finance department

**SELECT** FirstName, MiddleName, LastName

**FROM** EMPLOYEE **AS** E, DEPARTMENT **AS** D

**WHERE** E.StartTime = 8:00 **AND** D.D\_Name = ‘Finance’ **AND** E.D\_Number = D.D\_Number

1. Retrieve the average salary of all the employees who work for the human resources department and work more than 10 extra hours.

**SELECT AVG(**Salary)

**FROM** EMPLOYEE **AS** E, DEPARTMENT **AS** D

**WHERE** D.D\_Number = E.D\_Number **AND** D\_name = ‘Human Resources’ **AND** D.ExtraHours >= 10;

1. Retrieve the name of suppliers who reside in the same city as any customers with the location being Beirut, as well as there being a branch residing there

**SELECT** S.Name

**FROM** SUPPLIER **AS** S, CUSTOMER **AS** C

**WHERE** C.City = ‘Beirut’ **AND** C.City = S.City **AND ( SELECT** \*

**FROM** BRANCH

**WHERE** City = ‘Beirut’;

1. Retrieve the location of the branches that oversee factories with out of order machinery, besides factory 04.

**SELECT** Country, City, Street;

**FROM** BRANCH;

**WHERE** ( **EXISTS** ( **SELECT** \*;

**FROM** MACHINERY, FACTORY;

**WHERE** Service = N **AND** Factory\_Number <> ‘004’ **AND** Factory\_Number = FactoryNumber; )

1. Retrieve the name and type of products that have a production year before 2013

**SELECT** ProductName, Type

**FROM** PRODUCT

**WHERE** ProductionYear < ‘2013’;

**Phase 4:**

**Definition of the first normal form:**

Disallows relations within relations or relations as attribute values within tuples. The only

attribute values permitted by 1NF are single **atomic** (or **indivisible**) **values**.

**Definition of the second normal form:**A relation schema R is in 2NF if every nonprime attribute A in R is fully functionally dependent on the primary key of R.

**Definition of the third normal form:**

According to Codd’s original definition, a relation schema R is in

**3NF** if it satisfies 2NF and no nonprime attribute of R is transitively dependent

on the primary key.

**Definition of the Boyce-codd normal form:**

A relation schema R is in **BCNF** if whenever a nontrivial functional dependency X→A holds in R, then X is a superkey of R.

**EMPLOYEE: Relation schema**

**FD1**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First Name | MiddleName | Last Name | Gender | B\_ Day | B\_ Month | B\_Year | Salary | Address | Age |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| startTime | EndTime | NumbOfDays | ExtraHours | D\_ Number | Factory\_number | SH\_ID |

* EMPLOYEE relation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* EMPLOYEE relation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “ID”. Therefore, there is no possibility of having a partial functional dependency on a key.
* EMPLOYEE relation schema is in 3NF because for every non-trivial functional dependency X-> A, X is a superkey or A is a prime attribute “ID” is the primary key, so it is also a superkey.
* EMPLOYEE relation schema is in BCNF because there is no nontrivial functional dependency X->A that holds in it such that X is not a superkey. As mentioned above “ID” is the primary key, so FD1 satisfies the condition.

**PRODUCT: relational schema**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P\_ID | ProductName | ProductionMonth | ProductionYear | Volume | Type | ProductionTime | contractID | SH\_ID |

**FD1**

* PRODUCT relation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* PRODUCT relation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “P\_ID”. Therefore, there is no possibility of having a partial functional dependency on a key.
* PRODUCT relation schema is in 3NF because for every non-trivial functional dependency X-> A, that holds in it such that X is a superkey or A is a prime attribute.
* PRODUCT relation schema is in BCNF because there is no nontrivial functional dependency FD1, “P\_ID” is a superkey (primary key). Therefore, no decomposition is needed as the relation schema satisfies BCNF.

**CUSTOMER: relational schema**

**FD1**

|  |  |  |  |
| --- | --- | --- | --- |
| C\_ID | Name | Type | LoyaltyPoints |

* CUSTOMERrelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* CUSTOMERrelation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “C\_ID”. Therefore, there is no possibility of having a partial functional dependency on a key.
* CUSTOMERrelation schema is in 3NF because for every non-trivial functional dependency X-> A, that holds in it such that X is a superkey or A is a prime attribute.
* CUSTOMERrelation schema is in BCNF because there is no nontrivial functional dependency FD1, “C\_ID” is a superkey (primary key). Therefore, no decomposition is needed as the relation schema satisfies BCNF.

**FACTORY: relational schema**

**FD1**

|  |  |  |
| --- | --- | --- |
| FactoryNumber | NoOfEmployees | B\_Number |

* FACTORYrelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* FACTORYrelation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “FactoryNumber”. Therefore, there is no possibility of having a partial functional dependency on a key.
* FACTORYrelation schema is in 3NF because for every non-trivial functional dependency X-> A, that holds in it such that X is a superkey or A is a prime attribute.
* FACTORYrelation schema is in BCNF because there is no nontrivial functional dependency FD1, “FactoryNumber” is a superkey (primary key). Therefore, no decomposition is needed as the relation schema satisfies BCNF.

**DEPARTMENT: relational schema**

**FD1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| D\_number | D\_name | NoOfEmployees | EID | B\_Number |

**FD2**

**FD3**

* DEPARTMENTrelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* DEPARTMENT relation schema is in 2NF because no partial functional dependency on a key exists. Every non-prime attribute is fully functionally dependent on the single primary key “D\_number”, and on either the combination of “B\_number” and “D\_name” (which cannot function without one of them) and “EID” which refers to a manager employee ID of that department, both these cases are considered candidate keys.
* DEPARTMENT relation schema is in 3NF because for every non-trivial functional dependency X-> A, have X as a superkey or A as a prime attribute. In the three functional dependencies above, each of the “D\_number”, “EID” and the combination of “D\_name” and “B\_number” are superkeys since they are considered candidate keys.
* DEPARTMENT relation schema is in BCNF for there are no nontrivial functional dependencies X->A where X is not a superkey. In other words, we don’t have any non-prime attributes that can determine other prime attributes. Therefore, no further decomposition is needed to make the DEPARTMENT schema in the BCNF.

**BRANCH: relational schema**

**FD1**

|  |  |  |  |
| --- | --- | --- | --- |
| B\_number | NoOfEmployees | Phone\_number | EID |

**FD2**

* BRANCHrelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* BRANCH relation schema is in 2NF because no partial functional dependency on a key exists. Every non-prime attribute is fully functionally dependent on the single primary key “B\_number”, and “EID” which refers to a manager employee ID of that branch, which is considered a candidate key.
* BRANCH relation schema is in 3NF because for every non-trivial functional dependency X-> A, have X as a superkey or A as a prime attribute. In the two functional dependencies above, each of the “B\_number” and “EID” are superkeys since they are considered candidate keys.
* BRANCH relation schema is in BCNF for there are no nontrivial functional dependencies X->A where X is not a superkey. In other words we don’t have any non-prime attributes that can determine other prime attributes. Therefore, no further decomposition is needed to make the BRANCH schema in the BCNF.

**SUPPLY: relational schema**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Supply\_ID | SupplyName | Material | Type | ContractID |

**FD1**

* SUPPLY relation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* SUPPLY relation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “Supply\_ID”. Therefore, there is no possibility of having a partial functional dependency on a key.
* SUPPLY relation schema is in 3NF because for every non-trivial functional dependency X-> A, that holds in it such that X is a superkey or A is a prime attribute.
* SUPPLY relation schema is in BCNF because there is no nontrivial functional dependency FD1, “Supply\_ID” is a superkey (primary key). Therefore, no decomposition is needed as the relation schema satisfies BCNF.

**SUPPLIER: relational schema**

|  |  |  |  |
| --- | --- | --- | --- |
| SupplierID | Name | Contact\_name | Contact\_Title |

**FD1**

* SUPPLIERrelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* SUPPLIERrelation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “SupplierID”. Therefore, there is no possibility of having a partial functional dependency on a key.
* SUPPLIERrelation schema is in 3NF because for every non-trivial functional dependency X-> A, that holds in it such that X is a superkey or A is a prime attribute.
* SUPPLIERrelation schema is in BCNF because there is no nontrivial functional dependency FD1, “SupplierID” is a superkey (primary key). Therefore, no decomposition is needed as the relation schema satisfies BCNF.

**CONTRACT : relational schema**

**FD1**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ContractID | StartDate | EndDate | Quantity | Payment | Reward | C\_ID | Supplier\_ID |

* CONTRACT relation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* CONTRACT relation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “ContractID”. Therefore, there is no possibility of having a partial functional dependency on a key.
* CONTRACT relation schema is in 3NF because for every non-trivial functional dependency X-> A, X is a superkey or A is a prime attribute “ContractID” is the primary key, so it is also a superkey.
* CONTRACT relation schema is in BCNF because there is no nontrivial functional dependency X->A that holds in it such that X is not a superkey. As mentioned above “ContractID” is the primary key, so FD1 satisfies the condition.

**MACHINERY : relational schema**

**FD1**

|  |  |  |  |
| --- | --- | --- | --- |
| SerialNumber | Type | Service | Factory\_number |

* MACHINERY relation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* MACHINERY relation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “SerialNumber”. Therefore, there is no possibility of having a partial functional dependency on a key.
* MACHINERY relation schema is in 3NF because for every non-trivial functional dependency X-> A, X is a superkey or A is a prime attribute “SerialNumber” is the primary key, so it is also a superkey.
* MACHINERY relation schema is in BCNF because there is no nontrivial functional dependency X->A that holds in it such that X is not a superkey. As mentioned above “SerialNumber” is the primary key, so FD1 satisfies the condition.

**SCHEDULE : relational schema**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EID | StartTime | EndTime | NumberOfDays | ExtraHours |

**FD1**

* SCHEDULErelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* SCHEDULErelation schema is in 2NF because every non-prime attribute in it is fully functionally dependent on its single primary key, which is “EID”. Therefore, there is no possibility of having a partial functional dependency on a key.
* SCHEDULErelation schema is in 3NF because for every non-trivial functional dependency X-> A, that holds in it such that X is a superkey or A is a prime attribute.
* SCHEDULErelation schema is in BCNF because there is no nontrivial functional dependency FD1, “EID” is a superkey (primary key). Therefore, no decomposition is needed as the relation schema satisfies BCNF.

**SHOP : relational schema**

|  |  |  |  |
| --- | --- | --- | --- |
| SH\_ID | No\_OfEmployees | Phone\_No | EID |

**FD2**

**FD1**

* SHOPrelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* SHOP relation schema is not in the 2NF, because the non-prime attribute “EID” is partially determined by the candidate key {phone\_no,No\_ofEmployees}. A proper subset of this key is ”Phone\_No”, and this subset alone functionally determines the “EID” value, Hence, the relation schema is split into two, SHOP\_A and SHOP\_B, to solve this problem and satisfy 2NF.

**SHOP\_A**

|  |  |  |
| --- | --- | --- |
| SH\_ID | No\_OfEmployees | Phone\_No |

**SHOP\_B**

|  |  |
| --- | --- |
| EID | Phone\_no |

* After splitting the SHOP relation schema, the 3NF condition is also met, in FD1 and FD2, “Phone\_no” is a superkey since it is a primary key, so for every functional dependency X ->A, X is a superkey or A is a prime attribute.
* The relation schema is in BCNF because there is no notrivial functional dependency X->A such that X is not a superkey. As mentioned above “Phone\_no” is a superkey. So FD1 and FD2 satisfy the condition BCNF.

**PRODUCES : relational schema**

|  |  |  |
| --- | --- | --- |
| Product\_ID | FactoryNumber | Quantity |

* PRODUCESrelation schema satisfies the conditions of the NF1.It includes neither multivalued nor composite attributes, and all the values of the attributes of this schema are single and atomic.
* PRODUCESrelation schema is in 2NF because no proper subset of the primary key functionally determines the non-key attribute “Quantity”. The same producer may produce products in different quantities.
* The relation schema is in the 3NF because for all functional dependencies X->Y that exist, X is a superkey or Y is a prime attribute. In this case “Product” and “FactoryNumber” form a superkey.
* The relation schema is also in BCNF since all the functional dependencies X->Y have X as a superkey. Thus, no breaking down of the relation is needed for the relation to satisfy the BCNF.

**Relational schemas without non-prime attributes**

CONSUMES

|  |  |
| --- | --- |
| Supply\_ID | FactoryNumber |

DEPT\_PHONENUMBER

|  |  |
| --- | --- |
| Dnumber | DPhoneNumber |

EMP\_PHONENUMBER

|  |  |
| --- | --- |
| E ID | EPhoneNumber |

All these relational schemas are int the first, second, third and BC normal forms. They have multivalued or composite attributes, and since they have no non-prime attributes, all the conditions of the 2NF,3NF and BCNF are satisfied. Hence these relations remain as is without decomposition.