LAU
School of Arts and Sciences

Department of Computer Science & Mathematics

DATABASE DESIGN FOR A GAS COMPANY

STARK INDUSTRIES

By DATAVENGERS

Adam Kanj

Ahmad Malak

Angelo Abi Aad

Lionelli Maroun

REPORT

Submitted to DR. RAMZI HARATY

Course “CSC375: Database Management Systems” in Computer Science

Phase IIII

December 05, 2022

I. Table of Contents

II. Introduction

It was a normal day in Lebanon, power cuts were always occurring, rationing of limited resources were performed on the people, fuel and gas shortages were evident, and much more. It was when 4 guys Lionelli, Adam, Angelo and Ahmad were together in a car waiting in turn to fill up fuel due to the insufficient amount. Due to their bad luck, when their turn arrived, the gas station told them that they had dried out and cannot do anything about it. They got extremely frustrated wasting their time waiting for nothing and wanted to do something about it. Looking at the state of this country, they decided to come up with a solution. Much brainstorming was done until they came up with their idea of sharing their own Gas Company. They had in mind this big picture of owning a Gas company that would be supplying all over the world and most importantly in Lebanon solving most of their issues.

In order to have a plan for their goal, they visited famous gas companies in the country and tried to take important points on how to improve them. One of the first things they realized was the lack of technology being used. There wasn't any new updated technological equipment, and the used equipment were badly conditioned. There also lacked some coordination. Departments were not fully divided equally each one with a set of tasks and jobs. In addition, products were not safely monitored since different types of products were stored in the same storages which might eventually lead to an accident. And finally, each company stick to a specific location to extract from, no research and inspections were being made about possible extraction sites. With all this information, they had the perfect plan to do something great.

After years of planning, they finally created this gas company which is now one of the most successful factories in the world and is found in more than 40 countries around the globe shipping more than a hundred different types of natural products and resources.

And after years of success, this company has become popular and trending worldwide, creating a buzz and making it number one provider of natural gases and resources in the world. So none other than Tony Stark contacted them to have a partnership with them in order to have them supply the Avengers with any material they want to help protect the Earth. Although, Tony had one condition, he wanted the company to be named "Stark Industries". Which is where the company's famous name now is from.

This report is going to discuss a detailed summary of the database of "Starks Industries" that makes sure each product makes its way to the customer in the fastest way possible with no worries and help the Avengers in defending Earth against its intruders. This phase deals with ER diagram and database of the company.

III. System Description and Constraints

Various types of gases and fuel are required for the continued operation of various companies and businesses. This gas company is responsible for the supply of these gases and fuel. It is going to aid and serve thousands of people by supplying their needs.

This company possesses various different branches in different locations, where each branch contains different departments responsible for a certain area. Starks Industries has branched out to more than 50 countries being the number one provider of natural resources. Many departments exist like the HR department, storage department, technical department.... Each department is made up of several buildings having several levels and rooms depending on each one.

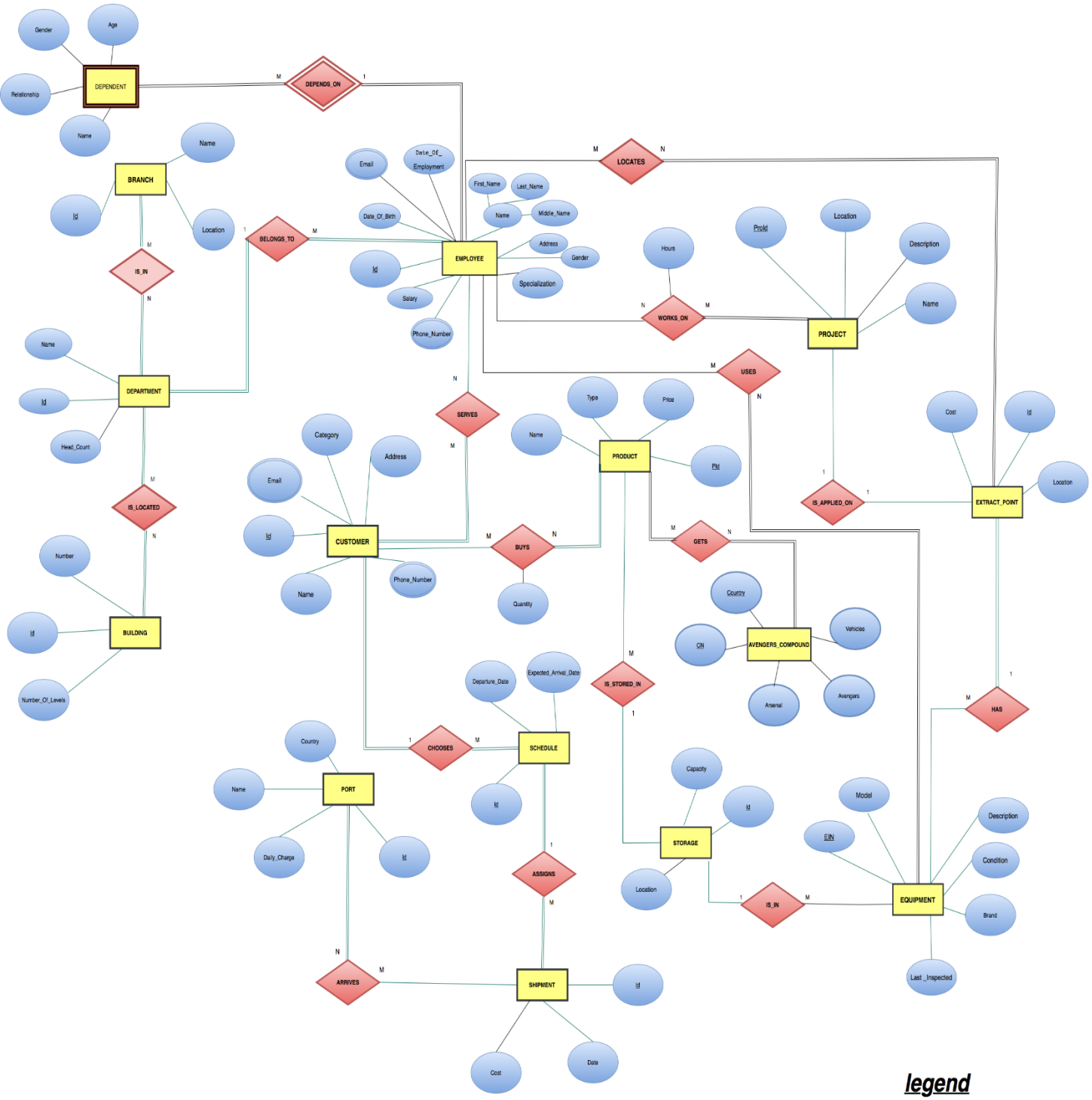
Our employees are the most hardworking ones out there. The staff consists of multiple employees in different jobs and roles, working in different departments. Some employees' main jobs, for example, are to conduct projects and research to extract input and gas from possible extraction points. They handle different equipment to increase their efficiency and also increase our stock.

Once we succeed in gathering our stock, we look for our nearest branch to go to and store it in our storage. There is a unique storage space for each type of product in every branch that is safely monitored in order to avoid any accidents. In addition, a summary review is performed by a manager who tracks everything that goes in and out of the storage.

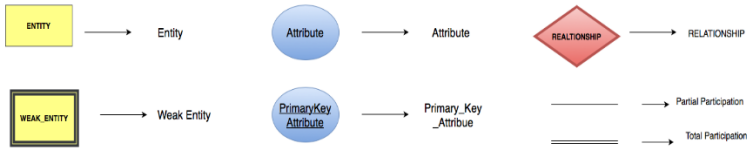
Now it is our turn to supply. Many different consumer companies and businesses contact us to order deliveries. Depending on our capabilities and availability, the nearest branch to our shipment point is contacted to transfer the preferred product ordered from the storage to the port using a container. A shipment is then scheduled to the consumer's location, which arrives on time with no worries. The consumer either pays the shipper the price of the products plus a delivery fee, or he/she can pay online.

But we are not no ordinary company, we are Stark Industries collaborating with Earth’s mightiest heroes that protect us from enemies. We do not only extract gases and natural resources anymore, we also extract rare rocks, stones and materials from other planets like Vibranium & Kryptonite and inspect them to find if they can aid us in supplying the Avengers Compound. Upon success, we use these rare materials in building and manufacturing new equipment, tech, weapons, vehicles, and suits for our heroes. We created Captain America’s unbreakable shield using Vibranium, found out a Kryptonite rock that is our rivals’ weaknesses, extracted a black venom slime that bonds to a body and gives him superhuman abilities, as well as many more. However, our biggest threat Thanos is looking for 6 different stones that when used together, might wipe out 50% of humanity so we were assigned to find these stones before him and make it our top priority.

IV. ER diagram for the Gas Company

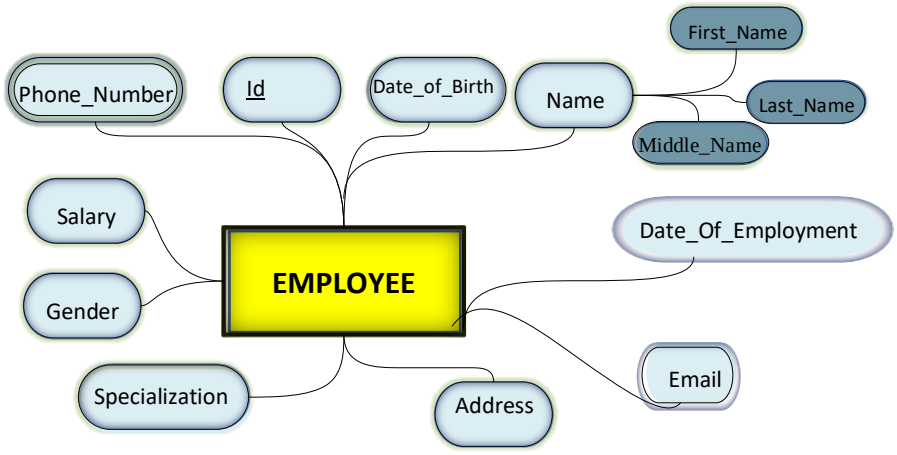


legend



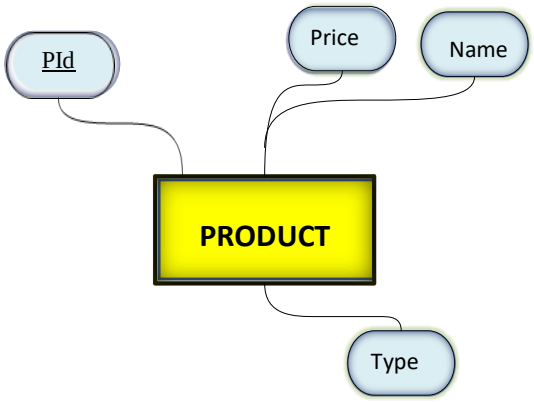
V- Entity types

1. Employee:



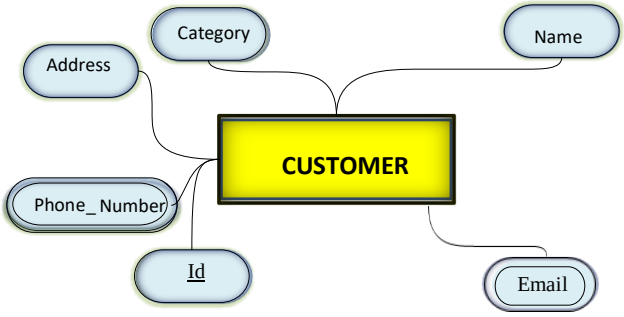
The employees are the ones who make this gas company stand on its feet. They work hard, and each one performs his or her task and role seriously. In our database, every employee is unique and special. An employee is identified by a unique Id, which is considered the primary key in this entity. The name of the employee is given as a composite attribute divided into three parts: first name, last name, and middle initial. He also has other attributes that describe him, such as gender, address, and date of birth, which provides us with his age. Every employee has a specific role and a different specialization working in their specific field, so we provide a specialization attribute. Some employees have been loyal to this company for many years, which in this case gives them a possible advantage over other employees. In this case, a Date_of_Employment attribute is given, which contains the date on which the employee was employed. In order to be able to contact the employee, a multivalued attribute phone number is given, which might have more than one value, including office number, home number, mobile, emergency number, etc. Moreover, customers can contact the employee using the multivalued attribute Email. And finally, a salary attribute is found in order to give back to him for his hardworking efforts after each month.

2. Product:



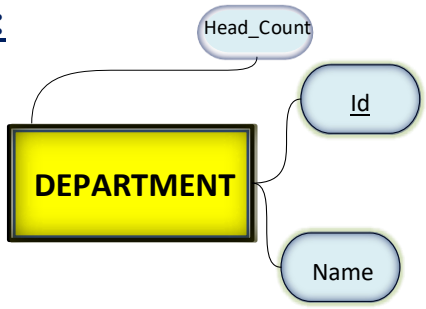
What kind of natural product you need we got it. Our customers order our products, and our job is to be able to supply them. We offer different types of products and gas, which is why we have a **Type** attribute to specify which gas. Each product also has its own product ID defined by the PId primary key. Obviously, depending on a certain product there exists a unique price, which may vary from time to time depending on environmental and political factors.

3. Customer:



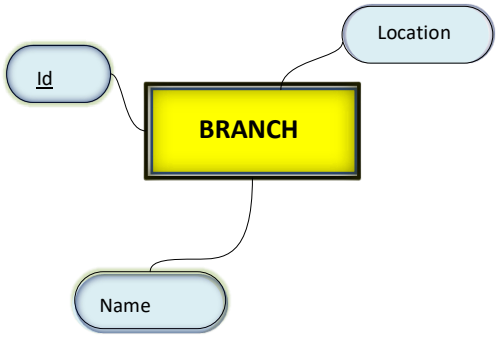
The customer is our main priority. Each customer is presented with a unique Id (primary key) that is automatically assigned to him upon ordering. The type of customer also varies. The customer may be an individual or a company, so we added a Category attribute to be able to identify them. The customer also has several other attributes, which are the customer's name (or company name), address. In addition, the customer can contact the employees via phone and email, which are multivalued attributes.

4. Department:



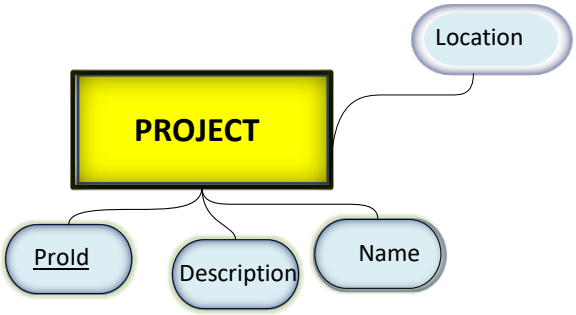
Our company has several departments that are always active and efficient. In our gas company, we may have many employees working in one department that specialize in a specific domain, like the technical department, fleet department, public and legal department, HR department, etc. Each department is represented by a unique Id which is our primary key.

5. Branch:



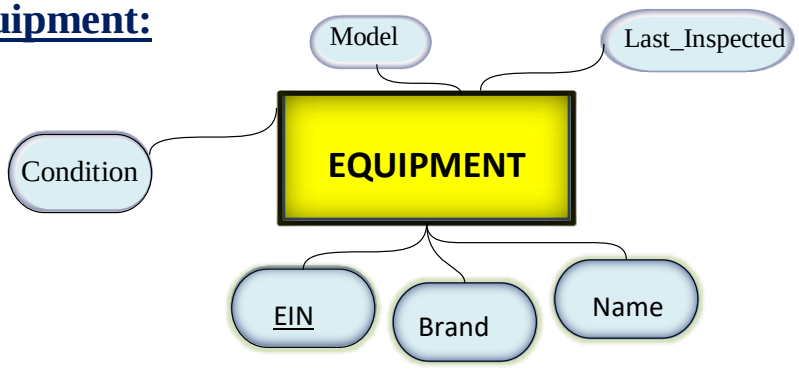
Branches are important and useful for a company in that they allow many of the client-specific administrative considerations to be conducted closest to clients. The more the branches the more we can branch out to more customers around the globe. A branch consists of many departments, and each branch has a primary key ID and a name representing what kind of department it is.

6. Project:



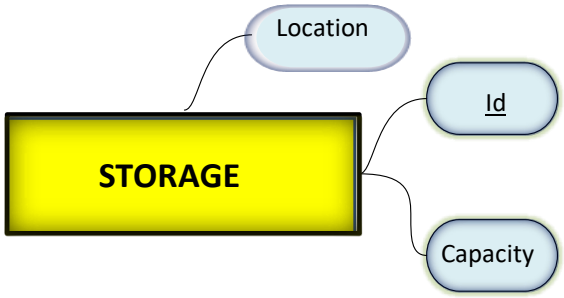
Projects are essential for the continuity of every company and to help in achieving a company's goal by enriching its capital. Each project is performed at a different location, which is given as an attribute. A gas company carries out a project by joining teams of employees from different departments to work on a certain objective regarding a topic such as expanding infrastructure, pipelines, production facilities, and liquefaction plants... Each project is represented by a unique ProId (primary key) and has a name in addition to a brief description.

7. Equipment:



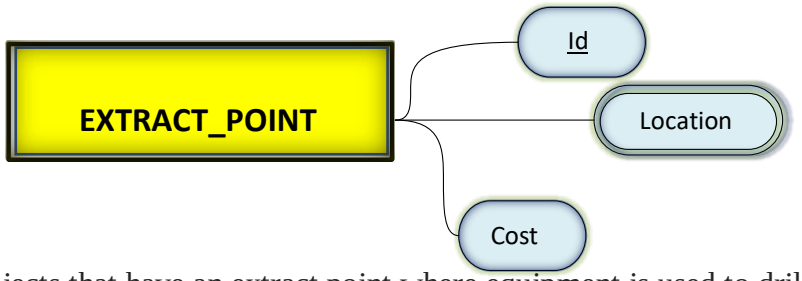
Equipment facilitates an employee's job drastically. Employees use many types of equipment and tools in order to complete numerous processes and procedures. Every tool is described by an EIN (equipment identification number), which is its primary key. It also has many different attributes, which are Model, name, and brand. In order to be familiar with the efficiency of a certain tool, each tool contains an attribute that specifies when it was last inspected, which determines the condition of the tool.

8. Storage:



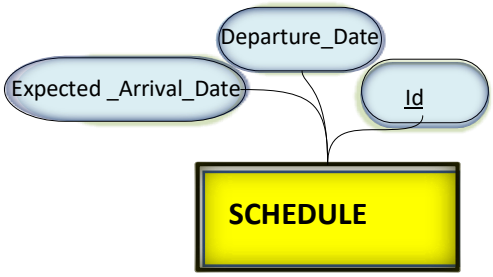
I have far too much storage, said no one ever. Our goal is to always increase the storage capacity for future purposes. The storages are responsible for storing the products in a safe place and having them ready for export. Each storage holds one type of product and is identified by an Id (primary key) and has two additional attributes specifying the capacity it can contain and the location.

9. Extract point:



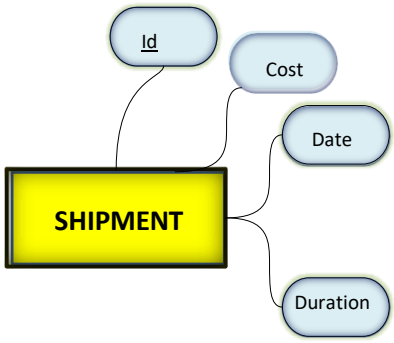
Employees are assigned to projects that have an extract point where equipment is used to drill and find natural gas. An attribute cost is assigned to this entity since each extract point requires different methods and thus might require more cost. Obviously, a location attribute is given, prescribing the exact location of the point of extraction. Finally, we assign a primary key of an Id to every extract point.

10. Schedule:



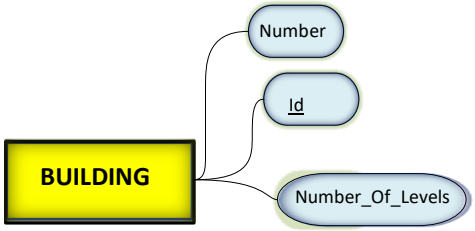
The customer selects the date and time he wants the shipment to depart as well as the time he needs it to arrive at the port. Here, the primary key is the Id, in addition to the attributes indicating the departure time and expected arrival of the product.

11. Shipment:



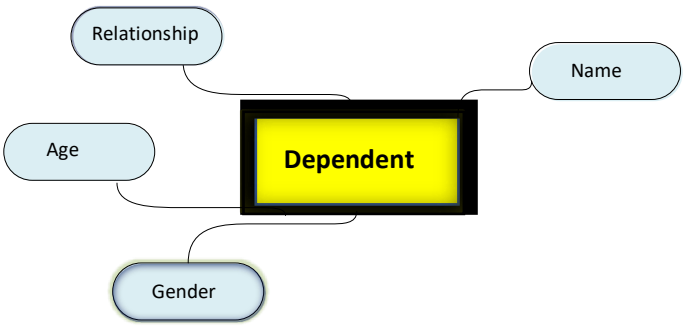
Shipments are assigned to customers in order to supply them with their orders. Each shipment contains a unique Id (primary key) which represents its primary key. The shipment also has a date of arrival, which is the date it is supposed to arrive at the port for the customer to claim. This date also affects its duration attribute, which contains the time it took to arrive since it boarded. And of course, a cost attribute regarding the price of each shipment was made.

12. Building:



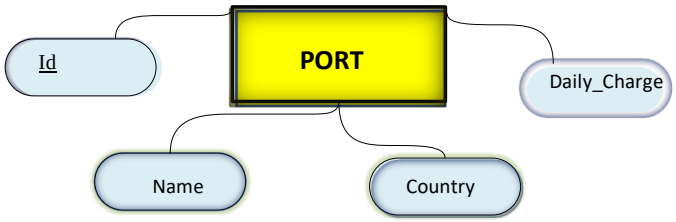
Each branch consists of several buildings and departments. Departments are located in a building that has many rooms distributed among its levels. A building is distinguished by its Id (primary key). In addition, it can contain different blocks numbered accordingly, which in this case is considered an attribute.

13. Dependent:



An employer should always keep track of his/her employees' dependents. Dependent is our weak entity which has people of various age groups related to a certain employee and depending on him to cover their fees. The combination of both the name of the dependent and the id of the related employee will make the primary key of the dependent entity.

14. Port:



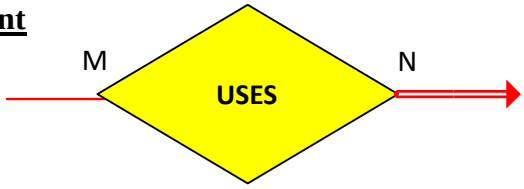
If one does not know to which port he is sailing, no wind is favorable. Thus, every shipment is assigned a port from which to board. The containers reach the port. Each port has a unique name and identification (primary key), and these ports can be from any nation. The port charges a daily fee for customer shipments.

V- Relationships

1) Employee - Project

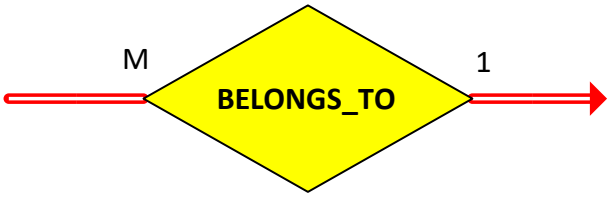
"WORKS_ON" relationship has to be created between **Employee Entity** and **Project Entity**. The participation is partial on the employee side because employees may not work on projects. They can, however, work on a variety of projects. On the other hand, there is total participation on the project side since every project should be worked on by at least one employee.

2) Employee - Equipment



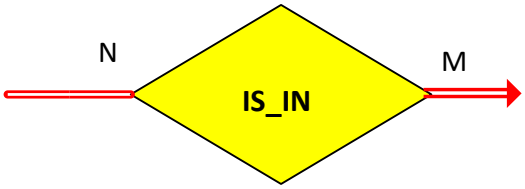
The employees use equipment for extracting and storing the products. So, between the **EMPLOYEE ENTITY** and the **EQUIPMENT ENTITY**, a "USES" many-to-many relationship needs to be established. Only employees are allowed to use the equipment. The participation is total on the equipment side. However, some employees lack the facilities to use the equipment, so employee participation is only partial.

3) Employee - Department



Each Employee works for a particular Department. Thus, a "**BELONGS_TO**" relationship has to be created between **Employee Entity** and **DEPARTMENT Entity**. The participation is total on both sides since each employee must belong to exactly one department and each department must contain at least one employee.

4) Department - Branch

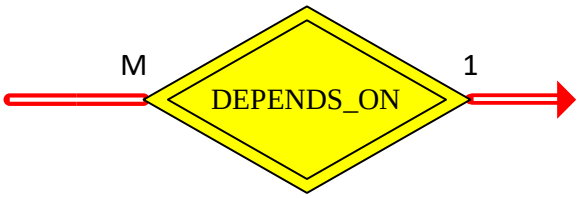


A branch exists for each department. As a result, an "IS IN" relationship needs to be established between the **BRANCH ENTITY** and the **DEPARTMENT ENTITY**. Since each branch must have at least one department, and each department must be a part of at least one branch, there is full participation on both sides.

5) Department - Building

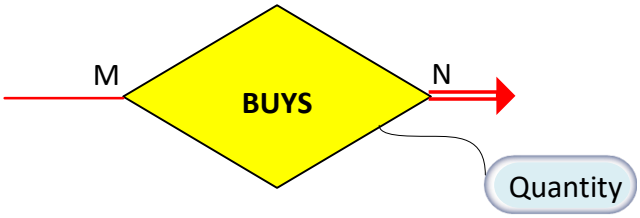
Every department is located in a different building. As a result, the relationship between the **DEPARTMENT ENTITY** and the **BUILDING ENTITY** is "IS_LOCATED". Since every department should be housed in one or more buildings, participation is total on both sides. Additionally, each building ought to have one or more departments.

6) Employee - Dependent



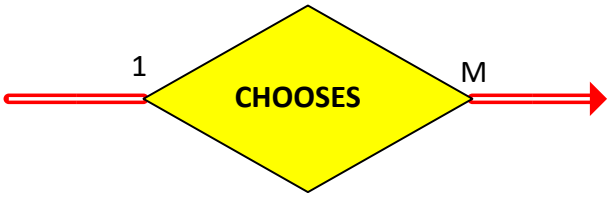
An employee has many people financially depending on him/her. As a result, the "DEPENDS_ON" identifying relationship has to be created between the **EMPLOYEE Entity** and the **weak DEPENDENT Entity**. Total participation is obtained since all dependents should be depending on a certain employee.

7) Customer - Product



Any product may be purchased by any customer in certain quantities. As a result, a "BUYS" many – to - many relationship must be established between the **CUSTOMER Entity** and the **PRODUCT Entity**. The participation is partial on the customer side because they have the option of choosing not to purchase a product, but the participation is total on the product side because only customers can buy any given product.

8) Customer - Schedule



Each customer decides when they want their ordered products to arrive. Thus, between the **CUSTOMER ENTITY** and the **SCHEDULE ENTITY**, a "CHOOSES" 1-to-many relationship is established. Participation is total on both sides. Every customer is required to select a schedule for both the departure information and an expected arrival date. Additionally, each schedule is chosen by the customers.

9) Schedule - Shipment

Following the selection of a schedule for the products' arrival, this schedule designates one or more shipments for the products. Thus, the **SCHEDULE ENTITY** and the **SHIPMENT ENTITY** must establish an "ASSIGN" one-to-many relationship. Depending on the customer's preference, the quantity of products, or other factors, each schedule allocates one or more shipments. Since shipments can only be assigned by schedule, which itself must assign shipments, the participation total on both sides.

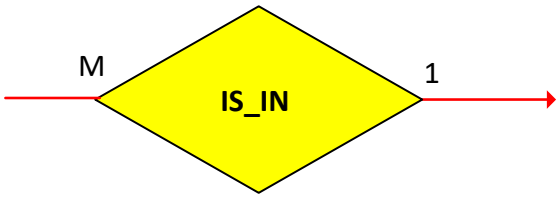
10) Shipment - Port

The port can accept many shipments, and a shipment can reach several ports. Thus , aa "ARRIVES" many - to - many relationships is created between **the SHIPMENT ENTITY** and the **PORT ENTITY**. Because the shipment might not need to be shipped into a port, there is partial participation on the shipment side. However, on the port side, there is full participation because there are only shipments there.

11) Product - Storage

Some products need to be stored. Thus, it is necessary to establish an " IS_STORED_IN" between the **PRODUCT ENTITY** and the **STORAGE ENTITY**. The products can be shipped right away without being stored after being extracted. Additionally, the storage area may have empty spaces or may contain equipment only. Therefore, the participation is partial on both sides. A product cannot be stored in more than one storage; each storage can hold one or more products.

12) Equipment - Storage

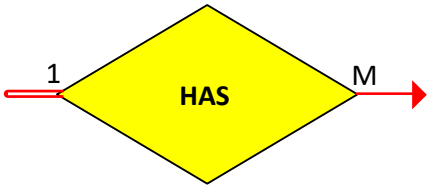


Some storage facilities require equipment in order to house the goods. As a result, a 1-to-many relationship called "IS_IN" is established between the **EQUIPMENT ENTITY** and the **STORAGE ENTITY**. The participation is partial on both sides, since some storage may not contain equipment. Additionally, some equipment might not be useful for storage.

13) Employee – Extract Point

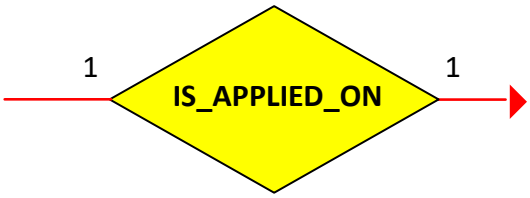
Locating the extraction points that have gas in them is the responsibility of some employees. Therefore, between the **EMPLOYEE ENTITY** and the **EXTRACT_POINT ENTITY**, a "LOCATES" many-to-many must be created. Finding the extract points is the responsibility of a specific employee, so employee participation is partial. However, since every single extract point is found by employees, participation is total on that side.

14) Extract Point - Equipment



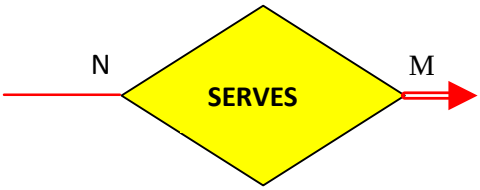
The employees use equipment at each extraction point to extract the gases. It is necessary to establish a "HAS" 1-to-many relationship between the **EXTRACT_POINT ENTITY** and the **EQUIPMENT ENTITY**. The participation is total on the extract point side since each one requires one or more pieces of equipment. However, because some equipment cannot be used at extract points, there is a partial participation on the equipment side.

15) Project – Extract Point



The project should start to executed in the real world as soon as the employee finishes designing it. As a result, an "IS_APPLIED_ON" relationship needs to be established between the **EXTRACT_POINT ENTITY** and the **PROJECT ENTITY**. Each extract point has exactly one project being applied to it. The participation is partial on both sides because some extract points may be located by employments but no project have been applied on yet. Additionally, not all projects may be ready for execution.

16) Employee – Customer

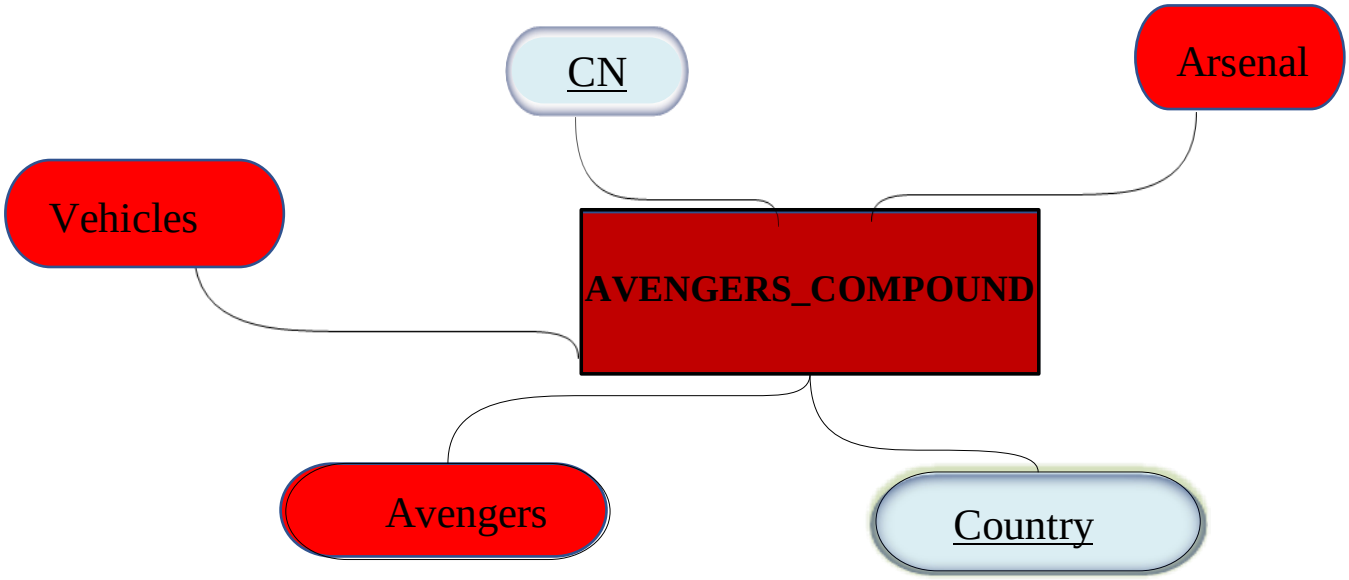


Customers should be served by employees. Therefore, a many-to-many relationship that "SERVES" both the **EMPLOYEE ENTITY** and the **CUSTOMER ENTITY** should be established. Employees serve every customer; thus the participation is total on the customer side. However, some staff are responsible for working in gas extraction rather than providing customer service, so employee participation is only partially complete.


CONFIDENTIAL

V. (ENTITY 15)

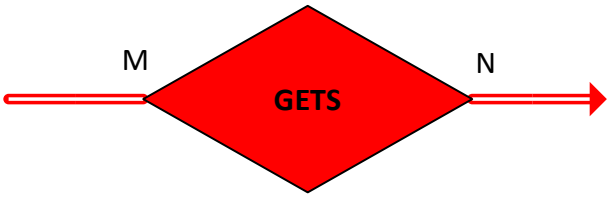
(Entity Type) Avengers Compound:



The Avengers are a team of extraordinary people who either possess superpowers or other unique qualities. The group's mission is to safeguard the stability of the world from threats, particularly those that are internal or external. Thanos, who is looking for the six infinity stones to destroy the planet, is the biggest threat to the Earth. They must locate those stones before Thanos in order for those heroes to be able to defend the world. To stay powerful, these heroes need shields, armor, vehicles, and weapons. In order to aid the Avengers in their fight against intruders, our company, "Stark Industries", donates to them the goods they require from our business, such as fuel and gases. Due to the fact that the Avengers team has several secret compounds spread across various nations, we have the country and CN (compound number) entities jointly produce a composite key.

(Relationship) Avengers Compound:

Avengers_Compound-Product:



Our company, "Stark Industries," donates to the Avengers the products they need from our company, like fuel and gases, to help them fight intruders. Therefore “GETS” many-to-many relationship should be created between the AVENGERS_COMPOUND ENTITY and the PRODUCT ENTITY. The ability to use all of our products in any Avengers compound was granted by our company. As a result, the participation is total on both sides.

VI. ER to Relational Mapping Algorithms

After designing the ER schema and having displayed the database for Stark Industries as a system of entities, attributes, and relationships, this high level design must be translated into a relational database design. In order to map the ER design to a relational database design, a seven-step algorithm needs to be followed. The following is a detailed description of applying the different steps to our database design.

STEP 1: Mapping of Regular Entity Types

In the first step, the regular entity types must be mapped into relations. Each regular entity is going to have its own relation that includes all of its simple attributes and a single primary key which is underlined. The regular (strong) entities in this database design for Stark Industries are: EMPLOYEE, PRODUCT, CUSTOMER, DEPARTMENT, BRANCH, PROJECT, EQUIPMENT, STORAGE, EXTRACT_POINT, SCHEDULE, SHIPMENT, BUILDING, PORT, AVENGERS_COMPOUND.

1. EMPLOYEE

Id	First_Name	Middle_Name	Last_Name	Gender	Date_Of_Birth
Address	Specialization	Salary	Email	Phone_Number	Date_Of_Employment

The EMPLOYEE entity contains simple and multivalued attributes. The multivalued attributes are Email and Phone Number. This relation includes all simple attributes and the primary key ***Id*** which is underlined. EMPLOYEE entity has simple attributes *First_Name, Last_Name, Middle_Name, Salary, Gender, Date_Of_Birth, Date_Of_Employment, Address, Specialization and Salary* are included in the relation.

2. PRODUCT

PId	Name	Type	Price

The PRODUCT entity contains simple attributes. This relation only includes simple attributes and the primary key ***PID*** which is underlined. The PRODUCT entity has simple attributes *Name, Type,* and *Price* are included in the relation.

3. CUSTOMER

Id	Name	Email	Category	Address	Phone_Number

The CUSTOMER entity contains simple attributes and multivalued attributes. The multivalued attributes are Email and Phone_Number. The simple attributes are Name, Category, Address, and ***Id*** which is underlined because it is a primary key.

4. DEPARTMENT

Id	Name	Head_Count

The DEPARTMENT entity contains only simple attributes, so we simply include in this relation the three attributes: *Name*, *Head_Count*, and ***Id*** which is underlined because it is a primary key.

5. BRANCH

Id	Name	Location

The BRANCH entity contains only simple attributes, so we simply include in this relation the three attributes: *Name*, *Location*, and ***Id*** which is underlined because it is a primary key.

6. PROJECT

ProId	Location	Description	Name

The PROJECT entity contains only simple attributes, so we simply include in this relation all four attributes: *Location*, *Description*, Name and ***ProId*** which is underlined because it is a primary key.

7. EQUIPMENT

EIN	Model	Description	Condition	Brand	Last_Inspected

The EQUIPMENT entity contains only simple attributes so we simply include in this relation all six attributes: *Model, Description, Condition, Brand Last_Inspected*, and ***EIN*** which is underlined because it is a primary key.

8. STORAGE

Id	Capacity	Location

The STORAGE entity contains only simple attributes so we simply include in this relation all two attributes: *Capacity, Location* and ***Id*** which is underlined because it is a primary key.

9. EXTRACT_POINT

Id	Cost	Location

The EXTRACT_POINT entity contains only simple attributes, so we simply include in this relation all three attributes: *Cost*, *Location*, and ***Id*** which is underlined because it is a primary key.

10. SCHEDULE

Id	Departure_Date	Expected_Arrival_Date

The SCHEDULE entity contains only simple attributes, so we simply include in this relation all three attributes: *Departure_Date*, *Expected_Arrival_Date*, and ***Id*** which is underlined because it is a primary key.

11. SHIPMENT

Id	Date	Cost

The SHIPMENT entity contains only simple attributes so we simply include in this relation all four attributes: *Date, Cost*, and ***Id*** which is underlined because it is a primary key.

12. BUILDING

Id	Number	Number_Of_Levels

The BUILDING entity contains only simple attributes, so we simply include in this relation all three attributes: *Number*, *Number_Of_Levels* and ***Id*** which is underlined because it is a primary key.

13. PORT

Id	Name	Country	Daily_Charge

The PORT entity contains only simple attributes so we simply include in this relation all four attributes: *Name*, *Country*, *Daily_Charge*, and ***Id*** which is underlined because it is a primary key.

14. AVENGERS_COMPOUND

CN	Country	Arsenal	Avengers	Vehicles

The AVENGERS_COMPOUND entity contains only simple attributes, so we simply include in this relation all five attributes: *Arsenal*, *Avengers*, *Vehicles*, *CN, and Country* which are both underlined because they form a composite key.

STEP 2: Mapping of Weak Entity Type

In this step, the weak entity types are mapped into relations. As in Step 1, only the simple attributes are included in the relations and not multivalued or derived attributes. Furthermore, weak entity relation has a foreign key attribute which is the primary key of the owner entity type. The combination of the foreign key added, and the partial key of the weak entity type represent the primary key of the relation. The weak entity in our database design is DEPENDENT.

1. DEPENDENT

Name	Age	Gender	Relationship	eid

The weak entity DEPENDENT. The weak entity does not have any derived attributes. The simple attributes included are the following: *Name, Gender, Age, Relationship*. Moreover, the Id, the primary key of the EMPLOYEE entity, is included as eid, and the partial key *Name* are combined to represent the primary key of this relation.

STEP 3: Mapping of Binary 1:1 Relationship Types

In this step, we are going to map the binary one-to-one relationships. In order to accomplish our goal, we can follow one of three approaches. The first approach, called foreign key approach is where we choose the entity on the total participation side of the relation, then we add as a foreign key the primary key of the other entity participating in this relation. The second approach, called merged relation approach is where we merge the two entities participating in the relationship into a single relation. This is only used when both participations are total and thus not useful in our case. The third approach, called cross reference or relationship relation approach is where we create a third relation which will include the primary keys of both entities participating in the relationship. We are going to follow the foreign key approach because it is the most useful in our case. The binary one-to-one relationships that need to be mapped are: APPLIED_ON.

1. EXTRACT_POINT (IS_APPLIED_ON)

Id	Cost	Location	Project_Id

Every Extract Point is Applied on a Project. The "IS APPLIED ON" relationship links the **PROJECT** entity and the **EXTRACT_POINT** entity. On both sides of the participating entities, we have partial participation. We chose the EXTRACT_POINT relation in which we added, as a foreign key, the primary key Id of the patient relation and we renamed it **Project_Id.**

STEP 4: Mapping of Binary1:M Relationship Types

In this step, we are going to map the binary one-to-many relationships. We add a foreign key in the entity type at the many sides of the relationship. This foreign key is the primary key of the other entity type participating in this relationship. We must also include any other simple attribute of the one-to-many relationship. The one-to-many relationships that need to be mapped are: BELONGS_TO, GETS, IS_STORED_IN, IS_IN, USES, CHOOSES, ASSIGN, and HAS.

1. EMPLOYEE (BELONGS_TO)

Id	First_Name	Mddle_Name	Last_Name	Gender	Date_Of_Birth
Address	Specialization	Salary	Date_Of_Employment	Phone_Number	DepartmentID

Many employees belong to one department. The "BELONGS_TO" relationship links the **EMPLOYEE** entity and the **DEPARTEMENT** entity. The EMPLOYEE entity is on the "many" side. Thus, we add to its relation the foreign key ID which is the primary key of the DEPARTEMNT entity and we rename it **DepartmentID**.

2. PRODUCT (IS_STORED_IN)

PId	Name	Type	Price	StorageID	CompoundN
COUNTRY					

Many products are stored in one storage room. The "IS_STORED_IN" relationship links the **PRODUCT** entity and the **STORAGE** entity. The PRODUCT entity is on the "many" side. Thus, we add to its relation a foreign key ID which is the primary key of the STORAGE entity and we rename it **StorageID**.

3. EQUIPMENT (IS_IN, HAS)

EIN	Model	Description	Condition	Brand	Last_Inspected
StorageID	ExtractID				

A storage may have more than one equipment stored in it. The "IS_IN" relationship links the **STORAGE** entity and the **EQUIPMENT** entity. The EQUIPMENT entity is on the "many" side so, we add to its relation a foreign key ID which is the primary key of the STORAGE entity and we rename it **StorageID**.

An extract point has more than one equipment. The "HAS" relationship links the **EXTRACT_POINT** entity and the **EQUIPMENT** entity. The EQUIPMENT entity is on the "many" side so, we add to its relation a foreign key ID which is the primary key of the EXTRACT_POINT entity and we rename it **ExtractID**.

4. SCHEDULE (CHOOSES)

Id	Departure_Date	Expected_Arrival_Date	CustomerID

A customer may choose more than one schedule for their shipment. The "CHOOSES" relationship links the **SCHEDULE** entity and the **CUSTOMER** entity. The SCHEDULE entity is on the many side so, we add to its relation the foreign key ID which is the primary key of the CUSTOMER entity and we rename it **CustomerID**.

5. SHIPMENT (ASSIGN)

Id	Cost	Date	ScheduleID

A schedule may be chosen for more than one shipment. The "CHOOSES" relationship links the **SCHEDULE** entity and the **SHIPMENT** entity. The SHIPMENT entity is on the many side so, we add to its relation the foreign key ID which is the primary key of the SCHEDULE entity and we rename it **ScheduleID**.

6. DEPENDENT (DEPENDS_ON)

Name	eid	Gender	Age	Relationship

A dependent person depends on a certain employee. The "DEPENDS_ON" relationship links the **DEPENDENT** entity and the **EMPLOYEE** entity. The DEPENDENT entity is on the many side so, we add to its relation the foreign key ID which is the primary key of the EMPLOYEE entity and we rename it **eid**.

STEP 5: Mapping of M:N Relationship Types

In this step, we are going to map the binary many-to-many relationships. For each many-to-many relationship we are going to create a new relation which includes, as foreign keys, the primary keys of all participating relations. Their combination will form the primary key of this newly created relation. We must also include any other simple attribute of the many-to-many relationship. The many-to-many relationships needed to be mapped are: IS_IN, IS_LOCATED, LOCATES, WORKS_ON, SERVES, BUYS, and ARRIVE.

1. IS_IN

DepartmentID	BranchID

Many departments are in many branches. The "IS_IN" relationship links the **BRANCH** entity and the **DEPARTMENT** entity. We create a new relation called "IS_IN" that includes the primary keys of the BRANCH and DEPARTMENT entities. The primary key of the DEPARTMENT entity, Id, is added to the "IS_IN" relation and renamed **DepartmentID**. Also, the primary key of the BRANCH entity, Id, is added and renamed **BranchID**. The combination of both added keys represents the primary key of the "IS_IN" relation and thus are underlined.

2. IS_LOCATED

DepartmentID	BuildingID

Many departments are located in many buildings. The “IS_LOCATED” relationship links the **DEPARTMENT** entity and the **BUILDING** entity. We create a new relation called “IS_LOCATED” in which we include the primary keys of the DEPARTMENT and BUILDING entities. The primary key of the DEPARTMENT entity, Id, is added to the “IS_LOCATED” relation and renamed **DepartmentID**. Also, the primary key of the BUILDING entity, Id, is added and renamed **BuildingID**. The combination of both added keys represents the primary key of the “IS_LOCATED” relation and thus are underlined.

3. LOCATES

EID	ExtractID

Many employees can locate many extract points. The “LOCATES” relationship links the **EMPLOYEE** entity and the **EXTRACT_POINT** entity. We create a new relation called “LOCATES” in which we include the primary keys of the EMPLOYEE and EXTRACT_POINT entity. The primary key of the EMPLOYEE entity, Id, is added to the “LOCATES” relation and renamed **EID** also the primary key of the EXTRACT_POINT entity, Id, is added and renamed **ExtractID**. The combination of both added keys represents the primary key of the “LOCATES” relation and thus are underlined.

4. WORKS_ON

EID	ProjectID

Many employees work as teams on many projects. The “WORKS_ON” relationship links the **EMPLOYEE** entity and the **PROJECT** entity. We create a new relation called the “WORKS_ON” in which we include the primary keys of the PROJECT and EMPLOYEE entities. The primary key of the EMPLOYEE entity, Id, is added to the “WORKS_ON” relation and renamed **EID**. Also, the primary key of the PROJECT entity, ProId, is added and renamed **ProjectID**. The combination of both keys added represents the primary key of the “WORKS_ON” relation.

5. SERVES

EID	CustomerID

Many employees serve many customers. The "SERVES" relationship links the **EMPLOYEE** entity and the **CUSTOMER** entity. We create a new relation called "SERVES" in which we include the primary keys of the EMPLOYEE and CUSTOMER entity. The primary key of the EMPLOYEE entity, Id, is added to the "SERVES" relation and renamed **EID**. Also, the primary key of the CUSTOMER entity, Id, is added and renamed **CustomerID**. The combination of added both keys represents the primary key of the "SERVES" relation and thus are underlined.

6. BUYS

CustomerID	ProductID	Quantity

Many customers buy many products. The "BUYS" relationship links the **CUSTOMER** entity and the **PRODUCT** entity. We create a new relation called "BUYS" in which we include the primary keys of the CUSTOMER and PRODUCT entities and the attribute Quantity. The primary key of the CUSTOMER entity, Id, is added to the "BUYS" relation and renamed **CustomerID**. Also, the primary key of the PRODUCT entity, PId, is added and renamed **ProductID**. The combination of both keys added represents the primary key of the "BUYS" relation and thus are underlined.

7. ARRIVE

PortID	ShipmentID

Many shipments arrive at many ports for later on to be claimed or distributed from there. The "ARRIVE" relationship links the **PORT** entity and the **SHIPMENT** entity. We create a new relation called "ARRIVE" in which we include the primary keys of the ARRIVE and SHIPMENT entitles. The primary key of the PORT entity, Id, is added to the "ARRIVE" relation and renamed **PortID**. Also, the primary key of the SHIPMENT entity, Id, is added and renamed **ShipmentID**. The combination of both keys added represents the primary key of the "ARRIVE" relation and thus are underlined.

8. GETS

PID	CompoundN	COUNTRY

Several compounds get many products. The "GETS" relationship links the **PRODUCT** entity and the **AVENGERS_COMPOUND** entity. We create a new relation called "GETS" in which we include the primary keys of the PRODUCT and AVENGERS_COMPOUND entities. The primary key of the PRODUCT entity, Pid, is added to the "GETS" relation and renamed **PID**. Also, the primary key of the AVENGERS_COMPOUND entity, CN and Country, are added and renamed **CompoundN** and **COUNTRY** respectfully. The combination of both keys added represents the primary key of the "GETS" relation and thus are underlined.

9. USES

Ein	Eid

A lot of equipment are used by many employees. The "USES" relationship links the **EMPLOYEE** entity and the **EQUIPMENT** entity. We create a new relation called "USES" that includes the primary keys of the EMPLOYEE and EQUIPMENT entities. The primary key of the EMPLOYEE entity, Id, is added to the "USES" relation and renamed **Eid**. Also, the primary key of the EQUIPMENT entity, EIN, is added and renamed **Ein**. The combination of both added keys represents the primary key of the "USES" relation and thus are underlined.

STEP 6: Mapping of Multivalued Attributes

In this step, we are going to map the multivalued attributes which we ignored before. For each multivalued attribute we create a new relation containing the related attribute and the primary key of the entity to which it belongs. Their combination will represent the primary key of the newly created relation. We have three multivalued attributes which are: the employee phone number, the employee email, the customer phone number, and the customer email.

1. EMPLOYEE_PHONE_NUMBER

EID	Phone_Number

The multivalued attribute Phone_Number belongs to the EMPLOYEE entity. To represent it, we create a relation called “EMPLOYEE_PHONE_NUMBER”. Its primary key is composed of **EID** which is the primary key of the employee entity, and the attribute **Phone_Number** which represents the multiple phone numbers an employee can have.

2. CUSTOMER _PHONE_NUMBER

CustomerID	Phone_Number

The multivalued attribute Phone_Number belongs to the CUSTOMER entity. To represent it, we create a relation called “CUSTOMER_PHONE_NUMBER”. Its primary key is composed of **CustomerID** which is the primary key of the customer entity, and the attribute **Phone_Number** which represents the multiple phone numbers a customer can have.

3. CUSTOMER_EMAIL

CustomerID	Email

The multivalued attribute Email belongs to the Customer entity. To represent it, we create a relation called “CUSTOMER_Email”. Its primary key is composed of **CustomerID** which is the primary key of the customer entity, and the attribute **Email** which represents the multiple email a customer can have.

4. EMPLOYEE_EMAIL

EID	Email

The multivalued attribute Email belongs to the Employee entity. To represent it, we create a relation called “EMPLOYEE_Email”. Its primary key is composed of **EID** which is the primary key of the employee entity, and the attribute **Email** which represents the multiple email an employee can have.

STEP 7: Mapping of N-ary Relationship Types

In this step, we are going to map the N-arry Relationship types. We should create a new relation containing the primary keys of all participating entities and any simple attributes of the relationship type. In our design we have no N-ary relationship types, so this step is not applicable here.

FINAL STEP: Final Displays

EMPLOYEE

Id	First_Name	Mddle_Name	Last_Name	Gender	Date_Of_Birth	Email
Address	Specialization	Salary	Date_Of_Employment	Phone_Number	DepartmentID	

PRODUCT (IS_STORED_IN)

PId	Name	Type	Price	StorageID

CUSTOMER

Id	Name	Email	Category	Address	Phone_Number

DEPARTMENT

Id	Name	Head_Count

BRANCH

Id	Name	Location

PROJECT

ProId	Location	Description	Name

EQUIPMENT (IS_IN, HAS)

EIN	Model	Description	Condition	Brand	Last_Inspected
StorageID	ExtractID				

STORAGE

Id	Capacity

EXTRACT_POINT (IS_APPLIED_ON)

Id	Cost	Location	Project_Id

SCHEDULE

Id	Departure_Date	Expected_Arrival_Date	CustomerID

SHIPMENT

Id	Cost	Date	ScheduleID

BUILDING

Id	Number	Number_Of_Levels

Port

Id	Name	Country	Daily_Charge

AVENGERS_COMPOUND

CN	Country	Arsenal	Avengers	Vehicles

DEPENDENT (DEPENDS_ON)

Name	eid	Gender	Age	Relationship

IS_IN

DepartmentID	BranchID

IS_LOCATED

DepartmentID	BuildingID

LOCATES

EID	ExtractID

WORKS_ON

EID	ProjectID

SERVES

EID	CustomerID

BUYS

CustomerID	ProductID	Quantity

ARRIVE

PortID	ShipmentID

GETS

PID	CompoundN	COUNTRY

USES

Ein	Eid

EMPLOYEE_PHONE_NUMBER

EID	Phone_Number

CUSTOMER _PHONE_NUMBER

CustomerID	Phone_Number

CUSTOMER_EMAIL

CustomerID	Email

EMPLOYEE_EMAIL

EID	Email

VII. Table Structure for Stark Industries:

After designing the ER diagram for Stark Industries and mapping this diagram into relational database design, now it is time to start creating the actual tables for our database on the Oracle Database Server. We will start by creating all tables and then inserting data into these tables. Finally, we will execute some queries to display the importance of the database and especially in a gas company.

1. Branch:

```
CREATE TABLE `BRANCH` (
        `Id` VARCHAR(6) NOT NULL,
        `Location` VARCHAR(15) NOT NULL,
        `Name` VARCHAR(15) NOT NULL,
        PRIMARY KEY (`Id`)
);
```

2. Department:

```
CREATE TABLE `DEPARTMENT` (
        `Id` VARCHAR(6) NOT NULL,
        `Name` VARCHAR(30) NOT NULL,
        `HeadCount` INT NULL,
        PRIMARY KEY (`Id`)
 );
```

3. IS_IN:

```
CREATE TABLE `IS_IN` (
        `DepartmentID` VARCHAR(6) NOT NULL,
        `BranchID` VARCHAR(6) NOT NULL,
        FOREIGN KEY (`DepartmentID`) REFERENCES DEPARTMENT(Id),
        FOREIGN KEY (`BranchID`) REFERENCES BRANCH(Id),
        PRIMARY KEY (`DepartmentID`, `BranchID`)
);
```

4. Building:

```
CREATE TABLE `BUILDING` (
        `Id` VARCHAR(6) NOT NULL,
        `Number` VARCHAR(45) ,
        `Number_Of_Levels` INT ,
        PRIMARY KEY (`Id`)
);
```

5. Is_Located:

```
CREATE TABLE `IS_LOCATED`(
        `DepartmentID` VARCHAR(6) NOT NULL,
        `BuildingID` VARCHAR(6) NOT NULL,
        FOREIGN KEY (`DepartmentID`) REFERENCES DEPARTMENT(Id),
        FOREIGN KEY (`BuildingID`) REFERENCES BUILDING(Id),
        PRIMARY KEY (`DepartmentID`,`BuildingID` )
);
```

6. Extract_Point:

```
CREATE TABLE `EXTRACT_POINT` (
        `Id` VARCHAR(6) NOT NULL,
        `Cost` INT ,
        `Location` VARCHAR(45) ,
        `Project_Id` VARCHAR(6) NOT NULL,
        PRIMARY KEY (`Id`)
);
```

7. Employee:

```
CREATE TABLE `Employee` (
        `Id` VARCHAR(6) NOT NULL,
        `First_Name` VARCHAR(15) NOT NULL,
        `Middle_Name` VARCHAR(15) ,
        `Last_Name` VARCHAR(15) NOT NULL,
        `Gender` VARCHAR(1) ,
        `Date_Of_Birth` DATE ,
        `Address` VARCHAR(45) ,
        `Specialization` VARCHAR(20) ,
        `Salary` INT ,
        `Date_Of_Employment` DATE ,
        `DepartmentID` VARCHAR(6)  ,
        PRIMARY KEY (`Id`)
 );
```

8. Locates:

```
CREATE TABLE `LOCATES` (
        `EID` VARCHAR(6) NOT NULL,
        `ExtractID` VARCHAR(6) NOT NULL,
        FOREIGN KEY (`EID`) REFERENCES Employee(Id),
        FOREIGN KEY (`ExtractID`) REFERENCES EXTRACT_POINT(Id),
        PRIMARY KEY (`EID`, `ExtractID`)
);
```

9. Project:

```
CREATE TABLE `PROJECT` (
        `ProId` VARCHAR(6) NOT NULL,
        `Location` VARCHAR(40) NULL,
        `Description` VARCHAR(100) NULL,
        `Name` VARCHAR(35) NULL,
        PRIMARY KEY (`ProId`)
 );
```

10. Works_On:

```
CREATE TABLE `WORKS_ON` (
        `EID` VARCHAR(6) NOT NULL,
        `ProjectID` VARCHAR(6) NOT NULL,
        `Hours` INT ,
        FOREIGN KEY (`EID`) REFERENCES Employee(Id),
        FOREIGN KEY (`ProjectID`) REFERENCES PROJECT(ProId),
        PRIMARY KEY (`EID`, `ProjectID`)
);
```

11. Customer:

```
CREATE TABLE `CUSTOMER` (
        `Id` VARCHAR(7) NOT NULL,
        `Name` VARCHAR(25) NOT NULL,
        `Category` VARCHAR(25) ,
        `Address` VARCHAR(45) ,
        PRIMARY KEY (`Id`)
);
```

12. Serves:

```
CREATE TABLE `SERVES` (
        `EID` VARCHAR(6) NOT NULL,
        `CustomerID` VARCHAR(6) NOT NULL,
        FOREIGN KEY (`EID`) REFERENCES Employee(Id),
        FOREIGN KEY (`CustomerID`) REFERENCES CUSTOMER(Id),
        PRIMARY KEY (`EID`, `CustomerID`)
);
```

13. Product:

```
CREATE TABLE `PRODUCT` (
        `PId` VARCHAR(6) NOT NULL,
        `Name` VARCHAR(15) NOT NULL,
        `Type` VARCHAR(20) ,
        `Price` INT ,
        `StorageID` VARCHAR(6) ,
        `CompoundN` VARCHAR(6) ,
        PRIMARY KEY (`PId`)
);
```

14. Buys:

```
CREATE TABLE `BUYS` (
        `CustomerID` VARCHAR(6) NOT NULL,
        `ProductID` VARCHAR(6) NOT NULL,
        `Quantity` INT(11),
        FOREIGN KEY (`CustomerID`) REFERENCES CUSTOMER(Id),
        FOREIGN KEY (`ProductID`) REFERENCES PRODUCT(PId),
        PRIMARY KEY (`CustomerID`, `ProductID`)
);
```

15. Shipment:

```
CREATE TABLE `SHIPMENT` (
        `Id` VARCHAR(6) NOT NULL,
        `Date` DATE ,
        `Cost` INT ,
        `ScheduleID` VARCHAR(6) NOT NULL,
        PRIMARY KEY (`Id`)
);
```

16. Port:

```
CREATE TABLE `PORT` (
        `Id` VARCHAR(6) NOT NULL,
        `Name` VARCHAR(25) ,
        `Country` VARCHAR(25),
        `Daily_Charge` INT ,
        PRIMARY KEY (`Id`)
 );
```

17. Arrives:

```
CREATE TABLE `ARRIVES` (
        `PortId` VARCHAR(6) NOT NULL,
        `ShipmentID` VARCHAR(6) NOT NULL,
        FOREIGN KEY (`PortId`) REFERENCES PORT(Id),
        FOREIGN KEY (`ShipmentID`) REFERENCES SHIPMENT(Id),
        PRIMARY KEY (`PortId`, `ShipmentID`)
 );
```

18. Storage:

```
CREATE TABLE `STORAGE` (
        `Id` VARCHAR(6) NOT NULL,
        `Capacity` VARCHAR(45) NULL,
        `Location` VARCHAR(45) NULL,
        PRIMARY KEY (`Id`)
 );
```

19. Avengers_Compound:

```
 CREATE TABLE `AVENGERS_COMPOUND` (
        `CN` VARCHAR(6) NOT NULL,
        `Arsenal` VARCHAR(30) ,
        `Avengers` VARCHAR(15) ,
        `Vehicles` VARCHAR(30) NULL,
        PRIMARY KEY (`CN`)
 );
```

20. Schedule:

```
CREATE TABLE `SCHEDULE` (
        `Id` VARCHAR(6) NOT NULL,
        `Departure_Date` DATE ,
        `Expected_Arrival_Date` DATE ,
        `CustomerID` VARCHAR(7) ,
        PRIMARY KEY (`Id`)
);
```

21. Equipment:

```
CREATE TABLE `EQUIPMENT` (
        `EIN` VARCHAR(6) NOT NULL,
        `Model` VARCHAR(45) ,
        `Description` VARCHAR(100) ,
        `Condition` VARCHAR(45) ,
        `Brand` VARCHAR(45) ,
        `Last_Inspect` DATE ,
        `StorageID` VARCHAR(6) NOT NULL,
        `ExtractID` VARCHAR(6) NOT NULL,
        PRIMARY KEY (`EIN`)
);
```

22. Dependent:

```
CREATE TABLE `DEPENDENT` (
        `Name` VARCHAR(30) NOT NULL,
        `eid` VARCHAR(6) NOT NULL,
        `RelationShip` VARCHAR(45) NULL,
        `Gender` VARCHAR(1) NULL,
        `Age` INT NULL,
        PRIMARY KEY (`Name`,`eid`)
 );
```

23. Employee_Phone_Number:

```
CREATE TABLE `EMPLOYEE_PHONE_NUMBER` (
        `EID` VARCHAR(6) NOT NULL,
        `Phone_Number` INT NOT NULL,
        FOREIGN KEY (`EID`) REFERENCES Employee(Id),
        PRIMARY KEY (`EID`, `Phone_Number`)
);
```

24. Employee_Email:

```
CREATE TABLE `EMPLOYEE_EMAIL` (
        `EID` VARCHAR(6) NOT NULL,
        `Email` VARCHAR(45) NOT NULL,
        FOREIGN KEY (`EID`) REFERENCES Employee(Id),
        PRIMARY KEY (`EID`, `Email`)
);
```

25. Customer_Email:

```
CREATE TABLE`CUSTOMER_EMAIL` (
        `CustomerID` VARCHAR(6) NOT NULL,
        `Email` VARCHAR(45) NOT NULL,
        FOREIGN KEY (`CustomerID`) REFERENCES CUSTOMER(Id),
        PRIMARY KEY (`CustomerID`, `Email`)
);
```

26. Customer_Phone_Number:

```
CREATE TABLE `CUSTOMER_PHONE_NUMBER` (
        `CustomerID` VARCHAR(6) NOT NULL,
        `Phone_Number` INT NOT NULL,
        FOREIGN KEY (`CustomerID`) REFERENCES CUSTOMER(Id),
        PRIMARY KEY (`CustomerID`, `Phone_Number`)
);
```

Some referential integrity constraints cannot be added directly upon creation of the table because the primary key it references is still not created because of that we are obliged to use the ALTER command.

```
ALTER TABLE `EMPLOYEE` ADD ( FOREIGN KEY (`DepartmentID`) REFERENCES `DEPARTMENT` (`Id`));

ALTER TABLE `PRODUCT` ADD ( FOREIGN KEY (`StorageID`) REFERENCES `STORAGE` (`Id`));

ALTER TABLE `PRODUCT` ADD ( FOREIGN KEY(`CompoundN`) REFERENCES `AVANGERS_COMPOUND`(`CN`));

ALTER TABLE `EQUIPMENT` ADD ( FOREIGN KEY(`StorageID`) REFERENCES `STORAGE`(`Id`));

ALTER TABLE `EQUIPMENT` ADD ( FOREIGN KEY(`ExtractID`) REFERENCES `EXTRACT_POINT`(`Id`));

ALTER TABLE `SCHEDULE` ADD ( FOREIGN KEY(`CustomerID`) REFERENCES `CUSTOMER`(`Id`));

ALTER TABLE `SHIPMENT` ADD ( FOREIGN KEY(`ScheduleID`) REFERENCES `SCHEDULE`(`Id`));
```

VIII. Table Descriptions:

After creating all the tables on the oracle database server we can view the description of each table in order to make sure everything is fine and no mistakes were made during creation of table.

In our database we have the following tables created on the oracle database server:

1. Customer:

Field	Type	Null	Key	Default	Extra
Id	varchar(7)	NO	PRI	NULL	
Name	**varchar(25)**	**NO**		**NULL**	
Category	**varchar(25)**	**YES**		**NULL**	
Address	**varchar(45)**	**YES**		**NULL**	

2. Product:

Field	Type	Null	Key	Default	Extra
PId	varchar(6)	NO	PRI	NULL	
Name	varchar(15)	NO		NULL	
Type	varchar(20)	YES		NULL	
Price	int(11)	YES		NULL	
StorageID	varchar(6)	YES	MUL	NULL	
CompoundN	varchar(6)	YES	MUL	NULL	

3. Branch:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Location	varchar(30)	NO		NULL	
Name	varchar(33)	NO		NULL	

4. Department:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Name	varchar(30)	NO		NULL	
HeadCount	int(11)	YES		NULL	

5. Building:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Number	varchar(45)	YES		NULL	
Number_Of_Levels	int(11)	YES		NULL	

6. Equipment:

Field	Type	Null	Key	Default	Extra
EIN	varchar(6)	NO	PRI	NULL	
Model	varchar(45)	YES		NULL	
Description	varchar(100)	YES		NULL	
Condition	varchar(45)	YES		NULL	
Brand	varchar(45)	YES		NULL	
Last_Inspect	varchar(45)	YES		NULL	
StorageID	varchar(6)	YES	MUL	NULL	
ExtractID	varchar(6)	YES	MUL	NULL	

7. Storage:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Capacity	**varchar(45)**	**YES**		**NULL**	
Location	**varchar(45)**	**YES**		**NULL**	

8. SHIPMENT:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Date	**date**	**YES**		**NULL**	
Cost	**int(11)**	**YES**		**NULL**	
ScheduleID	**varchar(6)**	**NO**	**MUL**	**NULL**	

9. Project:

Field	Type	Null	Key	Default	Extra
ProId	varchar(6)	NO	PRI	NULL	
Location	varchar(40)	YES		NULL	
Description	varchar(100)	YES		NULL	
Name	varchar(35)	YES		NULL	

10. Port:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Name	varchar(25)	YES		NULL	
Country	varchar(25)	YES		NULL	
Daily_Charge	int(11)	YES		NULL	

11. Employee:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
First_Name	varchar(15)	NO		NULL	
Middle_Name	varchar(15)	YES		NULL	
Last_Name	varchar(15)	NO		NULL	
Gender	varchar(1)	YES		NULL	
Date_Of_Birth	date	YES		NULL	
Address	varchar(45)	YES		NULL	
Specialization	varchar(20)	YES		NULL	
Salary	int(11)	YES		NULL	
Date_Of_Employment	date	YES		NULL	
DepartmentID	varchar(6)	YES	MUL	NULL	

12. Extract_Point:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Cost	int(11)	YES		NULL	
Location	varchar(45)	YES		NULL	
Project_Id	varchar(6)	NO		NULL	

13. Schedule:

Field	Type	Null	Key	Default	Extra
Id	varchar(6)	NO	PRI	NULL	
Departure_Date	date	YES		NULL	
Expected_Arrival_Date	date	YES		NULL	
CustomerID	varchar(6)	YES	MUL	NULL	

14. Avengers_Compound:

Field	Type	Null	Key	Default	Extra
CN	varchar(6)	NO	PRI	NULL	
Arsenal	varchar(30)	YES		NULL	
Avengers	varchar(15)	YES		NULL	
Vehicles	varchar(30)	YES		NULL	

15. DEPENDENT:

Field	Type	Null	Key	Default	Extra
Name	varchar(30)	NO	PRI	NULL	
eid	varchar(6)	NO	PRI	NULL	
RelationShip	varchar(45)	YES		NULL	
Gender	varchar(1)	YES		NULL	
Age	int(11)	YES		NULL	

16. SERVES:

Field	Type	Null	Key	Default	Extra
EID	varchar(6)	NO	PRI	NULL	
CustomerID	varchar(6)	NO	PRI	NULL	

17. WORKS_ON:

Field	Type	Null	Key	Default	Extra
EID	varchar(6)	NO	PRI	NULL	
ProjectID	varchar(6)	NO	PRI	NULL	
Hours	int(11)	YES		NULL	

18. IS_IN:

Field	Type	Null	Key	Default	Extra
DepartmentID	varchar(6)	NO	PRI	NULL	
BranchID	**varchar(6)**	**NO**	**PRI**	**NULL**	

19. BUYS:

Field	Type	Null	Key	Default	Extra
CustomerID	varchar(6)	NO	PRI	NULL	
ProductID	varchar(6)	NO	PRI	NULL	
Quantity	INT(11)	YES		NULL	

20. LOCATES:

Field	Type	Null	Key	Default	Extra
EID	varchar(6)	NO	PRI	NULL	
ExtractID	**varchar(6)**	**NO**	**PRI**	**NULL**	

21. ARRIVES:

Field	Type	Null	Key	Default	Extra
PortId	varchar(6)	NO	PRI	NULL	
ShipmentID	**varchar(6)**	**NO**	**PRI**	**NULL**	

22. IS_LOCATED:

Field	Type	Null	Key	Default	Extra
DepartmentID	varchar(6)	NO	PRI	NULL	
BuildingID	**varchar(6)**	**NO**	**PRI**	**NULL**	

23. EMPLOYEE_EMAIL:

Field	Type	Null	Key	Default	Extra
EID	varchar(6)	NO	PRI	NULL	
Email	**varchar(45)**	**NO**	**PRI**	**NULL**	

24. EMPLOYEE_PHONE_NUMBER:

Field	Type	Null	Key	Default	Extra
EID	varchar(6)	NO	PRI	NULL	
Phone_Number	**int(11)**	**NO**	**PRI**	**NULL**	

25. CUSTOMER_PHONE_NUMBER:

Field	Type	Null	Key	Default	Extra
CustomerID	varchar(6)	NO	PRI	NULL	
Phone_Number	**int(20)**	**NO**	**PRI**	**NULL**	

26. CUSTOMER_EMAIL:

Field	Type	Null	Key	Default	Extra
CustomerID	varchar(6)	NO	PRI	NULL	
Email	**varchar(45)**	**NO**	**PRI**	**NULL**	

IX. Inserting Data:

1. Customer:

```
INSERT INTO CUSTOMER VALUES('C-0001','Quandale Dingle','Household','Ohio');
INSERT INTO CUSTOMER VALUES('C-0002','Joe Biden','Country','Washington');
INSERT INTO CUSTOMER VALUES('C-0003','Medco','Gas Station','Tyre');
INSERT INTO CUSTOMER VALUES('C-0004','Tony Stark','Household','New York');
INSERT INTO CUSTOMER VALUES('C-0005','Charles Smith','Household','Australia');
INSERT INTO CUSTOMER VALUES('C-0006','Rania Dib' ,'Household','Beirut');
INSERT INTO CUSTOMER VALUES('C-0007','Jonesy','Household','Salty Springs');
INSERT INTO CUSTOMER VALUES('C-0008','ELectricite Du Liban' ,'Company','Beirut');
INSERT INTO CUSTOMER VALUES('C-0009','Seven Eleven','Gas Station',' New Jersey');
INSERT INTO CUSTOMER VALUES('C-0010','Hadi Mortada','Household','Beirut');
```

2. Product:

```
INSERT INTO PRODUCT VALUES('P-0001','Refinery Gas','Gas','50','SRG-07',NULL);
INSERT INTO PRODUCT VALUES('P-0002','Ethane','Dangerous Gas','75',NULL,'CN-001');
INSERT INTO PRODUCT VALUES('P-0003','Gasoline','Oil','100','SRG-03','CN-003');
INSERT INTO PRODUCT VALUES('P-0004','Diesel Oil','Oil','100',NULL,'CN-001');
INSERT INTO PRODUCT VALUES('P-0005','Fuel Oil','Fuel','120','SRG-01',NULL);
INSERT INTO PRODUCT VALUES('P-0006','LPG','Liquified Gas','40','SRG-07',NULL);
INSERT INTO PRODUCT VALUES('P-0007','Kerosene','Fuel','82','SRG-02',NULL);
INSERT INTO PRODUCT VALUES('P-0008','Petroleum Oil','Petrol','160',SRG-01',NULL);
INSERT INTO PRODUCT VALUES('P-0009','Oxygen Gas', 'Natural Gas','50',NULL,'CN-001');
INSERT INTO PRODUCT VALUES('P-0010','CNG','Natural Gas','75','SRG-01',NULL);
```

3. Branch:

```
INSERT INTO BRANCH VALUES('B-0001','Beirut','Stark Industries Beirut');
INSERT INTO BRANCH VALUES('B-0002','Barcelona','Stark Industries Barcelone');
INSERT INTO BRANCH VALUES('B-0003','KSA','Stark Industries KSA');
INSERT INTO BRANCH VALUES('B-0004','San Francisco’, ‘Stark Industries SF');
INSERT INTO BRANCH VALUES('B-0005','Australia','Stark Industries Australia');
INSERT INTO BRANCH VALUES('B-0006','Dubai','Stark Industries Dubai');
INSERT INTO BRANCH VALUES('B-0007','Germany','Stark Industries Germany');
INSERT INTO BRANCH VALUES('B-0008','Russia','Stark Industries Russia');
INSERT INTO BRANCH VALUES('B-0009','Brazil','Stark Industries Brazil');
INSERT INTO BRANCH VALUES('B-0010','Canada','Stark Industries Canada');
```

4. Department:

```
INSERT INTO DEPARTMENT VALUES('D-0001','Technical Department','32');
INSERT INTO DEPARTMENT VALUES('D-0002','HR Department','14');
INSERT INTO DEPARTMENT VALUES('D-0003','Research Department','44');
INSERT INTO DEPARTMENT VALUES('D-0004','Marketing Department','35');
INSERT INTO DEPARTMENT VALUES('D-0005','Sales Department','25');
INSERT INTO DEPARTMENT VALUES('D-0006','Finance Department','34');
INSERT INTO DEPARTMENT VALUES('D-0007','Shipping Department','31');
INSERT INTO DEPARTMENT VALUES('D-0008','General Management Department','19');
INSERT INTO DEPARTMENT VALUES('D-0009','Purchase Department','26');
INSERT INTO DEPARTMENT VALUES('D-0010','Administrative Department','30');
INSERT INTO DEPARTMENT VALUES('D-0011','Emergency Department','15');
```

5. Building:

```
INSERT INTO BUILDING VALUES('BLD-01','01','4');
INSERT INTO BUILDING VALUES('BLD-02','02','4');
INSERT INTO BUILDING VALUES('BLD-03','03','4');
INSERT INTO BUILDING VALUES('BLD-04','04','4');
INSERT INTO BUILDING VALUES('BLD-05','05','2');
INSERT INTO BUILDING VALUES('BLD-06','06','2');
INSERT INTO BUILDING VALUES('BLD-07','07','5');
INSERT INTO BUILDING VALUES('BLD-08','08','2');
INSERT INTO BUILDING VALUES('BLD-09','09','3');
INSERT INTO BUILDING VALUES('BLD-10','10','6');
```

6. Equipment:

```
INSERT INTO EQUIPMENT VALUES('EQ-001','2018','BullDozer','Good','BobCat','2018','SRG-01','EX-006');
INSERT INTO EQUIPMENT VALUES('EQ-002','2016','Asphalt Paver','Good','Volvo','2017','SRG-01',NULL);
INSERT INTO EQUIPMENT VALUES('EQ-003','2012','TeleHandlers','Bad','JCB','2013',NULL,'EX-005');
INSERT INTO EQUIPMENT VALUES('EQ-004','2019','Dump Truck','Great','KOMATSU','2020','SRG-03',NULL);
INSERT INTO EQUIPMENT VALUES('EQ-005','2022','Excavator','New','VOLVO','2022','SRG-02',NULL);
INSERT INTO EQUIPMENT VALUES('EQ-006','2008','ForkLift','Out of Use', 'Load Lifter','2010',NULL,'EX-006');
INSERT INTO EQUIPMENT VALUES('EQ-007','2017','Pile Boring Machine','Good','BobCat','2018','SRG-10',NULL);
INSERT INTO EQUIPMENT VALUES('EQ-008','2010','Concrete Mixer','Great','TRACEY','2017','SRG-09',NULL);
INSERT INTO EQUIPMENT VALUES('EQ-009','2021','PipeLayer','Bad','CATERPILLAR','2021',NULL,'EX-004');
INSERT INTO EQUIPMENT VALUES('EQ-010','2011','Drill RIg','Great','BobCat','2020',NULL,'EX-004');
```

7. Storage:

```
INSERT INTO STORAGE VALUES('SRG-01','300000','Beirut');
INSERT INTO STORAGE VALUES('SRG-02','200000','Brazil');
INSERT INTO STORAGE VALUES('SRG-03','3000000','Dubai');
INSERT INTO STORAGE VALUES('SRG-04','3000000','KSA');
INSERT INTO STORAGE VALUES('SRG-05','5000000','San Francisco');
INSERT INTO STORAGE VALUES('SRG-06','80000000','Russia');
INSERT INTO STORAGE VALUES('SRG-07','1500000','Barcelona');
INSERT INTO STORAGE VALUES('SRG-08','600000','Egypt');
INSERT INTO STORAGE VALUES('SRG-09','900000','Canada');
INSERT INTO STORAGE VALUES('SRG-01','1000000','Australia');
```

8. Shipment:

```
INSERT INTO SHIPMENT VALUES('SHP-01', '2022-12-05', 'SC-001');
INSERT INTO SHIPMENT VALUES('SHP-02' ,'2022-12-10', 'SC-002');
INSERT INTO SHIPMENT VALUES('SHP-03','2022-12-04', 'SC-003');
INSERT INTO SHIPMENT VALUES('SHP-04','2022-12-24', 'SC-004');
INSERT INTO SHIPMENT VALUES('SHP-05','2023-01-14', 'SC-005');
INSERT INTO SHIPMENT VALUES('SHP-06','2023-12-25', 'SC-006');
INSERT INTO SHIPMENT VALUES('SHP-07','2023-01-05', 'SC-007');
INSERT INTO SHIPMENT VALUES('SHP-08','2023-01-20', 'SC-008');
INSERT INTO SHIPMENT VALUES('SHP-09','2023-01-26', 'SC-009');
INSERT INTO SHIPMENT VALUES('SHP-10','2023-01-15', 'SC-010');
```

9. Project:

```
INSERT INTO PROJECT VALUES('PJ-001','Kana, Lebanon','Possible new extraction point that must be inspected and explored','Kana Oil Field');

INSERT INTO PROJECT VALUES('PJ-002','Beirut, Lebanon','Expansion of existing production facility','BEY-FAC. Expansion');

INSERT INTO PROJECT VALUES('PJ-003','zahle, Lebanon','Construction of a new oil refinery','Zahle oil Integrated Refinery');

INSERT INTO PROJECT VALUES('PJ-004','Cyprus-Lebanon','Building pipelines across to share resources easily','Mediterranean Export');

INSERT INTO PROJECT VALUES('PJ-005','Alexandria, Egypt','Extract Point Exploration','EGY-ExtractPT');

INSERT INTO PROJECT VALUES('PJ-006','Corpus Christi, TX, USA','Construction of liquefaction plants','CC-PLANT');

INSERT INTO PROJECT VALUES('PJ-007','Groom Lake, NV, USA','Designing and testing new weaponry for the US government','AREA51');

INSERT INTO PROJECT VALUES('PJ-008','Planet Titan','Extraction of rare rocks and material','Titan');

INSERT INTO PROJECT VALUES('PJ-009','New York','Creation of new Arsenal and vehicles for the Avengers','Avengers');

INSERT INTO PROJECT VALUES('PJ-010','Chernobyl, Ukraine','Disinfect the area of radiation and build secret storage tank','PROJECTX');
```

10. Port:

```
INSERT INTO PORT VALUES('PRT-01','Beirut Port','Lebanon','20');

INSERT INTO PORT VALUES('PRT-02','Barcelona Port','Spain','17');

INSERT INTO PORT VALUES('PRT-03','Sydney Port','Australia','9');

INSERT INTO PORT VALUES('PRT-04','Newark Port','USA','14');

INSERT INTO PORT VALUES('PRT-05','Frankfurt Port','Germany','6');

INSERT INTO PORT VALUES('PRT-06','Saint Petersburg Port','Russia','2');

INSERT INTO PORT VALUES('PRT-07','Salvador Port','Brazil','11');

INSERT INTO PORT VALUES('PRT-08','Montreal Port','Canada','22');

INSERT INTO PORT VALUES('PRT-09','Larnaca Port','Cyprus','16');

INSERT INTO PORT VALUES('PRT-10','Alexandria Port','Egypt','13');
```

11. Employee:

INSERT INTO EMPLOYEE **VALUES(**'E-0001'**,**'Ahmad'**,**'Mohamad'**,**'Malak'**,**'M','2000-03-09','Saida-Lebanon','IT','2500',2022-04-03, 70045344,'D-0001'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0002'**,**'Adam'**,**'Mohanad'**,**'Kanj'**,**'M','2001-12-18','Beirut-Lebanon','IT','3000',2021-12-20, 03637138,'D-0001'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0003'**,**'Lionelli'**,**'Angelo'**,**'Maroun'**,**'M','1998-11-07','Beirut-Lebanon','HR','2000',2022-01-07,71220302,'D-0002'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0004'**,**'Angelo'**,**'Lionelli'**,**'Abi Aad'**,**'M','1995-11-29'**,**'Aley-Lebanon','Research','2000',2022-02-26,70776505,'D-0003'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0005'**,**'Tala'**,**'Amer'**,**'Itani'**,**'F','2004-08-02','Beirut-Lebanon','Sales','1500',2022-10-29, 76458322,'D-0005'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0006'**,**'Hala'**,**'Hadi'**,**'Mortada'**,**'F','2004-04-17','Beirut-Lebanon','Finance','1750',2022-11-11,03345664,'D-0006'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0007'**,**'Tamer'**,**'Mohamad'**,**'Itani'**,**'M','1997-01-14','Tripoli-Lebanon','Management','2000',2022-04-20,70975446,'D-0008'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0008'**,**'Ragnar'**,**'Adam'**,**'Lothbrok'**,**'M','1990-06-07','Beqaa-Lebanon','Purcahse','3500',2021-04-16,03548944,'D-0009'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0009'**,**'Roman'**,**'Joe'**,**'Reigns'**,**'M','1996-06-19','Saida-Lebanon','Administration','4000', 2021-05-05,70047623,'D-0010'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0010'**,**'Paul'**,**'Simon'**,**'Walker'**,**'M','1982-12-18','Beirut-Lebanon','Shipping','3000',2021-05-22,03987456,'D-0007'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0011'**,**'Vin'**,**'John'**,**'Diesel'**,**'M','1978-03-07','Saida-Lebanon','Shipping','5000',2021-01-19,70979764,'D-0007'**);**

INSERT INTO EMPLOYEE **VALUES(**'E-0012'**,**'Kylie'**,**'Robert'**,**'Jenner'**,**'F','2000-12-01','Jounieh-Lebanon','Marketing','3500',2021-11-11,70298752,'D-0004'**);**

12. Extract_Point:

INSERT INTO EXTRACT_POINT **VALUES(**'EX-001'**,'**7000000','Heilongjiang, China','PJ-001'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-002'**,'**4000000','Liuhua, China','PJ-005'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-003'**,'**3500000','Gulf of Mexico','PJ-009'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-004'**,'**4800000','Orinoco, Venezuela','PJ-008'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-005'**,'**6250000','Cabinda, Angola',PJ-010'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-006'**,'**2320000','Congo Basin, Angola',PJ-007'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-007'**,'**9200000','Fateh, UAE','PJ-002'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-008'**,'**7600000','Dhahran, Saudi Arabia','PJ-006'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-009'**,'**5075000','Campos Basin, Brazil','PJ-004'**);**

INSERT INTO EXTRACT_POINT **VALUES(**'EX-010'**,'**9800000','Zafiro, Equatorial Guinea','PJ-003'**);**

13. Schedule:

```
INSERT INTO SCHEDULE VALUES('SC-001','2022-11-17','2022-12-05','C-0005');
INSERT INTO SCHEDULE VALUES('SC-002','2022-11-20','2022-12-10','C-0003');
INSERT INTO SCHEDULE VALUES('SC-003','2022-11-25','2022-12-04','C-0002');
INSERT INTO SCHEDULE VALUES('SC-004','2022-11-26','2022-12-24','C-0009');
INSERT INTO SCHEDULE VALUES('SC-005','2022-11-29','2023-01-14','C-0010');
INSERT INTO SCHEDULE VALUES('SC-006','2022-12-10','2023-12-20','C-0004');
INSERT INTO SCHEDULE VALUES('SC-007','2022-12-03','2023-01-05','C-0001');
INSERT INTO SCHEDULE VALUES('SC-008','2022-12-10','2023-01-20','C-0007');
INSERT INTO SCHEDULE VALUES('SC-009','2022-12-14','2023-01-28','C-0006');
INSERT INTO SCHEDULE VALUES('SC-010','2022-12-17','2023-01-15','C-0008');
```

14. Avengers_Compound:

```
INSERT INTO AVENGERS_COMPOUND VALUES('CN-001','Shields, ARs','Captain America','F-35 Lightning');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-002','Bows and arrows','Hawkeye','JEEPS and Quinjet');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-003','Gamma weapons','Hulk','Humvee');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-004','Sup. Prosthetics','Winter Soldier','F-35 Lightning');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-005','infinity stones','Thor','Acura, Helicarrier');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-006','Ironman Armory','Peper Potts','Stark Jet');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-007','Flying Equipment','Falcon','S.H.I.E.L.D.');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-008','Claws, Suits','Black Panther','Wakandan Tech');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-009','Web shooters','Spiderman','Acura Stark Car');
INSERT INTO AVENGERS_COMPOUND VALUES('CN-010','Ultron, Jarvis','Ironman','AI suits/Tech');
```

15. DEPENDENT:

```
INSERT INTO DEPENDENT VALUES('Karim Abi Aad','E-0004','Cousin','M','19');
INSERT INTO DEPENDENT VALUES('Tony Abi Aad','E-0004','Brother','M','12');
INSERT INTO DEPENDENT VALUES('Nour Elly Maroun','E-0003','Sister','F','22');
INSERT INTO DEPENDENT VALUES('Naya Malak','E-0001','Sister','F','18');
INSERT INTO DEPENDENT VALUES('Amer Itani','E-0007','Brother','M','18');
INSERT INTO DEPENDENT VALUES('Ram  Kanj','E-0002','Brother','M','12');
INSERT INTO DEPENDENT VALUES('Dwayne Johnson','E-0009','Uncle','M','49');
INSERT INTO DEPENDENT VALUES('Kim Kardashian','E-0012','Sister','F','42');
INSERT INTO DEPENDENT VALUES('Dom Toretto','E-0011','Family','M','39');
INSERT INTO DEPENDENT VALUES('Darth Vader','E-0006','Father','M','60');
```

16. SERVES:

```
INSERT INTO SERVES VALUES('E-0005','C-0001');
INSERT INTO SERVES VALUES('E-0010','C-0002');
INSERT INTO SERVES VALUES('E-0011','C-0003');
INSERT INTO SERVES VALUES('E-0005','C-0004');
INSERT INTO SERVES VALUES('E-0010','C-0005');
INSERT INTO SERVES VALUES('E-0011','C-0006');
INSERT INTO SERVES VALUES('E-0005','C-0007');
INSERT INTO SERVES VALUES('E-0010','C-0008');
INSERT INTO SERVES VALUES('E-0011','C-0009');
INSERT INTO SERVES VALUES('E-0005','C-0010');
```

17. WORKS_ON:

```
INSERT INTO WORKS_ON VALUES('E-0001',’PJ-001’,200);
INSERT INTO WORKS_ON VALUES('E-0001',’PJ-002’,110);
INSERT INTO WORKS_ON VALUES('E-0001',’PJ-003’,192);
INSERT INTO WORKS_ON VALUES('E-0001',’PJ-005’,92);
INSERT INTO WORKS_ON VALUES('E-0001',’PJ-007’,160);
INSERT INTO WORKS_ON VALUES('E-0001',’PJ-009’,190);
INSERT INTO WORKS_ON VALUES('E-0004',’PJ-004’,86);
INSERT INTO WORKS_ON VALUES('E-0004',’PJ-006’,60);
INSERT INTO WORKS_ON VALUES('E-0004',’PJ-008’,120);
INSERT INTO WORKS_ON VALUES('E-0004',’PJ-010’,78);
```

18. IS_IN:

```
INSERT INTO IS_IN VALUES('D-0001',’B-0005’);
INSERT INTO IS_IN VALUES('D-0003',’B-0003’);
INSERT INTO IS_IN VALUES('D-0005',’B-0001’);
INSERT INTO IS_IN VALUES('D-0008',’B-0001’);
INSERT INTO IS_IN VALUES('D-0006',’B-0004’);
INSERT INTO IS_IN VALUES('D-0004',’B-0002’);
INSERT INTO IS_IN VALUES('D-0002',’B-0010’);
INSERT INTO IS_IN VALUES('D-0010',’B-0007’);
INSERT INTO IS_IN VALUES('D-0009',’B-0008’);
INSERT INTO IS_IN VALUES('D-0006',’B-0006’);
```

19. BUYS:

```
INSERT INTO BUYS VALUES('C-0001','P-0005','77');
INSERT INTO BUYS VALUES('C-0003','P-0003','445');
INSERT INTO BUYS VALUES('C-0005','P-0001','55');
INSERT INTO BUYS VALUES('C-0008','P-0001','42');
INSERT INTO BUYS VALUES('C-0006','P-0004','634');
INSERT INTO BUYS VALUES('C-0004','P-0002','53');
INSERT INTO BUYS VALUES('C-0002','P-0010','55');
INSERT INTO BUYS VALUES('C-0010','P-0007','65464');
INSERT INTO BUYS VALUES('C-0009','P-0008','365');
INSERT INTO BUYS VALUES('C-0006','P-0006','556');
```

20. LOCATES:

```
INSERT INTO LOCATES VALUES('E-0001','EX-001');
INSERT INTO LOCATES VALUES('E-0001','EX-002');
INSERT INTO LOCATES VALUES('E-0004','EX-003');
INSERT INTO LOCATES VALUES('E-0004','EX-004');
INSERT INTO LOCATES VALUES('E-0001','EX-005');
INSERT INTO LOCATES VALUES('E-0001','EX-006');
INSERT INTO LOCATES VALUES('E-0004','EX-007');
INSERT INTO LOCATES VALUES('E-0004','EX-008');
INSERT INTO LOCATES VALUES('E-0001','EX-009');
INSERT INTO LOCATES VALUES('E-0001','EX-010');
```

21. ARRIVES:

```
INSERT INTO ARRIVES VALUES('PRT-01',’SHP-05’);
INSERT INTO ARRIVES VALUES('PRT-03',’SHP-03’);
INSERT INTO ARRIVES VALUES('PRT-05',’SHP-01’);
INSERT INTO ARRIVES VALUES('PRT-08',’SHP-01’);
INSERT INTO ARRIVES VALUES('PRT-06',’SHP-04’);
INSERT INTO ARRIVES VALUES('PRT-04',’SHP-02’);
INSERT INTO ARRIVES VALUES('PRT-02',’SHP-10’);
INSERT INTO ARRIVES VALUES('PRT-10',’SHP-07’);
INSERT INTO ARRIVES VALUES('PRT-09',’SHP-08’);
INSERT INTO ARRIVES VALUES('PRT-06',’SHP-06’);
```

22. IS_LOCATED:

```
INSERT INTO IS_LOCATED VALUES('D-0001',’BLD-05’);
INSERT INTO IS_LOCATED VALUES('D-0003',’BLD-03’);
INSERT INTO IS_LOCATED VALUES('D-0005',’BLD-01’);
INSERT INTO IS_LOCATED VALUES('D-0008',’BLD-01’);
INSERT INTO IS_LOCATED VALUES('D-0006',’BLD-04’);
INSERT INTO IS_LOCATED VALUES('D-0004',’BLD-02’);
INSERT INTO IS_LOCATED VALUES('D-0002',’BLD-10’);
INSERT INTO IS_LOCATED VALUES('D-0010',’BLD-07’);
INSERT INTO IS_LOCATED VALUES('D-0009',’BLD-08’);
INSERT INTO IS_LOCATED VALUES('D-0006',’BLD-06’);
```

23. EMPLOYEE_EMAIL:

```
INSERT INTO EMPLOYEE_EMAIL VALUES('E-0001','Ahmad.Malak@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0001','Ahmad.Malak@gmail.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0002','Adam.Kanj@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0002','Adam_Mohanad_Kanj@gmail.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0002','Adam.Kanj@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0002','Adam.Kanj@lau.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0003','Lionelli.Maroun@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0003','Lionelli.Maroun@lau.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0004','Angelo.AbiAad@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0005','Tala.Itani@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0006','Hala.Mortada@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0007','Tamer.Itani@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0008','Ragnar.Lothbrock@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0009','Roman.Reigns@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0010','Paul.Walker@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0011','Vin.Diesel@DA.com');

INSERT INTO EMPLOYEE_EMAIL VALUES('E-0012','Kylie.Jenner@DA.com');
```

24. EMPLOYEE_PHONE_NUMBER:

```
INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0001','70034562');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0002','03789654');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0003','70145236');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0004','76077762');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0004','07078887');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0005','98765432');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0005','75395145');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0005','98747897');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0006','65478932');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0007','44556666');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0008','01010101');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0009','21122111');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0010','11111111');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0010','11111112');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0011','00000000');

INSERT INTO EMPLOYEE_PHONE_NUMBER VALUES('E-0012','71111111');
```

25. CUSTOMER_PHONE_NUMBER:

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0001','12165130514');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0002','16505130514);

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0003','55555444');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0003','70000008')

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0004','12188787887');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0005','101112131415');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0006','03666555');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0007','778896621447');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0008',' 01442720');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0008',' 08823455');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0008',' 08823455');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0009',' 101112131222');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0009',' 101112131222');

INSERT INTO CUSTOMER _PHONE_NUMBER **VALUES**('C-0010',' 76716733');

26. CUSTOMER_EMAIL:

```
INSERT INTO CUSTOMER_EMAIL VALUES('C-0001','Quandle.Dingle@gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0002','Joe.Biden@gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0002','USA.President@gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0003','Medco@gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0004','Tony.Stark@gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0004','Stark.Industries@gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0004','IronMan@Avengers.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0005','Charles.Smith @gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0005','Charles.Smith @NBA.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0006','Rania.Dib @hotmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0007','Jonesyyyy @support.epicgames.com);

INSERT INTO CUSTOMER_EMAIL VALUES('C-0008','EDL @gov.lb');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0009','Seven.Eleven @gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0010','Hadi.Mortada@gmail.com');

INSERT INTO CUSTOMER_EMAIL VALUES('C-0010','Hadi.Mortada@lau.edu);
```

X. Final Tables State:

1. Customer:

Id	Name	Category	Address
C-0001	Quandale Dingle	Household	Ohio
C-0002	Joe Biden	Country	Washington
C-0003	Medco	Gas Station	Tyre
C-0004	Tony Stark	Household	New York
C-0005	Charles Smith	Household	Australia
C-0006	Rania Dib	Household	Beirut
C-0007	Jonesy	Household	Salty Springs
C-0008	ELectricite Du Liban	Company	Beirut
C-0009	Seven Eleven	Gas Station	New Jersey
DR-0010	Hadi Mortada	Household	Beirut

2. Product:

PId	Name	Type	Price	StorageID	CompoundN
P-0001	Refinery Gas	Gas	50	SRG-07	NULL
P-0002	Ethane	Dangerous Gas	75	NULL	CN-001
P-0003	Gasoline	Oil	100	SRG-03	CN-003
P-0004	Diesel Oil	Oil	100	NULL	CN-001
P-0005	Fuel Oil	Fuel	120	SRG-01	NULL
P-0006	LPG	Liquified Gas	40	SRG-07	NULL
P-0007	Kerosene	Fuel	82	SRG-02	NULL
P-0008	Petroleum Oil	Petrol	160	SRG-01	NULL
P-0009	Oxygen Gas	Natural Gas	50	NULL	CN-001
P-0010	CNG	Natural Gas	75	SRG-01	NULL

3. Branch:

Id	Location	Name
B-0001	Beirut	Stark Industries Beirut
B-0002	Barcelona	Stark Industries Barcelone
B-0003	KSA	Stark Industries KSA
B-0004	San Francisco	Stark Industries SF
B-0005	Australia	Stark Industries Australia
B-0006	Dubai	Stark Industries Dubai
B-0007	Germany	Stark Industries Germany
B-0008	Russia	Stark Industries Russia
B-0009	Brazil	Stark Industries Brazil
B-0010	Canada	Stark Industries Canada

4. Department:

Id	Name	HeadCount
D-0001	Technical Department	32
D-0002	HR Department	14
D-0003	Research Department	44
D-0004	Marketing Department	35
D-0005	Sales Department	25
D-0006	Finance Department	34
D-0007	Shipping Department	31
D-0008	General Management Department	19
D-0009	Purchase Department	26
D-0010	Administrative Department	30
D-0011	Emergency Department	15

5. Building:

Id	Number	Number_Of_Levels
BLD-01	1	4
BLD-02	2	4
BLD-03	3	4
BLD-04	4	4
BLD-05	5	2
BLD-06	6	2
BLD-07	7	5
BLD-08	8	2
BLD-09	9	3
BLD-10	10	6

6. Equipment:

EIN	Model	Description	Condition	Brand	Last_Inspect	StorageID	ExtractID
EQ-001	2018	BullDozer	Good	BobCat	2018	SRG-01	NULL
EQ-002	2016	Asphalt Paver	Good	Volvo	2017	SRG-01	NULL
EQ-003	2012	TeleHandlers	Bad	JCB	2013	NULL	EX-005
EQ-004	2019	Dump Truck	Great	KOMATSU	2020	SRG-03	NULL
EQ-005	2022	Excavator	New	VOLVO	2022	SRG-02	NULL
EQ-006	2008	ForkLift	Out of Use	Load Lifter	2010	NULL	EX-006
EQ-007	2017	Pile Boring Machine	Good	BobCat	2018	SRG-10	NULL
EQ-008	2010	Concrete Mixer	Great	TRACEY	2017	SRG-09	NULL
EQ-009	2021	PipeLayer	Bad	CATERPILLAR	2021	NULL	EX-004
EQ-010	2011	Drill Rlg	Great	BobCat	2020	NULL	EX-004

7. Storage:

Id	Capacity	Location
SRG-01	300000	Beirut
SRG-02	200000	Brazil
SRG-03	3000000	Dubai
SRG-04	3000000	KSA
SRG-05	5000000	San Francisco
SRG-06	80000000	Russia
SRG-07	1500000	Barcelona
SRG-08	600000	Egypt
SRG-09	900000	Canada
SRG-10	1000000	Australia

8. SHIPMENT:

Id	Date	Cost	ScheduleID
SHP-01	12/5/2022	234	SC-001
SHP-02	12/10/2022	564	SC-002
SHP-03	12/4/2022	554	SC-003
SHP-04	12/24/2022	547	SC-004
SHP-05	1/14/2023	66	SC-005
SHP-06	12/25/2023	35	SC-006
SHP-07	1/5/2023	556	SC-007
SHP-08	1/20/2023	54	SC-008
SHP-09	1/26/2023	670	SC-009
SHP-10	1/15/2023	4142	SC-010

9. Project:

ProId	Location	Description	Name
PJ-001	Kana, Lebanon	Possible new extraction point that must be inspected and explored	Kana Oil Field
PJ-002	Beirut, Lebanon	Expansion of existing production facility	BEY-FAC. Expansion
PJ-003	zahle, Lebanon	Construction of a new oil refinery	Zahle oil Integrated Refinery
PJ-004	Cyprus-Lebanon	Building pipelines across to share resources easily	Mediterranean Export
PJ-005	Alexandria, Egypt	Extract Point Exploration	EGY-ExtractPT
PJ-006	Corpus Christi, TX, USA	Construction of liquefaction plants	CC-PLANT
PJ-007	Groom Lake, NV, USA	Designing and testing new weaponry for the US government	AREA51
PJ-008	Planet Titan	Extraction of rare rocks and material	Titan
PJ-009	New York	Creation of new Arsenal and vehicles for the Avengers	Avengers
PJ-010	Chernobyl, Ukraine	Disinfect the area of radiation and build secret storage tank	PROJECTX

10. Port:

Id	Name	Country	Daily_Charge
PRT-01	Beirut Port	Lebanon	20
PRT-02	Barcelona Port	Spain	17
PRT-03	Sydney Port	Australia	9
PRT-04	Newark Port	USA	14
PRT-05	Frankfurt Port	Germany	6
PRT-06	Saint Petersburg Port	Russia	2
PRT-07	Salvador Port	Brazil	11
PRT-08	Montreal Port	Canada	22
PRT-09	Larnaca Port	Cyprus	16
PRT-10	Alexandria Port	Egypt	13

11. Schedule:

Id	Departure_Date	Expected_Arrival_Date	CustomerID
SC-001	11/17/2022	12/5/2022	C-0005
SC-002	11/20/2022	12/10/2022	C-0003
SC-003	11/25/2022	12/4/2022	C-0002
SC-004	11/26/2022	12/24/2022	C-0009
SC-005	11/29/2022	1/14/2023	DR-0010
SC-006	12/10/2022	12/20/2023	C-0004
SC-007	12/3/2022	1/5/2023	C-0001
SC-008	12/10/2022	1/20/2023	C-0007
SC-009	12/14/2022	1/28/2023	C-0006
SC-010	12/17/2022	1/15/2023	C-0008

12. Extract_Point:

Id	Cost	Location	Project_Id
EX-001	7000000	Heilongjian China	PJ-001
EX-002	4000000	iuhua ,China	PJ-005
EX-003	3500000	Gulf of Mexico	PJ-009
EX-004	4800000	Orinoco, Venezuela	PJ-008
EX-005	6250000	Cabinda Angola	PJ-010
EX-006	2320000	Congo Basin , Angola	PJ-007
EX-007	9200000	Fateh, UAE	PJ-008
EX-008	7600000	Dhahran, Saudi Arabia	PJ-002
EX-009	5075000	Campos Basin, Brazil	PJ-004
EX-010	9800000	Zafiro, Equatorial Guinea	PJ-003

13. Avengers_Compound:

CN	Arsenal	Avengers	Vehicles
CN-001	Shields, ARs	Captain America	F-35 Lightning
CN-002	Bows and arrows	Hawkeye	JEEPS and Quinjet
CN-003	Gamma weapons	Hulk	Humvee
CN-004	Sup. Prosthetics	Winter Soldier	F-35 Lightning
CN-005	infinity stones	Thor	Acura, Helicarrier
CN-006	Ironman Armory	Peper Potts	Stark Jet
CN-007	Flying Equipment	Falcon	S.H.I.E.L.D.
CN-008	Claws, Suits	Black Panther	Wakandan Tech
CN-009	Web shooters	Spiderman	Acura Stark Car
CN-010	Ultron, Jarvis	Ironman	AI suits/Tech

14. Employee:

Id	First_Name	Middle_Name	Last_Name	Gender	Date_Of_Birth	Address	Specialization	Salary	Date_Of_Employment	DepartmentID
E-0001	Ahmad	Mohamad	Malak	M	3/9/2000	Saida-Lebanon	IT	2500	4/3/2022	D-0003
E-0002	Adam	Mohanad	Kanj	M	12/18/2001	Beirut-Lebanon	IT	3000	12/20/2021	D-0001
E-0003	Lionelli	Angelo	Maroun	M	11/7/1998	Beirut-Lebanon	HR	2000	1/7/2022	D-0002
E-0004	Angelo	Lionelli	Abi Aad	M	11/29/1995	Aley-Lebanon	Research	2000	2/26/2022	D-0003
E-0005	Tala	Amer	Itani	F	8/2/2004	Beirut-Lebanon	Sales	1500	10/29/2022	D-0005
E-0006	Hala	Hadi	Mortada	F	4/17/2004	Beirut-Lebanon	Finance	1750	11/11/2022	D-0006
E-0007	Tamer	Mohamad	Itani	M	1/14/1997	Tripoli-Lebanon	Management	2000	4/20/2022	D-0008
E-0008	Ragnar	Adam	Lothbrok	M	6/7/1990	Beqaa-Lebanon	Purcahse	3500	4/16/2021	D-0009
E-0009	Roman	Joe	Reigns	M	6/19/1996	Saida-Lebanon	Administration	4000	5/5/2021	D-0010
E-0010	Paul	Simon	Walker	M	12/18/1982	Beirut-Lebanon	Shipping	3000	5/22/2021	D-0007
E-0011	Vin	John	Diesel	M	3/7/1978	Saida-Lebanon	Shipping	5000	1/19/2021	D-0007
E-0012	Kylie	Robert	Jenner	F	12/1/2000	Jounieh-Lebanon	Marketing	3500	11/11/2021	D-0004

15. DEPENDENT:

Name	eid	RelationShip	Gender	Age
Amer Itani	E-0007	Brother	M	18
Darth Vader	E-0006	Father	M	60
Dom Toretto	E-0011	Family	M	39
Dwayne Johnson	E-0009	Uncle	M	49
Karim Abi Aad	E-0004	Cousin	M	19
Kim Kardashian	E-0012	Sister	F	42
Naya Malak	E-0001	Sister	F	18
Nour Maroun	E-0003	Sister	F	22
Ram Kanj	E-0002	Brother	M	12
Tony Abi Aad	E-0004	Brother	M	12

16. SERVES:

EID	CustomerID
E-0005	C-0001
E-0010	C-0002
E-0011	C-0003
E-0005	C-0004
E-0010	C-0005
E-0011	C-0006
E-0005	C-0007
E-0010	C-0008
E-0011	C-0009
E-0005	C-0010

17. WORKS_ON:

EID	ProjectID	Hours
E-0001	PJ-005	92
E-0001	PJ-007	160
E-0004	PJ-006	60
E-0004	PJ-004	86
E-0001	PJ-003	192
E-0001	PJ-002	110
E-0004	PJ-010	78
E-0004	PJ-008	120
E-0001	PJ-001	200
E-0001	PJ-009	190

18. IS_IN:

DepartmentID	BranchID
D-0005	B-0001
D-0008	B-0001
D-0004	B-0002
D-0003	B-0003
D-0006	B-0004
D-0001	B-0005
D-0006	B-0006
D-0010	B-0007
D-0009	B-0008
D-0002	B-0010

19. BUYS:

CustomerID	ProductID	Quantity
C-0005	P-0001	77
C-0008	P-0001	445
C-0004	P-0002	55
C-0003	P-0003	42
C-0006	P-0004	634
C-0001	P-0005	53
C-0006	P-0006	55
C-0010	P-0007	65464
C-0009	P-0008	365
C-0002	P-0010	556

20. LOCATES:

EID	ExtractID
E-0001	EX-001
E-0001	EX-002
E-0004	EX-003
E-0004	EX-004
E-0001	EX-005
E-0001	EX-006
E-0004	EX-007
E-0004	EX-008
E-0001	EX-009
E-0001	EX-010

21. ARRIVES:

PortId	ShipmentID
PRT-05	SHP-01
PRT-08	SHP-01
PRT-04	SHP-02
PRT-03	SHP-03
PRT-06	SHP-04
PRT-01	SHP-05
PRT-06	SHP-06
PRT-10	SHP-07
PRT-09	SHP-08
PRT-02	SHP-10

22. IS_LOCATED:

DepartmentID	BuildingID
D-0005	BLD-01
D-0008	BLD-01
D-0004	BLD-02
D-0003	BLD-03
D-0006	BLD-04
D-0001	BLD-05
D-0006	BLD-06
D-0010	BLD-07
D-0009	BLD-08
D-0002	BLD-10

23. EMPLOYEE_EMAIL:

EID	Email
E-0001	Ahmad.Malak@DA.com
E-0001	Ahmad.Malak@gmail.com
E-0002	Adam_Mohanad_Kanj@gmail.com
E-0002	Adam.Kanj@DA.com
E-0002	Adam.Kanj@DAA.com
E-0002	Adam.Kanj@lau.com
E-0003	Lionelli.Maroun@DA.com
E-0003	Lionelli.Maroun@lau.com
E-0004	Angelo.AbiAad@DA.com
E-0005	Tala.Itani@DA.com
E-0006	Hala.Mortada@DA.com
E-0007	Tamer.Itani@DA.com
E-0008	Ragnar.Lothbrock@DA.com
E-0009	Roman.Reigns@DA.com
E-0010	Paul.Walker@DA.com
E-0011	Vin.Diesel@DA.com
E-0012	Kylie.Jenner@DA.com

24. EMPLOYEE_PHONE_NUMBER:

EID	Phone_Number
E-0001	70034562
E-0002	3789654
E-0003	70145236
E-0004	7078887
E-0004	76077762
E-0005	75395145
E-0005	98747897
E-0005	98765432
E-0006	65478932
E-0007	44556666
E-0008	1010101
E-0009	21122111
E-0010	11111111
E-0010	11111112
E-0012	71111111

25. CUSTOMER_PHONE_NUMBER:

CustomerID	Phone_Number
C-0001	21651305
C-0002	16505130
C-0003	55555444
C-0003	70000008
C-0004	12188787
C-0005	10111213
C-0006	3666555
C-0007	77881447
C-0008	1442720
C-0008	7823455
C-0008	8823455
C-0009	10111222
C-0009	12131222
C-0010	76716733

26. CUSTOMER_EMAIL:

CustomerID	Email
C-0001	Quandle.Dingle@gmail.com
C-0002	Joe.Biden@gmail.com
C-0002	USA.President@gmail.com
C-0003	Medco@gmail.com
C-0004	IronMan@Avengers.com
C-0004	Stark.Industries@gmail.com
C-0004	Tony.Stark@gmail.com
C-0005	Charles.Smith @gmail.com
C-0005	Charles.Smith @NBA.com
C-0006	Rania.Dib @hotmail.com
C-0007	Jonesyyyy @support.epicgames.com
C-0008	EDL @gov.lb
C-0009	Seven.Eleven @gmail.com
C-0010	Hadi.Mortada@gmail.com
C-0010	Hadi.Mortada@lau.edu

XIV. Queries:

1. Very competent work is produced by Stark Industry. They desire the best items, best services, and quickest delivery times in order to preserve our clients' contentment and make the best impression. We will track down the clients' names and shipping dates for the shipments that arrived on schedule.

   - **SQL QUERY:**

```
SELECT C.Name , SH.Date, S.Expected_Arrival_Date
FROM CUSTOMER  C, SCHEDULE  S, SHIPMENT  SH
WHERE S.CustomerID=C.Id and SH.ScheduleID=S.Id and SH.Date!=S.Expected_Arrival_Date;
```

OUTPUT:

Name	Date	Expected_Arrival_Date
Charles Smith	2022-12-05	2022-12-05
Medco Gasoline	2022-12-10	2022-12-10
Joe Biden	2022-12-04	2022-12-04
Seven Eleven	2022-12-24	2022-12-24
Hadi Mortada	2023-01-14	2023-01-14
Quandale Dingle	2023-01-05	2023-01-05
Jonesy	2023-01-20	2023-01-20
ELectricite Du Liban	2023-01-15	2023-01-15

2. Our work is extremely sensitive and crucial. The objective is to determine whether the product was delivered to the customer on time and according to plan. Therefore, the names of consumers whose shipments didn't arrive on time will be retrieved. We must thus express our apologies to these clients and seek a solution.

- **SQL QUERY:**

```
SELECT C.Name
FROM CUSTOMER  C, SCHEDULE  S, SHIPMENT  SH
WHERE S.CustomerID=C.Id and SH.ScheduleID=S.Id and SH.Date!=S.Expected_Arrival_Date;
```

OUTPUT:

Name
Tony Stark
Rania Dib

3. We must inspect the availability of the products in storage at the end of each month. As a result, we select the product ID and check to see if it is in storage. So, get the storage name and the product names that are stored in.

- **SQL QUERY:**

```
SELECT P.Name , S.Id

FROM PRODUCT P, STORAGE S

WHERE P.StorageID=S.Id;
```

OUTPUT:

```
Name                 Id
-------------------  ----------
Refinery Gas         SRG-07

Gassolne             SRG-03

Fuel Oil             SRG-01

LPG                  SRG-07

Kerosene             SRG-02

Petroleum Oil        SRG-01

CNG                  SRG-01
```

4. The number of hours spent working for an employee can show a lot. It can highlight his/her trustworthiness and reliability as an employee. Consistently being punctual can also help show that this employee meet the standards of professionalism, which can increase his/her value and help advance the employee's career. Thus, we trace the hard workers in our company and seek the ones who come up with brightest ideas and achieve these projects to later on be promoted and hounoured.

- **SQL QUERY:**

```
SELECT E.First_Name, E.Middle_Name, E.Last_Name, (D.Name) AS Department_Name, (P.Name) AS
Project_Name, Hours

FROM WORKS_ON W, EMPLOYEE E, PROJECT P, DEPARTMENT D

WHERE W.EID=E.Id AND W.ProjectID=P.ProId AND E.DepartmentID=D.Id;
```

OUTPUT:

First_Name	Middle_Name	Last_Name	Department_Name	Project_Name	Hours
Ahmad	Mohamad	Malak	Research Department	Kana Oil field	200
Ahmad	Mohamad	Malak	Research Department	Bey-Fac Expansion	110
Ahmad	Mohamad	Malak	Research Department	Zahle oil integr--ated Refinery	192
Ahmad	Mohamad	Malak	Research Department	EGY-ExtractPT	92
Ahmad	Mohamad	Malak	Research Department	Area51	160
Ahmad	Mohamad	Malak	Research Department	Avengers	190
Angelo	Lionelli	Abi Aad	Research Department	Mediterranean Export	86
Angelo	Lionelli	Abi Aad	Research Department	CC-PLANT	60
Angelo	Lionelli	Abi Aad	Research Department	Titan	120
Angelo	Lionelli	Abi Aad	Research Department	PROJECTX	78

5. The Avengers compound should always be fully stacked with various arsenal and vehicles for the avengers to always be prepared for a fight with the enemies that threaten the world. Therefore, numerous products are needed to assemble and build those weapons that best suits each avenger. Here are some of the products that each avenger gets from stark industries in order to manufacture those weapons.

- **SQL QUERY:**

```
SELECT A.Arsenal, P.Name, A.Avengers
FROM AVENGERS_COMPOUND A, PRODUCT P
WHERE P.CompoundN=A.CN;
```

OUTPUT:

ARSENAL	Name	AVENGERS
Shields, ARs	Ethane	Captain America
Gamma weapons	Gasoline	Hulk
Shields, Ars	Diesel Oil	Captain America
Shields, Ars	Oxygen Gas	Captain America

6. Drilling for oil is an incredibly dangerous task. Last year, a Horrifying disaster occurred in one of our competitor's oil rigs. It was caused of several equipment malfunction that resulted in a gas leakage which triggered multiple explosions on the platform. Thankfully no workers were harmed. Therefore, Stark Industries are always checking the condition of each equipment being used on every extraction point in order to be replaced or upgraded if needed.

- **SQL QUERY:**

```
SELECT E.Description, E.`Condition`, EX.Location
FROM EQUIPMENT E, EXTRACT_POINT EX
WHERE E.ExtractID=EX.Id;
```

OUTPUT:

Description	Condition	Location
TeleHandlers	Bad	Cabinda Angola
ForkLift	Out of Use	Congo Basin, Angola
PipeLayer	Bad	Orinoco, Venezuela
Drill Rig	Great	Orinoco, Venezuela

7. An employer should keep track of the products his/her customers are buying in specific locations and in what quantities. In order to suffice the customer's demands and satisfy their needs.

- **SQL QUERY:**

```
SELECT C.Name, P.Name, B.Quantity, (P.Price*B.Quantity) as Total_Price,C.Address
FROM CUSTOMER C, BUYS B, PRODUCT P
where B.CustomerID=C.Id and B.ProductID=P.PId;
```

OUTPUT:

Name	Name	Quantity	Total_Price	Address
Quandale Dingle	Fuel Oil	77	9240	Ohio
Joe Biden	CNG	445	33375	Washington
Medco	Gasoline	55	5500	Tyre
Tony Stark	Ethane	42	3150	New York
Charles Smith	Refinery Gas	634	31700	Australia
Rania Dib	Diesel Oil	53	5300	Beirut
Rania Dib	LPG	55	2200	Beirut
Electricite Du Liban	Refinery Gas	65464	3273200	Beirut
Seven Eleven	Petroleum Oil	365	58400	New Jersey

8. The war between Russia and Ukraine is growing worse by day. Yesterday, the S.H.I.E.L.D. agency provided top secret information to Russian troops that Saint Petersburg port is targeted and may potentially be destroyed by Ukraine and are willing to decimate a vast swath of the city. Thus, the city's residents were requested to evacuate and that all shipments arriving to the port to be canceled or delayed.

- **SQL QUERY:**

```
SELECT C.Name

FROM SHIPMENT  SH, SCHEDULE S, CUSTOMER C, PORT P,ARRIVES A

WHERE S.CustomerID=C.Id and SH.ScheduleID=S.Id and A.PortId=P.Id and
A.ShipmentID=SH.Id and P.Name='Saint Petersburg Port';
```

OUTPUT:

Name:
SEVEN ELEVEN
TONY STARK

9. In Stark Industries, one of our major concerns and importance is our customers' feedback. In addition, we would like to see which one of our employees whom deal with customers is working the hardest and serving a number of different employees. Thus a query is needed that checks which employees serve customers and extract their name, ID, as well as the customer's name they are serving.

- **SQL QUERY:**

```
SELECT E.Id, E.FirstName, E.MiddleName, E.LastName, C.Name

FROM EMPLOYEE  E, SERVES S, CUSTOMER C

WHERE S.EmployeeID=E.Id and S.CustomerID=C.Id
```

OUTPUT:

ID	First_Name	Middle_Name	Last_Name	Name
E-0005	Tala	Amer	Itani	Quandale Dingle
E-0010	Paul	Simon	Walker	Joe Biden
E-0011	Vin	John	Diesel	Medco
E-0005	Tala	Amer	Itani	Tony Strak
E-0010	Paul	Simon	Walker	Charles Smith
E-0011	Vin	John	Diesel	Rania Dib
E-0005	Tala	Amer	Itani	Jonesy
E-0010	Paul	Simon	Walker	Electricite Du Liban
E-0011	Vin	John	Diesel	Seven Eleven

10. In our company, we are performing numerous projects and extracting from many different extract points around the world. In order to do so, we must be equipped with some of the newest most updated equipment and machines in order to facilitate our heavy work. Our equipment must be recently inspected for safe of use but if the last time it was inspected was prior to 2019, we want to demand inspection for this equipment. Thus, a query is performed to collect all the equipment that need inspection.

- **SQL QUERY:**

```
SELECT E.EIN, E.Description, E.Last_Inspect

FROM EQUIPMENT  E

WHERE E.Last_Inspect<2019;
```

OUTPUT:

EIN	Description	Last_Inspect
EQ-001	BullDozer	2018
EQ-002	Asphalt Paver	2017
EQ-003	TeleHandlers	2013
EQ-006	ForkLift	2010
EQ-007	Pile Boring Machine	2018
EQ-008	Concrete Mixer	2017

XIV. Normalization Up to The BCNF Normal Form:

After creating all relations, we should improve them by normalizing according to several normal forms. Here we are going to normalize our database up to the fourth normal form which is the Boyce-Codd Normal Form. On each relation we are going to apply the four normal forms. We start with the first then second then third and at last the BCNF normal form. Let us first start by a general description to each normal form.

First Normal Form:

This form does not allow multivalued attributes, composite attributes, and their combinations to exist in a relation.

1. Only attribute values permitted are single atomic values.
2. Domain of an attribute must only include atomic values and the value of an attribute in a tuple must be a single value from the domain of that attribute.
3. Disallows having a set of values as an attribute value for a single tuple.

Second Normal Form:

The Second normal form is based on the concept of full functional dependency. Before explaining the second form let us define some concepts used in this form and other forms also.

Functional Dependencies: A constraint between two sets of attributes from the database. The values of the Y component of a tuple in relation R depend on or are determined by the values of an X component. We say that Y is functionally dependent on X.

Prime attribute: An attribute that is a member of a candidate key in a relation R. An attribute is called non prime if it is not a prime attribute that is, if it is not a member of any candidate key.

Full functional dependency: A functional dependency X →Y is a full functional dependency if removal of any attribute A from X means that the dependency does not hold anymore.

Partial Dependency: A functional dependency X →Y is a partial functional dependency if removal of any attribute A from X means that the dependency still holds.

A relation schema R is in the second normal form if every nonprime attribute in R is fully functionally dependent on the every key of R and every nonprime attribute A in R is not partially dependent on any key in R.

Third Normal Form:

The third normal form is based on the concept of transitive dependency. So let us first define a transitive dependency.

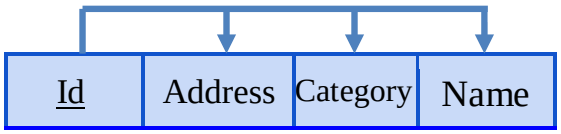
Transitive Dependency: A functional dependency X →Y in a relation schema R is a transitive dependency if there exists a set of attributes Z in R that is neither a candidate key nor a subset of any key of R, and both X →Z and Z →Y hold.

A relation schema R is in the third normal form if it satisfies the second normal form, and no nonprime attribute of R is transitively dependent on the primary key. For every nontrivial functional dependency X →Y either X should be a super key or Y is a prime attribute.

Boycee-Codd Normal Form:

The Boycee-Codd normal form is a stricter form than the third normal form. The BCNF differs from the definition of the third normal form in only one condition. The third normal form allows the right-hand side of the functional dependency to be a prime attribute while BCNF does not allow that.

1. CUSTOMER:



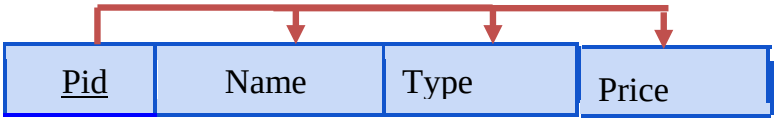
A. The **CUSTOMER** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **CUSTOMER** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **CUSTOMER r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **CUSTOMER** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

2. PRODUCT:



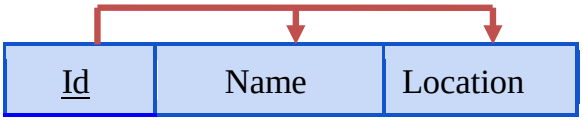
A. The **PRODUCT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **PRODUCT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Pid"**.

C. The **PRODUCT r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"Pid"**.

D. The **PRODUCT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

3. BRANCH:



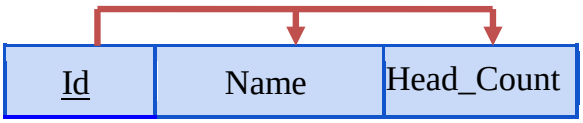
A. The **BRANCH** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **BRANCH** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **BRANCH r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **BRANCH** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

4. DEPARTMENT:



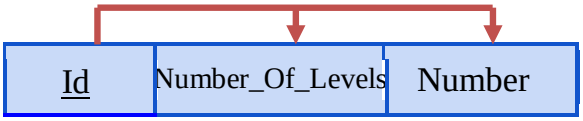
A. The **DEPARTMENT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **DEPARTMENT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **DEPARTMENT r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **DEPARTMENT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

5. BUILDING:



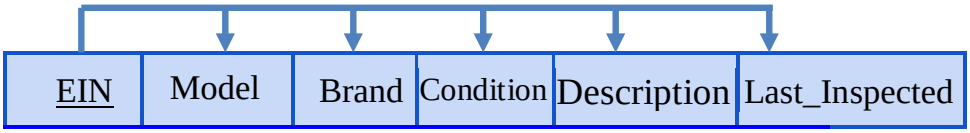
A. The **BUILDING** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **BUILDING** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **BUILDING r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **BUILDING** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

6. EQUIPMENT:



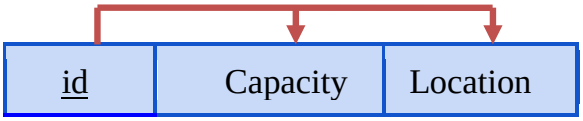
A. The **EQUIPMENT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **EQUIPMENT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"EIN"**.

C. The **EQUIPMENT r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"EIN"**.

D. The **EQUIPMENT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

7. STORAGE:



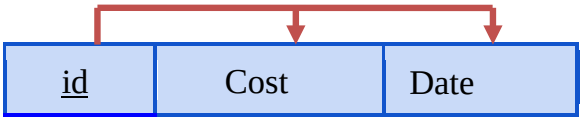
A. The **STORAGE** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **STORAGE** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"id"**.

C. The **STORAGE r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"id"**.

D. The **STORAGE** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

8. SHIPMENT:



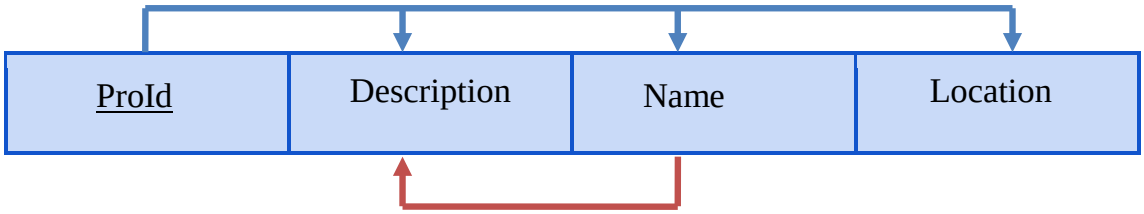
A. The **SHIPMENT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **SHIPMENT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"id"**.

C. The **SHIPMENT r**elation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"id"**.

D. The **SHIPMENT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

9. PROJECT:

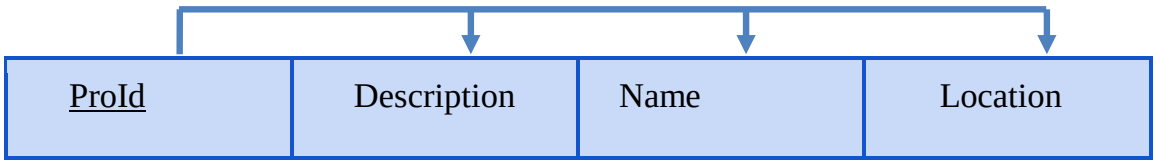


A. The **PROJECT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

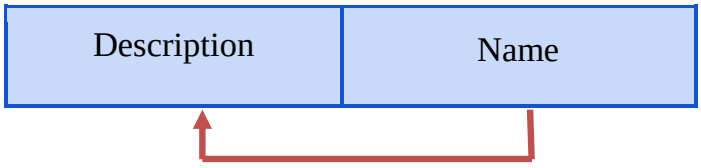
B. The **PROJECT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"ProId"**.

C. The **PROJECT** relation schema does not satisfy all conditions of the 3NF because the functional dependency represented by **Project_Name→Description** is a functional dependency were neither Project_Name is a super key nor Description is a prime attribute. Thus, further decomposition is needed.

PROJECT_A

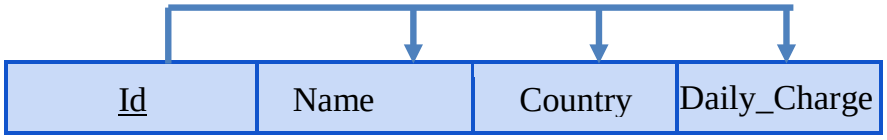


PROJECT_B



D. The **PROJECT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

10. Port:



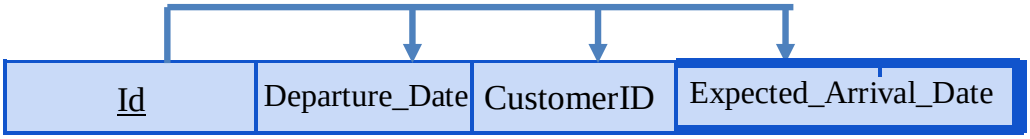
A. The **PORT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **PORT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **PORT** relation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there is no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **PORT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

11. SCHEDULE:



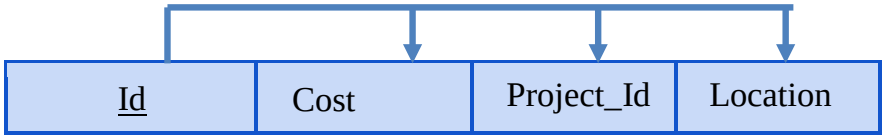
A. The **SCHEDULE** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **SCHEDULE** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **SCHEDULE** relation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there is no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **SCHEDULE** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

12. EXTRACT_POINT:



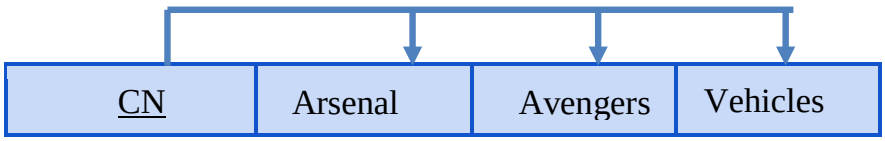
A. The **EXTRACT_POINT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **EXTRACT_POINT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **EXTRACT_POINT** relation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **EXTRACT_POINT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

13. AVENGERS_COMPOUND:



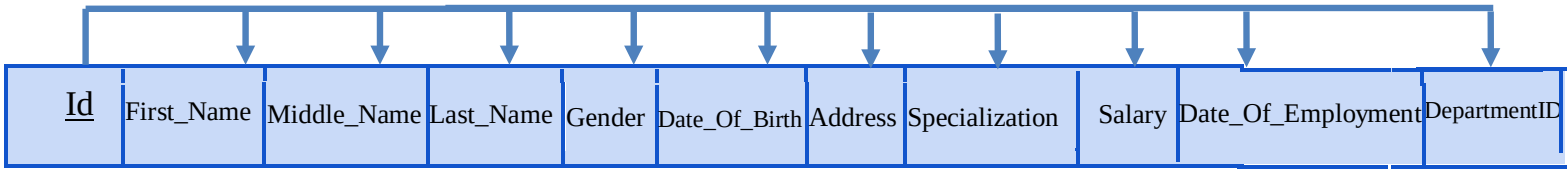
A. The **AVENGERS_COMPOUND** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **AVENGERS_COMPOUND** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"CN"**.

C. The **AVENGERS_COMPOUND** relation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there is no non-prime attributes that are transitively dependent on the primary key **"CN"**.

D. The **AVENGERS_COMPOUND** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

14. EMPLOYEE:



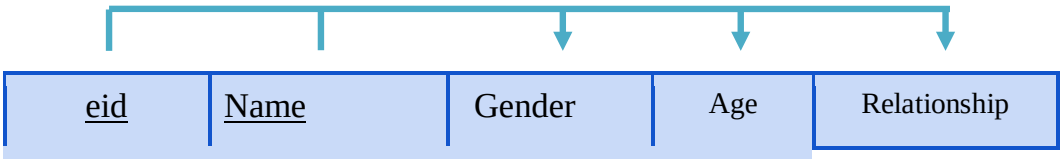
A. The **EMPLOYEE** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **EMPLOYEE** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"Id"**.

C. The **EMPLOYEE** relation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there are no non-prime attributes that are transitively dependent on the primary key **"Id"**.

D. The **EMPLOYEE** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

15. DEPENDENT:



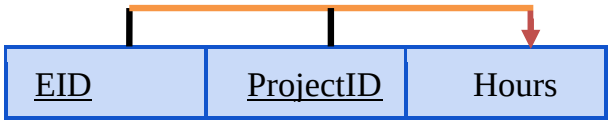
A. The **DEPENDENT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **DEPENDENT** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"eid and Name"**.

C. The **DEPENDENT** relation schema satisfies all conditions of the 3NF because it satisfies the 2NF and there is no non-prime attributes that are transitively dependent on the primary key **"eid and Name"**.

D. The **DEPENDENT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key or A is a prime attribute and X not a super key.

16. WORKS_ON:



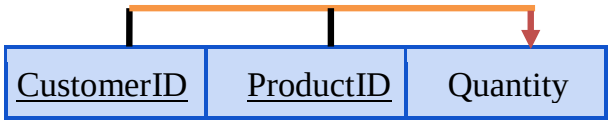
A. In the **WORKS_ON** table there is no multivalued and composite attribute thus 1NF is satisfied.

B. As for the 2NF all attributes depend on the Primary keys "EID" and "ProjectID" and if one primary key is dropped the FD cannot hold.

C. 3NF is satisfied since there is no Transitive Functional Dependency in the entity **WORKS_ON**, no non-prime attribute is transitively dependent on the primary keys "EID" and "ProjectID".

D. BCNF is satisfied since there is no FD X->A such that X is a not a super key.

17. BUYS:



A. In the **BUYS** table there is no multivalued and composite attribute thus 1NF is satisfied.

B. Since 1NF is satisfied and all Non-prime attributes are Fully Functional Dependent on the primary key "CustomerID" and "ProductID" then the 2NF is satisfied.

C. In the **BUYS** table the conditions are satisfied for the 3NF because it satisfies the 2NF and there are no nonprime attributes that are transitively dependent on the primary key "CustomerID" and "ProductID".

D. BCNF is satisfied since there is no FD X->A such that X is a not a super key.

18. Relation Schemas without non-prime attributes:

SERVES:

IS_IN:

DepartmenID	BranchID

ARRIVES:

PortID	EXTRACTID

LOCATES:

EID	ExtractID

IS_LOCATED

DepartmentID	BuildingID

EMPLOYEE_PHONE_NUMBER

EID	Phone_Number

CUSTOMER _PHONE_NUMBER

CustomerID	Phone_Number

CUSTOMER_EMAIL

CustomerID	Email

EMPLOYEE_EMAIL

EID	Email

XV. Conclusion:

A huge firm with hundreds of people, locations, departments, and clients need a database for effective management and operation. A gas company is no different because it receives hundreds of visits each year from clients looking for gas supplies, both commercial and residential. Before the goods arrives at the clients, the customers must go through a process. The first step in the employees' service to the customer is to examine the type and cost of gas. The consumer then pays in the division in charge of client payments and services if they are ready to purchase the product. The customer will then set a time for the anticipated day of the shipment's arrival. Ships will be the means of transportation used to get the goods to the closest port if shipping by water is an option and the distance is great. High-quality management and diligent workers are required for this process. Considering giving the business with modern equipment. Errors shouldn't be made in the workplace since they could produce an explosion that kills thousands of people. The business must also ensure that its own infrastructure and people resources are well-maintained and structured; additionally, its equipment must be cutting-edge and delicately functional. There must be no scheduling difficulties and the employees must be well-paid. Because of this, it's important to closely monitor a gas company using a database system to ensure that all daily operations, such as following safety standards and doing equipment maintenance, are performed smoothly.

XVI. Instructor's feedback and evaluation

This page should be used by our instructor DR. RAMZI HARATY to write any comments, feedback, and suggestions for our database system. This page will be used to improve our final report.