Lebanese American University

CSC375

NKIY - Phase IV

A Database Design for GoGreen Plant Nursery

The Team:

Dana Nasser

Wafic Khalife

Miguel Ibrahim

Charbel Abi Younes

December 1, 2021

Contents

A-Phase IV Updates:

- Normalize the database up to the BCNF Normal Form.
- Explain the normalization process.

B- Copyright Notice:

wafic.khalife@lau.edu,
charbel.abiyounes02@lau.edu,
dana.nasser01@lau.edu ,
miguel.ibrahim@lau.edu.

C- Terms and Conditions:

-You vow to keep this information safe and secure and not allow any leakage.

-You vow not to copy this information and use it with your database design or any database design.

-You vow not to destroy, incinerate, tear, and even feed it to your dog.

-You may use this report for the use of public good, but never for self-benefit, and even in the public good (government), you must keep aware of the people that you are dealing with.

-The information disclosed to you is for Dr. Ramzi Haraty alone (maybe viewed with family members and pets alone at home but not in public)

Name: Signature

____ _____

D-Background Info:

NKIY or, in short, Nasser Khalifeh Ibrahim Younes database management systems to data manage several plant nurseries locally. The idea or goal of our team is to help plant nurseries in repopulating our planet Earth with rare and threatened to be extinct species of plants, flowers, and much more.

In our report, we summarize what we have done during our work process. We were assigned to create a database design for GoGreen nursery. Overall, our work experience has been constructive and useful for our careers. This report will explore the different aspects of what we have done when designing the database for this local nursery, as they were preparing to expand.

In short, our goal is to help this local nursery achieve its goal by designing database systems with the idea of keeping things simple for GoGreen's employees, departments, and customers. We would cordially thank GoGreen nursery for giving us the chance to be part of this amazing project.

E- System Description and Constraints:

The importance of plants to humans and just about all other life on Earth is staggering. Plants bring life to not only our company but the earth as well, so it is vital to keep repopulating planet Earth with plants as we head to a more technological and possibly environmental threatening future. Protecting and taking care of plants in recent times should be a priority for several organizations and not only our company. GoGreen plant nursery is continuously expanding and repopulating our planet with trees and plants that were once lost due to natural and unnatural processes while providing customers with a wide spectrum of plants from fruitless to trees.

GoGreen's staff consists of hundreds of botanists and agricultures, who make sure that plants are taken care of, protected, and raised to a particular age. In addition, there exist office employees that ensure everything is organized and well equipped for the plants, farmers, and customers. Also, all employees in this nursery are provided with schedules that will help prevent time conflicts and reduce time management errors.

- Manager of the Agricultural Department
- CEO
- Intern (pat and pending)
- Greenhouse supervisor
- Head of Customer Service
- Botanist
- Farmer

The primary function of this nursery is to assist plants and grow them until they reach a particular age, and then they are sold at GoGreen's branches in pots to be later grown in fields or remained in pots. This nursery must be made up of multiple departments to ensure that the previous process is accomplished smoothly. Each Department has its responsibility to manage the various aspects of the plant nursery such as employees, greenhouse, vehicle, branch, and many more and may as

- Agricultural department
- Customer Service/Support
- Branch Manager
- HR
- Transportation Management

GoGreen nursery is nothing without its greenhouses. Greenhouses are distributed in several regions and come in different types such as freestanding, attached as well as different sizes of land. In these greenhouses, we have storage that store nutrient and equipment required for the growth of the plants.

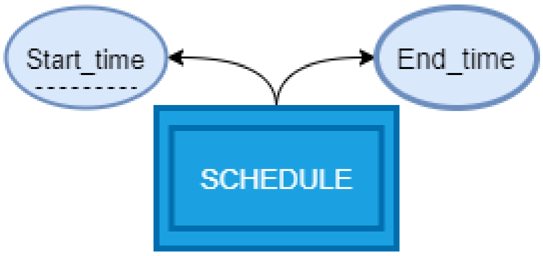
The normal course of events in the nursery revolves around plants. Plants can either be provided by suppliers or directly grown in one of the greenhouses. Once a plant enters this nursery, it is assigned a code to uniquely identify her. After that, the plant will be in a particular greenhouse where it will be taken care of using a certain dosage of fertilizers, pesticides, and nutrients. Once it reaches a specified age, this plant will be moved using one of our vehicles or trucks to the nearest branch, where it will be sold to either local farmers or citizens.

A plant might need a different amount of nutrients, and some don't need any nutrients at all; such plants require as well as different dosages of the nutrient as follows:
-Rosemary needs a 50ng (nano-grams) of Nitrogen-Magnesium fertilizers.

GoGreen also takes into consideration the client's point of view. It has built branches in several regions to reduce customers' traveling expenses in these difficult times. Customers can also contact the Customer Support department in case of any problem during their purchases and can also contact the agricultural department for tips or guides on how to take care of their plans.

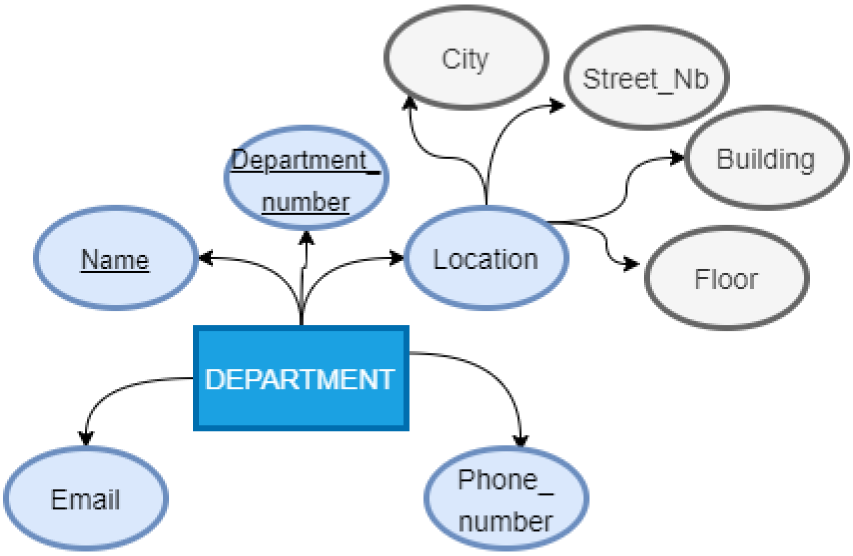
F- Entity Types and Attributes:

1- Schedule:



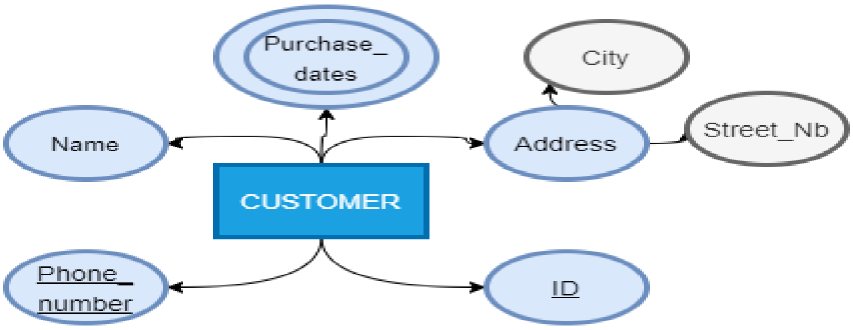
The SCHEDULE is an entity type mainly for the employees, and it is the weak entity type of this database. It has the start and end time of their shift. All employees have their independent SCHEDULE.

2- Department:



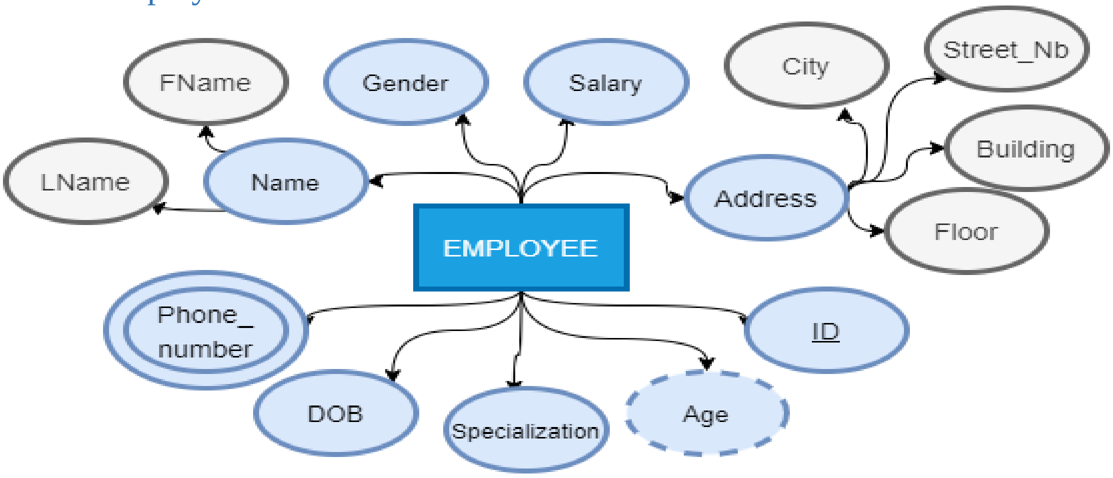
Departments can work more like well-oiled machines, focusing time and energy on productive tasks. Thus, an entity type DEPARTMENT was needed to increase productivity and avoid chaos in the company. This entity type is uniquely determined by a number of 2 digits and by its name. For example, we can have Sales, Agriculture, HR departments. The DEPARTMENT has a location which is composed of city, street number, building, floor number. It also has an email, phone number to promote communications between departments and other entities. For instance, employees can easily contact departments in case they face a certain problem.

3- Customer:



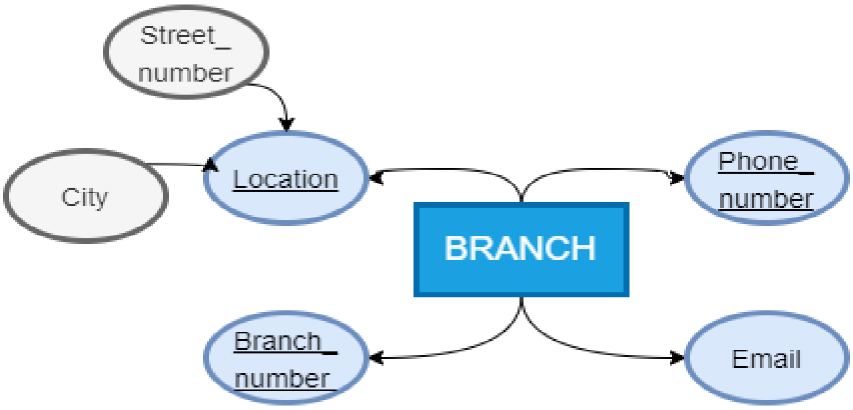
Customers are the people that purchase plants from the branches of the nursery. After purchasing an item, the CUSTOMER gets a unique ID of 8 digits which is considered the key attribute of this entity type. Naturally, every CUSTOMER has a name. This entity type takes the customer's phone number and address (composite attribute) as attributes to contact or pass by the customer if any problem occurs. Also, the purchase dates will be recorded to help the branches track the transactions that occurred between the branch and customers.

4- Employee:



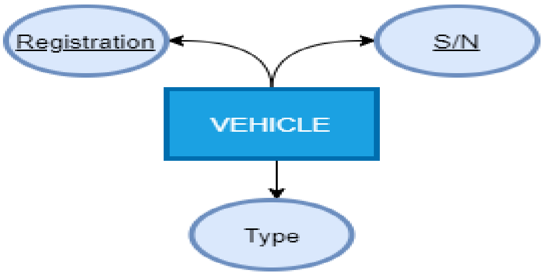
Employees aren't just workers. They're the backbone of this business. They are responsible for taking care of our plants, greenhouses, branches, and even departments. Hence, creating an EMPLOYEE entity type is a necessity for this database. This entity type records the employee's personal information. It keeps track of the name, which is composed of first and last name, gender, date of birth, a derived attribute age, phone number which may have several values (home, mobile), and an address composed of the city, street name, building and floor. Each EMPLOYEE has a unique ID formed of 10 digits to uniquely identify each employee, and a specialization in the company, for example, gardener, plant biologist, or simply a driver. Finally, employees receive monthly salaries to reward them for their hard work.

5- Branch:



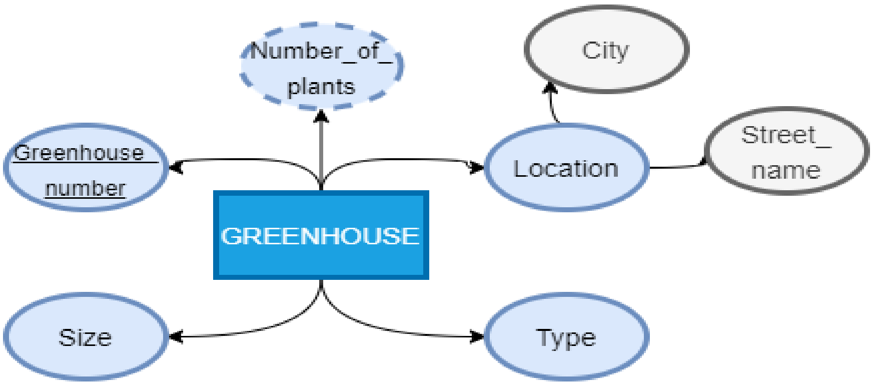
The BRANCH is a shop where all the plants are sold at. Every BRANCH has its unique number formed of 2 digits. It is also uniquely determined by its location (composed of city and street numbers) since we only have one branch in a particular area. All BRANCHs also have a phone number and email to be easily contacted with.

6- Vehicle:



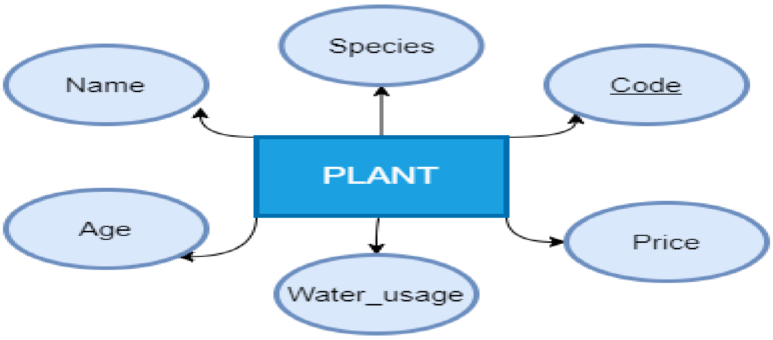
The VEHICLE is used to transport plants from different locations or to transport equipment. The key is the registration number. The registration number will allow the departments to track the VEHICLE during transportation. Another key attribute is the serial number. We also store in our database the type of VEHICLE. The type will allow us to track the vehicles even faster, so we have fewer vehicles to sort and filter through. This will also allow us to know what vehicles to use according to the items transported. For example, if the plants are large, we would require large trucks.

7- Greenhouse:



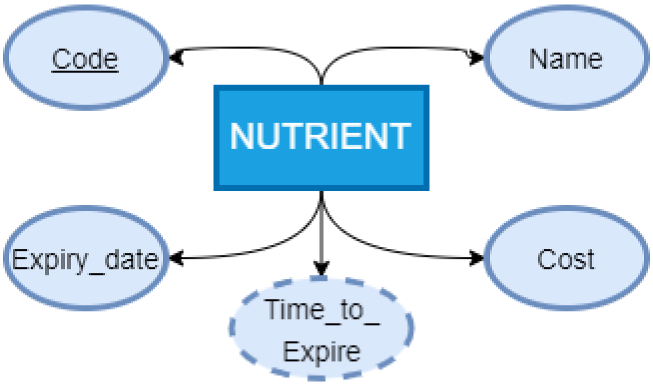
A GREENHOUSE is a place in which plants are taken care of. Each GREENHOUSE has its very own number made of 2 digits; therefore, it represents the key attribute of this entity type. Greenhouses come with different types such as freestanding, attached as well as different sizes of land. This entity type keeps track of the number of plants available in the GREENHOUSE, which is a derived attribute, along with its location, which consists of the city and street name.

8- Plant:



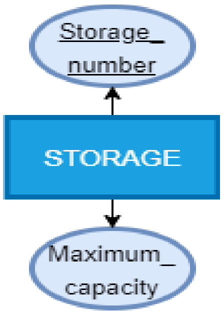
Plants are the main focus of this nursery. They have different species and names. Once stored in our nursery, a unique code, composed of 4 digits, is assigned to each PLANT to differentiate them. This entity type keeps track of the age of the PLANT and its water usage to inform botanists and even customers about the amount of water needed. We also keep the price of the PLANT if we want to sell them at one of our branches.

9- Nutrient:



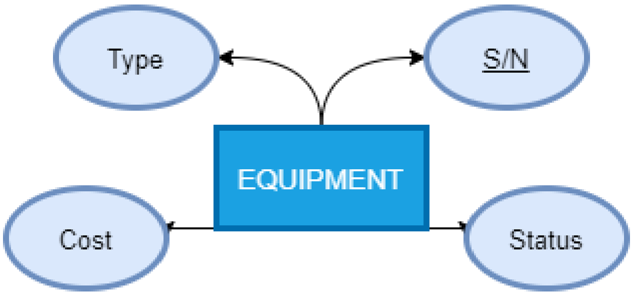
Nutrients are essential in this nursery since plants can't grow effectively without them. Each NUTRIENT has a name, and once stored in our nursery, a unique code, formed of 4 digits, is assigned to keep track of them. This entity type also has an expiry date and a derived attribute time to expire that informs the agriculture department and botanists about the products that are close to being expired. We also store the cost of nutrients to keep track of financial records and to be able to approximate the market price of those products.

10-Storage:



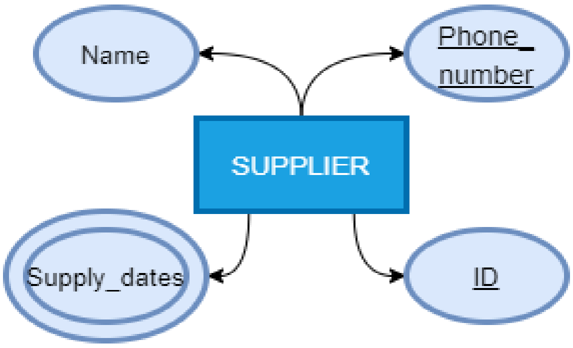
The STORAGE entity type represents the place where equipment and nutrients are stored. It is characterized by a storage number, made of 2 digits, as well as a maximum capacity attribute.

11-Equipment:



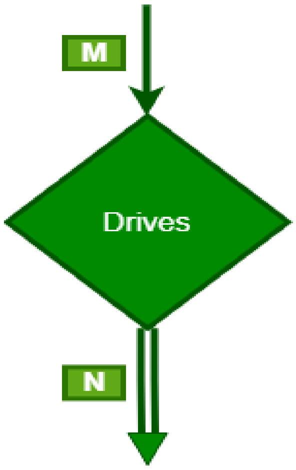
Equipment is necessary to the plant nursery. They are stored in the storage rooms and used in the greenhouse when needed. The primary key of the EQUIPMENT in our database is the serial number. Each kind of EQUIPMENT has a serial number that we can store and look at when needed. Using the serial number, the employees will know more detail about the tools needed. Other keys include the type of equipment, status, cost, and quantity. The type of equipment will allow the employees to categorize the EQUIPMENT properly and organize them according to the similarity in the storage rooms. The status in our database will let the employees know if the EQUIPMENT is being used, available for use, or even under maintenance.

12-Supplier:

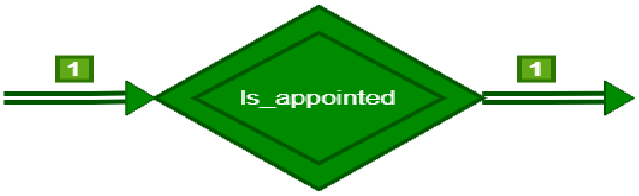


Suppliers are the ones who provide us with equipment, resources, and even plants. They supply the plant nursery with the essential material to keep working. The primary key is the supplier's ID. The ID of the supplier is a unique number of 9 digits given to each SUPPLIER and lets the employees know which SUPPLIER supplied what. Our database also keeps track of the suppliers' names. The name will allow the employees to track the whole process and the transactions. This entity type also includes attributes such as phone numbers and supply dates. The phone number is another key attribute as it allows the employees to contact the SUPPLIER if anything goes wrong, and supply dates are stored to know when exactly these transactions occurred.

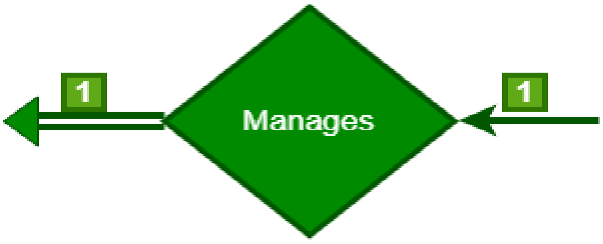
G- Relationships



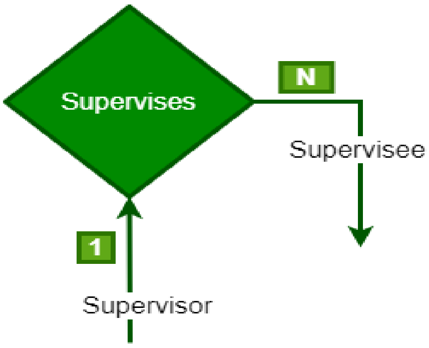
Vehicles are used to transport plants in this nursery. Therefore, a relationship called "Drives" between the EMPLOYEE and VEHICLE entity type has been made. All vehicles are driven by employees, but not all employees drive vehicles. Thus, the participation is total from one side, which is the vehicle's side, and is partial from the other side. Also, many employees drive many vehicles, which explains the many-to-many relationship.



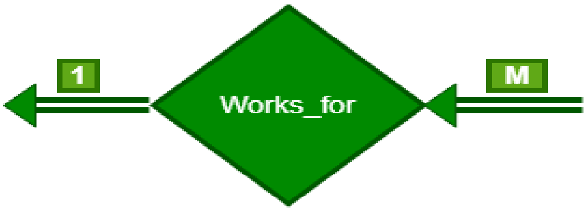
Every employee has one schedule assigned to him. Thus, an identifying relationship “Is_appointed” has to be made between the weak entity type SCHEDULE and EMPLOYEE entity type. The participation is total on both sides since all employees have schedules and all schedules are assigned to employees. Moreover, each employee has one schedule assigned to him, and a schedule is given to only one employee. Therefore, this relationship is one-to-one.



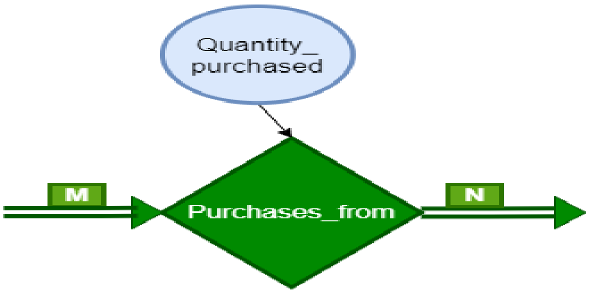
Due to their importance, departments need to be supervised and managed by the responsible employee. Thus, a "Manages" relationship has to occur between the DEPARTMENT and EMPLOYEE entity types. One Employee can manage one department and a department can have one manager. Therefore, it is one to one relationship. Moreover, every department needs to be managed but not all employees supervise departments; thus, there is total participation only from the department side.



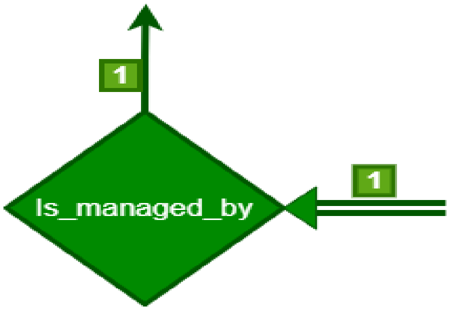
A supervisor is needed to supervise the work of other employees. This explains why we have a self-referencing relationship “Supervises” on EMPLOYEE. One employee may be the supervisor of many other employees; hence, this is a one-to-many relationship. In addition, not every employee is supervised, and not every employee is a supervisor; hence, we have partial participation of the two entity types in this relationship.



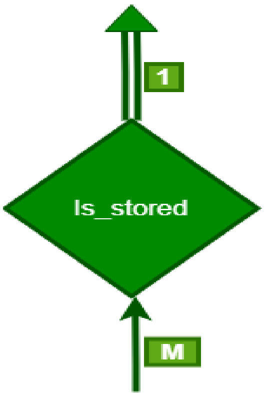
The division of employees into departments allows them to specialize. Hence, a relationship “Works_for” is needed to connect EMPLOYEE to DEPARTMENT. Many employees can work in a single department; thus, we have a many-to-one relationship between the two entity types. Since every employee belongs in a department and every department contains employees, we have total participation of both entity types in this relationship.



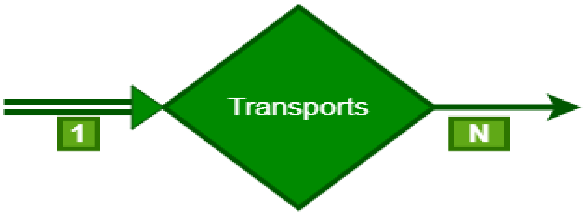
Customers can purchase well-treated plants from the branches. Thus, a "Purchases_from" relationship has to occur between the CUSTOMER and BRANCH Entities. All customers buy from branches and all branches sell to customers. Therefore, there is total participation on both sides of this entity type. Moreover, many customers buy from many branches. Thus, a many-to-many relationship is presented. This relationship has an attribute called quantity purchased which purpose is to specify the number of items that the customer has bought from the branch.



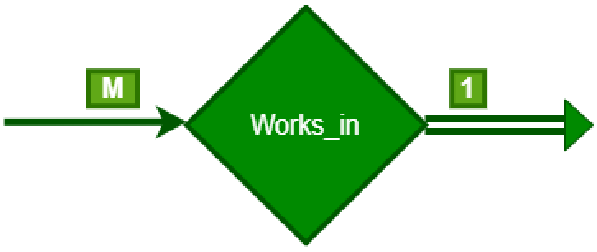
A manager is an essential component of a branch. This explains why we need a relationship “Is_managed_by” between BRANCH and EMPLOYEE. Since all branches have a manager but not all employees are branch managers, we only total participation of BRANCH in this relationship. Also, a branch has one manager, and an employee can manage at most one branch hence, we have a one-to-one relationship.



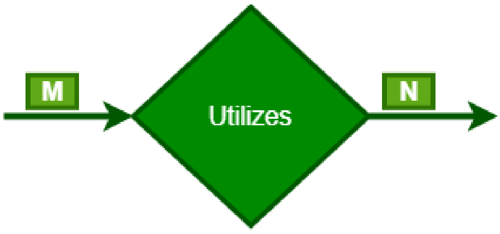
Many plants can be stored in branches, hence, creating a many to one relationship “Is_Stored” between PLANT and BRANCH was a necessity. We have total participation of BRANCH since all branches store plants. However, not all plants are stored in branches since they might be at a greenhouse, hence, we partial participation of PLANT in this relationship.



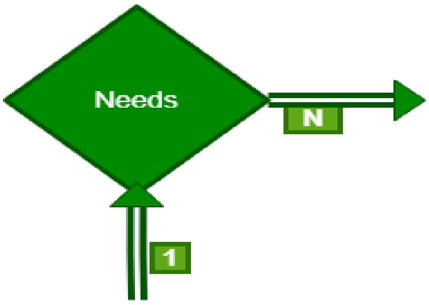
Each vehicle is used to transport many plants in our nursery. Thus, the one-to-many relationship "Transports" must be created between VEHICLE and PLANT entity types. We have total participation of vehicles in this relationship because all vehicles in our nursery are used to transport plants only. However, not all plants are being transported since some remain stored. Hence, plants participate partially in this relationship.



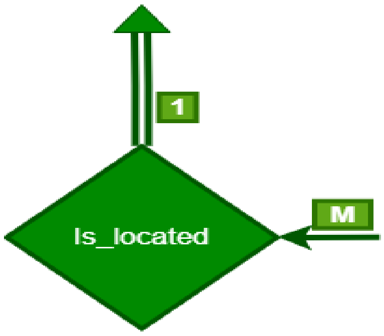
Many employees can work in a greenhouse. Hence, the many to one relationship "Works_in" between EMPLOYEE and GREENHOUSE was needed. Not all employees work in a greenhouse, but all greenhouses have employees working in them. This justifies why we only have total participation of GREENHOUSE in this relationship.



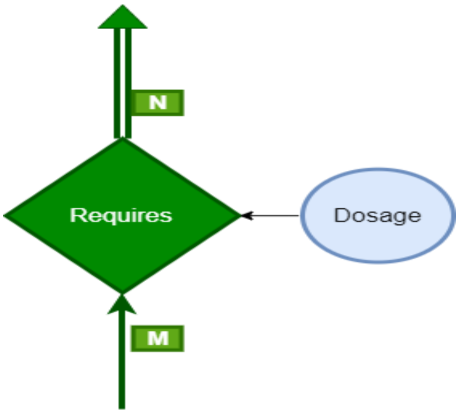
Equipment is being utilized by the employees working in greenhouses. Thus, a “Utilizes” relationship between the entity types EMPLOYEE and EQUIPMENT is needed. Not all equipment is used by employees since some are left to work alone like automatic water drills, also not all employees use equipment. Moreover, several types of equipment can be utilized by many employees: therefore, a many-to-many relationship.



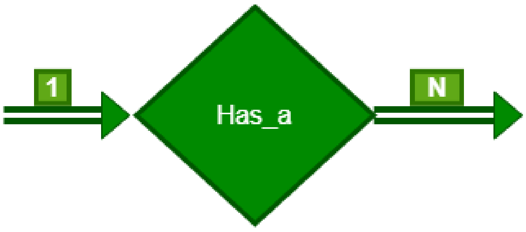
A greenhouse needs to be well equipped with several types of equipment like water drills and other machines to store and help plants grow. Thus, a one-to-many relationship “Needs” was created to connect GREENHOUSE and EQUIPMENT. All greenhouses have different types of equipment to take care of plants, and all equipment is used in a greenhouse, thus we have total participation on both sides.



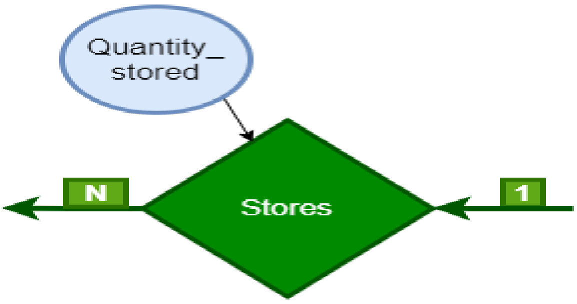
Many plants might be located in a greenhouse where they are taken care of. This justifies the need for a many-to-one relationship “Is_located” between PLANT and GREENHOUSE. Like we previously said plants can be stored in branches hence, not all plants are located in a greenhouse. However, all greenhouses contain plants, hence, we have a partial participation from PLANT, but total participation of GREENHOUSE is this relationship.



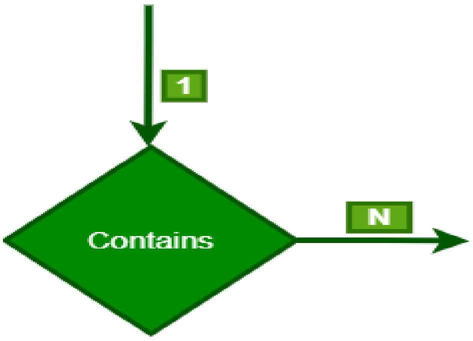
A plant may require nutrients, and nutrients can be given to many different plants; thus, the many-to-many relationship "Requires" between PLANT and NUTRIENT was needed. Plants don't totally participate in this relationship since not all plants require nutrients. However, all nutrients in our nursery are used on plants. Hence, we have total participation of nutrients in this relationship. This relationship has an attribute called dosage which purpose is to specify the quantity of a nutrient required by the plant.



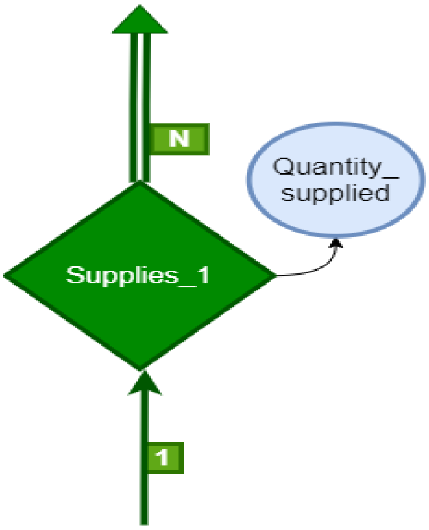
A greenhouse needs to have one or many storages. Therefore, a "has_a" relationship occurs between the STORAGE and GREENHOUSE entity type. Each greenhouse individually has many storages; therefore, a one-to-many relationship occurs between the two. Moreover, there is total participation on both sides because all greenhouses have storage, and all stores are located in greenhouses.



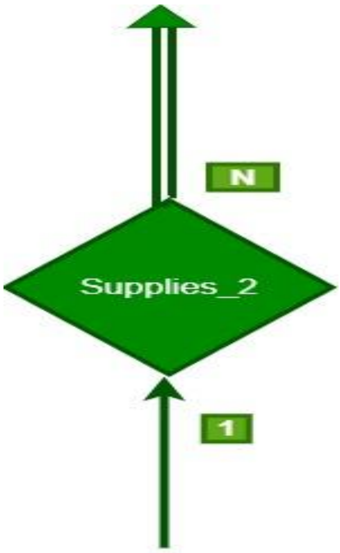
Storage is the place where nutrients might be stored. Thus a "Stores" relationship was needed between STORAGE and NUTRIENT. Storages store many nutrients, but not all of them are stored and not all storages contain nutrients; therefore, total participation is not occurring on both sides. One storage can store many nutrients; thus, a one-to-many relationship occurs between the two. This relationship has an attribute called quantity stored which purpose is to specify the quantity of a nutrient stored in each storage.



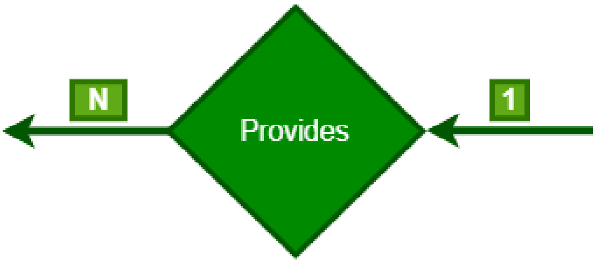
Storages might also contain equipment. Thus, a “contains” relationship is needed between the STORAGE and EQUIPMENT entity types. storages contain several types of equipment, but not all of them are stored and not all storages contain equipment; therefore, total participation is not occurring on both sides. One storage can store several pieces of equipment; thus, a one-to-many relationship occurs between the two.



Suppliers supply the plant nursery with nutrients. Thus, a relationship named "Supplies_1" must be made between the NUTRIENT and SUPPLIER entity types. Many nutrients can be supplied by suppliers. Therefore, many-to-many relationship occur between the two. Furthermore, all nutrients are supplied by a supplier, and not all suppliers supply nutrients; thus, a total participation occurs from one side. This relationship has an attribute called quantity supplied which is the number of nutrients supplied by the supplier.

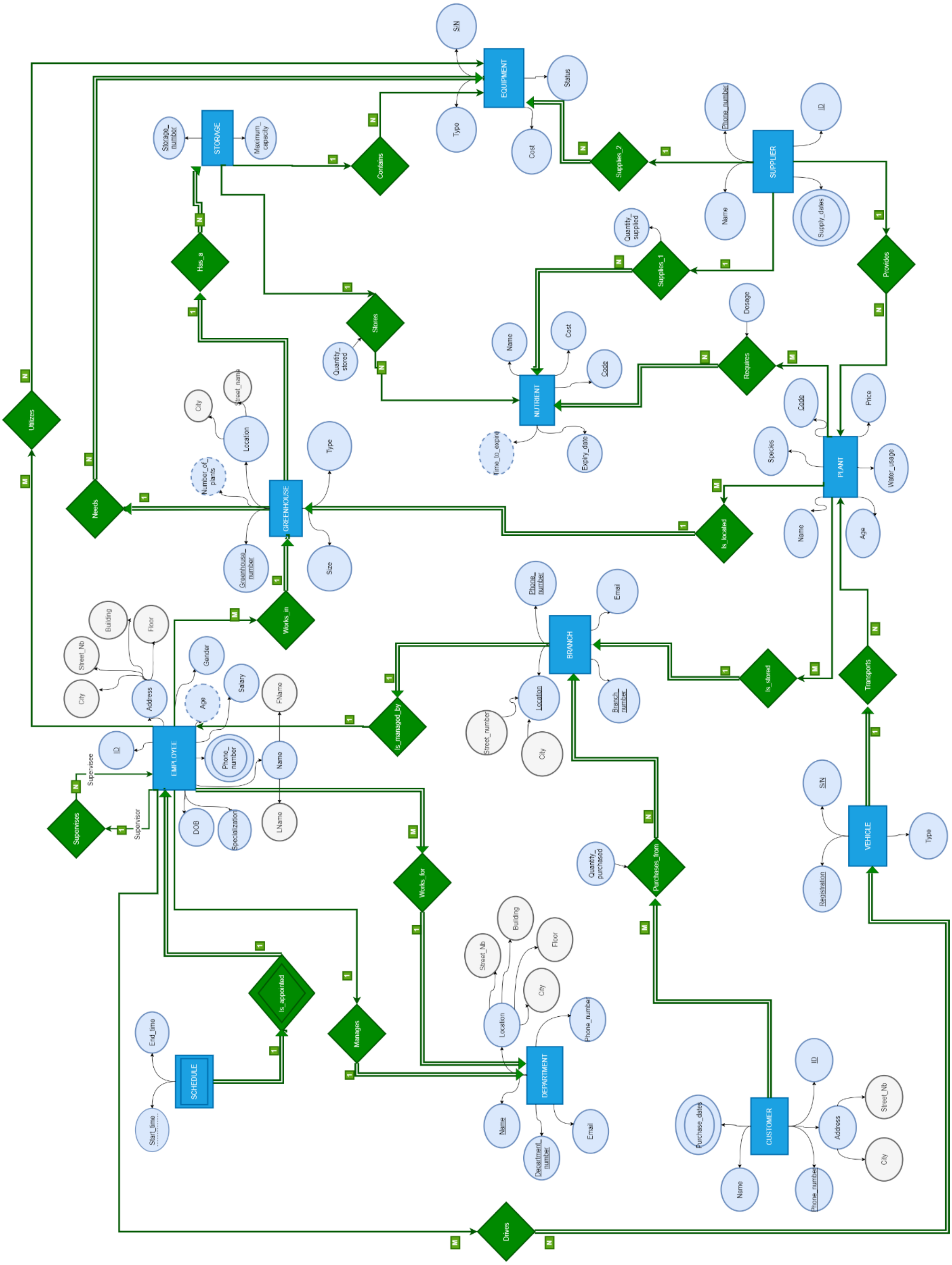


Suppliers also supply the plant nursery with equipment. Thus, a relationship named "Supplies_2" must be made between the EQUIPMENT and SUPPLIER entity types. Several types of equipment can be supplied by suppliers. Therefore, many-to-many relationship occur between the two. Furthermore, a piece of equipment is supplied by a supplier, and not all suppliers supply equipment; thus, a total participation occurs from one side.



The supplier provides seeds of plants. Therefore, a relationship called “provides” is needed between the PLANT and SUPPLIER entity types. this relationship is considered one too many because one supplier provides many plants. And there is no total participation in this relationship because not all suppliers provide plants and not all the plants are provided by a supplier.

H- ER Diagram:


EQUIPMENT
S/N
Status
Type
Cost
STORAGE
Storage_number
Maximum_capacity
Contains
Supplies_2
SUPPLIER
Phone_number
ID
Name
Supply_dates
Quantity_supplied
Supplies_1
Provides
Has_a
Stores
Quantity_stored
NUTRIENT
Name
Cost
Code
Time_to_expire
Expiry_date
Requires
Dosage
PLANT
Code
Price
Species
Water_usage
Name
Age
Utilizes
Needs
GREENHOUSE
City
Street_name
Location
Number_of_plants
Type
Greenhouse_number
Size
Is_located
Works_in
BRANCH
Phone_number
Email
Location
Branch_number
Street_number
City
Is_managed_by
Is_stored
Transports
EMPLOYEE
Building
Floor
Street_Nb
City
Address
Gender
Salary
Age
FName
ID
Phone_number
Name
DOB
Specialization
LName
Supervisee
Supervisor
Supervises
Works_for
Purchases_from
Quantity_purchased
VEHICLE
S/N
Registration
Type
Is_appointed
Manages
DEPARTMENT
Building
Floor
Street_Nb
City
Location
Phone_number
Name
Department_number
Email
SCHEDULE
End_time
Start_time
CUSTOMER
ID
Street_Nb
Address
City
Purchase_dates
Name
Phone_number
Drives

I - ER to Relational Mapping Algorithms:

After designing the abstract and simple mapping of the data structure to an ER diagram, the next step is to transform them into relational database design which requires a seven-step procedure. 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 relation that includes all of its simple attributes and a single primary key that is underlined. The regular (strong) entities in this database design are DEPARTMENT, CUSTOMER, EMPLOYEE, BRANCH, VEHICLE, GREENHOUSE, PLANT, NUTRIENT, STORAGE, EQUIPMENT, and SUPPLIER.

1- DEPARTMENT:

Department-number	Name	Email	Phone_ number	City	Street_ Nb	Building
Floor						

The DEPARTMENT entity type contains simple, and composite attributes in addition to a primary key which is underlined, for this case we will be using *'Department_number'* as our main primary key. The composite attribute location that has *'Street_Nb'*, *'Building'*, *'Floor'*, and *'City'* are included in the relation.

2- CUSTOMER:

ID	Phone_number	Name	City	Street_Nb

This entity type CUSTOMER contains the primary key *‘ID’*, simple attributes *‘Name’* and *‘Phone_Number’* (which was originally a candidate key), and a composite attribute address that has *‘City’* and *‘Street_Nb’*. This entity type also has a multi-valued attribute that won’t be added here due to it being part of another step.
Something we didn’t mention here but is also vital is that the customer is always right.

3- EMPLOYEE:

ID	F_Name	L_Name	Gender	Salary	DOB	Age
Specialization	City	Street_Nb	Building	Floor		

The biggest and most confusing entity type is the EMPLOYEE. It contains simple, multi-valued, derived composite and primary attributes of which: *‘Phone_Number’* is the multi-valued (not included here since it's not part of step 1), *‘Age’* that is derived, Name that is composite of *‘L_Name’*, *‘F_Name’*, Address is also composite of *‘City’*, *‘Street_Nb’*, *‘Building’*, *‘Floor’*, and *‘ID’* being the main primary key.
Note: Age = Date now – Year of Birth.

4- BRANCH:

Branch_number	City	Street_number	Email	Phone_number

In this entity type, there exists a primary key *'Branch_Number'*, simple keys: *'Email'*, and *'Phone_number'*. The composite attribute location has been reduced to both *'City'* and *'Street_number'* for this step.

5- VEHICLE:

S/N	Registration	Type

Like we previously said this entity type is used to move plants from the greenhouse to the branch. In this entity type, we have *'S/N'*, which is a primary key, *'Registration'* and *'Type'* as simple key attributes.

6- GREENHOUSE:

Greenhouse-number	City	Street_name	Type	Size	Number_of_plants

In this entity type we have a primary key *'Greenhouse_number'*, derived key attribute *'Number_of_plants'**, a composite attribute location that has been sub-divided into both *'City'* and *'Street_name'*, finally, we have simple attributes *'Size'* and *'Type'*.
Note: The number of plants depends on the size of the plant and how well we can efficiently organize the space to fit the plants.

7- PLANT:

Code	Species	Name	Price	Age	Water_usage

No plant nursery is a plant nursery without its plants. In this entity type, plants have different attributes. The primary attribute *'Code'* and the simple attributes such as its *'Name'*, *'Age'*, '*Species*', '*Water_usage'* and *'Price'*.

8- NUTRIENT:

Code	Name	Cost	Expiry_date	Time_to_Expire

This entity type consists of its primary key *'Code'*, the simple keys *'Name'*, *'Cost'* and *'Expiry_date'*, and it consists of a derived attribute *'Time_to_expire'**.
*: To calculate the time to expire, we run the expiry date with the current date of time, if Expiry date ex: 10/23/21 and today is 10/23/21 then it is expired --> 0.

9- STORAGE:

Storage_number	Maximum_capacity

It is the smallest entity type in this company. The storage consists of two attributes, one primary attribute *'Storage_number'*, and one simple attribute *'Maximum_capacity'*.

10- EQUIPMENT:

S/N	Type	Cost	Status

The equipment helps increase the harvest while keeping things efficient. It is given a primary key attribute called ‘*S/N* ‘(serial number in short), and simple attributes ‘*Type*’, ‘*Quantity*’, ‘*Status*’, and ‘*Cost*’.

11- SUPPLIER:

ID	Name	Phone_number

Finally, we have reached the end of Step 1, the final entity type to be translated is SUPPLIER. This entity type has a multi-valued attribute ‘*Supply_dates’* that will be mentioned in step 6 of the translation, a simple attribute *‘Phone_Number’*, and *‘Name’*, and a key attribute *‘ID’*.

STEP 2: Mapping of Weak Entity Types

In this step, the weak entity types are mapped into relations. As in Step 1, only the simple attributes are included in the relations. Furthermore, weak entity type 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 represents the primary key of the relation. The weak entity type in our database design is SCHEDULE.

1- SCHEDULE:

Emp-ID	Start-time	End-time

The weak entity type SCHEDULE has three simple attributes which are *'Start_time'*, *'End_time'*, and *'Hours_worked'*. It does not have a multi-valued nor a derived attribute. Moreover, the primary key of the owner entity type *'Emp-ID'* is included. *'Emp-ID'* and the partial key *'Date'* 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. To accomplish our goal, we can follow one of three approaches. The first approach, called the foreign key approach is where we choose the entity type on the total participation side of the relationship, then we add as a foreign key the primary key of the other entity type participating in this relationship. The second approach, called the 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. 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. The binary one-to-one relationships that need to be mapped are: Is_appointed, Manages, and Is_managed_by. In Is_appointed we have total participation on both sides; hence, we will use the merged approach to map it. For the others, we are going to follow the foreign key approach because it is the most useful in our case.

1- DEPARTMENT (Manages):

Department-number	Name	Email	Phone_number	City	Street_Nb	Building	Floor	EMP-ID

Each employee manages different departments. The "MANAGES" is a relationship between the EMPLOYEE entity type and the DEPARTMENT entity type. Here, we don't have full participation between the entity types. Thus, the approach followed here is the foreign key approach. The full participation in the department entity type, which explains why we used the relation DEPARTMENT. In this case, we added, as a foreign key, the primary key ID of the employee entity type and named it *'EMP-ID'*.

2- BRANCH (Is_managed_by):

Branch_number	City	Street_number	Email	Phone_number	EMP-ID

Each branch is managed by different employees. The "IS_MANAGED_BY" is a relationship between the EMPLOYEE entity type and the BRANCH entity type. Here, we don't have full participation between the entity types. Thus, the approach followed here is the foreign key approach. The full participation is the BRANCH entity type, which explains why we used the relation BRANCH. In this case, we added, as a foreign key, the primary key ID of the employee entity type and named it *'EMP-ID'*.

STEP 4: Mapping of Binary 1:N 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: Supervises, Works_for, Is_stored, Transports, Works_in, Needs, Is_located, Has_a, Stores, Contains, Supplies_1, Supplies_2, and Provides.

1- EMPLOYEE (Supervises):

ID	F_Name	L_Name	Gender	Salary	DOB	Age
Specialization	City	Street_Nb	Building	Floor	Supervisor _ID	

An employee can oversee many other employees. The “Supervises” is a self-referencing relationship on the EMPLOYEE entity type. Therefore, we use the relation EMPLOYEE. Furthermore, we add to the relation the foreign key ID which is the primary key of EMPLOYEE entity type and we name it ‘*Superviser_ID’*.

2- EMPLOYEE (Works_for):

ID	F_Name	L_Name	Gender	Salary	DOB	Age
Specialization	City	Street_Nb	Building	Floor	Supervisor _ID	DPT_Nb

Many employees work for a particular department. The "Works_for" is a relationship between the EMPLOYEE entity type and the DEPARTMENT entity type. In this case, the "many" side is the EMPLOYEE. Therefore, we use the previously modified relation "EMPLOYEE".

Furthermore, we add to the relation the foreign key number which is the primary key of DEPARTMENT entity type, and name it *'DPT_Nb'*.

3- PLANT (Is_stored):

Code	Species	Name	Price	Age	Water_usage	BRANCH_Nb

Many plants are stored in one branch. The "IS_STORED" is a relationship between the PLANT entity type and the BRANCH entity type. In this case, the "many" side is the PLANT entity type. Therefore, we use the relation "PLANT". Furthermore, we add to the relation the foreign key number which is the primary key of the BRANCH entity type, and name it *'BRANCH_NB'*.

4- PLANT (Transports):

Code	Species	Name	Price	Age	Water_usage	BRANCH_Nb
VHC-S/N						

One vehicle transports many plants. The "TRANSPORTS" is a relationship between the PLANT entity type and the VEHICLE entity type. In this case, the "many" side is the PLANT entity type. Therefore, we use the previously modified relation "PLANT". Furthermore, we add to the relation the foreign key serial number which is the primary key of the VEHICLE entity type, and name it *'VHC_S/N'*.

5- EMPLOYEE (Works_in):

ID	F_Name	L_Name	Gender	Salary	DOB	Age
Specialization	City	Street_Nb	Building	Floor	Supervisor _ID	DPT_Nb
GRN_Nb						

Some employees might work in greenhouses like botanists and agricultures. Hence, the "Works_in" is a relationship between the EMPLOYEE and GREENHOUSE entity types. In this case, the EMPLOYEE is the "many" side. Therefore, we use the previously modified relation "EMPLOYEE". Furthermore, we add to the relation the foreign key number which is the primary key of GREENHOUSE and name it *'GRN_Nb'*.

6- EQUIPMENT (Needs):

S/N	Type	Cost	Status	GRN_Nb

One greenhouse needs several types of equipment. The "NEED" is a relationship between the GREENHOUSE entity type and the EQUIPMENT entity type. In this case, the "many" side is the EQUIPMENT entity type. Therefore, we use the relation "EQUIPMENT". Furthermore, we add to the relation the foreign key number which is the primary key of the GREENHOUSE entity type, and name it *'GRN_Nb'*.

7- PLANT (Is_located):

Code	Species	Name	Price	Age	Water_usage	BRANCH_Nb	VHC_S/N	GRN_Nb

Many plants are located in one greenhouse. The “IS_LOCATED” is a relationship between the GREENHOUSE entity type and the PLANT entity type. In this case, the “many” side is the PLANT entity type. Therefore, we use the previously modified relation “PLANT”. Furthermore, we add to the relation the foreign key number which is the primary key of the GREENHOUSE entity type, and name it *‘GRN_Nb’*.

8- STORAGE (Has_a):

Storage_number	Maximum_capacity	GRN_Nb

One greenhouse has many storages. The “HAS_A” is a relationship between the GREENHOUSE entity type and the STORAGE entity type. In this case, the “many” side is the STORAGE entity type. Therefore, we use the relation “STORAGE”. Furthermore, we add to the relation the foreign key number which is the primary key of the GREENHOUSE entity type, and name it *‘GRN_Nb’*.

9- NUTRIENT (Stores):

Code	Name	Cost	Expiry_date	Time_to_Expire	STRG_Nb	Quantity_stored

One storage stores many nutrients. The “STORES” is a relationship between the NUTRIENT entity type and the STORAGE entity type. In this case, the “many” side is the NUTRIENT entity type. Therefore, we use the relation "NUTRIENT". Furthermore, we add to the relation the foreign key number which is the primary key of the STORAGE entity type, and name it *‘STRG_Nb’* in addition to the simple attribute of Stores *‘Quantity_stored’*.

10- EQUIPMENT (Contains):

S/N	Type	Cost	Status	GRN_Nb	STRG_Nb

One storage contains several types of equipment. The "CONTAIN" is a relationship between the STORAGE entity type and the EQUIPMENT entity type. In this case, the "many" side is the EQUIPMENT entity type. Therefore, we use the previously modified relation "EQUIPMENT". Furthermore, we add to the relation the foreign key number which is the primary key of the STORAGE entity type, and name it *‘STRG_Nb’*.

11- NUTRIENT (Supplies_1):

Code	Name	Cost	Expiry_ date	Time_ to_Expire	STRG_Nb	Quantity_ stored
SUPP_ID	Quantity_ supplied					

One supplier supplies many nutrients. The “SUPPLIES_1” is a relationship between the NUTRIENT entity type and the SUPPLIER entity type. In this case, the “many” side is the NUTRIENT entity type. Therefore, we use the previously modified relation "NUTRIENT". Furthermore, we add to the relation the foreign key ID which is the primary key of the SUPPLIER entity type, and name it *‘SUPP_ID’*, in addition to the simple attribute of the relation called *‘Quantity_supplied’*.

12- EQUIPMENT (Supplies_2):

S/N	Type	Cost	Status	GRN_Nb	STRG_Nb
Quantity_ stored	SUPP_ID				

One supplier supplies several types of equipment. The "SUPPLIES_2" is a relationship between the EQUIPMENT entity type and the SUPPLIER entity type. In this case, the "many" side is the EQUIPMENT entity type. Therefore, we use the previously modified relation "EQUIPMENT". Furthermore, we add to the relation the foreign key ID which is the primary key of the SUPPLIER entity type, and name it *'SUPP_ID'*.

13- PLANT (Provides):

Code	Species	Name	Price	Age	Water_usage	BRANCH_Nb
VHC_S/N	GRN_Nb	SUPP_ID				

One supplier provides many plants. The "PROVIDES" is a relationship between the PLANT entity type and the SUPPLIER entity type. In this case, the "many" side is the PLANT entity type. Therefore, we use the previously modified relation "PLANT". Furthermore, we add to the relation the foreign key ID which is the primary key of the SUPPLIER entity type, and name it *'SUPP_ID'*.

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 relationship that 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: Drives, Purchases_from, Utilizes, and Requires.

1- Drives:

EMP-ID	VHC-S/N

Many employees drive many vehicles. The “DRIVES” relationship links the EMPLOYEE entity type and the VEHICLE entity type. We create a new relation called "DRIVES" that includes the primary key of both entities EMPLOYEE and VEHICLE. The primary key of the EMPLOYEE entity type, ID, is added to this relation and renamed *‘EMP-ID’*. Also, the primary key of VEHICLE, S/N, is added and renamed *‘VHC-S/N’*. The combination of those two keys forms the primary key of this relation.

2- PURCHASES_FROM:

CUST-ID	BRANCH-Nb	Quantity_purchased

Many customers purchase from many branches. The “PURCHASE_FROM” relationship links the CUSTOMER entity type and the BRANCH entity type. We create a new relation called "PURCHASE_FROM" that includes the primary key of both entities CUSTOMER and BRANCH. The primary key of the CUSTOMER entity type, ID, is added to this relation and renamed *‘CUST-ID’*. Also, the primary key of BRANCH, branch_number, is added and renamed *‘BRANCH_Nb’*. The combination of those two keys forms the primary key of this relation. We also add the simple attribute of this relation called *‘Quantity_purchased’*.

3- UTILIZES:

EMP-ID	EQUIP-S/N

Many employees use several types of equipment. The "UTILIZES" relationship links the EMPLOYEE entity type and the EQUIPMENT entity type. We create a new relation called "UTILIZES" that includes the primary key of both entities EMPLOYEE and EQUIPMENT. The primary key of the EMPLOYEE entity type, ID, is added to this relation and renamed *‘EMP-ID’*. Also, the primary key of EQUIPMENT, S/N, is added and renamed *‘EQUIP-S/N’*. The combination of those two keys forms the primary key of this relation.

4- REQUIRES:

Plant-Code	Nutrient-Code	Dosage

Many plants require many nutrients. The “REQUIRES” relationship links the PLANT entity type and the NUTRIENT entity type. We create a new relation called "REQUIRES" that includes the primary key of both entities PLANT and NUTRIENT. The primary key of the PLANT entity type, code, is added to this relation and renamed *‘Plant-Code’*. Also, the primary key of NUTRIENT, S/N, is added and renamed *‘Nutrient-Code’*. The combination of those two keys forms the primary key of this relation. We also add the simple attribute of this relation called *‘Dosage’*.

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 type to which it belongs. Their combination will represent the primary key of the newly created relation. We have three multivalued attributes which are: CUST_PURCHASE_DATES, EMP_PHONE_NUMBER, and SUPP_SUPPLY_DATES.

1- CUST_PURCHASE_DATES:

CUST-ID	Purchase-dates

The multivalued attribute Purchase-dates belong to the CUSTOMER entity type. To illustrate it, we create a relation called "CUST_PURCHASE_DATES". Its primary key is made of the primary key of the CUSTOMER entity type, which is *‘CUST_ID’*, as well as the *‘Purchase-dates’* attribute that represents the multiple dates that customers have purchased on.

2- EMP_PHONE_NUMBER:

EMP-ID	Phone-number

The multivalued attribute Phone-number belongs to the EMPLOYEE entity type. To illustrate it, we create a relation called "EMP_PHONE_NUMBER". Its primary key is made of the primary key of the EMPLOYEE entity type, which is *‘EMP-ID’*, and the ‘*Phone-number’* attribute.

3- SUPP_SUPPLY_DATES:

SUPP-ID	Supply-dates

The multivalued attribute Supply-dates belongs to the SUPPLIER entity type. To illustrate it, we create a relation called “SUPP_SUPPLY_DATES”. Its primary key is made of the primary key of the SUPPLIER entity type, which is *‘SUPP-ID’*, as well as the *‘Supply-dates’* attribute to track the dates of when these supplies are being purchased.

STEP 7: Mapping of N-ary Relationship Types

In this step, we are going to map the N-ary 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

DEPARTMENT:

Department-number	Name	Email	Phone_number	City	Street_Nb	Building
Floor	Emp-ID					

CUSTOMER:

ID	Phone_number	Name	City	Street_Nb

EMPLOYEE:

ID	F_Name	L_Name	Gender	Salary	DOB	Age
Specialization	City	Street_Nb	Building	Floor	Supervisor _ID	DPT_Nb
GRN_Nb						

SCHEDULE:

Emp-ID	Start-time	End-time

BRANCH:

Branch_number	City	Street_number	Email	Phone_number	Emp-ID

VEHICLE:

S/N	Registration	Type

GREENHOUSE:

Greenhouse-number	City	Street_name	Type	Size	Number_of_plants

PLANT:

Code	Species	Name	Price	Age	Water_usage	BRANCH_Nb
VHC_S/N	GRN_Nb	SUPP_ID				

NUTRIENT:

Code	Name	Cost	Expiry_date	Time_to_Expire	STRG_Nb	Quantity_stored
SUPP_ID	Quantity_supplied					

STORAGE:

Storage_number	Maximum_capacity	GRN_Nb

EQUIPMENT:

S/N	Type	Cost	Status	GRN_Nb	STRG_Nb
SUPP_ID					

SUPPLIER:

ID	Name	Phone_number

Drives:

EMP-ID	VHC-S/N

PURCHASES_FROM:

CUST-ID	BRANCH-Nb	Quantity_purchased

UTILIZES:

EMP-ID	EQUIP-S/N

REQUIRES:

Plant-Code	Nutrient-Code	Dosage

CUST_PURCHASE_DATES:

CUST-ID	Purchase-dates

EMP_PHONE_NUMBER:

EMP-ID	Phone-number

SUPP_SUPPLY_DATES:

SUPP-ID	Supply-dates

J- Table Structure for GoGreen Plant Nursery:

After designing the ER diagram for GoGreen plant nursery and mapping this diagram into relational database design, now it is time to start creating the concrete tables for our database on the Oracle Database Server. We will begin by creating all tables and then inserting data into them. Finally, we will execute some queries to show the value of the database in a plant nursery.

1- DEPARTMENT:

```
CREATE TABLE DEPARTMENT
        (
                Department_number INT PRIMARY KEY,
                Name VARCHAR(30) NOT NULL,
                Email VARCHAR(50) NOT NULL,
                Phone_number CHAR(14),
                City VARCHAR(15),
                Street_Nb INT,
                Building VARCHAR(30),
                Floor INT,
                Emp_ID INT
        );
        ALTER TABLE DEPARTMENT
        ADD FOREIGN KEY(EMP_ID)
        REFERENCES EMPLOYEE(ID);
```

2- CUSTOMER:

```
CREATE TABLE CUSTOMER
        (
                ID INT PRIMARY KEY,
                Phone_Number CHAR(14),
                Name VARCHAR(30),
                City VARCHAR(30),
                Street_Nb INT);
```

3- EMPLOYEE:

```
CREATE TABLE EMPLOYEE
        (
                ID INT PRIMARY KEY,
                F_Name VARCHAR(15),
                L_Name VARCHAR(15),
                Gender CHAR(1),
                Salary INT,
                DOB  DATE NOT NULL,
                AGE INT,
                Specialization VARCHAR(30),
                City VARCHAR(30),
                Street_Nb INT,
                Building VARCHAR(50),
                Floor INT,
                Supervisor_ID INT,
                DPT_Nb INT,
                GRN_Nb INT,
        FOREIGN KEY(Supervisor_ID)
        REFERENCES EMPLOYEE(ID),
        FOREIGN KEY(DPT_Nb)
        REFERENCES DEPARTMENT(Department_number)
        );
                ALTER TABLE EMPLOYEE
                ADD FOREIGN KEY(GRN_Nb)
                REFERENCES GREENHOUSE(Greenhouse_number);
```

4- SCHEDULE:

SCHEDULE:

```
CREATE TABLE SCHEDULE
        (
                Emp_ID INT NOT NULL,
                Start_time TIMESTAMP,
                End_time TIMESTAMP,
        FOREIGN KEY(Emp_ID)
        REFERENCES EMPLOYEE(ID)
        );
```

5- BRANCH:

```
CREATE TABLE BRANCH
        (
                Branch_number INT PRIMARY KEY,
                City VARCHAR(30),
                Street_number INT,
                Email VARCHAR(50),
                Phone_number CHAR(14),
                Emp_ID INT,
        FOREIGN KEY(Emp_ID)
        REFERENCES EMPLOYEE(ID)
        );
```

6- VEHICLE:

```
CREATE TABLE VEHICLE
        (
                S_N CHAR(17) PRIMARY KEY,
                Registration VARCHAR(7),
                Type VARCHAR(10)
        );
```

7- GREENHOUSE:

```
CREATE TABLE GREENHOUSE
        (
                Greenhouse_number INT PRIMARY KEY,
                City VARCHAR(30),
                Street_name VARCHAR(50),
                Type VARCHAR(40),
                Size_of_Greenhouse VARCHAR(10),
                Number_of_plants INT
        );
```

8- PLANT:

```
CREATE TABLE PLANT
(
        Code INT PRIMARY KEY,
        Species VARCHAR(30),
        Name VARCHAR(50),
        Price INT,
        Water_usage VARCHAR(4),
        BRANCH_Nb INT,
        VHC_S_N CHAR(17),
        GRN_Nb INT,
        SUPP_ID INT,
        Age INT,
FOREIGN KEY(BRANCH_Nb)
REFERENCES BRANCH(Branch_number),
FOREIGN KEY(VHC_S_N)
REFERENCES VEHICLE(S_N)
);
        ALTER TABLE PLANT
        ADD FOREIGN KEY(GRN_Nb)
        REFERENCES GREENHOUSE(Greenhouse_number);
        ALTER TABLE PLANT
        ADD FOREIGN KEY(SUPP_ID)
        REFERENCES SUPPLIER(ID);
```

9- SUPPLIER:

```
CREATE TABLE SUPPLIER
        (
                ID INT PRIMARY KEY,
                Name VARCHAR(30),
                Phone_number CHAR(13)
        );
```

10- STORAGE:

```
CREATE TABLE STORAGE
        (
                Storage_number INT PRIMARY KEY,
                Maximum_capacity INT,
                GRN_Nb INT,
        FOREIGN KEY(GRN_Nb)
        REFERENCES GREENHOUSE(Greenhouse_number));
```

11- NUTRIENT:

```
CREATE TABLE NUTRIENT
        (
                Code INT PRIMARY KEY,
                Name VARCHAR(15),
                Cost INT,
                Expiry_date DATE,
                Time_to_Expire INT,
                STRG_Nb INT,
                Quantity_Stored INT,
                SUPP_ID INT,
                Quantity_supplied INT,
        FOREIGN KEY(STRG_Nb)
        REFERENCES STORAGE(Storage_number),
        FOREIGN KEY(SUPP_ID)
        REFERENCES SUPPLIER(ID)
        );
```

12- EQUIPEMENT:

```
CREATE TABLE EQUIPMENT
        (
                S_N VARCHAR(15) PRIMARY KEY,
                Type VARCHAR(20),
                Cost INT,
                Status VARCHAR(20),
                GRN_Nb INT,
                STRG_Nb INT,
                SUPP_ID INT,
        FOREIGN KEY (GRN_Nb)
        REFERENCES GREENHOUSE(Greenhouse_number),
        FOREIGN KEY(STRG_Nb)
        REFRENCES STORAGE(Storage_number),
        FOREIGN KEY(SUPP_ID)
        REFERENCES SUPPLIER(ID)
        );
```

13- DRIVES:

```
CREATE TABLE DRIVES
        (
                EMP_ID INT NOT NULL,
                VHC_S_N CHAR(17) NOT NULL,
        FOREIGN KEY(EMP_ID)
        REFERENCES EMPLOYEE(ID),
        FOREIGN KEY(VHC_S_N)
        REFERENCES VEHICLE(S_N)
        );
```

14- PURCHASES_FROM:

```
CREATE TABLE PURCHASES_FROM
        (
                CUST_ID INT,
                BRANCH_Nb INT,
                Quantity_purchased INT,
        FOREIGN KEY(CUST_ID)
        REFERENCES CUSTOMER(ID),
        FOREIGN KEY(BRANCH_Nb)
        REFERENCES BRANCH(Branch_number)
        );
```

15- UTILIZES:

```
CREATE TABLE UTILIZES
        (
                EMP_ID INT,
                EQUIP_S_N VARCHAR(17),
        FOREIGN KEY(EMP_ID)
        REFERENCES EMPLOYEE(ID),
        FOREIGN KEY(EQUIP_S_N)
        REFERENCES EQUIPMENT(S_N)
        );
```

16- REQUIRES:

```
CREATE TABLE REQUIRES
        (
                Plant_Code INT,
                Nutrient_Code INT,
            Dosage VARCHAR(10),
        FOREIGN KEY(Plant_Code)
        REFERENCES PLANT(Code),
        FOREIGN KEY(Nutrient_Code)
        REFERENCES NUTRIENT(Code));
```

17- CUST_PURCHASE_DATES:

```
    CREATE TABLE SUPP_SUPPLY_DATES
        (
            SUPP_ID INT,
            Supply_dates DATE,
            FOREIGN KEY(SUPP_ID)
            REFERENCES SUPPLIER (ID)
        );
```

18- EMP_PHONE_NUMBER:

```
    CREATE TABLE EMP_PHONE_NUMBER
        (
            EMP_ID INT,
            Phone_number CHAR(14),
            FOREIGN KEY(EMP_ID)
            REFERENCES EMPLOYEE(ID)
        );
```

19- SUPP_SUPPLY_DATES:

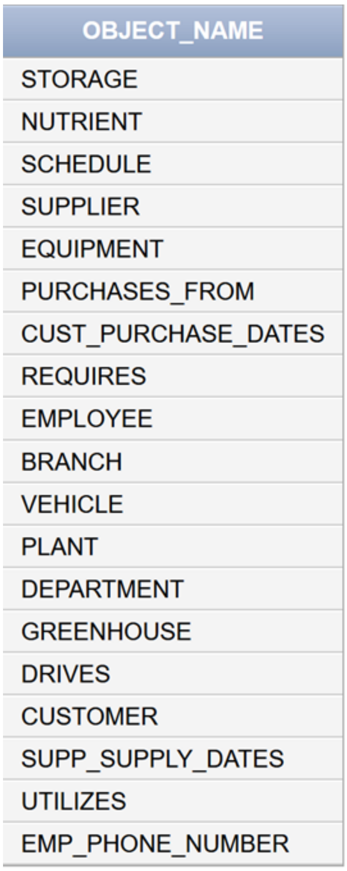
```
    CREATE TABLE SUPP_SUPPLY_DATES
        (
            SUPP_ID INT,
            Supply_dates DATE PRIMARY KEY,
        FOREIGN KEY(SUPP_ID)
        REFERENCES SUPPLIER (ID)
        );
```

K-Table Description:

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

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

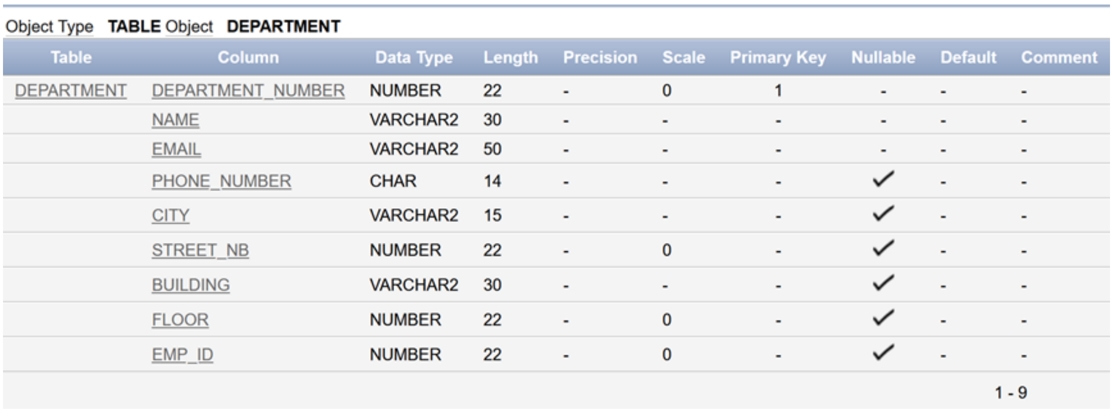
```
SQL>    SELECT DISTINCT OBJECT_NAME
        FROM USER_OBJECTS
        WHERE OBJECT_TYPE='TABLE';
```



OBJECT_NAME
STORAGE
NUTRIENT
SCHEDULE
SUPPLIER
EQUIPMENT
PURCHASES_FROM
CUST_PURCHASE_DATES
REQUIRES
EMPLOYEE
BRANCH
VEHICLE
PLANT
DEPARTMENT
GREENHOUSE
DRIVES
CUSTOMER
SUPP_SUPPLY_DATES
UTILIZES
EMP_PHONE_NUMBER

1- DEPARTMENT:

SQL> DESC DEPARTMENT;



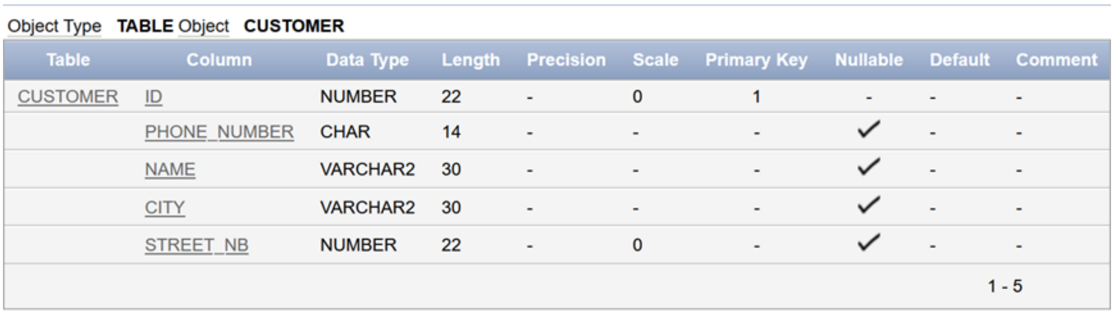
Object Type TABLE Object DEPARTMENT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPARTMENT	DEPARTMENT_NUMBER	NUMBER	22	-	0	1	-	-	-
	NAME	VARCHAR2	30	-	-	-	-	-	-
	EMAIL	VARCHAR2	50	-	-	-	-	-	-
	PHONE_NUMBER	CHAR	14	-	-	-	✓	-	-
	CITY	VARCHAR2	15	-	-	-	✓	-	-
	STREET_NB	NUMBER	22	-	0	-	✓	-	-
	BUILDING	VARCHAR2	30	-	-	-	✓	-	-
	FLOOR	NUMBER	22	-	0	-	✓	-	-
	EMP_ID	NUMBER	22	-	0	-	✓	-	-

1 - 9

2- CUSTOMER:

SQL> DESC CUSTOMER;



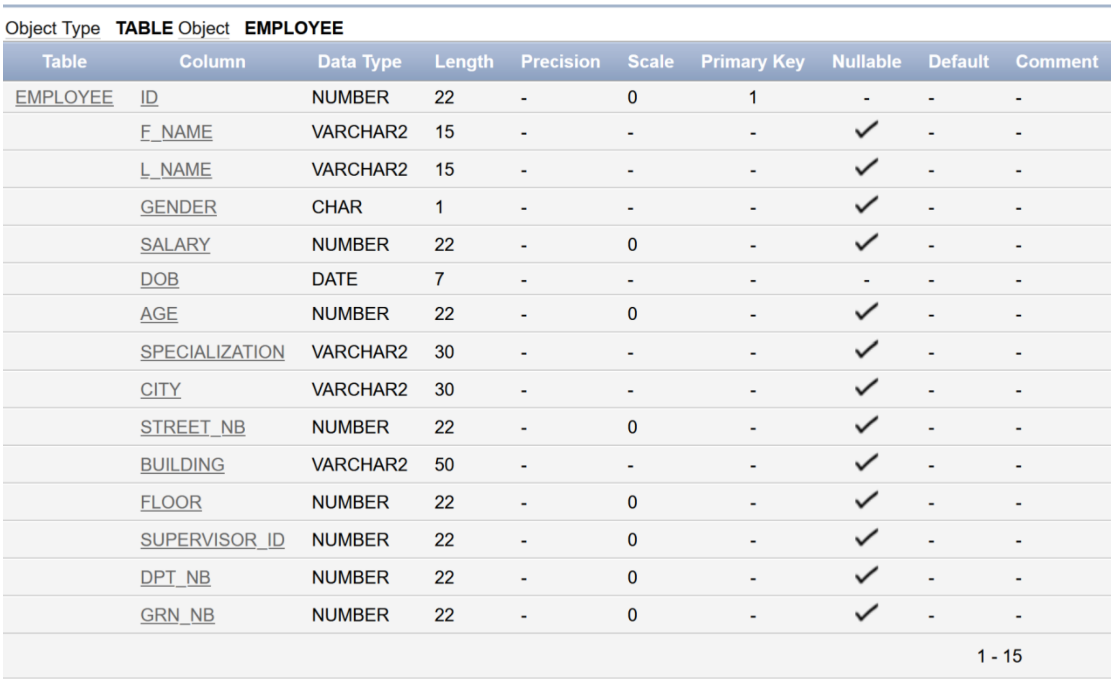
Object Type TABLE Object CUSTOMER

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUSTOMER	ID	NUMBER	22	-	0	1	-	-	-
	PHONE_NUMBER	CHAR	14	-	-	-	✓	-	-
	NAME	VARCHAR2	30	-	-	-	✓	-	-
	CITY	VARCHAR2	30	-	-	-	✓	-	-
	STREET_NB	NUMBER	22	-	0	-	✓	-	-

1 - 5

3- EMPLOYEE:

SQL> DESC EMPLOYEE;



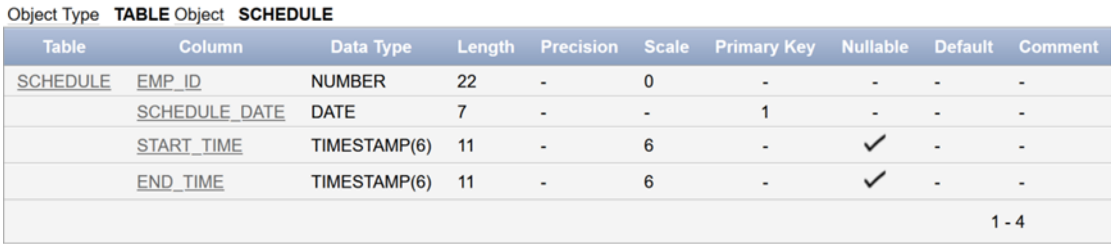
Object Type **TABLE** Object **EMPLOYEE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMPLOYEE	ID	NUMBER	22	-	0	1	-	-	-
	F_NAME	VARCHAR2	15	-	-	-	✓	-	-
	L_NAME	VARCHAR2	15	-	-	-	✓	-	-
	GENDER	CHAR	1	-	-	-	✓	-	-
	SALARY	NUMBER	22	-	0	-	✓	-	-
	DOB	DATE	7	-	-	-	-	-	-
	AGE	NUMBER	22	-	0	-	✓	-	-
	SPECIALIZATION	VARCHAR2	30	-	-	-	✓	-	-
	CITY	VARCHAR2	30	-	-	-	✓	-	-
	STREET_NB	NUMBER	22	-	0	-	✓	-	-
	BUILDING	VARCHAR2	50	-	-	-	✓	-	-
	FLOOR	NUMBER	22	-	0	-	✓	-	-
	SUPERVISOR_ID	NUMBER	22	-	0	-	✓	-	-
	DPT_NB	NUMBER	22	-	0	-	✓	-	-
	GRN_NB	NUMBER	22	-	0	-	✓	-	-

1 - 15

4- SCHEDULE:

SQL> DESC SCHEDULE;



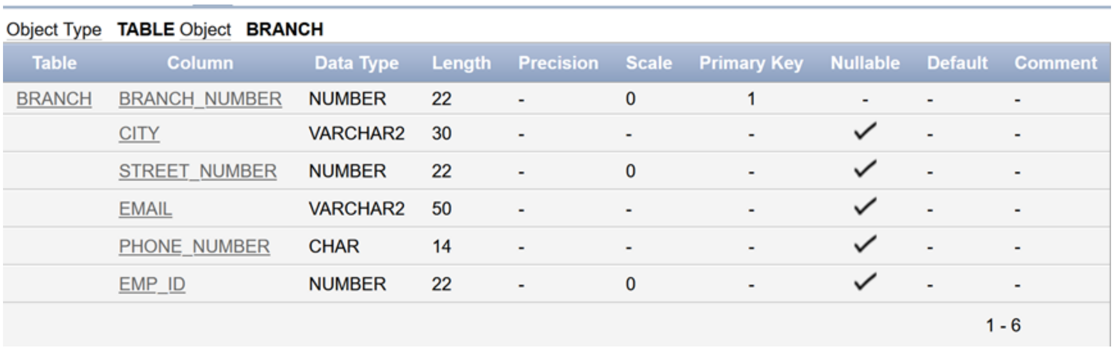
Object Type **TABLE** Object **SCHEDULE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SCHEDULE	EMP_ID	NUMBER	22	-	0	-	-	-	-
	SCHEDULE_DATE	DATE	7	-	-	1	-	-	-
	START_TIME	TIMESTAMP(6)	11	-	6	-	✓	-	-
	END_TIME	TIMESTAMP(6)	11	-	6	-	✓	-	-

1 - 4

5- BRANCH:

SQL> DESC BRANCH;



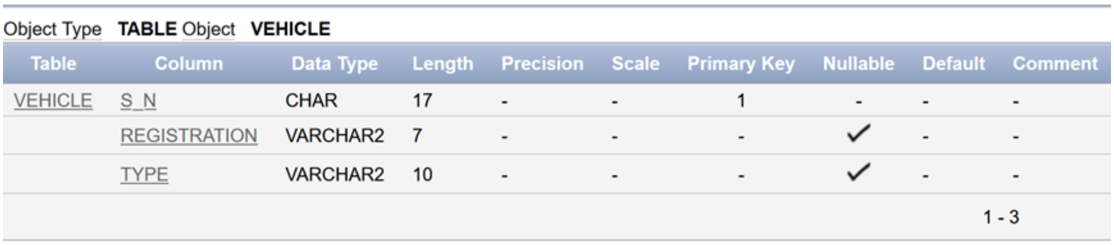
Object Type **TABLE** Object **BRANCH**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
BRANCH	BRANCH_NUMBER	NUMBER	22	-	0	1	-	-	-
	CITY	VARCHAR2	30	-	-	-	✓	-	-
	STREET_NUMBER	NUMBER	22	-	0	-	✓	-	-
	EMAIL	VARCHAR2	50	-	-	-	✓	-	-
	PHONE_NUMBER	CHAR	14	-	-	-	✓	-	-
	EMP_ID	NUMBER	22	-	0	-	✓	-	-

1 - 6

6- VEHICLE:

SQL> DESC VEHICLE;



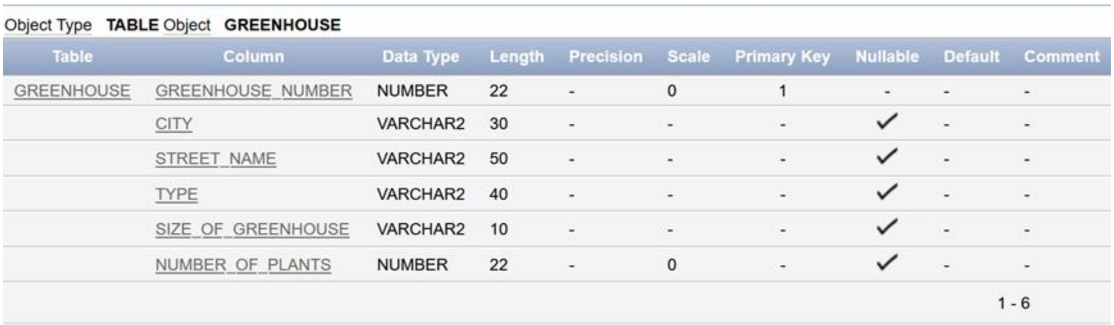
Object Type **TABLE** Object **VEHICLE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
VEHICLE	S_N	CHAR	17	-	-	1	-	-	-
	REGISTRATION	VARCHAR2	7	-	-	-	✓	-	-
	TYPE	VARCHAR2	10	-	-	-	✓	-	-

1 - 3

7- GREENHOUSE:

SQL> DESC GREENHOUSE;



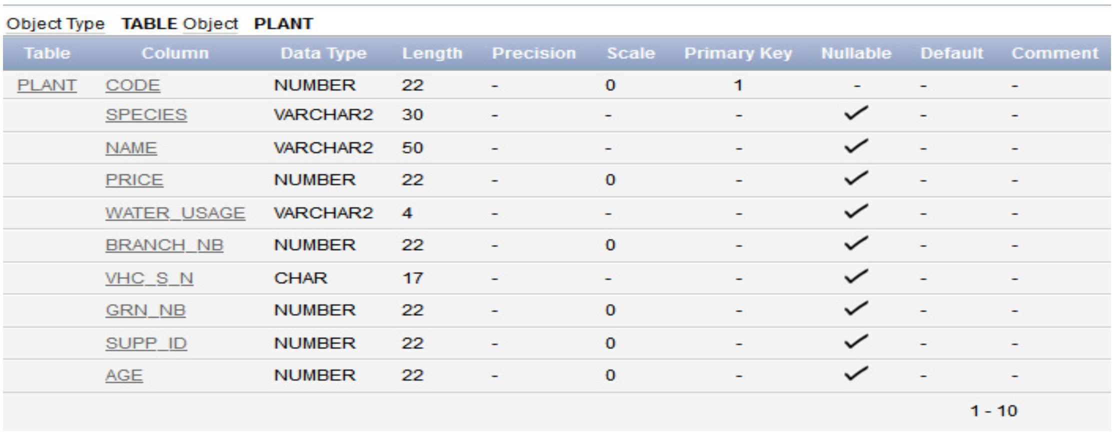
Object Type **TABLE** Object **GREENHOUSE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GREENHOUSE	GREENHOUSE_NUMBER	NUMBER	22	-	0	1	-	-	-
	CITY	VARCHAR2	30	-	-	-	✓	-	-
	STREET_NAME	VARCHAR2	50	-	-	-	✓	-	-
	TYPE	VARCHAR2	40	-	-	-	✓	-	-
	SIZE_OF_GREENHOUSE	VARCHAR2	10	-	-	-	✓	-	-
	NUMBER_OF_PLANTS	NUMBER	22	-	0	-	✓	-	-

1 - 6

8- PLANT:

SQL> DESC PLANT;



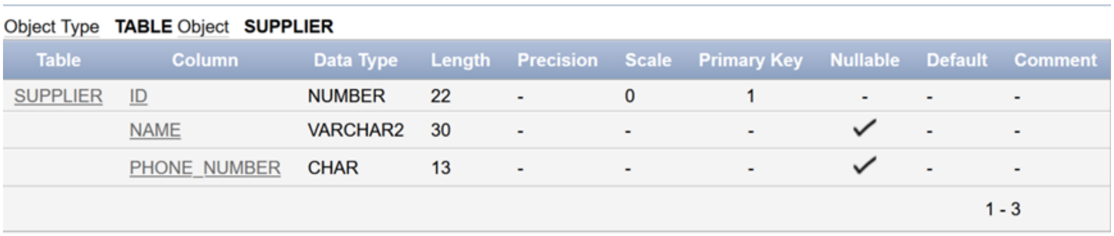
Object Type **TABLE** Object **PLANT**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PLANT	CODE	NUMBER	22	-	0	1	-	-	-
	SPECIES	VARCHAR2	30	-	-	-	✓	-	-
	NAME	VARCHAR2	50	-	-	-	✓	-	-
	PRICE	NUMBER	22	-	0	-	✓	-	-
	WATER_USAGE	VARCHAR2	4	-	-	-	✓	-	-
	BRANCH_NB	NUMBER	22	-	0	-	✓	-	-
	VHC_S_N	CHAR	17	-	-	-	✓	-	-
	GRN_NB	NUMBER	22	-	0	-	✓	-	-
	SUPP_ID	NUMBER	22	-	0	-	✓	-	-
	AGE	NUMBER	22	-	0	-	✓	-	-

1 - 10

9- SUPPLIER:

SQL> DESC SUPPLIER;



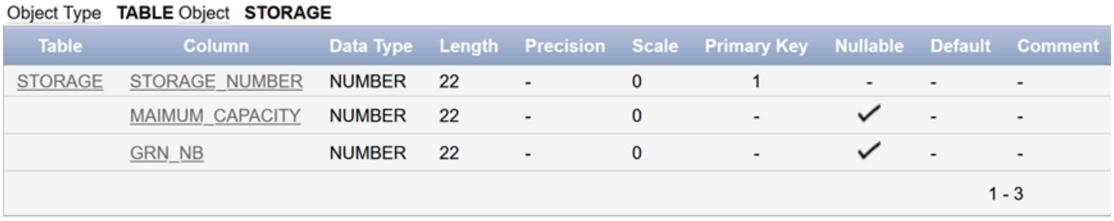
Object Type **TABLE** Object **SUPPLIER**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SUPPLIER	ID	NUMBER	22	-	0	1	-	-	-
	NAME	VARCHAR2	30	-	-	-	✓	-	-
	PHONE_NUMBER	CHAR	13	-	-	-	✓	-	-

1 - 3

10- STORAGE:

SQL> DESC STORAGE;



Object Type **TABLE** Object **STORAGE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STORAGE	STORAGE_NUMBER	NUMBER	22	-	0	1	-	-	-
	MAIMUM_CAPACITY	NUMBER	22	-	0	-	✓	-	-
	GRN_NB	NUMBER	22	-	0	-	✓	-	-

1 - 3

11- NUTRIENT:

SQL> DESC NUTRIENT;

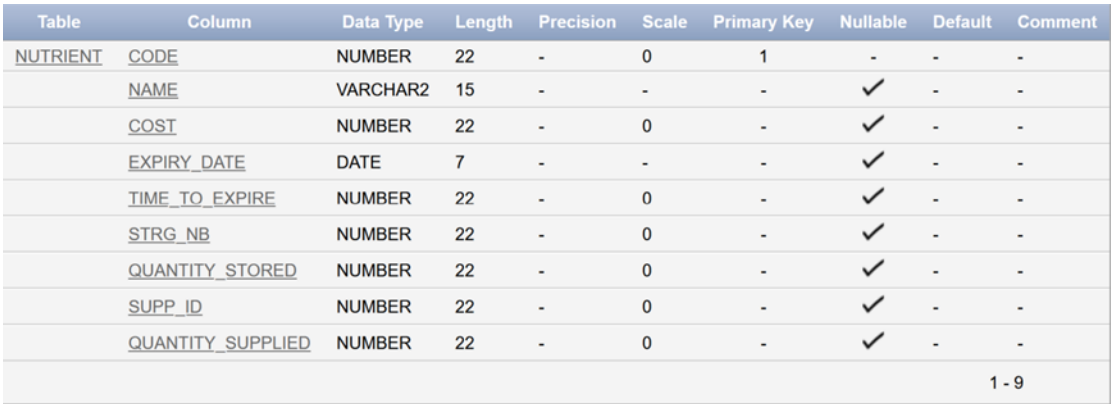
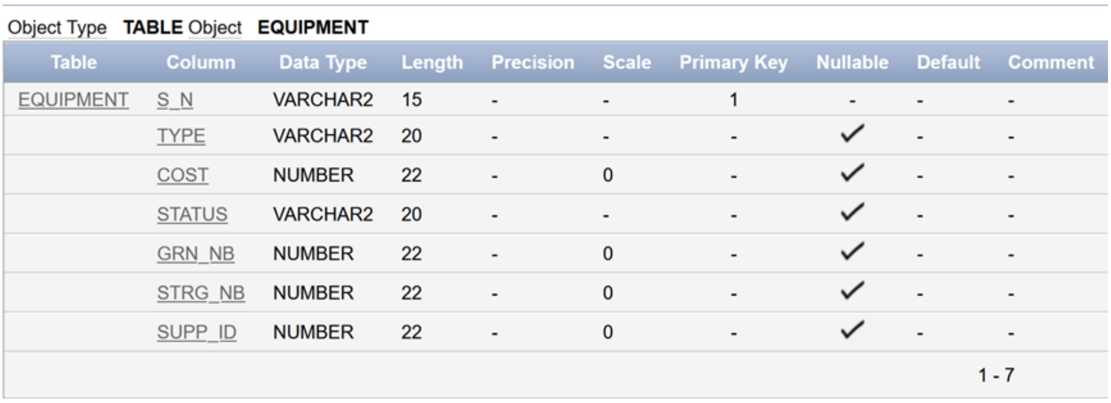


Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
NUTRIENT	CODE	NUMBER	22	-	0	1	-	-	-
	NAME	VARCHAR2	15	-	-	-	✓	-	-
	COST	NUMBER	22	-	0	-	✓	-	-
	EXPIRY_DATE	DATE	7	-	-	-	✓	-	-
	TIME_TO_EXPIRE	NUMBER	22	-	0	-	✓	-	-
	STRG_NB	NUMBER	22	-	0	-	✓	-	-
	QUANTITY_STORED	NUMBER	22	-	0	-	✓	-	-
	SUPP_ID	NUMBER	22	-	0	-	✓	-	-
	QUANTITY_SUPPLIED	NUMBER	22	-	0	-	✓	-	-

1 - 9

12- EQUIPMENT:

SQL> DESC EQUIPMENT;



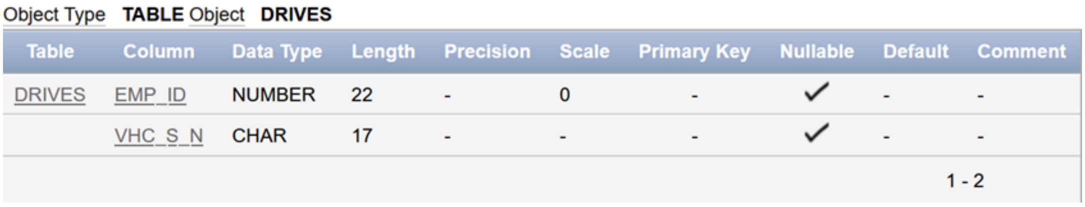
Object Type TABLE Object EQUIPMENT

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EQUIPMENT	S_N	VARCHAR2	15	-	-	1	-	-	-
	TYPE	VARCHAR2	20	-	-	-	✓	-	-
	COST	NUMBER	22	-	0	-	✓	-	-
	STATUS	VARCHAR2	20	-	-	-	✓	-	-
	GRN_NB	NUMBER	22	-	0	-	✓	-	-
	STRG_NB	NUMBER	22	-	0	-	✓	-	-
	SUPP_ID	NUMBER	22	-	0	-	✓	-	-

1 - 7

13- DRIVES:

SQL> DESC DRIVES;



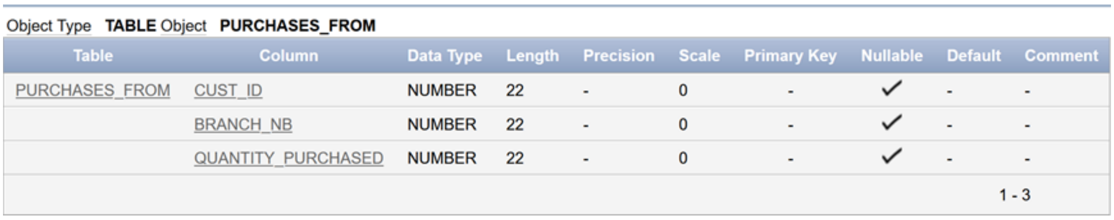
Object Type TABLE Object DRIVES

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DRIVES	EMP_ID	NUMBER	22	-	0	-	✓	-	-
	VHC_S_N	CHAR	17	-	-	-	✓	-	-

1 - 2

14- PURCHASES_FROM:

SQL> DESC PURCHASES_FROM;



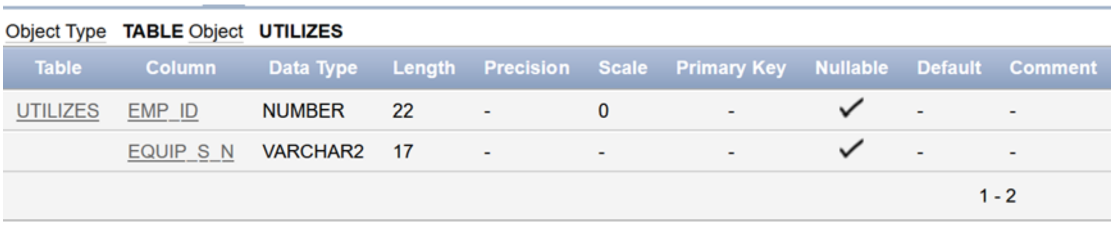
Object Type TABLE Object PURCHASES_FROM

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PURCHASES_FROM	CUST_ID	NUMBER	22	-	0	-	✓	-	-
	BRANCH_NB	NUMBER	22	-	0	-	✓	-	-
	QUANTITY_PURCHASED	NUMBER	22	-	0	-	✓	-	-

1 - 3

15- UTILIZES:

SQL> DESC UTILIZES;



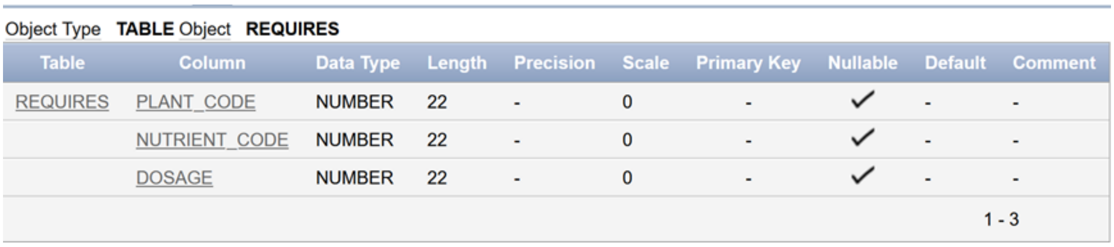
Object Type TABLE Object UTILIZES

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
UTILIZES	EMP_ID	NUMBER	22	-	0	-	✓	-	-
	EQUIP_S_N	VARCHAR2	17	-	-	-	✓	-	-

1 - 2

16- REQUIRES:

SQL> DESC REQUIRES;



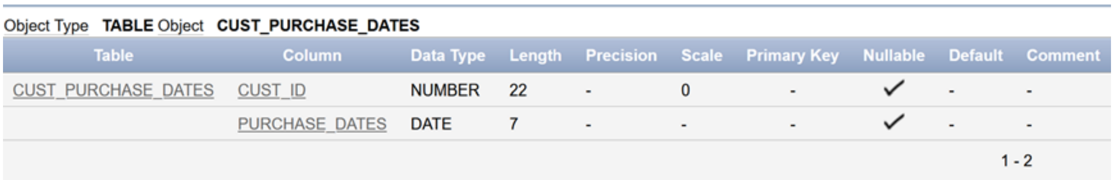
Object Type TABLE Object REQUIRES

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
REQUIRES	PLANT_CODE	NUMBER	22	-	0	-	✓	-	-
	NUTRIENT_CODE	NUMBER	22	-	0	-	✓	-	-
	DOSAGE	NUMBER	22	-	0	-	✓	-	-

1 - 3

17- CUST_PURCHASE_DATES:

SQL> DESC CUST_PURCHASE_DATES;



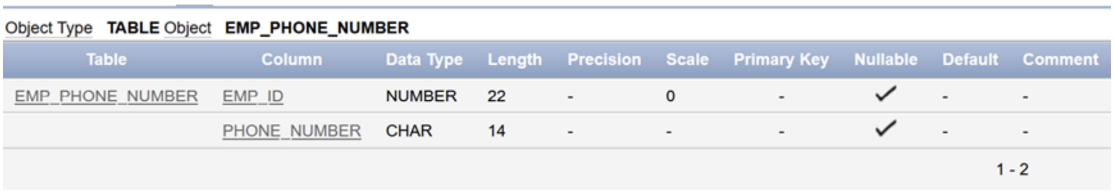
Object Type TABLE Object CUST_PURCHASE_DATES

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CUST_PURCHASE_DATES	CUST_ID	NUMBER	22	-	0	-	✓	-	-
	PURCHASE_DATES	DATE	7	-	-	-	✓	-	-

1 - 2

18- EMP_PHONE_NUMBER:

SQL> DESC EMP_PHONE_NUMBER;



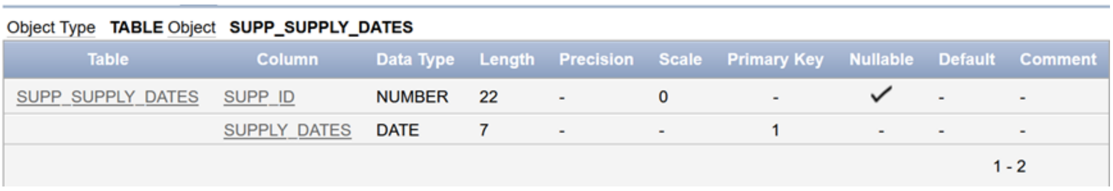
Object Type TABLE Object EMP_PHONE_NUMBER

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMP_PHONE_NUMBER	EMP_ID	NUMBER	22	-	0	-	✓	-	-
	PHONE_NUMBER	CHAR	14	-	-	-	✓	-	-

1 - 2

19- SUPP_SUPPLY_DATES:

SQL> DESC SUPPLY_DATES;



Object Type TABLE Object SUPP_SUPPLY_DATES

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SUPP_SUPPLY_DATES	SUPP_ID	NUMBER	22	-	0	-	✓	-	-
	SUPPLY_DATES	DATE	7	-	-	1	-	-	-

1 - 2

L- Inserting data:

1- DEPARTMENT:

```
INSERT INTO DEPARTMENT VALUES (11, ‘Agriculture’, ‘agriculture@gogreen.co’, ‘+961 01-333444’, ‘Beirut’, 21, ‘Safadi Building’, 5, 1845905447);

INSERT INTO DEPARTMENT VALUES (12, ‘Sales’, ‘sales@gogreen.co’, ‘+961 01-222555’, ‘Beirut’, 21, ‘Nicola Building’, 3, 1122334455);

INSERT INTO DEPARTMENT VALUES (13, ‘HR’, ‘hr@gogreen.co’, ‘+961 01-955777’, ‘Beirut’, 21, ‘Go Nature Building’, 1, 1234512345);

INSERT INTO DEPARTMENT VALUES (14, ‘Research’, ‘sciresearch@gogreen.co’, ‘+961 01-433366’, ‘Beirut’, 21, ‘Beirut local building’, 2, null);

INSERT INTO DEPARTMENT VALUES (15, ‘Finance’, ‘finance@gogreen.co’, ‘+961 01-777234’, ‘Beirut’, 21, ‘Nicola Building’, 2, 5544332211);

INSERT INTO DEPARTMENT VALUES (16, ‘Nursery’, ‘nurserydepartment@gogreen.co’, ‘+961 01-297198’, ‘Beirut’, 21, ‘Beirut local building’, 1, 2307411820);

INSERT INTO DEPARTMENT VALUES (17, ‘Customer Service’, ‘customerservice@gogreen.co’, ‘+961 01-589238’, ‘Beirut’, 21 ‘Safadi Building’, 2, 4892211133);

INSERT INTO DEPARTMENT VALUES (18, ‘Branches Management’, ‘branchesmanagement@gogreen.co’, ‘+961 01-499219’, ‘Beirut’, 21, ‘Managerial Building’, 2, null);
```

INSERT INTO DEPARTMENT VALUES (19, ‘Vehicle Management’, ‘transportationdepartment@gogreen.co’, ‘+961 01-398138’, ‘Beirut’, 21, ‘Managerial Building’, 3, null);

INSERT INTO DEPARTMENT VALUES (20, ‘Supplies Management’, ‘suppliesdepartment@gogreen.co’, ‘+961 81-483012’, ‘Beirut’, 21, ‘Managerial Building’, 4, 5431254321);

2- CUSTOMER:

INSERT INTO CUSTOMER VALUES (58147405, ‘+961 81-483924’, ‘Leyla Thana’, ‘Beirut’, 23);

INSERT INTO CUSTOMER VALUES (84397160, ‘+961 81-483308’, ‘Naziha Adila’, ‘Beqaa’, 43);

INSERT INTO CUSTOMER VALUES (95747451, ‘+961 81-293249’, ‘Samia Salman’, ‘Beqaa’, 34);

INSERT INTO CUSTOMER VALUES (61996164, ‘+961 81-492209’, ‘Anwar Shahd’, ‘Beirut’, 23);

INSERT INTO CUSTOMER VALUES (‘17664996’, ‘+961 81-394398’, ‘Talib Amina’, ‘Jnoub’, 23);

INSERT INTO CUSTOMER VALUES (42284032, ‘+961 81-129328’, ‘Rafiq Jalil’, ‘Beirut’, 23);

INSERT INTO CUSTOMER VALUES (94405296, ‘+961 81-437792’, ‘Salim Safaa’, ‘Beirut’, 53);

```
INSERT INTO CUSTOMER VALUES (58019557, ‘+961 81-329249’, ‘Dalia Noura’, ‘Beirut’, 51);
```

```
INSERT INTO CUSTOMER VALUES (63679144, ‘+961 81-393923’, ‘Fatin Amjad’, ‘Beirut’, 25);
```

```
INSERT INTO CUSTOMER VALUES (97303832, ‘+961 81-329293’, ‘Yasmeen Anwar’, ‘Jbeil’, 72);
```

```
INSERT INTO CUSTOMER VALUES (202, NULL, ‘Ramzi Harati’, ‘Hamra’, NULL);
```

3- EMPLOYEE:

```
INSERT INTO EMPLOYEE VALUES (6784705218, ‘Latif’, ‘Ghadir’, ‘M’, 500, ‘12-01-1990’, 31, ‘Greenhouse staff’, ‘Beirut’, 21, ‘Red Ribbon’, 1, 1845905447, 11, 31);
```

```
INSERT INTO EMPLOYEE VALUES (4349797788, ‘Mariam’, ‘Wasim’, ‘F’, 1000, ’01-01-1989’, 32, ‘Branch manager’, ‘Jbeil’, 72, ‘Aqua Building’, 2, 6784705218, 18, null);
```

```
INSERT INTO EMPLOYEE VALUES (5409979193, ‘Marwa’, ‘Noura’, ‘F’, 1000, ’03-07-1988’, 33, ‘Branch manager’, ‘Beirut’, 26, ‘Smith Building’, 1, null, 18, null);
```

```
INSERT INTO EMPLOYEE VALUES (6963258256, ‘Sani’, ‘Naaji’, ‘M’, 2000, ’09-10-1972’, 49, ‘Botanist’, ‘Beirut’, 21, ‘Erwin building’, 1, 1845905447, 11, 33);
```

```
INSERT INTO EMPLOYEE VALUES (1117618988, ‘Jaffar’, ‘Sanaa’, ‘F’, 3500, ‘12-05-1970’, 51, ‘Specialized Botanist’, ‘Jbeil’, 21, ‘Younes building’, 1, 1845905447, 11, 35);
```

INSERT INTO EMPLOYEE VALUES (2074524800, ‘Esmail’, ‘Naqi’, ‘M’, 300, ‘02-10-1999’, 22, ‘Customer service’, ‘Jbeil’, 82, ‘Nasser building’, 2, 4892211133, 17, null);

INSERT INTO EMPLOYEE VALUES (1845905447, ‘Safa’, ‘Gabr’, ‘F’, 5000, ‘12-12-1978’, 43, ‘Supervisor’, ‘Beirut’, 21, ‘Wafic Building’, 1, null, 11, null);

INSERT INTO EMPLOYEE VALUES (2307411820, ‘Samad’, ‘Faruq’, ‘M’, 5500, ‘12-06-1976’, 45, ‘Supervisor’, ‘Khalde’, ‘Near Choifeit Street’, ‘Wafic Building’,1, null, 16, null);

INSERT INTO EMPLOYEE VALUES (7487344706, ‘Danial’, ‘Maryam’, ‘M’, 4000, ‘12-07-1995’, 26, ‘Greenhouse staff’, ‘Beirut’, 21, ‘Dana Building’, 3, 1845905447, 11, 34);

INSERT INTO EMPLOYEE VALUES (8610870297, ‘Ahmad’, ‘Mukhtar’, ‘M’, 3900, ‘12-11-1996’, 25, ‘Greenhouse staff’, ‘Beirut’, 29, ‘Dana Building’, 4, 1845905447, 11, 34);

INSERT INTO EMPLOYEE VALUES (4892211133, ‘Marian’, ‘Kaouk’, ‘F’, 5000, ‘09-12-1978’, 43, ‘Supervisor’, ‘Beirut’, 23, ‘Younes Building’, 1, null, 17, null);

INSERT INTO EMPLOYEE VALUES (1122334455, ‘Farah’, ‘Bizri’, ‘F’, 5000, ‘11-12-1978’, 43, ‘Supervisor’, ‘Khalde’, 27, ‘Khaddage Building’, 1, null, 12, null);

INSERT INTO EMPLOYEE VALUES (1234512345, ‘Paul’, ‘Maroun’, ‘M’, 5100, ‘11-12-1978’, 43, ‘Supervisor’, ‘Hamra’, 14, ‘Smeha Building’, 1, null, 13, null);

INSERT INTO EMPLOYEE VALUES (5544332211, ‘Ali’, ‘Mawla’, ‘M’, 5100, ‘11-11-1979’, 42, ‘Supervisor’, ‘Jbeil’, 88, ‘Kwizatz Building’, 5, null, 15, null);

```
INSERT INTO EMPLOYEE VALUES (5431254321, ‘Moustapha’, ‘Nasser’, ‘M’,
5100, ‘11-11-1981’, 40, ‘Supervisor’, ‘Jbeil’, 89, ‘Hazeratch Building’, 3, null, 20,
null);
```

4- SCHEDULE:

```
INSERT INTO SCHEDULE VALUES (6784705218, TIMESTAMP ‘2019-12-1
09:00:00’, TIMESTAMP ‘2020-12-1 17:00:00’);

INSERT INTO SCHEDULE VALUES (4349797788, TIMESTAMP ‘2020-12-1
10:00:00’, TIMESTAMP ‘2021-12-1 17:00:00’);

INSERT INTO SCHEDULE VALUES (5409979193, TIMESTAMP ‘2019-12-1
09:00:00’, TIMESTAMP ‘2020-12-1 18:00:00’);

INSERT INTO SCHEDULE VALUES (6963258256, TIMESTAMP ‘2019-12-1
08:00:00’, TIMESTAMP ‘2020-12-1 17:00:00’);

INSERT INTO SCHEDULE VALUES (1117618988, TIMESTAMP ‘2019-12-1
09:00:00’, TIMESTAMP ‘2020-12-1 17:00:00’);

INSERT INTO SCHEDULE VALUES (2074524800, TIMESTAMP ‘2019-12-1
09:00:00’, TIMESTAMP ‘2020-12-1 18:00:00’);

INSERT INTO SCHEDULE VALUES (1845905447, TIMESTAMP ‘2020-12-1
09:00:00’, TIMESTAMP ‘2021-12-1 17:00:00’);

INSERT INTO SCHEDULE VALUES (2307411820, TIMESTAMP ‘2020-12-1
09:00:00’, TIMESTAMP ‘2021-12-1 19:00:00’);

INSERT INTO SCHEDULE VALUES (7487344706, TIMESTAMP ‘2020-12-1
11:00:00’, TIMESTAMP ‘2021-12-1 19:00:00’);
```

```
INSERT INTO SCHEDULE VALUES (8610870297, TIMESTAMP ‘2019-12-1 11:00:00’, TIMESTAMP ‘2020-12-1 19:00:00’);
```

5- BRANCH:

```
INSERT INTO BRANCH VALUES (21, ‘Beirut Hamra’, 13, ‘gogreenbeiruthamra@gogreen.co’, ‘+961 81-937298’, 6784705218);

INSERT INTO BRANCH VALUES (22, ‘Jbeil’, 81, ‘gogreenjbeil@gogreen.co’, ‘+961 81-4929439’, 2074524800);

INSERT INTO BRANCH VALUES (23, ‘Baalbek’, 18, ‘gogreenbaalbek@gogreen.co’, ‘+961 81-935941’, 6784705218);

INSERT INTO BRANCH VALUES (24, ‘Beqaa’, 21, ‘gogreenbeqaa@gogreen.co’, ‘+961 81-483484’, 2074524800);

INSERT INTO BRANCH VALUES (25, ‘Beirut Verdun’, 73, ‘gogreenbeirutverdun@gogreen.co’, ‘+961 81-494942’, 6784705218);

INSERT INTO BRANCH VALUES (26, ‘Beirut Gemeiyze’, 32, ‘gogreenbeirutgemeiyze@gogreen.co’, ‘+961 81-289394’, 2074524800);

INSERT INTO BRANCH VALUES (27, ‘Tyre’, 41, ‘gogreentyre@gogreen.co’, ‘+961 81-923937’, 2307411820);

INSERT INTO BRANCH VALUES (28, ‘Tripoli’, 63, ‘gogreentrablous@gogreen.co’, ‘+961 81-148937’, 2307411820);

INSERT INTO BRANCH VALUES (29, ‘Aley’, 79, ‘gogreenaley@gogreen.co’, ‘+961 81-652837’, 7487344706);
INSERT INTO BRANCH VALUES (30, ‘Beirut Mar Mikhael’, 55, ‘gogreenbeirutmarmakhiel@gogreen.co’, ‘+961 76-575180’, 7487344706);
```

6- VEHICLE:

```
INSERT INTO VEHICLE VALUES (‘1N4BL2AP8BN503925’, ‘M392218’, ‘Nissan’);

INSERT INTO VEHICLE VALUES (‘KNDJN2A27E7740516’, ‘M113179’, ‘Ford’);

INSERT INTO VEHICLE VALUES (‘JTJZK1BA8A2403790’, ‘A123456’, ‘Nissan’);

INSERT INTO VEHICLE VALUES (‘1GKET16S426107309’, ‘B682683’, ‘Toyota’);

INSERT INTO VEHICLE VALUES (‘5NPE34AF8FH082224’, ‘B428473’, ‘Hunda’);

INSERT INTO VEHICLE VALUES (‘2HNYD18753H504973’, ‘G385580’, ‘Audi’);

INSERT INTO VEHICLE VALUES (‘4V4NC9TK36N430345’, ‘O168579’, ‘Hunda’);

INSERT INTO VEHICLE VALUES (‘2G1WX15K429306075’, ‘B459078’, ‘Hunda’);

INSERT INTO VEHICLE VALUES (‘19UUA66274A046775’, ‘M422982’, ’Nissan’);

INSERT INTO VEHICLE VALUES (‘3C3CFFBR3CT340684’, ‘G100404’, ’Hunda’);
```

7- GREENHOUSE:

```
INSERT INTO GREENHOUSE VALUES (31, ‘Beirut’, ‘Hamra street’, ‘Freestanding’, ‘250m^2’, 3000);
```

INSERT INTO GREENHOUSE VALUES (32, ‘Jbeil’, ‘Byblos street’, ‘Freestanding’, ‘300m^2’, 3500);

INSERT INTO GREENHOUSE VALUES (33, ‘Beqaa’, ‘Chtoura street’, ‘Freestanding’, ‘400m^2’, 4000);

INSERT INTO GREENHOUSE VALUES (34, ‘Beirut’, ‘Verdun street’, ‘Attached’, ‘150m^2’, 2000);

INSERT INTO GREENHOUSE VALUES (35, ‘Jbeil’, ‘Mastita street’, ‘Freestanding’, ‘600m^2’, 6000);

INSERT INTO GREENHOUSE VALUES (36, ‘Aley’, ‘Bsous street’, ‘Attached’, ‘200m^2’, 2500);

INSERT INTO GREENHOUSE VALUES (37, ‘Tyre’, ‘El Kouds street’, ‘Attached’, ‘250m^2’, 3000);

INSERT INTO GREENHOUSE VALUES (38, ‘Tripoli’, ‘El Thakafa street’, ‘Freestanding’, ‘600m^2’, 6000);

INSERT INTO GREENHOUSE VALUES (39, ‘Faraiya’, ‘Faraya street’, ‘Attached’, ‘250m^2’, 3000);

INSERT INTO GREENHOUSE VALUES (40, ‘Kfardebian’, ‘Mazraat Kfardibiane’, ‘Freestanding’, ‘400m^2’, 4000);

8- PLANT:

INSERT INTO PLANT VALUES (9091, ‘Acer’, ‘amplum’, 60, ‘3%’, 21, ‘1N4BL2AP8BN503925’, 31, 313369229, 2);

```
INSERT INTO PLANT VALUES (9100, ‘Acer’, ‘barbinerve’, 50, ‘2%’, 22,
‘KNDJN2A27E7740516’, 32, 647010725, 1);

INSERT INTO PLANT VALUES (9119, ‘Narcissus’, ‘dubius’, 34, ‘3%’, 23,
‘JTJZK1BA8A2403790’, 33, 633281215, 3);

INSERT INTO PLANT VALUES (9491, ‘Acer’, ‘argutum’, 90, ‘4%’, 24,
‘1GKET16S426107309’, 34, 716552266, 5);

INSERT INTO PLANT VALUES (9161, ‘Narcissus’, ‘tazetta’, 20, ‘1%’, 25,
‘5NPE34AF8FH082224’, 35, 653634641, 1);

INSERT INTO PLANT VALUES (9661, ‘Bamboo’, ‘acidosasa’, 88, ‘3%’, 26,
‘2HNYD18753H504973’, 36, 636446549, 3);

INSERT INTO PLANT VALUES (6996, ‘Oxalis’, ‘alpina’, 19, ‘1%’, 27,
‘4V4NC9TK36N430345’, 37, 653464364, 1);

INSERT INTO PLANT VALUES (6975, ‘Oxalis’, ‘alata’, 22, ‘4%’, 28,
‘2G1WX15K429306075’, 38, 128373291, 1);

INSERT INTO PLANT VALUES (6361, ‘Oxalis’, ‘acetosella’, 24, ‘1%’, 29,
‘19UUA66274A046775’, 39, 238237842, 2);

INSERT INTO PLANT VALUES (6496, ‘Dahlia’, ‘excelsa’, 69, ‘5%’, 30,
‘3C3CFFBR3CT340684’, 40, 283282497, 3);

INSERT INTO PLANT VALUES (6493, ‘Dahlia’, ‘imperialis’, 63, ‘5%’, 21,
‘1N4BL2AP8BN503925’, 31, 313369229, 4);
```

```
INSERT INTO PLANT VALUES (8655, ‘Ficus’, ‘abelii’, 69, ‘5%’, 22, ‘KNDJN2A27E7740516’, 32, 647010725, 3);

INSERT INTO PLANT VALUES (1870, ‘Ficus’, ‘adelpha’, 63, ‘5%’, 23, ‘JTJZK1BA8A2403790’, 33, 633281215, 3);

INSERT INTO PLANT VALUES (9861, ‘Ficus’, ‘lecardii’, 68, ‘5%’, 24, ‘1GKET16S426107309’, 34, 716552266, 3);

INSERT INTO PLANT VALUES (6936, ‘salix’, ‘acutifolia’, 62, ‘5%’, 25, ‘5NPE34AF8FH082224’, 35, 653634641, 3);
```

9- Storage:

```
INSERT INTO STORAGE VALUES (41,3500,31);

INSERT INTO STORAGE VALUES (42,4000,32);

INSERT INTO STORAGE VALUES (43,5000,33);

INSERT INTO STORAGE VALUES (44,3000,34);

INSERT INTO STORAGE VALUES (45,7000,35);

INSERT INTO STORAGE VALUES (46,3000,36);

INSERT INTO STORAGE VALUES (47,4000,37);

INSERT INTO STORAGE VALUES (48,6500,31);

INSERT INTO STORAGE VALUES (49,5050,39);
INSERT INTO STORAGE VALUES (50,5000,40);
```

10- Nutrient:

```
INSERT INTO NUTRIENT VALUES (2121, ‘Iron’, 913, ‘10/10/2030’, 9, 41, 200, 313369229, 100);

INSERT INTO NUTRIENT VALUES (2122, ‘Manganese’, 513, ‘12/20/2022’, 1, 42, 200, 647010725, 300);

INSERT INTO NUTRIENT VALUES (2123, ‘Boron’, 303, ‘12/12/2024’, 3, 43, 200, 633281215,300);

INSERT INTO NUTRIENT VALUES (2124, ‘Chloride’, 213, ‘11/11/2023’, 2, 44, 500, 716552266, 500);

INSERT INTO NUTRIENT VALUES (2126, ‘Calcium’, 213, ‘12/30/2023’, 2, 46, 200, 636446549, 300);

INSERT INTO NUTRIENT VALUES (2127, ‘Potassium’, 503, ‘12/31/2024’, 3, 47, 500, 653464364, 500);

INSERT INTO NUTRIENT VALUES (2128, ‘Sulfur’, 273, ‘1/1/2022’, 1, 48, 100, 128373291, 400);

INSERT INTO NUTRIENT VALUES (2129, ‘Nitrogen’, 214, ‘10/20/2023’, 2, 49, 1025, 238237842, 1025);

INSERT INTO NUTRIENT VALUES (2130, ‘Magnesium’, 55, ‘2/2/2025’, 4, 50, 500, 283282497, 500);
```

INSERT INTO NUTRIENT VALUES (2131, ‘Zinc’, 213, ‘12/12/2020’, 0, 42, 100, 313369229, 100);

11- Supplier:

INSERT INTO SUPPLIER VALUES (313369229, ‘Ruya Taliba’, ‘+961 71884141’);

INSERT INTO SUPPLIER VALUES (647010725, ‘Joe Helo’, ‘+961 81743718’);

INSERT INTO SUPPLIER VALUES (633281215, ‘Hashem Safadi’, ‘+961 81324032’);

INSERT INTO SUPPLIER VALUES (716552266, ‘Ibrahim Arayssi’, ‘+961 76324232’);

INSERT INTO SUPPLIER VALUES (653634641, ‘Firas Jabar’, ‘+961 70324132’);

INSERT INTO SUPPLIER VALUES (636446549, ‘Ellyas Mahfoos’, ‘+961 81689689’);

INSERT INTO SUPPLIER VALUES (653464364, ‘Genevieve Khoury’, ‘+961 76131611’);

INSERT INTO SUPPLIER VALUES (128373291, ‘Jomana Abo Al Hosn’, ‘+961 81323232’);

INSERT INTO SUPPLIER VALUES (238237842, ‘Iman Sleiman’, ‘+961 91424032’);

INSERT INTO SUPPLIER VALUES (283282497, ‘Ali Hareb’, ‘+961 71324245’);

12- Equipment:

```
INSERT INTO EQUIPMENT VALUES (‘ABC872902072866’, ‘Pruner’, 100, ‘Used’,
31, 48, 313369229);

INSERT INTO EQUIPMENT VALUES (‘FEA819837167660’, ‘Glimour_thumb’,
200, ‘Used’, 31, 41, 633281215);

INSERT INTO EQUIPMENT VALUES (‘ABC255829920279’, ‘FC50A_ferticart’,
100, ‘Used’, 34, 44, 313369229);

INSERT INTO EQUIPMENT VALUES (‘FEA574455109197’, ‘gallon_container’, 20,
‘Available’, 36, 46, 128373291);

INSERT INTO EQUIPMENT VALUES (‘ABC445294540734’, ‘hoe’, 40, ‘Available’,
31, 41, 128373291);

INSERT INTO EQUIPMENT VALUES (‘KEG943070065519’, ‘Shovel’, 45,
‘Available’, 31, 48, 313369229);

INSERT INTO EQUIPMENT VALUES (‘HRC770764478370’, ‘Axes’,45,
‘Available’, 34, 44,283282497);

INSERT INTO EQUIPMENT VALUES (‘HRC194778195255’, ‘Trowels’, 45,
‘Available’, 31, 41,128373291);

INSERT INTO EQUIPMENT VALUES (‘KEG577302200145’, ‘cloches’, 40,
‘Available’, 34, 44, 283282497);

INSERT INTO EQUIPMENT VALUES (‘DVN418938152080’, ‘Rake’, 40,
‘Available’, 39, 49, 128373291);
```

13- Drives:

```
INSERT INTO DRIVES VALUES (1845905447, ‘1N4BL2AP8BN503925’);

INSERT INTO DRIVES VALUES (5409979193, ‘3C3CFFBR3CT340684’);

INSERT INTO DRIVES VALUES (1845905447, ‘4V4NC9TK36N430345’);

INSERT INTO DRIVES VALUES (5409979193, ‘1N4BL2AP8BN503925’);

INSERT INTO DRIVES VALUES (1845905447, ‘JTJZK1BA8A2403790’);

INSERT INTO DRIVES VALUES (5409979193, ‘4V4NC9TK36N430345’);

INSERT INTO DRIVES VALUES (2074524800, ‘JTJZK1BA8A2403790’);

INSERT INTO DRIVES VALUES (2074524800, ‘3C3CFFBR3CT340684’);

INSERT INTO DRIVES VALUES (2074524800, ‘2HNYD18753H504973’);

INSERT INTO DRIVES VALUES (2074524800, ‘1N4BL2AP8BN503925’);
```

14- PURCHASES_FROM

```
INSERT INTO PURCHASES_FROM VALUES (58147405, 21, 2);

INSERT INTO PURCHASES_FROM VALUES (84397160, 22, 1);

INSERT INTO PURCHASES_FROM VALUES (95747451, 23, 3);

INSERT INTO PURCHASES_FROM VALUES (61996164, 24, 4);
INSERT INTO PURCHASES_FROM VALUES (17664996, 24, 1);

INSERT INTO PURCHASES_FROM VALUES (42284032, 25, 2);

INSERT INTO PURCHASES_FROM VALUES (94405296, 25, 5);

INSERT INTO PURCHASES_FROM VALUES (58019557, 25, 1);

INSERT INTO PURCHASES_FROM VALUES (63679144, 28, 3);

INSERT INTO PURCHASES_FROM VALUES (97303832, 29, 7);
```

15- UTILIZES

```
INSERT INTO UTILIZES VALUES (6784705218, ‘ABC872902072866’);

INSERT INTO UTILIZES VALUES (6784705218, ‘FEA819837167660’);

INSERT INTO UTILIZES VALUES (6784705218, ‘KEG943070065519’);

INSERT INTO UTILIZES VALUES (6784705218, ‘ABC445294540734’);

INSERT INTO UTILIZES VALUES (7487344706, ‘ABC255829920279’);

INSERT INTO UTILIZES VALUES (7487344706, ‘HRC770764478370’);

INSERT INTO UTILIZES VALUES (7487344706, ‘KEG577302200145’);

INSERT INTO UTILIZES VALUES (8610870297, ‘ABC255829920279’);

INSERT INTO UTILIZES VALUES (8610870297, ‘KEG577302200145’);
INSERT INTO UTILIZES VALUES (8610870297, ‘HRC770764478370’);
```

16- REQUIRES:

```
INSERT INTO REQUIRES VALUES (9091, 2121, ‘100mg’);

INSERT INTO REQUIRES VALUES (9100, 2122, ‘130mg’);

INSERT INTO REQUIRES VALUES (6996, 2127, ‘123mg’);

INSERT INTO REQUIRES VALUES (9091, 2128, ‘90mg’);

INSERT INTO REQUIRES VALUES (6496, 2130, ‘80mg’);

INSERT INTO REQUIRES VALUES (8655, 2122, ‘120mg’);

INSERT INTO REQUIRES VALUES (9100, 2131, ‘110mg’);

INSERT INTO REQUIRES VALUES (9861, 2124, ‘130mg’);

INSERT INTO REQUIRES VALUES (6493, 2121, ‘100mg’);

INSERT INTO REQUIRES VALUES (6493, 2128, ‘80mg’);
```

17- CUST_PURCHASE_DATES:

```
INSERT INTO CUST_PURCHASE_DATES VALUES (58147405, ‘07-11-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (84397160, ‘09-17-2021’);
INSERT INTO CUST_PURCHASE_DATES VALUES (95747451, ‘09-17-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (61996164, ‘09-20-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (17664996, ‘09-27-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (42284032, ‘03-08-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (94405296, ‘08-17-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (58019557, ‘09-04-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (63679144, ‘07-21-2021’);

INSERT INTO CUST_PURCHASE_DATES VALUES (97303832, ‘08-28-2021’);
```

18- EMP_PHONE_NUMBER:

```
INSERT INTO EMP_PHONE_NUMBER VALUES (6784705218, ‘+961 76-575180’);

INSERT INTO EMP_PHONE_NUMBER VALUES (4349797788, ‘+961 09-795922’);

INSERT INTO EMP_PHONE_NUMBER VALUES (5409979193, ‘+961 76-257312’);

INSERT INTO EMP_PHONE_NUMBER VALUES (6963258256, ‘+961 09-159632’);

INSERT INTO EMP_PHONE_NUMBER VALUES (1117618988, ‘+961 03-886442’);

INSERT INTO EMP_PHONE_NUMBER VALUES (2074524800, ‘+961 03-987543’);

INSERT INTO EMP_PHONE_NUMBER VALUES (1845905447, ‘+961 76-014532’);
INSERT INTO EMP_PHONE_NUMBER VALUES (2307411820, ‘+961 09-358324’);

INSERT INTO EMP_PHONE_NUMBER VALUES (7487344706, ‘+961 76-575191’);

INSERT INTO EMP_PHONE_NUMBER VALUES (8610870297, ‘+961 76-757919’);
```

19- SUPP_SUPPLY_DATES:

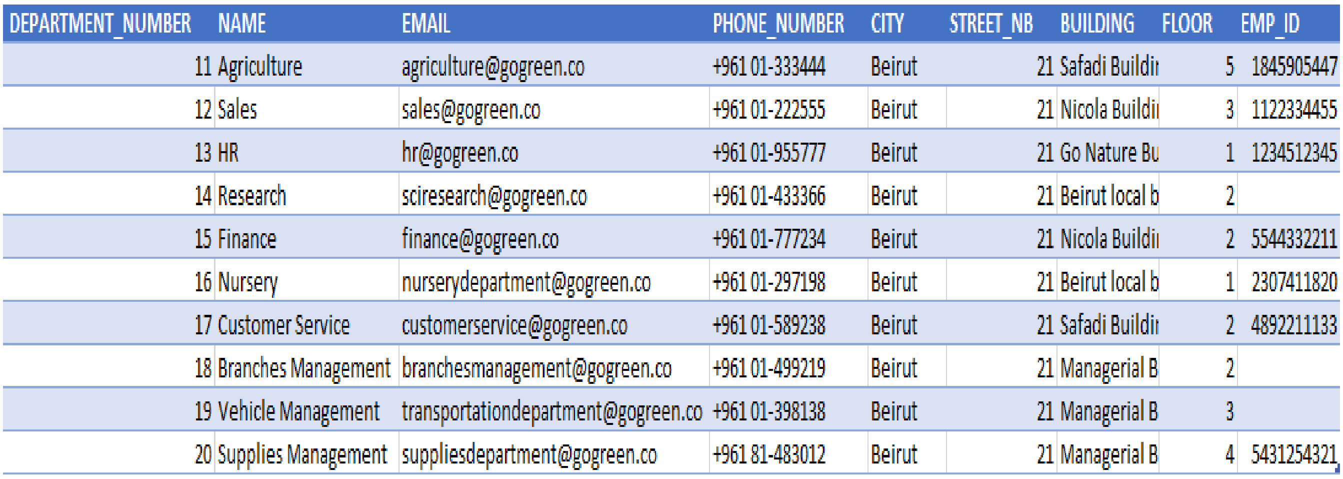
```
INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(313369229, ‘07-11-2021’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(313369229, ‘04-11-2021’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(633281215, ‘22-09-2020’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(313369229, ‘19-03-2020’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(653464364, ‘21-06-2019’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(633281215, ‘10-10-2020’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(653464364, ‘11-11-2019’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES
(283282497, ‘05-12-2020’);
```

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES (283282497, ‘15-01-2021’);

INSERT INTO SUPP_SUPPLY_DATES (SUPP_ID, SUPPLY_DATES) VALUES (283282497, ‘13-04-2021’);

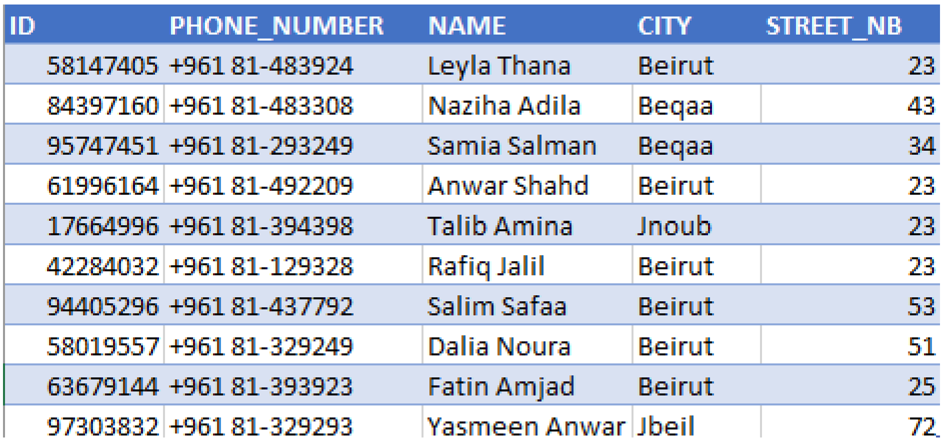
M- Final Tables State:

1- DEPARTMENT:



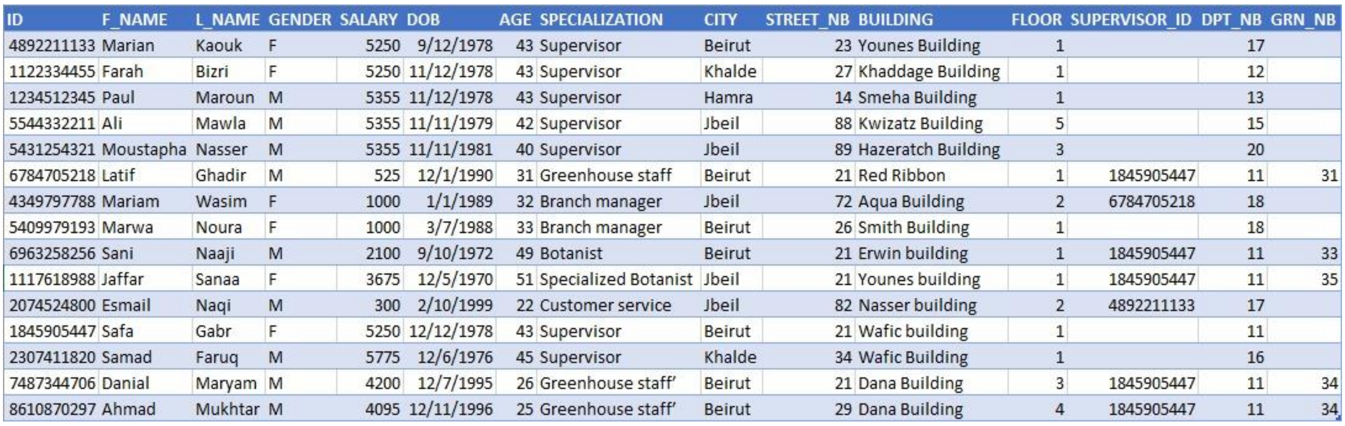
DEPARTMENT_NUMBER	NAME	EMAIL	PHONE_NUMBER	CITY	STREET_NB	BUILDING	FLOOR	EMP_ID
11	Agriculture	agriculture@gogreen.co	+961 01-333444	Beirut	21	Safadi Buildi	5	1845905447
12	Sales	sales@gogreen.co	+961 01-222555	Beirut	21	Nicola Buildi	3	1122334455
13	HR	hr@gogreen.co	+961 01-955777	Beirut	21	Go Nature Bu	1	1234512345
14	Research	sciresearch@gogreen.co	+961 01-433366	Beirut	21	Beirut local b	2	
15	Finance	finance@gogreen.co	+961 01-777234	Beirut	21	Nicola Buildi	2	5544332211
16	Nursery	nurserydepartment@gogreen.co	+961 01-297198	Beirut	21	Beirut local b	1	2307411820
17	Customer Service	customerservice@gogreen.co	+961 01-589238	Beirut	21	Safadi Buildi	2	4892211133
18	Branches Management	branchesmanagement@gogreen.co	+961 01-499219	Beirut	21	Managerial B	2	
19	Vehicle Management	transportationdepartment@gogreen.co	+961 01-398138	Beirut	21	Managerial B	3	
20	Supplies Management	suppliesdepartment@gogreen.co	+961 81-483012	Beirut	21	Managerial B	4	5431254321

2- CUSTOMER:



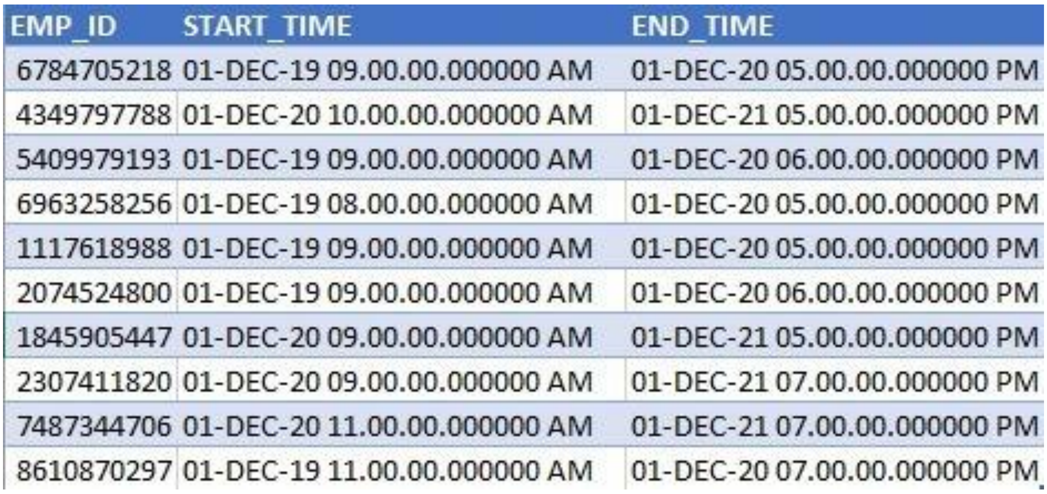
ID	PHONE_NUMBER	NAME	CITY	STREET_NB
58147405	+961 81-483924	Leyla Thana	Beirut	23
84397160	+961 81-483308	Naziha Adila	Beqaa	43
95747451	+961 81-293249	Samia Salman	Beqaa	34
61996164	+961 81-492209	Anwar Shahd	Beirut	23
17664996	+961 81-394398	Talib Amina	Jnoub	23
42284032	+961 81-129328	Rafiq Jalil	Beirut	23
94405296	+961 81-437792	Salim Safaa	Beirut	53
58019557	+961 81-329249	Dalia Noura	Beirut	51
63679144	+961 81-393923	Fatin Amjad	Beirut	25
97303832	+961 81-329293	Yasmeen Anwar	Jbeil	72

3- EMPLOYEE:



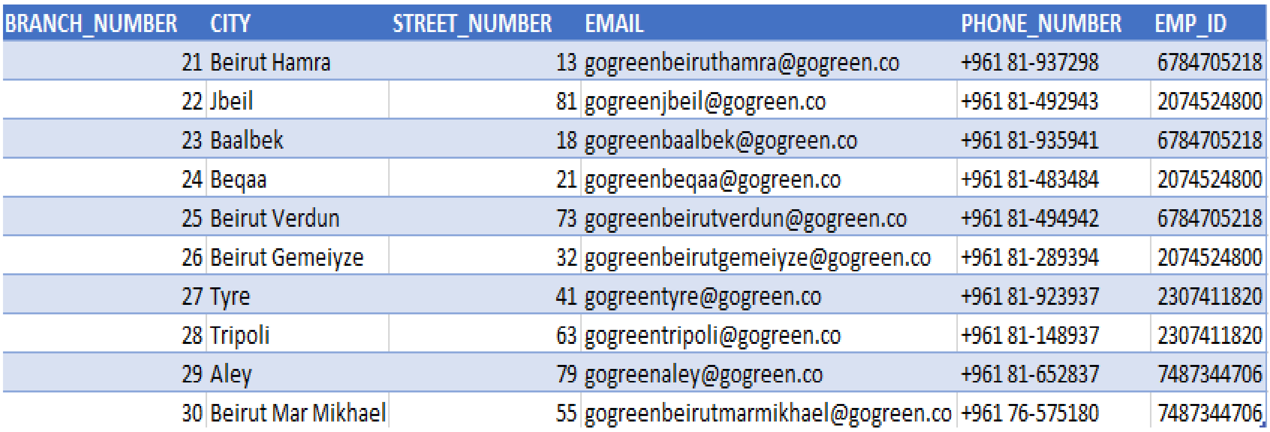
ID	F_NAME	L_NAME	GENDER	SALARY	DOB	AGE	SPECIALIZATION	CITY	STREET_NB	BUILDING	FLOOR	SUPERVISOR_ID	DPT_NB	GRN_NB
4892211133	Marian	Kaouk	F	5250	9/12/1978	43	Supervisor	Beirut	23	Younes Building	1		17	
1122334455	Farah	Bizri	F	5250	11/12/1978	43	Supervisor	Khalde	27	Khaddage Building	1		12	
1234512345	Paul	Maroun	M	5355	11/12/1978	43	Supervisor	Hamra	14	Smeha Building	1		13	
5544332211	Ali	Mawla	M	5355	11/11/1979	42	Supervisor	Jbeil	88	Kwizatz Building	5		15	
5431254321	Moustapha	Nasser	M	5355	11/11/1981	40	Supervisor	Jbeil	89	Hazeratch Building	3		20	
6784705218	Latif	Ghadir	M	525	12/1/1990	31	Greenhouse staff	Beirut	21	Red Ribbon	1	1845905447	11	31
4349797788	Mariam	Wasim	F	1000	1/1/1989	32	Branch manager	Jbeil	72	Aqua Building	2	6784705218	18	
5409979193	Marwa	Noura	F	1000	3/7/1988	33	Branch manager	Beirut	26	Smith Building	1		18	
6963258256	Sani	Naaji	M	2100	9/10/1972	49	Botanist	Beirut	21	Erwin building	1	1845905447	11	33
1117618988	Jaffar	Sanaa	F	3675	12/5/1970	51	Specialized Botanist	Jbeil	21	Younes building	1	1845905447	11	35
2074524800	Esmail	Naqi	M	300	2/10/1999	22	Customer service	Jbeil	82	Nasser building	2	4892211133	17	
1845905447	Safa	Gabr	F	5250	12/12/1978	43	Supervisor	Beirut	21	Wafic building	1		11	
2307411820	Samad	Faruq	M	5775	12/6/1976	45	Supervisor	Khalde	34	Wafic Building	1		16	
7487344706	Danial	Maryam	M	4200	12/7/1995	26	Greenhouse staff’	Beirut	21	Dana Building	3	1845905447	11	34
8610870297	Ahmad	Mukhtar	M	4095	12/11/1996	25	Greenhouse staff’	Beirut	29	Dana Building	4	1845905447	11	34

4- SCHEDULE:



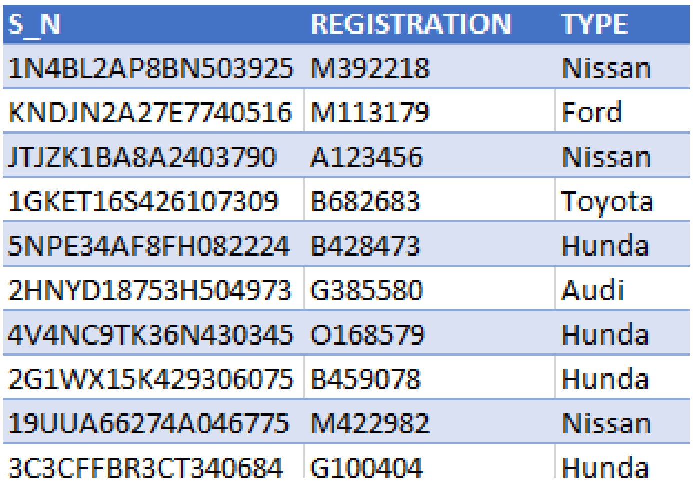
EMP_ID	START_TIME	END_TIME
6784705218	01-DEC-19 09.00.00.000000 AM	01-DEC-20 05.00.00.000000 PM
4349797788	01-DEC-20 10.00.00.000000 AM	01-DEC-21 05.00.00.000000 PM
5409979193	01-DEC-19 09.00.00.000000 AM	01-DEC-20 06.00.00.000000 PM
6963258256	01-DEC-19 08.00.00.000000 AM	01-DEC-20 05.00.00.000000 PM
1117618988	01-DEC-19 09.00.00.000000 AM	01-DEC-20 05.00.00.000000 PM
2074524800	01-DEC-19 09.00.00.000000 AM	01-DEC-20 06.00.00.000000 PM
1845905447	01-DEC-20 09.00.00.000000 AM	01-DEC-21 05.00.00.000000 PM
2307411820	01-DEC-20 09.00.00.000000 AM	01-DEC-21 07.00.00.000000 PM
7487344706	01-DEC-20 11.00.00.000000 AM	01-DEC-21 07.00.00.000000 PM
8610870297	01-DEC-19 11.00.00.000000 AM	01-DEC-20 07.00.00.000000 PM

5- BRANCH:



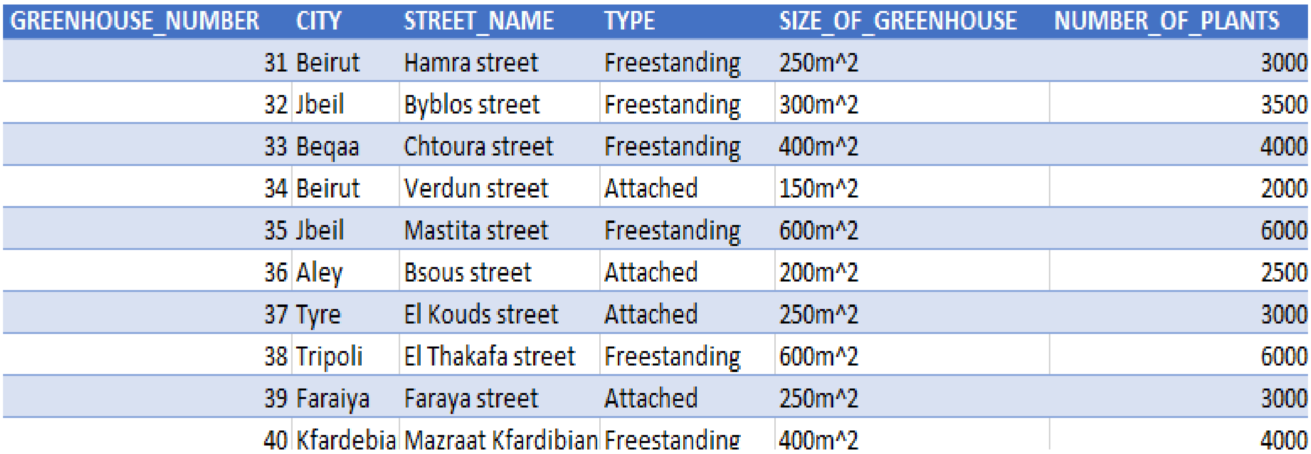
BRANCH_NUMBER	CITY	STREET_NUMBER	EMAIL	PHONE_NUMBER	EMP_ID
21	Beirut Hamra	13	gogreenbeiruthamra@gogreen.co	+961 81-937298	6784705218
22	Jbeil	81	gogreenjbeil@gogreen.co	+961 81-492943	2074524800
23	Baalbek	18	gogreenbaalbek@gogreen.co	+961 81-935941	6784705218
24	Beqaa	21	gogreenbeqaa@gogreen.co	+961 81-483484	2074524800
25	Beirut Verdun	73	gogreenbeirutverdun@gogreen.co	+961 81-494942	6784705218
26	Beirut Gemeiyze	32	gogreenbeirutgemeiyze@gogreen.co	+961 81-289394	2074524800
27	Tyre	41	gogreentyre@gogreen.co	+961 81-923937	2307411820
28	Tripoli	63	gogreentripoli@gogreen.co	+961 81-148937	2307411820
29	Aley	79	gogreenaley@gogreen.co	+961 81-652837	7487344706
30	Beirut Mar Mikhael	55	gogreenbeirutmarmikhael@gogreen.co	+961 76-575180	7487344706

6- VEHICLE:



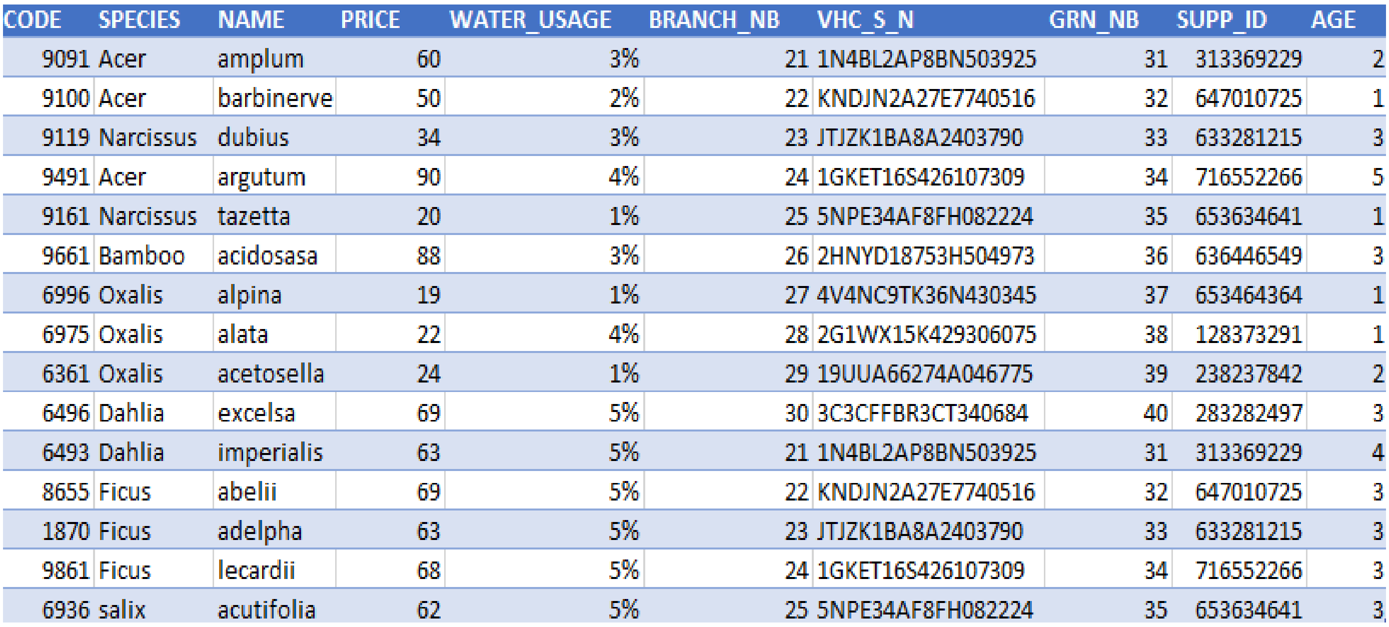
S_N	REGISTRATION	TYPE
1N4BL2AP8BN503925	M392218	Nissan
KNDJN2A27E7740516	M113179	Ford
JTJZK1BA8A2403790	A123456	Nissan
1GKET16S426107309	B682683	Toyota
5NPE34AF8FH082224	B428473	Hunda
2HNYD18753H504973	G385580	Audi
4V4NC9TK36N430345	O168579	Hunda
2G1WX15K429306075	B459078	Hunda
19UUA66274A046775	M422982	Nissan
3C3CFFBR3CT340684	G100404	Hunda

7- GREENHOUSE:



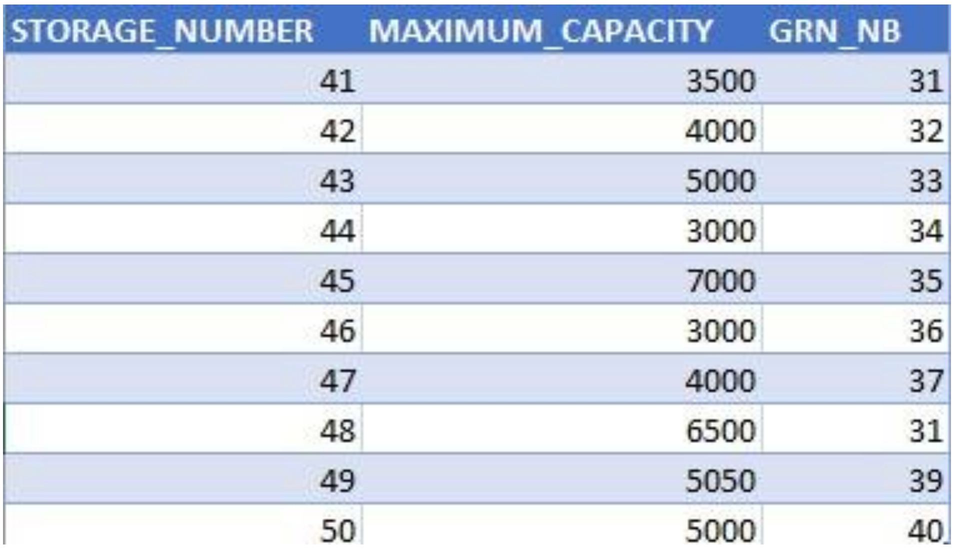
GREENHOUSE_NUMBER	CITY	STREET_NAME	TYPE	SIZE_OF_GREENHOUSE	NUMBER_OF_PLANTS
31	Beirut	Hamra street	Freestanding	250m^2	3000
32	Jbeil	Byblos street	Freestanding	300m^2	3500
33	Beqaa	Chtoura street	Freestanding	400m^2	4000
34	Beirut	Verdun street	Attached	150m^2	2000
35	Jbeil	Mastita street	Freestanding	600m^2	6000
36	Aley	Bsous street	Attached	200m^2	2500
37	Tyre	El Kouds street	Attached	250m^2	3000
38	Tripoli	El Thakafa street	Freestanding	600m^2	6000
39	Faraiya	Faraya street	Attached	250m^2	3000
40	Kfardebia	Mazraat Kfardibian	Freestanding	400m^2	4000

8- PLANT:



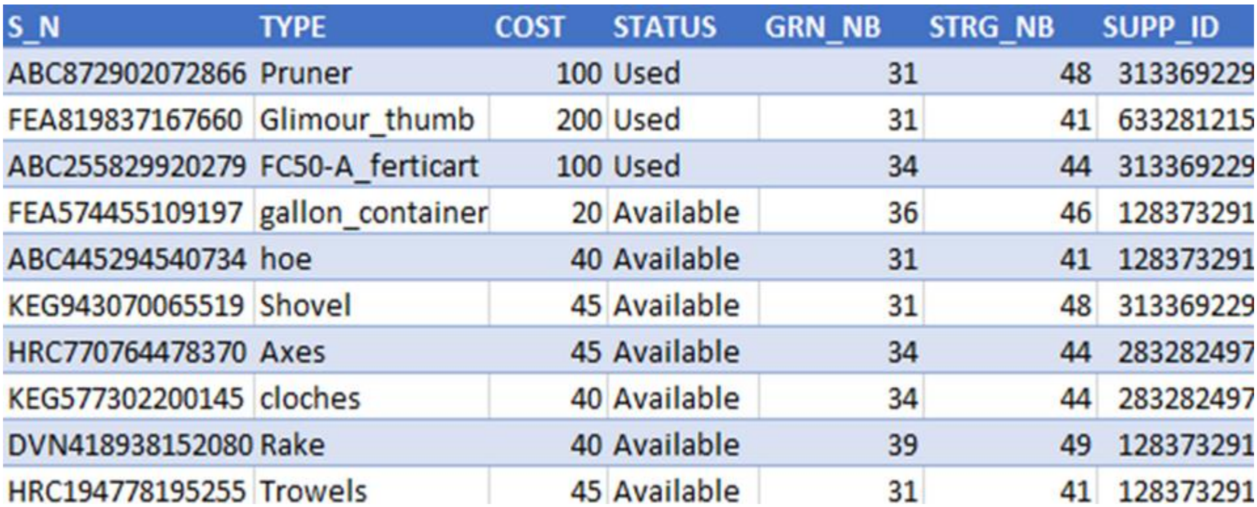
CODE	SPECIES	NAME	PRICE	WATER_USAGE	BRANCH_NB	VHC_S_N	GRN_NB	SUPP_ID	AGE
9091	Acer	amplum	60	3%	21	1N4BL2AP8BN503925	31	313369229	2
9100	Acer	barbinerve	50	2%	22	KNDJN2A27E7740516	32	647010725	1
9119	Narcissus	dubius	34	3%	23	JTJZK1BA8A2403790	33	633281215	3
9491	Acer	argutum	90	4%	24	1GKET16S426107309	34	716552266	5
9161	Narcissus	tazetta	20	1%	25	5NPE34AF8FH082224	35	653634641	1
9661	Bamboo	acidosasa	88	3%	26	2HNYD18753H504973	36	636446549	3
6996	Oxalis	alpina	19	1%	27	4V4NC9TK36N430345	37	653464364	1
6975	Oxalis	alata	22	4%	28	2G1WX15K429306075	38	128373291	1
6361	Oxalis	acetosella	24	1%	29	19UUA66274A046775	39	238237842	2
6496	Dahlia	excelsa	69	5%	30	3C3CFFBR3CT340684	40	283282497	3
6493	Dahlia	imperialis	63	5%	21	1N4BL2AP8BN503925	31	313369229	4
8655	Ficus	abelii	69	5%	22	KNDJN2A27E7740516	32	647010725	3
1870	Ficus	adelpha	63	5%	23	JTJZK1BA8A2403790	33	633281215	3
9861	Ficus	lecardii	68	5%	24	1GKET16S426107309	34	716552266	3
6936	salix	acutifolia	62	5%	25	5NPE34AF8FH082224	35	653634641	3

9- STORAGE:



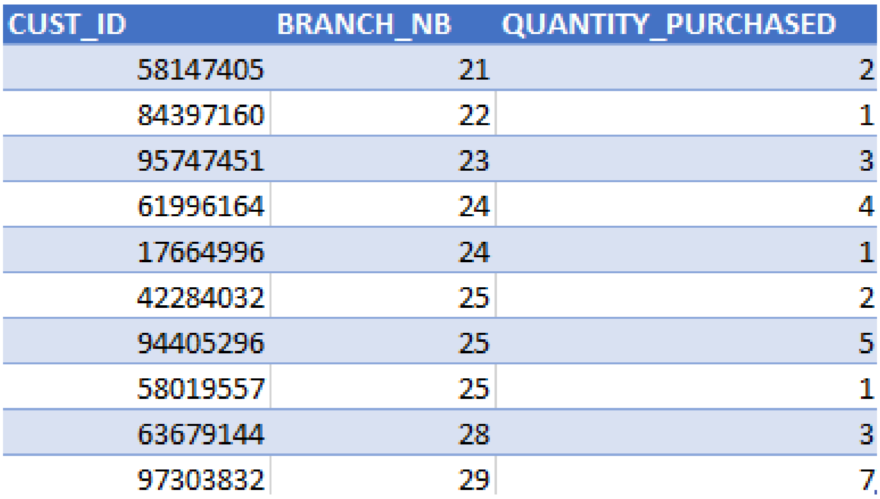
STORAGE_NUMBER	MAXIMUM_CAPACITY	GRN_NB
41	3500	31
42	4000	32
43	5000	33
44	3000	34
45	7000	35
46	3000	36
47	4000	37
48	6500	31
49	5050	39
50	5000	40

10- EQUIPMENT:



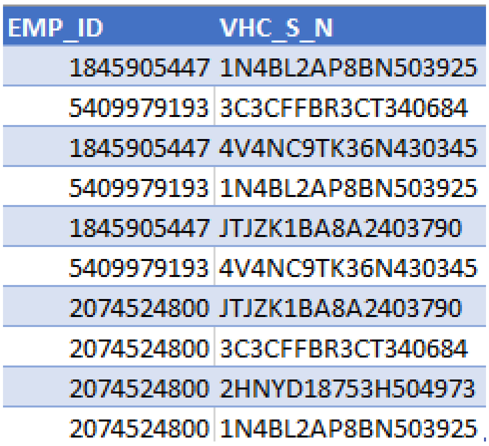
S_N	TYPE	COST	STATUS	GRN_NB	STRG_NB	SUPP_ID
ABC872902072866	Pruner	100	Used	31	48	313369229
FEA819837167660	Glimour_thumb	200	Used	31	41	633281215
ABC255829920279	FC50-A_ferticart	100	Used	34	44	313369229
FEA574455109197	gallon_container	20	Available	36	46	128373291
ABC445294540734	hoe	40	Available	31	41	128373291
KEG943070065519	Shovel	45	Available	31	48	313369229
HRC770764478370	Axes	45	Available	34	44	283282497
KEG577302200145	cloches	40	Available	34	44	283282497
DVN418938152080	Rake	40	Available	39	49	128373291
HRC194778195255	Trowels	45	Available	31	41	128373291

11- PURCHASED_FROM:



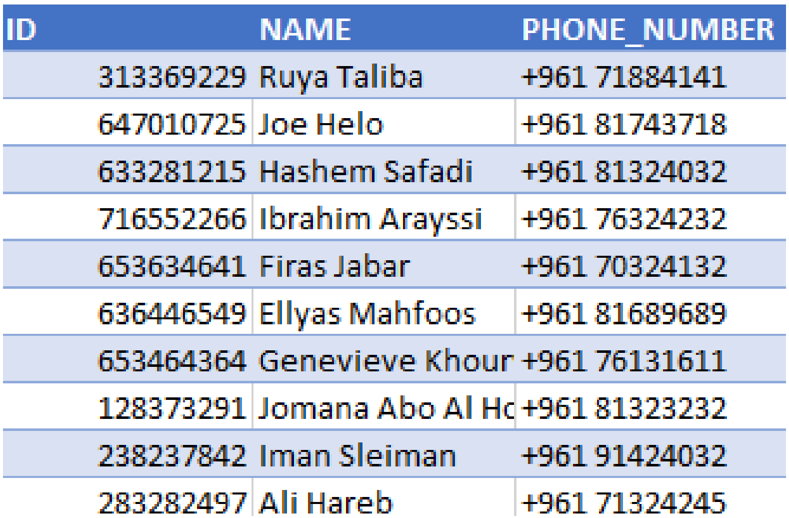
CUST_ID	BRANCH_NB	QUANTITY_PURCHASED
58147405	21	2
84397160	22	1
95747451	23	3
61996164	24	4
17664996	24	1
42284032	25	2
94405296	25	5
58019557	25	1
63679144	28	3
97303832	29	7

12- DRIVES:



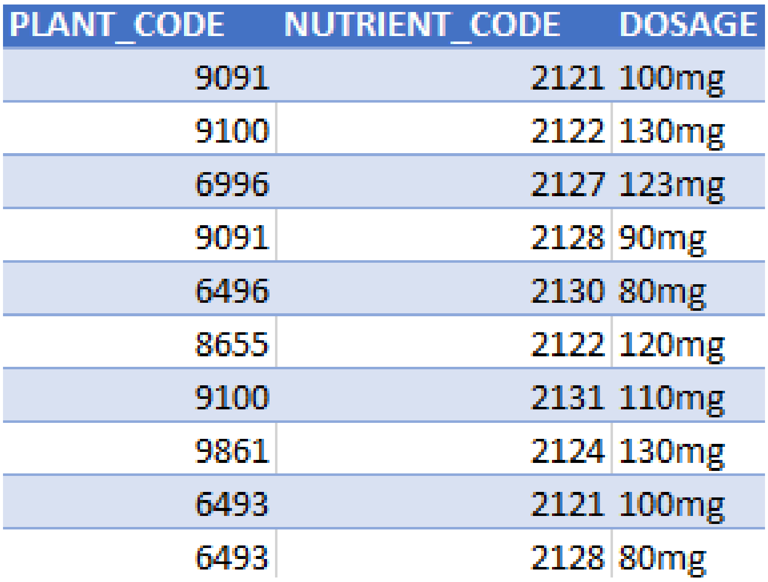
EMP_ID	VHC_S_N
1845905447	1N4BL2AP8BN503925
5409979193	3C3CFFBR3CT340684
1845905447	4V4NC9TK36N430345
5409979193	1N4BL2AP8BN503925
1845905447	JTJZK1BA8A2403790
5409979193	4V4NC9TK36N430345
2074524800	JTJZK1BA8A2403790
2074524800	3C3CFFBR3CT340684
2074524800	2HNYD18753H504973
2074524800	1N4BL2AP8BN503925

13- SUPPLIER:



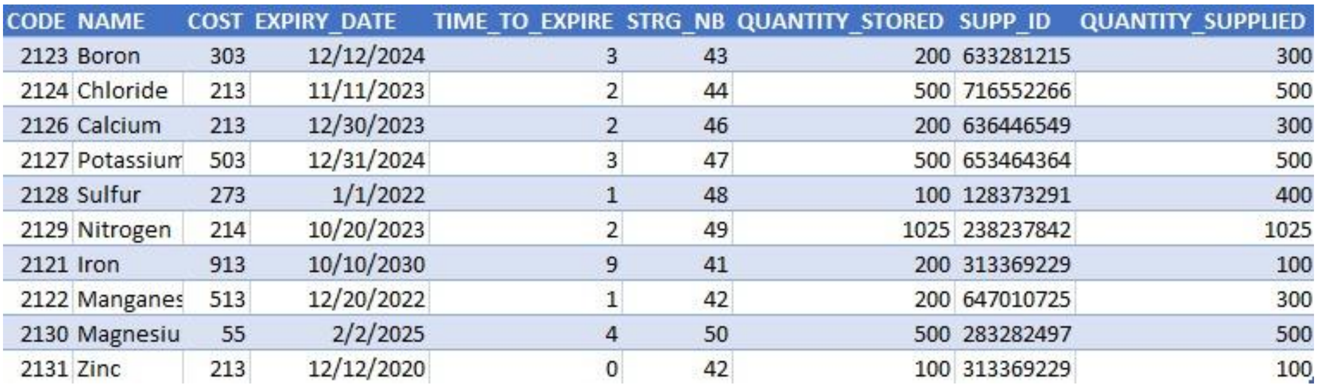
ID	NAME	PHONE_NUMBER
313369229	Ruya Taliba	+961 71884141
647010725	Joe Helo	+961 81743718
633281215	Hashem Safadi	+961 81324032
716552266	Ibrahim Arayssi	+961 76324232
653634641	Firas Jabar	+961 70324132
636446549	Ellyas Mahfoos	+961 81689689
653464364	Genevieve Khour	+961 76131611
128373291	Jomana Abo Al Hc	+961 81323232
238237842	Iman Sleiman	+961 91424032
283282497	Ali Hareb	+961 71324245

14- REQUIRES:



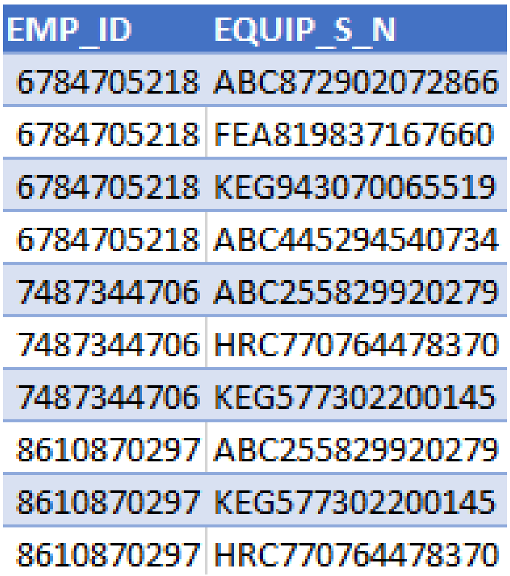
PLANT_CODE	NUTRIENT_CODE	DOSAGE
9091	2121	100mg
9100	2122	130mg
6996	2127	123mg
9091	2128	90mg
6496	2130	80mg
8655	2122	120mg
9100	2131	110mg
9861	2124	130mg
6493	2121	100mg
6493	2128	80mg

15- NUTRIENT:



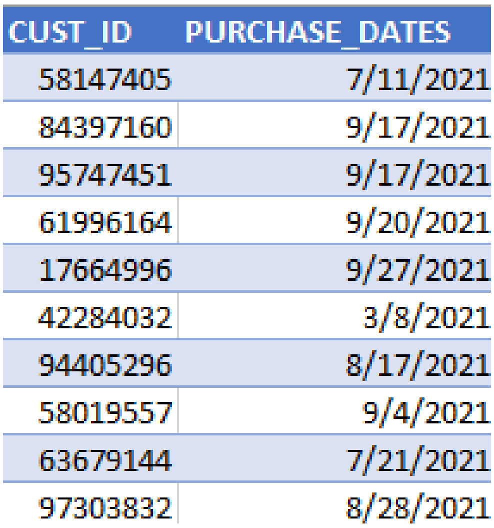
CODE	NAME	COST	EXPIRY_DATE	TIME_TO_EXPIRE	STRG_NB	QUANTITY_STORED	SUPP_ID	QUANTITY_SUPPLIED
2123	Boron	303	12/12/2024	3	43	200	633281215	300
2124	Chloride	213	11/11/2023	2	44	500	716552266	500
2126	Calcium	213	12/30/2023	2	46	200	636446549	300
2127	Potassium	503	12/31/2024	3	47	500	653464364	500
2128	Sulfur	273	1/1/2022	1	48	100	128373291	400
2129	Nitrogen	214	10/20/2023	2	49	1025	238237842	1025
2121	Iron	913	10/10/2030	9	41	200	313369229	100
2122	Manganes	513	12/20/2022	1	42	200	647010725	300
2130	Magnesiu	55	2/2/2025	4	50	500	283282497	500
2131	Zinc	213	12/12/2020	0	42	100	313369229	100

16.UTILIZES:



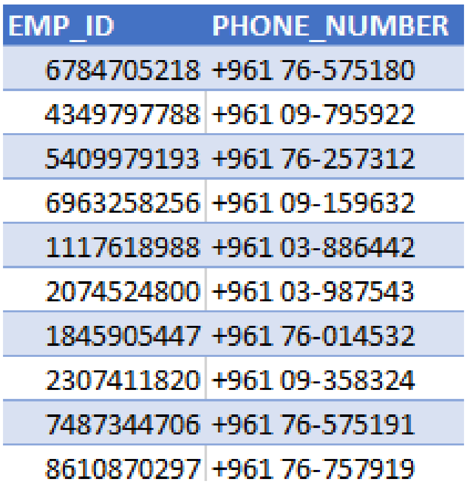
EMP_ID	EQUIP_S_N
6784705218	ABC872902072866
6784705218	FEA819837167660
6784705218	KEG943070065519
6784705218	ABC445294540734
7487344706	ABC255829920279
7487344706	HRC770764478370
7487344706	KEG577302200145
8610870297	ABC255829920279
8610870297	KEG577302200145
8610870297	HRC770764478370

17. CUST_PURCHASE_DATES:



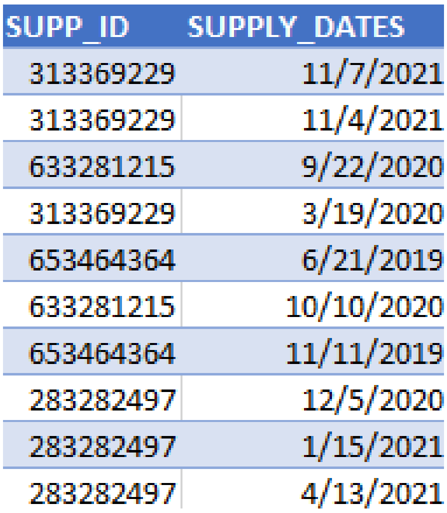
CUST_ID	PURCHASE_DATES
58147405	7/11/2021
84397160	9/17/2021
95747451	9/17/2021
61996164	9/20/2021
17664996	9/27/2021
42284032	3/8/2021
94405296	8/17/2021
58019557	9/4/2021
63679144	7/21/2021
97303832	8/28/2021

18.EMP_PHONE_NUMBER:



EMP_ID	PHONE_NUMBER
6784705218	+961 76-575180
4349797788	+961 09-795922
5409979193	+961 76-257312
6963258256	+961 09-159632
1117618988	+961 03-886442
2074524800	+961 03-987543
1845905447	+961 76-014532
2307411820	+961 09-358324
7487344706	+961 76-575191
8610870297	+961 76-757919

19. SUPP_SUPPLY_DATES:



SUPP_ID	SUPPLY_DATES
313369229	11/7/2021
313369229	11/4/2021
633281215	9/22/2020
313369229	3/19/2020
653464364	6/21/2019
633281215	10/10/2020
653464364	11/11/2019
283282497	12/5/2020
283282497	1/15/2021
283282497	4/13/2021

N-Queries:

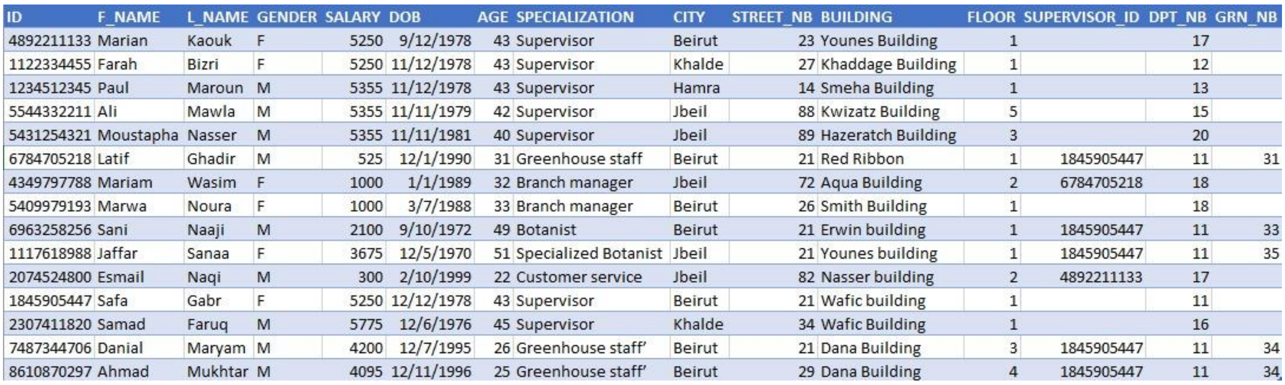
Query 1:

The nursery recently made a lot of profit and decided to award its hard-working employees. Increase the salary by 5% for all employees working in the ‘Agriculture’ department or supervising other employees.

SQL Query:

```
UPDATE EMPLOYEEE
SET SALARY=1.05*Salary
WHERE ID IN (SELECT E.ID
                          FROM EMPLOYEE AS E, DEPARTMENT AS D
                         WHERE D.Name=’Agriculture’ AND
                         E.DPT_Nb=D.Department_number)
UNION (SELECT E.ID
               FROM EMPOYEE as E
               WHERE E. Specialization=’Supervisor’);
```

OUTPUT:



ID	F_NAME	L_NAME	GENDER	SALARY	DOB	AGE	SPECIALIZATION	CITY	STREET_NB	BUILDING	FLOOR	SUPERVISOR_ID	DPT_NB	GRN_NB
4892211133	Marian	Kaouk	F	5250	9/12/1978	43	Supervisor	Beirut	23	Younes Building	1		17	
1122334455	Farah	Bizri	F	5250	11/12/1978	43	Supervisor	Khalde	27	Khaddage Building	1		12	
1234512345	Paul	Maroun	M	5355	11/12/1978	43	Supervisor	Hamra	14	Smeha Building	1		13	
5544332211	Ali	Mawla	M	5355	11/11/1979	42	Supervisor	Jbeil	88	Kwizatz Building	5		15	
5431254321	Moustapha	Nasser	M	5355	11/11/1981	40	Supervisor	Jbeil	89	Hazeratch Building	3		20	
6784705218	Latif	Ghadir	M	525	12/1/1990	31	Greenhouse staff	Beirut	21	Red Ribbon	1	1845905447	11	31
4349797788	Mariam	Wasim	F	1000	1/1/1989	32	Branch manager	Jbeil	72	Aqua Building	2	6784705218	18	
5409979193	Marwa	Noura	F	1000	3/7/1988	33	Branch manager	Beirut	26	Smith Building	1		18	
6963258256	Sani	Naaji	M	2100	9/10/1972	49	Botanist	Beirut	21	Erwin building	1	1845905447	11	33
1117618988	Jaffar	Sanaa	F	3675	12/5/1970	51	Specialized Botanist	Jbeil	21	Younes building	1	1845905447	11	35
2074524800	Esmail	Naqi	M	300	2/10/1999	22	Customer service	Jbeil	82	Nasser building	2	4892211133	17	
1845905447	Safa	Gabr	F	5250	12/12/1978	43	Supervisor	Beirut	21	Wafic building	1		11	
2307411820	Samad	Faruq	M	5775	12/6/1976	45	Supervisor	Khalde	34	Wafic Building	1		16	
7487344706	Danial	Maryam	M	4200	12/7/1995	26	Greenhouse staff’	Beirut	21	Dana Building	3	1845905447	11	34
8610870297	Ahmad	Mukhtar	M	4095	12/11/1996	25	Greenhouse staff’	Beirut	29	Dana Building	4	1845905447	11	34

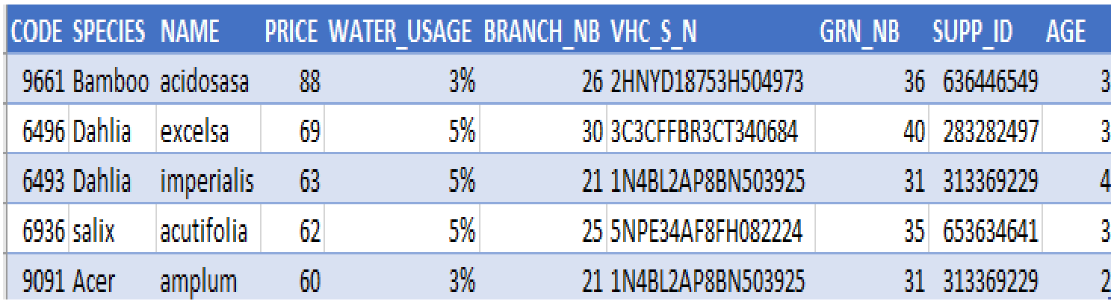
Query 2:

A customer wanted to buy a plant from a branch in Beirut but had a specific budget. He cannot go over 100$ and is willing to pay at least 60$ for a plant. Retrieve all the information about plants which are stored in branches located in Beirut and whose prices ranges between 60 and 100 dollars.

SQL Query:

```
SELECT P.*
FROM PLANT AS P, BRANCH AS B
WHERE B.City LIKE ‘Beirut%’ AND P.BRANCH_Nb=B.Branch_number AND
(P.Price BETWEEN 60 AND 100);
```

OUTPUT:



CODE	SPECIES	NAME	PRICE	WATER_USAGE	BRANCH_NB	VHC_S_N	GRN_NB	SUPP_ID	AGE
9661	Bamboo	acidosasa	88	3%	26	2HNYD18753H504973	36	636446549	3
6496	Dahlia	excelsa	69	5%	30	3C3CFFBR3CT340684	40	283282497	3
6493	Dahlia	imperialis	63	5%	21	1N4BL2AP8BN503925	31	313369229	4
6936	salix	acutifolia	62	5%	25	5NPE34AF8FH082224	35	653634641	3
9091	Acer	amplum	60	3%	21	1N4BL2AP8BN503925	31	313369229	2

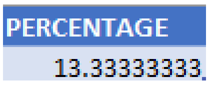
Query 3:

Knowing how many plants need nutrient helps identify the amount of nutrients that the plant nursery must buy. Return the percentage of plants that require iron to grow.

SQL Query:

```
SELECT 100*COUNT(*)/(SELECT COUNT(*) FROM PLANT) AS Percentage
FROM PLANT AS P, REQUIRES AS R, NUTRIENT AS N
WHERE R.Plant_Code=P.Code AND R.Nutrient_Code=N.Code AND
N.Name=’Iron’
GROUP BY N.Name;
```

OUTPUT:



PERCENTAGE
13.33333333

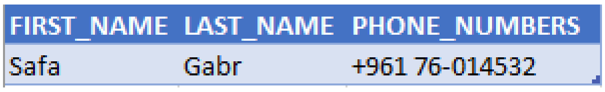
Query 4:

The plant nursery needed urgently to contact the employees in ‘HR’ department. So, their supervisors were responsible for contacting them. However, some of them are not supervised by other employees. Retrieve the phone numbers and names of all employees working in ‘HR’ and are not supervised by other employees.

SQL Query:

```
SELECT E.F_Name AS First Name, E.L_Name AS Last Name, PN. Phone_number
AS Phone numbers
FROM EMPLOYEE AS E, EMP_PHONE_NUMBER AS PN, DEPARTMENT AS
D
WHERE E.DPT_Nb= D.Department_number AND D.Name=’HR’ AND E.ID=PN.
EMP_ID AND E. Supervisor_ID IS NULL;
```

OUTPUT:



FIRST_NAME	LAST_NAME	PHONE_NUMBERS
Safa	Gabr	+961 76-014532

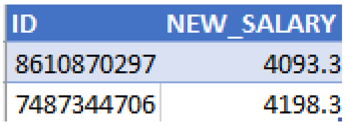
Query 5:

The equipment with serial numbers ‘KEG577302200145’ and ‘HRC770764478370’ were found to be stolen. So, the nursery decided to punish all employees who recently used both equipment. Show the expected decrease in salary of employees who used both the equipment if the decrease is equal to 2% of the sum of cost of both equipment.

SQL Query:

```
SELECT DISTINCT E.ID AS ID,  E.Salary-0.02*((SELECT Cost FROM
EQUIPMENT WHERE S_N= 'KEG577302200145')+( SELECT Cost FROM
EQUIPMENT WHERE S_N= 'HRC770764478370')) as New_Salary
FROM EMPLOYEE E, EQUIPMENT
WHERE E.ID IN (SELECT UTILIZES.EMP_ID
FROM UTILIZES
WHERE  EQUIP_S_N= 'HRC770764478370'
INTERSECT
SELECT UTILIZES.EMP_ID
FROM UTILIZES
WHERE EQUIP_S_N= 'KEG577302200145') AND (EQUIPMENT.S_N=
'KEG577302200145' OR EQUIPMENT.S_N= 'HRC770764478370');
```

OUTPUT:



ID	NEW_SALARY
8610870297	4093.3
7487344706	4198.3

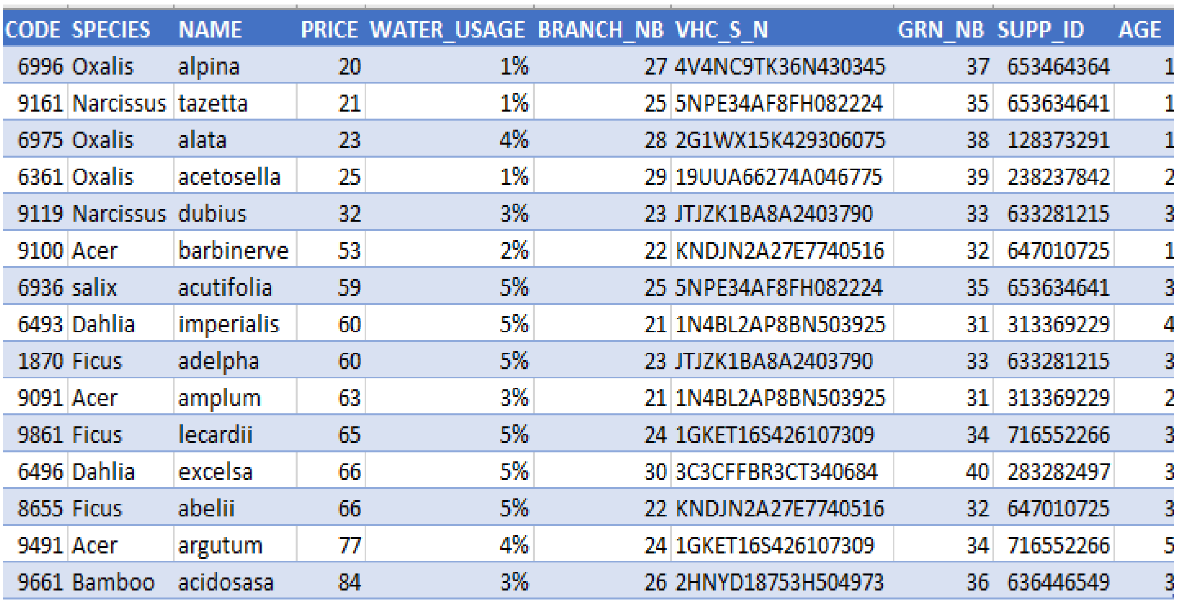
Query 6:

Young plants are destined to live longer than the old ones; thus, decreasing the price of old plants should be done in the branches so that customers would buy them before they die. Increase the price of plants ranging between 0 and 2 by 5%. Decrease the price of plants ranging between 3 and 4 by 5%. Decrease the price of plants ranging between 5 and 10 by 15%.

SQL Query:

```
UPDATE PLANT
SET Cost =
CASE WHEN Age BETWEEN 0 AND 2 THEN 1.05*Cost
          WHEN Age BETWEEN 3 AND 4 THEN 0.95*Cost
          WHEN Age BETWEEN 5 AND 7 THEN 0.85*Cost;
```

OUTPUT:



CODE	SPECIES	NAME	PRICE	WATER_USAGE	BRANCH_NB	VHC_S_N	GRN_NB	SUPP_ID	AGE
6996	Oxalis	alpina	20	1%	27	4V4NC9TK36N430345	37	653464364	1
9161	Narcissus	tazetta	21	1%	25	5NPE34AF8FH082224	35	653634641	1
6975	Oxalis	alata	23	4%	28	2G1WX15K429306075	38	128373291	1
6361	Oxalis	acetosella	25	1%	29	19UUA66274A046775	39	238237842	2
9119	Narcissus	dubius	32	3%	23	JTJZK1BA8A2403790	33	633281215	3
9100	Acer	barbinerve	53	2%	22	KNDJN2A27E7740516	32	647010725	1
6936	salix	acutifolia	59	5%	25	5NPE34AF8FH082224	35	653634641	3
6493	Dahlia	imperialis	60	5%	21	1N4BL2AP8BN503925	31	313369229	4
1870	Ficus	adelpha	60	5%	23	JTJZK1BA8A2403790	33	633281215	3
9091	Acer	amplum	63	3%	21	1N4BL2AP8BN503925	31	313369229	2
9861	Ficus	lecardii	65	5%	24	1GKET16S426107309	34	716552266	3
6496	Dahlia	excelsa	66	5%	30	3C3CFFBR3CT340684	40	283282497	3
8655	Ficus	abelii	66	5%	22	KNDJN2A27E7740516	32	647010725	3
9491	Acer	argutum	77	4%	24	1GKET16S426107309	34	716552266	5
9661	Bamboo	acidosasa	84	3%	26	2HNYD18753H504973	36	636446549	3

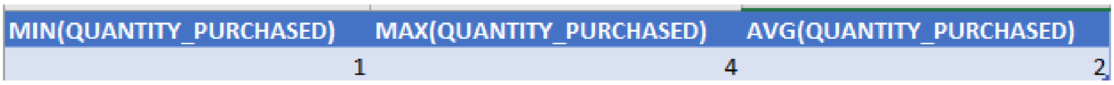
Query 7:

The nursery decided to run some statistics regarding its sales in the month of September 2021. Retrieve the max, min and average number of plants purchased by a customer in September 2021.

SQL Query:

```
 SELECT MIN(Quantity_purchased), MAX(Quantity_purchased),
AVG(Quantity_purchased)
FROM PURCHASES_FROM  P, CUST_PURCHASE_DATES D
WHERE P.CUST_ID=D.CUST_ID AND to_char(D.Purchase_dates, 'MM-
YYYY')='09-2021';
```

OUTPUT:



MIN(QUANTITY_PURCHASED)	MAX(QUANTITY_PURCHASED)	AVG(QUANTITY_PURCHASED)
1	4	2

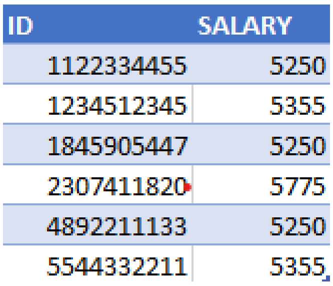
Query 8:

Due to the economic crisis, GOGREEN decided to kick some employees, so they decided to kick the old employees. To help GOGREEN with this decision, select the ID, and salary of all employees born in the 1970s and salary greater than 5000$.

SQL Query:

```
SELECT ID, Salary
FROM EMPLOYEE
WHERE ID IN (SELECT E.ID
FROM EMPLOYEE E
WHERE ID IN( SELECT E.ID FROM EMPLOYEE E WHERE TO_CHAR(E.DOB,
'YYYY') LIKE '__7_') INTERSECT SELECT E.ID FROM EMPLOYEE E WHERE
E.SALARY >5000);
```

OUTPUT:



ID	SALARY
1122334455	5250
1234512345	5355
1845905447	5250
2307411820	5775
4892211133	5250
5544332211	5355

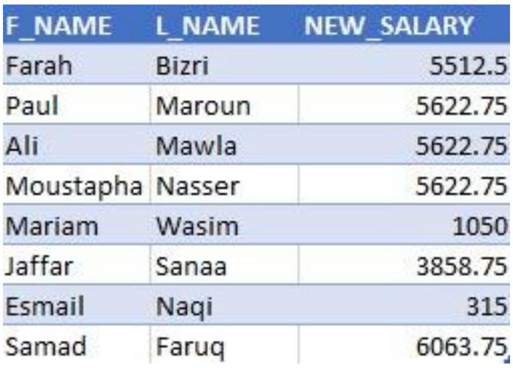
Query 9:

Due to the fuel crisis, the nursery decided to help all employees working away from their homes. Retrieve the name of all employees who work in a department in Beirut but don’t live in Beirut and increase their salary by 5%.

SQL Query:

```
SELECT F_NAME, L_NAME, 1.05*Salary as New_Salary
FROM EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE.City!='Beirut' AND DPT_NB=DEPARTMENT_NUMBER
AND DEPARTMENT.City='Beirut';
```

OUTPUT:



F_NAME	L_NAME	NEW_SALARY
Farah	Bizri	5512.5
Paul	Maroun	5622.75
Ali	Mawla	5622.75
Moustapha	Nasser	5622.75
Mariam	Wasim	1050
Jaffar	Sanaa	3858.75
Esmail	Naqi	315
Samad	Faruq	6063.75

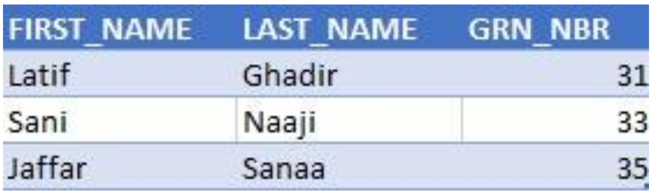
Query 10:

There is a lack of human resources in the greenhouses. Therefore, the nursery plans on increasing the number of employees in it. To help them, retrieve the greenhouse numbers and name employees working there, if the greenhouse has less than 2 employees working there.

SQL Query:

```
SELECT E.F_NAME AS FIRST_NAME, E.L_NAME AS LAST_NAME,
G.Greenhouse_number AS GRN_NBR
FROM EMPLOYEE E, GREENHOUSE G
WHERE EXISTS (SELECT COUNT(*)
FROM EMPLOYEE
WHERE EMPLOYEE.GRN_Nb=G.Greenhouse_number
GROUP BY G.Greenhouse_number
HAVING COUNT(*)<2)AND E.GRN_NB=G.Greenhouse_number;
```

OUTPUT:



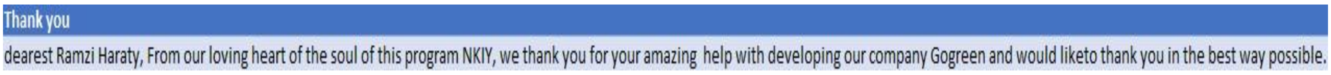
FIRST_NAME	LAST_NAME	GRN_NBR
Latif	Ghadir	31
Sani	Naaji	33
Jaffar	Sanaa	35

Query 11:

SQL Query:

```
INSERT INTO CUSTOMER VALUES (202, NULL, 'Ramzi Harati', 'Hamra',
NULL);
```

OUTPUT:



Thank you
dearest Ramzi Haraty, From our loving heart of the soul of this program NKIY, we thank you for your amazing help with developing our company Gogreen and would liketo thank you in the best way possible.

O- 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 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.

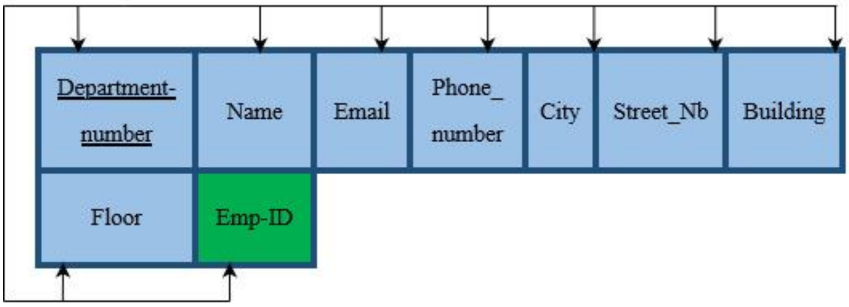
Transitive Dependency: A functional dependency X →Y in a relation schema R is a transitive dependency if there exists a set of attributes Z in R that is neither a candidate key nor a subset of any key of R, and both X →Z and Z →Y hold.

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- DEPARTMENT:



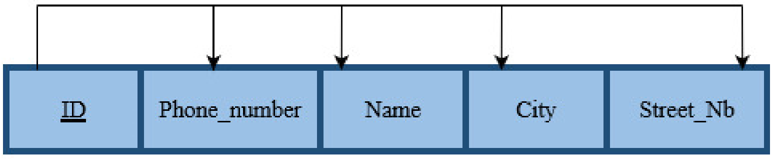
A. The **DEPARTMENT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

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

C. The **DEPARTMENT** 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 **"Department-number"**.

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.

2- CUSTOMER:



ID	Phone_number	Name	City	Street_Nb

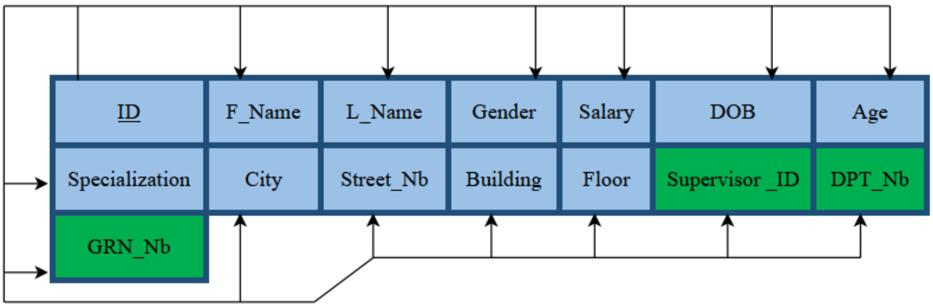
A. The **CUSTOMER** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

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** 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 **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.

3- EMPLOYEE:



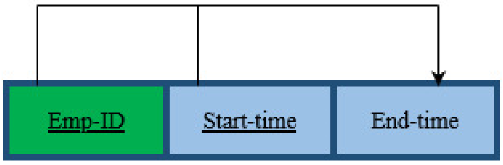
A. The **EMPLOYEE** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

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 is 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.

4- SCHEDULE:



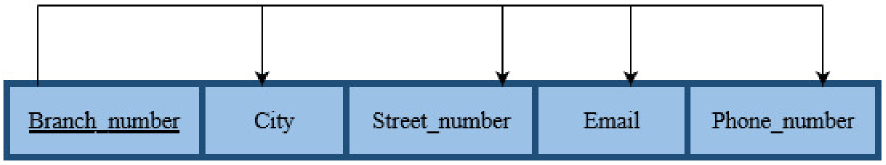
A. The **SCHEDULE** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **SCHEDULE** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **“{Emp-ID , Start-time}”**.

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 **“{Emp-ID , Start-time}”**.

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.

5- BRANCH:



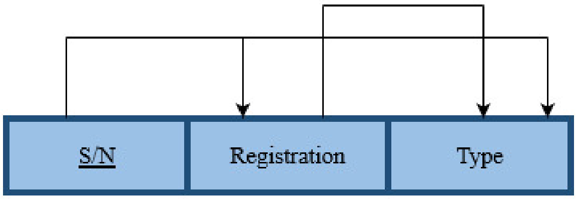
A. The **BRANCH** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

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

C. The **BRANCH** 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 **“Branch_number”**.

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.

6- VEHICLE:



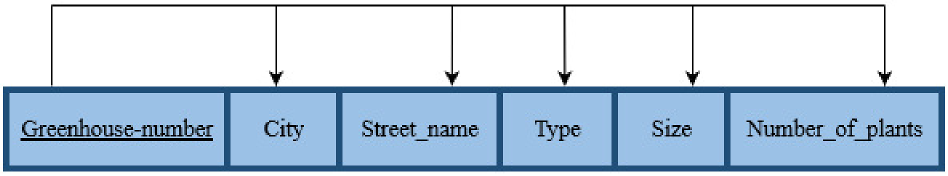
E. The **VEHICLE** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

F. The **VEHICLE** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"S/N"**.

G. The **VEHICLE** 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 **"S/N"**.

H. The **VEHICLE** 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- GREENHOUSE:



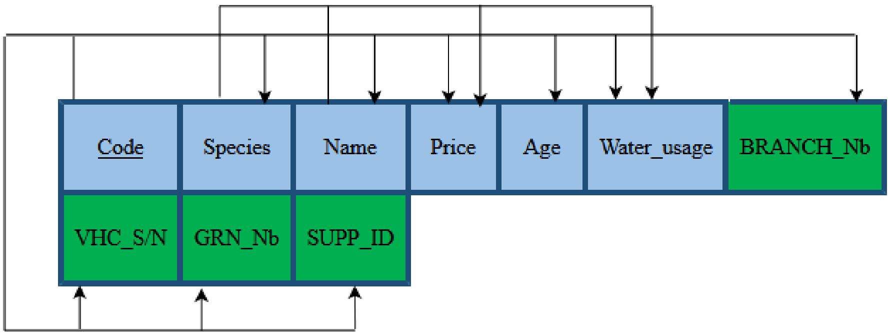
A. The **GREENHOUSE** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **GREENHOUSE** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **“Greenhouse-number”**.

C. The **GREENHOUSE** 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 **“Greenhouse-number”**.

D. The **GREENHOUSE** 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- PLANT:



Our client would love it if we could get the price and Water_usage of a plant using its species and name

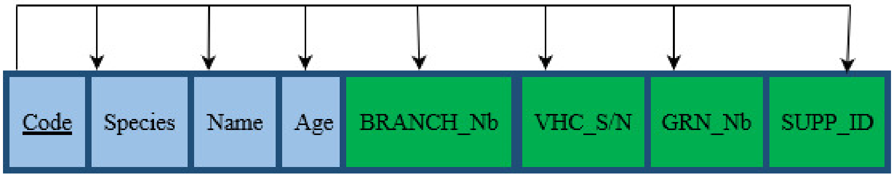
A. The **PLANT** relation schema satisfies all conditions of the 1NF because it has neither multi-valued attributes nor composite attributes. All attributes are single and atomic.

B. The **PLANT** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on **"Code"**.

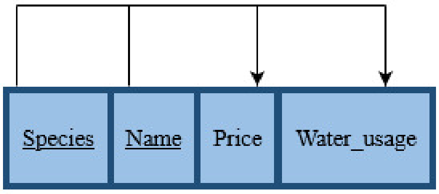
C. Unfortunately, the 3NF doesn't pass this test since {Species,Name} is not a superkey and in the same case the attributes its getting which are Price and Water_usage prime attributes.

According to the client's requirements we must construct two tables like below:

PLANT1:



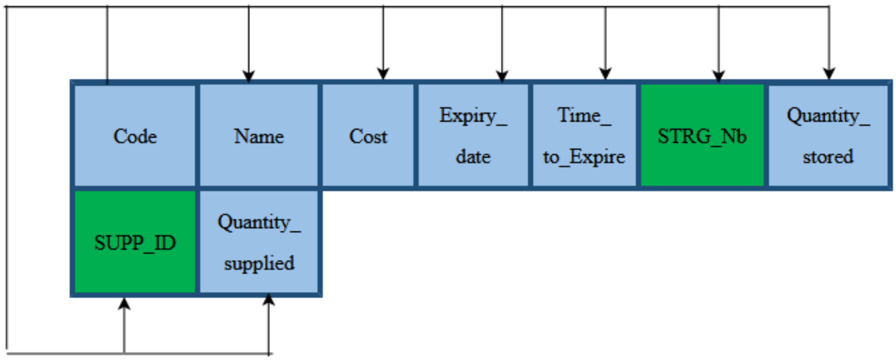
PLANT2:



According to table **PLANT2**, "Price" and "Water_usage" are fully dependent on bothSpecies and Name thus satisfying the 3NF.

D. The **PLANT1** and **PLANT2** relation schemas satisfy 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- NUTRIENT:



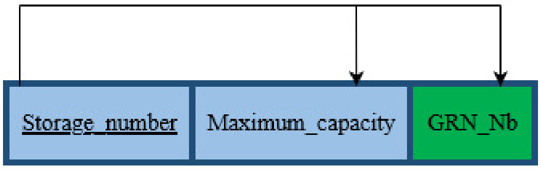
A. The **NUTRIENT** relation schema satisfies all conditions of the 1NF because it has neither multi-valued attributes nor composite attributes.

B. The **NUTRIENT** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on “Code”.

C. The **NUTRIENT** relation schema satisfies all conditions of the 3NF since it satisfies 2NF and since there are no non-prime attributes that are transitively dependent on “Code”.

D. The **NUTREINT** 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-STORAGE:



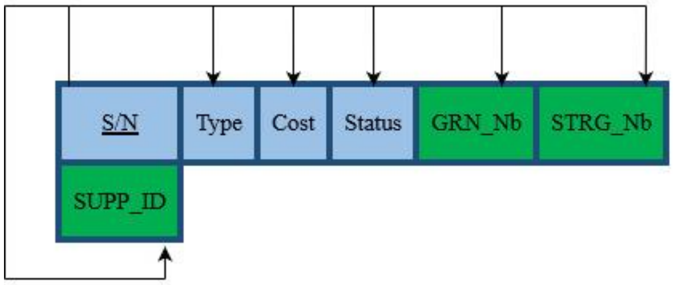
A. The **STORAGE** relation schema satisfies all conditions of the 1NF because it has neither multi-valued attributes nor composite attributes.

B. The **STORAGE** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on "Storage_number".

C. The **STORAGE** relation schema satisfies all conditions of the 3NF since it satisfies 2NF and since there are no non-prime attributes that are transitively dependent on "Storage_number".

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.

11- EQUIPMENT:



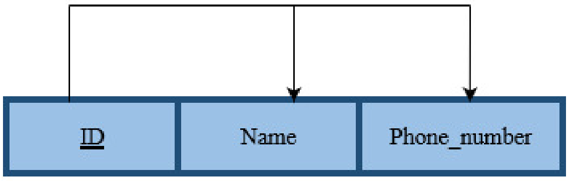
A. The **EQUIPMENT** relation schema satisfies all conditions of the 1NF because it has neither multi-valued attributes nor composite attributes.

B. The **EQUIPMENT** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on “S/N”.

C. The **EQUIPMENT** relation schema satisfies all conditions of the 3NF since it satisfies 2NF and since there are no non-prime attributes that are transitively dependent on “S/N”.

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.

12- SUPPLIER:



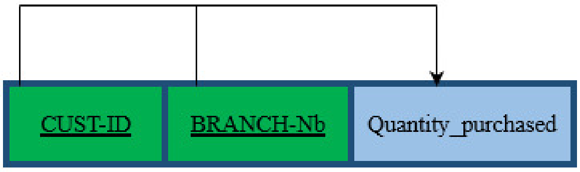
A. The **SUPPLIER** relation schema satisfies all conditions of the 1NF because it has neither multi-valued attributes nor composite attributes.

B. The **SUPPLIER** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on “ID”.

C. The **SUPPLIER** relation schema satisfies all conditions of the 3NF since it satisfies 2NF and since there are no non-prime attributes that are transitively dependent on “ID”.

D. The **SUPPLIER** 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-PURCHASES_FROM:



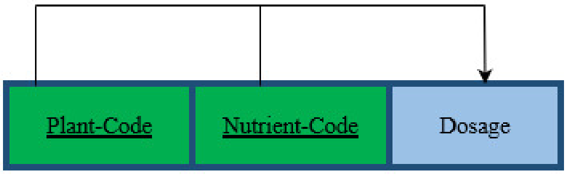
A. The **PURCHASES_FROM** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

B. The **PURCHASES_FROM** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"{CUST-ID, BRANCH-Nb}"**.

C. The **PURCHASES_FROM** 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 **"{CUST-ID, BRANCH-Nb}"**.

D. The **PURCHASES_FROM** 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-REQUIRES:



A. The **REQUIRES** 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 **REQUIRES** relation schema satisfies all conditions of the 2NF because every nonprime attribute is fully functionally dependent on the primary key **"{Plant_Code, Nutrient-Code}"**.

C. The **REQUIRES** 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 **"{Plant_Code, Nutrient-Code}"**.

D. The **REQUIRES** 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.

Relation Schemas without non-prime attributes:

15-DRIVES:

EMP-ID	VHC-S/N

16-UTILIZES:

EMP-ID	EQUIP-S/N

17-CUST_PURCHASE_DATES:

CUST-ID	Purchase-dates

18-EMP_PHONE_NUMBER:

EMP-ID	Phone-number

19-SUPP_SUPPLY_DATES:

SUPP-ID	Supply-dates

P- Conclusion:

A database is crucial to the establishment and functioning of an organization, containing hundreds of data from staff members, departments, and clients. In our case, this plant nursery database is used to show how plants are grown in greenhouses and taken care of by the employees specialized in the field. Plants are being sold in multiple retail branches that are related to the plant nursery itself. Moreover, this database classifies each component needed to do certain tasks, such as delivering plants to the retail branches via vehicles. It also shows the number of nutrients that each plant needs (such as nutrients and water), and this displays where equipment and nutrients are being stored. Additionally, this database takes into consideration the client's point of view via a customer-focused entity type that will get to know the clients better. Lastly, this database provides schedules for each employee, which will prevent time conflicts and time management errors.

Q- Instructor’s Comments and Evaluation:

This page should be filled by our instructor DR. RAMZI HARATY regarding any comments, and improvements for our database system.