SAPIENZA UNIVERSITY - ROMA

Students: Alessio Borgi, Elena Maria Muià

Reference email: borgi.1952442@studenti.uniroma1.it, muia.1938610@studenti.uniroma1.it

DELIVERY DATE: June 2022 the 5th

Applied Computer Science and Artificial Intelligence

DATABASE AND MANAGEMENT SYSTEM: UNIT 2

"THE HELPDESK COMPANY"

PRESENTATION VIDEO:

THE PRESENTATION VIDEO IS LINKED BELOW. NOTE THAT IT IS A PRIVATE YOUTUBE VIDEO, THEREFORE ONLY USERS ALLOWED CAN SEE IT. WE HAVE ALLOWED TO SEE THE VIDEO TO THIS ACCOUNT: **pirro@di.uniroma1.it**.
YOU SHOULD BE LOGGED WITH THIS ACCOUNT WHEN YOU SEE THE VIDEO. FOR ANY PROBLEM PLEASE DO NOT ESITATE TO CONTACT US.

CLICK HERE YOUTUBE VIDEO

CONCEPTUAL SCHEME:

DRAWING1:

CLICK ON THE BELOW SITE IN ORDER TO VIEW THE DRAWING PRE-RESTRUCTURIZATION!
CLICK HERE

DRAWING2:

CLICK ON THE BELOW SITE IN ORDER TO VIEW THE DRAWING POST-RESTRUCTURIZATION!
CLICK HERE RESTRUCTURED

OVERVIEW: TASK's EXPLANATION

We are requested from a Hypothetical Company "**HelpDesk**" to build up a **Ticketing System**. "HelpDesk", gives us the task to build up all the DataBase and Framework parts of the company. After some interviews with our client, through which we collect all the necessary information to build up our Database, we are explained how the company is internally structured, what are the different tasks of the employees, what are the links with external companies, and most importantly, how the company approaches its clients. The company HelpDesk, in order to build up the Database for its internal organization, asks us to organize and handle the assistance part, in which customers ask for help, which is provided by HelpDesk's Technicians if they encounter problems with some products. Moreover, we are also requested to handle all the processes from the start to the closure of a Ticket.

INTERNAL STRUCTURE:

First of all, we have explained the Internal Structure of the HelpDesk, from an employment point of view. There is a big distinction in the hierarchy in which we have the Leadership Team, composed of all the directors and executors of the Company, and the Workforce, in which we include, Managers, Technicians, and Cross-Functional Employees. We also know that there is another sub distinction in each of the three Workforce Divisions. We have that Technicians are divided into 1st Level and 2nd Level Technicians, Managers are divided into Account and Business Managers and that the Cross-Functionals are divided into Finance employees and HR(Human Resources) employees. This is the Detailed Distinction:

In the **Leadership Team**, indeed, there are the most important employees. Thanks to an interview, we have individuated the following roles:

- **CEO**: Chief Executive Officer. He/She is the Boss of the entire HelpDesk company.
- **CFO**: Chief Financial Officer. He/She is the Boss of the Finance division in the Workforce.
- **CHRO**: Chief Human Resources Officer. He/She is the Boss of the HR Division.
- **CAMO**: Chief Account Manager Officer. He/She is the Boss of all Account Managers.
- **CBMO**: Chief Business Manager Officer. He/She is the Boss of all Business Managers.
- **CTECO**: Chief Technician Officer. He/She is the Boss of the Technician Division.

In the **Workforce** Division instead, we have all the Employees, except bosses. The workforce has an ulterior division. Every subdivision has another subdivision(We reach depth 3 if we view the hierarchy as a tree):

- **TECHNICIANS**: These are the employees that have a direct approach to the clients. Every time a customer opens a Ticket, a **1-st Level Technician** tries to solve the problem described by the Customer, and if necessary, it contacts him. If the 1-st level Technician understands that on-site intervention is necessary, he delegates a **2-nd Level Technician** that will promptly intervene on-site. Let's make a clearer distinction between these two figures:

    - **1st Level Technician:** As it has already been defined, they have the first approach with the client, so they have a WorkStation Number which defines their desk position. They have to View the Ticket opened by the Customer and, if the task is solvable from remote, they can write the Report and wait for the Customer to Close the Ticket. In the case in which the Customer's request is not satisfied by this Division, they delegate an expert from the 2nd Level Technician Division who can give on-site support to the Customer.
    - **2nd Level Technician:** They are called based on their specializations in order to solve the issues of the Customer right on-site. In order to reach the destination they are assigned to a Vehicle and based on the product that has to be adjusted, on the distance from the destination, and on their driving license, they are assigned to a different type of Vehicle between Motorcycle or Car. Furthermore, they are assigned to proper PPE (Personal Protective Equipment), based on the type of intervention that will be actuated, in our case the options are: Gloves, Helmet, Safety Jacket, and Safety Shoes.

- **MANAGERS**: These are the Managing part of the company. They are divided into two main categories. We have **Account Managers**, whose role is to supervise Technicians and to handle customers in the most efficient way possible. They manage and are responsible to handle and verify the Customer Satisfaction, which is one of the fundamental Basic-Principles of "HelpDesk". We then have the **Business Managers** whose role is to acquire new External Companies(i.e. Apple, Samsung, etc.) that will hire "HelpDesk" that will handle its assistance requests. Business Managers have the role to close affairs and to convince External Companies to sign Contracts. Once External Companies sign a Contract, the Finance Division is responsible to effectively close the affair, and to therefore register the contract.
- **CROSS-FUNCTIONALS**: They are, by definition, groups of people from various departments in an organization. In our case, we are requested to implement the figures for **Human Resources** and of **Finance** Employees. The first ones are the only ones who can hire new Employees and give them the credentials to log into our HelpDesk website. Furthermore, they handle the contracts and the payments of the Technicians. Instead, the second ones support the Business Manager in the financial side of HelpDesk management and are requested to register the External Companies' Contracts.

Everytime that a new Employee is hired, he/she has to sign a **Contract** and they are assigned with an ID_Employee. The HR Division handles all these recruiter procedures. First of all, an Employee has to sign a Contract in which are indicated all of the basic information which legally defines all of its tasks and benefits. The Contract indicates the Type of the Contract, the Expiry_Date, the Salary, the Holidays, Benefits, the Assurance (which changes based on the role of the Employee), and the Bonus accumulated with positive feedback or working achievements.

BENEFITS:

Based on the role of the Employee, different Benefits are assigned to it. After the interviewing part, we have individuated the following Benefits that HelpDesk may provide:

- **Food Stamp:** They are assigned to every Employee of HelpDesk. They have an Expiry Date and a certain Value, these tickets can be spent only in certain Stores. which are affiliated with HelpDesk.
- **Courses Bonus:** They are assigned to all the Employees, in order to give to all of them the capability of increasing their culture.
- **PC Benefit:** They are assigned to every Employee of HelpDesk, except for the 2nd Level Technician Division. It is particularly important for the 1st Level Technician Division because it is the interface between Customers and HelpDesk.
- **IPAD Benefit:** They are assigned to every 2nd Level Technician in such a way they have the possibility to compile, from the site of intervention, the Report.
- **Phone Benefit:** The phone is assigned to every Employee of "HelpDesk" except for Technicians(both 1st and 2nd level).
- **Car Benefit:** The car is assigned only to Finance Employees, Managers (both Account and Business Managers) and to the LeadershipTeam. Notice that, linked to cars, only Business Managers and Leadership Team Employees have a reserved place in the "HelpDesk" park.
- **Credit Card:** They are assigned only to 2nd Level-Technicians(in order to allow them to buy fuel for their Company Vehicle), to Business Managers, in order to allow them to bring new customers to have dinner, and to the LeadershipTeam
- **Book Bonus:** They are assigned only to Managers (both Account and Business Managers) and to the LeadershipTeam.
- **Stock Options:** They are assigned only to the Leadership Team.

BENEFIT TABLE

	Food	Courses	Car	Book	Stock	Credit	Phone	PC	IPAD
Tech_1	✓	✓						✓	
Tech_2	✓	✓				✓			✓
Acc_Man	✓	✓	✓	✓			✓	✓	
Bus_Man	✓	✓	✓	✓		✓	✓	✓	
HR	✓	✓					✓	✓	
Finance	✓	✓	✓				✓	✓	
Lead	✓	✓	✓	✓	✓	✓	✓	✓	

.

EXTERNAL COMPANIES:

External Companies have been mentioned many times. They are linked in many ways to the HelpDesk company. They hire HelpDesk to let HelpDesk manage the product-assistance side. When they hire HelpDesk they sign an **External_Contract** with the Finance Division, which will define all the quibbles of the agreement between the two companies. All the future communication with HelpDesk will be managed by the Business Managers. When they sign the Contract, they have to establish an insurance product defined as default in UE for all of their sellings. This has to happen because it will have an impact on the cost that the Customer will have to pay for the broken Products, which have been manufactured by the External Company.

The amount of their Subscription payments depends on many factors: their dimension and the amount of intervention requested on their products, this is because HelpDesk imagined that many of the products that have to be adjusted would have been under insurance, so there would have been no income in that intervention. This is why the amount of interventions has a weight on the amount of monthly Subscription to the HelpDesk service.

CUSTOMER:

The Customers have a totally different point of view of what is happening, they do not know what is happening beyond the surface, they only want that their product is adjusted as fast as possible.

First of all, they register themselves on the website, furthermore, they have to register the product that has to be adjusted (once they have registered they can add as many products as necessary just by logging in). When they register a product, they also Open a Ticket to inform the 1st Level Technicians about their issue. They can check how their problem is being handled and in how much time by Inspecting the Report and when the issue has been solved they can Validate the Report, add feedback and Close the Ticket, they are the only ones who can do that.

RESTRUCTURING:

We then proceed in Restructuring our schema by:

- Bank with Stock_Company and Credit_Company: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “Bank_Type” Attribute that will handle this distinction. This attribute can only take two values: Stock_Company/Credit_Company.
- Store with Book Store and Shop: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “Store_Type” Attribute that will handle this distinction. This attribute can only take two values: Book Store/Shop.
- Ticket with Closed_Ticket: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “Closed_Ticket” Attribute that will be a Boolean Attribute. In addition to this, we have merged on Ticket also the Attribute that Closed_Ticket had: “Closure_Data_and_Time” and we managed to set it to be Optional by putting the (0,1) indication, of “timestamp without timezone” type. (Note that the latter attribute will be allowed to be set to a value only when “Closed_Ticket” is set to True.)
- PPE with Helmet, Safety_Jacket, Safety_Shoes and Gloves: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “PPE_Type” Attribute that will handle this distinction. This attribute can only take four values: Helmet / Safety_Jacket / Safety_Shoes / Gloves. In addition to this, we have merged on PPE also the Attribute that Helmet had: “Visor” and we managed to set it to be Optional by putting the (0,1) indication, and it is a Boolean type. (Note that the latter attribute will be allowed to be set to True only when “PPE_Type” is set to “Helmet”).
- Vehicle with Van and Motorcycle: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “Vehicle_Type” Attribute that will handle this distinction. This attribute can only take two values: Van/Motorcycle. In addition to this, we have merged on Vehicle also the Attributes that both Motorcycle(Security_Clothing, GPS_Assigned, Weight_Limit) and Van(Attention_Assist and Camera_Assist) had. We managed to set them to be Optional by putting the (0,1) indication. We set “Attention_Assist”, “Camera_Assist”, “Security_Clothing” and “GPS_Assigned” to be a Boolean type, whilst “Weight_Limit” to be an Integer value.(Note that the Security_Clothing, GPS_Assigned, Weight_Limit attributes will be allowed to be set only when “Vehicle Type” is set to “Motorcycle”. Same thing for Attention_Assist and Camera_Assist attributes that will be allowed to be set only when “Vehicle Type” is set to “Van”).
- Parking with Free and Reserved: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “Parking_Type” Attribute that will handle this distinction. This attribute can only take two values: Free/Reserved. In addition to this, we have merged on Parking also the Attribute that Reserved Parking had, “ID_Employee_Reserved”, and we managed to set it to be Optional by putting the (0,1) indication. Indeed, it wil have to be set only if the choice in “Parking Type” is set to “Reserved”.

- Employee, WorkForce and Leadership, Technicians, Managers and Cross Functionals: (Method 4: Hybrid Solution):
    - Employee with WorkForce and Leadership: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added an “Employee_Type” Attribute that will handle this distinction. This attribute can only take two values: WorkForce_Employee / Leadership_Employee. In addition to this, we have merged on Employee also the Attribute that Leadership Employee had, “Parking_Number”, and we managed to set it to be Optional by putting the (0,1) indication. Indeed, it wil have to be set only if the choice in “Employee_Type” is set to “Leadership_Employee”. We also changed the Relationship that there was between these two subentities, “Leadership”. We therefore placed it as a Recursive Relationship.
    - Technician with 1st_Level_Technician and 2nd_Level_Technician (Method 2: Merging the parent into the child entities). We added to every sub-entity the Attributes that the super-entity “Technician” had: “ID_Account_Manager”, and “Company_Assigned_Expert”. From this follows that every relationship we had with “Technician” (in this case “Supervision”, “Management” and “Compilation”) will become a relationship for every sub-entities.
    - Manager with Account_Manager and Business_Manager: Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “Manager_Type” Attribute that will handle this distinction. This attribute can only take two values: Account_Manager / Business_Manager. We also changed the Relationship that there was between these two subentities, “Supervision_1st_Level_Tech”, “Supervision_2nd_Level_Tech” and “Responsible”. We therefore placed them all starting from the “Manager” new entity and we let all of them to be optional Relationships. Indeed, we will have that the first and the second relationhips listed above need to be mandatory only if the choice for “Manager_Type” is “Account_Manager”, whilst in the case we choose “Manager_Type” to be “Business_Manager”, we have that the third relationship need to be mandatory .
    - Cross_Functionals with Finance and Human_Resources(HR): Elimination of generalization (Method 1: Merging children of the generalization into the parent). We added a “Cross_Functionals_Type” attribute that will handle this distinction. This attribute can only take two values: Finance / Human_Resources. We also changed the Relationship that there was between these two subentities, “Administration”, “Registration” and “Handler”. We therefore placed them all starting from the “Cross_Functionals” new entity and we let all of them to be optional Relationships. Indeed, we will have that the first relationhip listed above need to be mandatory only if the choice for “Cross_Functionals_Type” is “Finance”, whilst in the case we choose “Cross_Functionals_Type” to be “Human_Resources”, we have that the second and third relationships need to be mandatory.

    - Employee with 1_st_Level_Technician, 2_nd_Level_Technician, Manager and Cross_Functionals: Elimination of generalization (Method 3: Substitution with relationships). We added a Rrelationship between “Employee” and each of the Subentities listed above. The relationships will folow the following schema: ”*Correspondence_NameSubEntity*”. We added the fact that the relationships will start form employee, with an optional link and then we will have that we have that all the subentities have a Foreign Key that is the ID_Employee.
    -
- Benefit with Courses_Bonus, IPAD_Benefit, PC_Benefit, Phone_Benefit, Car_Benefit, Book_Bonus, Food_Stamp, Stock_Option, Credit_Card: Elimination of generalization (Method 2: Merging the parent into the child entities). We added to every sub-entity the Attributes that the super-entity “Benefit” had: “ID_Benefit”, “ID_Contract”, “ID_Employee”. From this follows that every relationship we had with “Benefit” (in this case only “Inclusion”) will become a relationship for every sub-entities. Every Sub-entity will inherit the Key that will follow the following schema: *“ID_Benefit_NameSubEntity”* Thus we need to create in total 9 relationships(They will follow the following schema “*Inclusion_NameSubEntity*”.

GLOSSARY OF TERMS:

Concept	Meaning
Customer	The person who asks for HelpDesk services
Watching_Report	Each time a customer accesses the reports to see its updates
Convalidation_Report	The customer has to convalidate the report after that the Technicians have solved his/her issue
Customer_Living	Where does each customer lives
Cus_Owning	How to contact by phone each customer
Cus_Contact	How to contact by email each customer
Phone_Customer	Phone of each Customer
Email_Costumer	Email of each Customer
Address_Customer	Address of each Customer
Report	Here the Technicians will explain what they have one, how and in how much time
Inspection	Each time an employee accesses to the reports to see its updates
Ticket	This is the mean used by the customer to arise his need for HelpDesk service, in this document the Technicians may see all the needed information about the customer (address, contact)
Closure_Ticket	This is the entity representing each ticket that has been closed by the customer
Revisited_Ticket	Each time a ticket has been revisited by a customer
Submission_Ticket	Each time a Customer submits a ticket
Compilation_1st_Tech	The 1st level technician can compile the ticket with the info requested

Compilation_2nd_Tech	The 2nd level technician can compile the ticket with the info requested
Viewing_Ticket	Each time a ticket is accessed to see its updates
Management_1st_Tech	The 1st level technicians handle the progress of the tickets
Management_2nd_Tech	The 2nd level technicians handle the progress of the tickets
Employee	All the workers in the HelpDesk Company
Living	Relation connecting an employee and the relative address
Owning	Relation connecting an employee and the relative phone number
Communicate	Relation connecting an employee and the relative email
Address_Employee	Address of each employee
Phone_Employee	Phone number of each employee
Email_Employee	Email address of each employee
Supervision_1st_Tech	The 1st Level Tech supervised by Account Managers
Supervision_2nd_Tech	The 2nd Level Tech supervised by Account Managers
1st Level Technician	These are the Technicians who approach to the client from remote
Delegation_Technician	If the 1st Level Technician isn't able to solve the issue from remote, the problem is delegated to the 2nd Level Technician
2nd Level Technician	These are the employee who solve the issues delegated from the 1st Level Technician and they will solve the issue on-site

Required	Required driving licenses in order to drive a certain vehicle
Drive_License	Available Driving License owned by 2nd Level Technicians
Drive_Vehicle	Since the 2nd _Level Technician have to reach the location of the Costumer, maybe bringing some supportive tools, they will have to drive to the destination
Vehicle	This is the entity containing all the information about the vehicle available for the 2nd Level Technicians
Positioning	The Vehicles used by the Technicians are positioned in the free spots of the company's Parking
Parking	This is the Parking used by the Technicians and the Employees who have a reserved space
Assignment	Some cars given has a benefit have a reserved space in the parking
PPE_Assigned	Since some interventions may be dangerous, to follow the legal procedures, Technicians of 2nd Level are assigned with 1 or more Personal Protective Equipment
PPE	This is the entity containing all the general information about the PPE available for technicians
Made_of_Material	Material of each PPE
Material	List of some materials used for clothes
Manager	An employee having the role of Manager, this team is composed by Account Managers and Business Managers
Correspondence_Manager	If the role of the employee is :Manager

Responsible	As we said, this is the moment of connection between the Business Managers and the External Companies
External Companies	These are all the Companies who buy the services of HelpDesk
Manufacture	The External Companies are also those who manufacture the product that the costumer ask us to repair, following the guidelines of the EU insurance
Product	This is the product broken, which has been produced by one of the external Companies that will be registered in HelpDesk DataBase
Record	The Customer register all of his/her broken products
Signature	All the external companies that requires for HelpDesk services, have to sign a contract managed by one of HelpDesk employee
External Contract	This is the entity representing the contract that has to be signed by External Companies, it declares all the duties and rights of both the External Company, both HelpDesk employees
Administration	The External Contract are Administrated from the Finance Team
Correspondence_Cross_Functional	If the role of the employee is: Cross Functional
Cross Functional	The Cross Functional is the one directly below the leadership, it is composed of Human Resources Team and Finance team
Registration	Each new hired employee has to be recorded on the DataBase by the Human Resources employee

Handler_Contract	The HR teeam, also has to handle the contract of each employee under his/her supervision
Contract	This is the entity containing all the information about the contract signed by every role, of course the information of it changes basing on the role of the employee
Sign	Each Employee has to sign the Contract proposed by HelpDesk
Inclusion_Courses_Bonus	Each course bonus connected to a certain contract, and it is represented in this relation
Courses Bonus	This bonus may be spent by some categories in order to buy some courses which may update them about all the news in their working field
Usage	These Course Bonus may be used to buy some courses, each of them has the same price, but different roles have different valued bonuses
Course	These are the courses proposed to the employees
Organization	Each Course is organized by a Course Company affiliated with HelpDesk
Course Company	These are the companies affiliated with Helpdesk that offer courses for the employees
Release	Each course and course company, realeas a valid ceertificate at the end of the course
Certification	This is the certification achieved by the employees by following the courses
Achievement_Certification	Each certification is achieved by an employee

Inclusion_IPAD_Benefit	Each IPAD benefit connected to a certain contract, and it is represented in this relation
IPAD Benefit	Some employees receive an IPAD, which has to be used to help them in the working activities
Sale IPAD	These IPAD are sold by an affiliated Technology Company
Inclusion_PC_Benefit	Each PC benefit connected to a certain contract, and it is represented in this relation
PC Benefit	Some employees receive a PC, which has to be used to help them in the working activities
Sale PC	These PC are sold by an affiliated Technology Company
Inclusion_Phone_Benefit	Each Phone benefit connected to a certain contract, and it is represented in this relation
Phone Benefit	Some employees receive a Phone, which has to be used to help them in the working activities
Sale TEL	These Phones are sold by an affiliated Technology Company
Technology Company	This is the affiliated company that gives to HelpDesk the technological benefits
Inclusion_Car_Benefit	Each Car benefit connected to a certain contract, and it is represented in this relation
Car Benefit	Some employees have a car, also for marketing purposes, and a few of those also have a served place in the parking
Rent	The car given as a benefit to the employees are just rented from a fleet company
Fleet Company	This is the company that rents car to the HelpDesk company

Buying	Instead, the car used by the 2nd level technincian are bought, always from the same fleet company
Inclusion_Book_Bonus	Each book bonus connected to a certain contract, and it is represented in this relation
Book Bonus	Some employees mau have a bonus which can be used to buy some books, maybe also related to their working field
Partnership	To buy this books, they have to present it to some stores in partnership with HelpDesk
Store	These are all the stores affiliated with HelpDesk, both for Book Bonuses, both for Food Stamps
Inclusion_Food_Stamp	Each food stamp connected to a certain contract, and it is represented in this relation
Food Stamp	All the employees have some Food Stamps that they can use in affiliated markets or even restaurants
Affiliation_Food	These food stamps may be spent in places affiliated with HelpDesk
Stock Options	Some employees have a portion of the company's stock available for investments
Exchanged	These stock options are exchanged in some bank affiliated with HelpDesk
Bank	These are the banks affiliated with HelpDesk
Credit Card	Some employees have the corporate credit card which can be used for purposes useful to the HelpDesk company (add gas to the vehicle, take the clients at dinner, etc.)
Affiliation_CC	These Credit Card are all affiliated with a Credit Compan

EXTERNAL CONSTRAINTS:

1. Each Account Manager handles at least 1 Technicians.
2. Each Human Resources employee handles at least 1 employees.
3. The ratio between Reports and Tickets must be 1.5.
4. The Contract value of any employee of the Workforce must not be higher than the value of the Contract of any member of the Leadership team.
5. The amount of monthly FoodStamp must be equal to the number of Employees in HelpDesk multiplied by the days in the considered month.
6. The number of places in HelpDesk's parking is made by the sum of the Van used by the Technicians of Second Level for their interventions, plus the Motorcycles used by the Technicians of Second Level for their interventions, plus the number of employees that have reserved space in the parking as a benefit.
7. The number of reserved spaces in the Parking is made of the number of Leadership employees, plus the number of Business managers.
8. The number of types of Contracts must be equal to the number of different Employee roles present in the HelpDesk company.
9. Each Business Manager must administrate at least 1 External Companies. Any Employee can never reach a yearly bonus higher than 50% of their monthly salary.

VOLUME TABLE:

After the Interviewing part, we obtained some other information that helped us to have an idea of the Volumes that our Database will have to withstand:

- We have been told that the Database will have to register a number of *Clients* of about **3000** individuals. Clearly, this is only an indicative volume and it is going to increase because every month, we are told that about **100** new clients register to the Platform. Knowing these facts we have decided to refresh our Volume Table each month, in order to have a clear vision of the system as a whole.
- We know that each Customer may open more than one *Ticket* if they need HelpDesk to repair more than one object, so we have been told that, based on previous statistics, the predicted amount of monthly *Tickets* is **10000**. From the statistics, we predict to have about **750** new tickets every month.
- Following the statistic trend of the HelpDesk company, we have noticed that the ratio between *Tickets* and *Report* is 1.5, which means that if the amount of *Tickets* is **10000**, the amount of *Reports* is **15000**
- The amount of *Products* instead, depends on how many times the same *Product* will request for HelpDesk assistance and on the amount of *Product* manufactured by the *External Companies*. Following the statistics, we have noticed that the amount of products is 5 times the amount of *External Companies* which means that in this case we have **550** Products, given the fact that HelpDesk has a Contract with **110** Companies.
- We know that the amount of *Employees* is **171**, which is composed of:
    - **6** employees in the *Leadership* (CEO, CFO, CHRO, CAMO, CBMO, CTECO)
    - **175** employees working in the *WorkForce*, where there are:
        - **100** *Technicians*
            - **30** *Technicians of the 1st Level*
            - **70** *Technicians of the 2nd Level* → the amount of *2nd Level Tech* is much higher because HelpDesk has supposed that online support would take much less time than the time needed for onsite support, thus, to cover all the *Customer*'s needs they had to have many more *2nd Level technicians*
        - **30** Managers
            - **10** *Account Managers* who will handle 10 *Technicians* each (100/10 = 10)
            - **20** *Business Managers*, each of them handling at least **5** *External Company*, for a total of **110** *External Companies* who have decided to get affiliated with us. Since each *External Company* signs one and only one contract, *External_Contract* will have the same volume in our DataBase
        - **35** Cross-Functionals
            - **20** employees involved in the *Finance* field

        - **15** employees in the *Human Resources*, who will *Handle* **171** hiring and contracts
    - Which means that we will have **7** types of *Contract*, one for every type of role, so all the *Signed* Contracts will be **171**
- We know that each *2nd Level Technician* has some tools which are necessary for him/her to execute his/her job, *PPE* and a *Vehicle*:
    - In the parking, there are **70** *Vehicles*, in this way each technician is always sure that there will be a car ready for him/her. These *Vehicles* are divided into **30** *Motorcycles* and **50** *Vans*, because we know that is more probable that some are able to drive a car rather than a motorcycle and that it is always better to have a car available in case the technicians would need more space
    - There are **80** *PPE* available, **20** for each type (*Gloves, Safety Jacket, Safety Shoes, Helmet*), because HelpDesk supposed that there are never more than 100 *Customers* needing for security support at the same time.
- As we have already mentioned in the previous pages of this documentation, to each role is assigned a certain *Benefit*, after some calculations, we have defined that the total amount of *Benefits Included* in the total is 10794
    - [*FoodStamp* (= Employees * days in the Current month) + *PcBenefit* (= 1st Level Tech+CrossFunctional+Leadership+Managers) + *IpadBenefit* (= 2nd Level Tech) + *PhoneBenefit* (= Managers + CrossFunctionals + Leadership) + *CarBenefit* (= Finance + Managers + Leadership) + *BookBonus*(= Managers + Leadership) + *CreditCard* (= 2nd Lev Tech + BusinessMangers + Leadership) + *StockOptions* (= Leadership) + *CourseBonus* (Finance + Managers + Leadership)]
- All of these *Benefits* are released by some companies affiliated with HelpDesk:
    - We know that all the employees who want (and can) spend their *CourseBonus* can choose between **300** *Courses* all at the same price, it is obvious that based on their role, their Bonus will have a higher value. These courses are all released by a *Course Company*, our employees can choose between courses released by **10** companies, all specialized in different fields; considering that almost all the courses correspond to a *Certification*, it is natural that the amount of possible certification can be considered equal to the number of available courses, which is **300**. Our statistics have defined that up until now our employees have *Achieved* **1000** courses in total.
    - Both the Ipads, the PCs, and the Phones are released by a *Technology Company*. HelpDesk has decided to be affiliated with only **1** Company which will give them all the necessary products.
    - Furthermore, HelpDesk *Rents* the car that will be used by our employees, from a *Fleet Company*, which is the same one from which the *Buy* the *Vehicles* used by the 2nd level Technicians.
    - All of the *BookBonus* and *FoodStamps* are spendable in some affiliated *Stores*, we know that HelpDesk is now affiliated with **50** Stores, **10** *BookStores* (to use *BookBonus*), and **40** *Shops* (Market, Restaurants, etc.)

- In the end, we have the *CreditCard* and the *StockOptions*, which are both affiliated with some *Banks***,** in their case **20** Banks, **10** *StockCompanies,* and **10** are *CreditCompanies* .

PRE-RESTRUCTURING

Concept	**Type**	**Volume**
Customer	E	3000
Report	E	15000
Ticket	E	10000
Closed Tickets	E	10000
Employees	E	171
Technicians	E	400
1st Level Technician	E	100
2nd Level Technician	E	300
WorkForce	E	475
Cross Functionals	E	35
Managers	E	10
Account Manager	E	20
Business Manager	E	20
Finance	E	20
Human Resources	E	15
Leadership	E	6
Vehicle	E	70
Van	E	50
Parking	E	96

Free	E	70
Reserved	E	26
Motorcycle	E	20
Product	E	550
PPE	E	80
Gloves	E	20
Safety Jackets	E	20
Safety Shoes	E	20
Helmet	E	20
External Company	E	110
Contract	E	7
Certification	E	300
Course Company	E	10
External Contract	E	110
Technology Company	E	1
Fleet Company	E	1
Store	E	50
Book Store	E	10
Shop	E	40
Bank	E	20
Stock Company	E	10
Credit Company	E	10
Benefit	E	10794
Courses Bonus	E	56
Course	E	5301
PC Benefit	E	101

IPAD Benefit	E	70
Phone Benefit	E	71
Car Benefit	E	56
Book Bonus	E	36
Food Stamp	E	5301
Stock Option	E	6
Credit Card	E	96
Inspection	R	60000
Viewing_Ticket	R	50000
Revisited	R	30000
Convalidation	R	15000
Watching	R	60000
Compilation_	R	15000
Submission	R	10000
Closure	R	4000
Record	R	550
Manufacture	R	550
Management	R	10000
Delegation_Technician	R	2100
PPE_Assigned	R	2000
Drive_Vehicle	R	2100
Supervision	R	400
Signature	R	110
Responsible	R	110
Administration	R	110
Leadership_Hierarchy	R	6

Sign	R	171
Handler_Contract	R	171
Registration	R	171
Positioning	R	70
Buying	R	70
Assignment	R	26
Achievement_Certification	R	1000
Release	R	100
Organization	R	100
Inclusion	R	33
Usage	R	60
Sale IPAD	R	500
Sale PC	R	200
Sale TEL	R	100
Rent	R	100
Partnership	R	50
Affiliation_Food	R	6000
Exchanged	R	20
Affiliation_CC	R	350

POST RESTRUCTURING

Concept	Type	Volume
Customer	E	3000
Address_Customer	E	3050
Email_Customer	E	3050
Phone_Customer	E	3000
Report	E	15000
Ticket	E	10000
Employees	E	171
Address_Employee	E	171
Phone_Employee	E	171
Email_Employee	E	171
1st Level Technician	E	100
2nd Level Technician	E	300
Cross Functionals	E	35
Managers	E	10
Vehicle	E	70
Drive_License	E	10
Parking	E	96
Product	E	550
PPE	E	80
Material	E	20
External Company	E	110
Contract	E	7
Certification	E	300
Course Company	E	10

External Contract	E	110
Technology Company	E	1
Fleet Company	E	1
Store	E	50
Bank	E	20
Benefit	E	10794
Courses Bonus	E	56
Course	E	5301
PC Benefit	E	101
IPAD Benefit	E	70
Phone Benefit	E	71
Car Benefit	E	56
Book Bonus	E	36
Food Stamp	E	5301
Stock Option	E	6
Credit Card	E	96
Cus_Contact	R	3050
Cus_Owning	R	3000
Customer_Living	R	3050
Inspection	R	60000
Viewing_Ticket	R	50000
Revisited_Ticket	R	30000
Closure_Ticket	R	10000
Submission_Ticket	R	10000
Convalidation_Report	R	15000
Watching_Report	R	60000

Compilation_1st_Tech	R	7000
Compilation_2nd_Tech	R	8000
Record	R	550
Manufacture	R	550
Management_1st_Tech	R	4000
Management_2nd_Tech	R	6000
Delegation_Technician	R	2100
PPE_Assigned	R	2000
Made_Of_Material	R	80
Drive_Vehicle	R	2100
Required	R	70
Supervision_1st_Tech	R	30
Supervidion_2nd_Tech	R	70
Signature	R	110
Responsible	R	110
Administration	R	110
Leadership_Hierarchy	R	6
Sign	R	171
Living	R	171
Owning	R	171
Communicate	R	171
Handler_Contract	R	171
Registration	R	171
Positioning	R	70
Buying	R	70
Assignment	R	26

Achievement_Certification	R	1000
Release	R	100
Organization	R	100
Inclusion_IPAD_Benefit	R	33
Inclusion_Phone_Benefit	R	71
Inclusion_PC_Benefit	R	101
Inclusion_Car_Benefit	R	56
Inclusion_Book_Bonus	R	36
Inclusion_Course_Bonus	R	56
Inclusion_Food_Stamp	R	5301
Inclusion_Credit_Card	R	96
Inclusion_Stock_Option	R	15
Usage	R	60
Sale IPAD	R	500
Sale PC	R	200
Sale TEL	R	100
Rent	R	100
Partnership	R	50
Affiliation_Food	R	6000
Exchanged	R	20
Affiliation_CC	R	350

OPERATIONS:

General:

- Select the Customer rate of the report referred to Tickets that have been closed on the same day on which they have been opened.
- Find the number of Tickets having requests about products manufactured from a certain external company.
- Select all the companies that hired us that have at least a certain number of employees.
- Find the address of all the Customers where the day of start and end of the intervention is different.
- Find the Customer Feedback of all the Customers where the day of start and end of the intervention is different.
- Find the Company's names of the type of products that have requested more interventions.
- Find all the Closed Tickets that in the Products have defined that the Insurance on the product has expired.
- Select the dates of the rent expiry on the HelpDesk cars used as benefits.
- Select the model of all the technological benefits that HelpDesk has offered, which are older than a certain threshold.
- Insert all the data of the client and the product needing assistance.
- Select all the Tickets with the highest level(1st Level) of priority still not closed.

1st Level Technician:

- Assign a WorkStation to each 1st Level Technician.
- Find how many times a certain 1st Level Technician has viewed a Certain Ticket.
- Find how many times a certain 1st Level Technician has inspected a Certain Report.
- Insert in the Report all the ID Customer and personal data of 1st Level Technicians that have closed a ticket without delegating to a 2nd Level Technicians.
- Select the Course Companies corresponding to the certifications released to 1st Level Technicians.

2nd Level Technician:

- Find the personal data of all the 2nd Level Technicians who compiled a Report.
- Select all the Driving Licences Levels of the 2nd Level Technicians who have chosen a van.
- Select the Course Companies correspondent to the certifications released to 2nd Level Technicians.

- Select all the PPE used by the 2nd Level Technicians to repair Products manufactured by a certain external company.
- Find the Costumer Rate of a 2nd Level Technician who spent a certain amount of money using the Benefit Credit Card.

Account Manager:

- Select the Customer Star of the Technicians supervised by the Account Manager with the highest amount of Bonus.
- Select the number of closed Tickets of all the Technicians under the responsibility of a certain Manager.
- Select the number of Reports compiled by all the 1st Level Technicians under the responsibility of a certain Manager.
- Select the Course Companies corresponding to the certifications released to Account Managers.

Business Manager:

- Select the number of External Companies supervised by the Business Manager with the highest amount of bonuses.
- Insert the data of each Business manager who viewed a Ticket.
- Find the number of reserved parking spaces assigned to Business managers.
- Select the Course Companies corresponding to the certifications released to Business Managers.

Finance:

- Insert all the info of the External Company that will be used to fill out their Contract.
- Select the personal data about the Finance employee who defined the details of the oldest Contract signed by an External Contract.
- Select the model of the Car Benefit assigned to the Finance employee.
- Select the Course Bonus spent, correspondent to the amount of certifications released to Finance employees.

Human Resources:

- Find the personal information of the new Employees hired from a certain employee of the Human Resources.
- Find the amount of the Contract handled by the HR employee with the highest number of Bonus.
- Select the model of the PC Benefit assigned to the Human Resources employee.

- Select the Course Companies corresponding to the certifications released to Human Resources.

Leadership:

- Inserts the data of all the Leadership Employee in the HelpDesk company.
- Insert the credentials of the Credit Card given to the Leadership employee.
- Select the amount of Certification released to Leadership employees.
- Select the Course Companies corresponding to the certifications released to Leadership employees.

OPERATION TABLE:

- Consider that all the values in the Frequency column as **daily**

Operation	Type	Frequency
General 1	B	322
General 2	I	50
General 3	I	10
General 4	B	322
General 5	B	500
General 6	I	10
General 7	B	322
General 8	B	10
General 9	I	10
General 10	B	322
General 11	I	200
1st Lev Tech - 1	B	30
1st Lev Tech - 2	I	100
1st Lev Tech - 3	I	100
1st Lev Tech - 4	I	200
1st Lev Tech - 5	I	10
2nd Lev Tech - 1	B	500
2nd Lev Tech - 2	I	100
2nd Lev Tech - 3	I	10
2nd Lev Tech - 4	I	50

2nd Lev Tech - 5	I	10
Account Manager 1	I	50
Account Manager 2	I	75
Account Manager 3	I	20
Account Manager 4	I	10
Business Manager 1	I	10
Business Manager 2	B	50
Business Manager 3	B	10
Business Manager 4	I	10
Finance 1	B	1
Finance 2	I	10
Finance 3	B	10
Finance 4	I	10
HR 1	B	5
HR 2	I	5
HR 3	I	10
HR 4	I	10
Leadership 1	B	6
Leadership 2	B	10
Leadership 3	I	50
Leadership 4	I	10

READING KEYS:

- ***Customer → Submission_Ticket → Ticket (0, N):*** Every Customer may want to Submit N Tickets. (Our Customer has to Register himself on the Platform if he wants to be assisted by HelpDesk's Technicians. However, he may also want to Register himself, even if he will never open a Ticket(very rare case, but possible)).
- ***Ticket → Submission_Ticket → Customer (1,1):*** Every Ticket must be Submitted by one Customer.
- ***Customer → Closure_Ticket → Ticket (1, N):*** Every Customer is obliged to close N Tickets. (Our Customer, provided that he received the assistance service that solved the problem, is obliged to Close the tickets. Thus, the Customer has the power to open N Tickets, but he has also to close them. (We know that if the Customer doesn't close the Tickets after a certain amount of time, he will receive two messages after 10 and 15 days the Ticket is opened and if the Customer doesn't answer it, the system will automatically close the Ticket)).
- ***Ticket → Closure_Ticket → Customer (1,1):*** Every Ticket must be Closed after the Customer has opened it.
- ***Customer → Revisited_Ticket → Ticket (0, N):*** Each Customer may revisit up to N Tickets while they are updated
- ***Ticket → Revisited_Ticket → Customer (0,1)*****:** Each Ticket may or may not be revised by one Customer
- ***Employee → Viewing_Ticket → Ticket (0, N):*** Every Employee may want to view up to N Tickets.
- ***Ticket → Viewing_Ticket → Employee (0, N)*****:** Every Ticket may be viewed by up to N Employees.
- ***Employee→ Inspection → Report (0, N):*** Every Customer may want to Watch up to N Reports.
- ***Report→ Inspection → Employee(0, N)*****:** Every Report may be Watched by the Customer who has opened the Ticket to which the Report is associated.
- ***Customer → Convalidation_Report → Report (0, N)*****:** Every Customer may want to Convalidate up to N Reports. (These Reports will be linked to interventions done by Technicians trying to solve the Ticket's Problem.)
- ***Report → Convalidation_Report → Customer (0,1)*****:** Every Report may be Convalidated by the Customer who has opened the Ticket.
- ***Customer → Watching_Report → Report (0, N)*****:** Every Employee may want to Inspect up to N Reports.
- ***Report → Watching_Report → Customer (0, 1)*****:** Every Report may or may not be inspected by the Customer that created the Ticket
- ***Customer → Record → Product (0, N):*** Every Customer may Record N Products on the Platform. (He may want to Record them as soon as he buys them or when he has some problem and therefore he has to open a Ticket.)
- ***Product → Record → Customer (1,1):*** Every Product must be Recorded by one Customer.

- ***1st_Level_Technician → Compilation_1st_Tech → Report (1, N):*** Every 1st Level Technician must compile up to N Reports linked to a set of Intervents done in order to solve the Ticket's Problem.
- ***Report → Compilation_1st_Tech → 1st_Level_Technician (0,1):*** Every Report may be compiled by one 1st Level Technician.
- ***2nd_Level_Technician → Compilation_2nd_Tech → Report (1, N):*** Every 2nd Level Technician must compile up to N Reports linked to a set of Intervents done in order to solve the Ticket's Problem.
- ***Report → Compilation_2nd_Tech → 2nd_Level_Technician (0,1):*** Every Report must be compiled by one 2nd Level Technician.
- ***1st_Level_Technician → Management_1st_Tech → Ticket (1, N):*** Every 1st Level Technician must manage up to N Tickets.
- ***Ticket → Management_1st_Tech → 1st_Level_Technician(0, N):*** Every Ticket may be managed up to N 1st Level Technician.
- ***2nd_Level_Technician → Management_2nd_Tech → Ticket (1, N):*** Every 2nd Level Technician must manage up to N Tickets.
- ***Ticket → Management_2nd_Tech → 2nd_Level_Technician(0, N):*** Every Ticket may be managed up to N 2nd Level Technician.
- ***1st_Level_Technician → Delegation_Technician → 2nd_Level_Technician (0, N):*** Every 1st Level Technician may delegate up to N 2nd Level Technicians if he is not able to solve the problem from remote.
- ***2nd_Level_Technician → Delegation_Technician → 1st_Level_Technician (1, N):*** Every 2nd Level Technician must be Delegated by up to N 1st Level Technicians.
- ***2nd_Level_Technician → PPE_Assigned → PPE (0, N):*** Every 2nd Level Technician may be assigned with up to N Personal Protective Equipment depending on the type of intervention it has to offer.
- ***PPE → PPE_Assigned → 2nd_Level_Technician (0, N):*** Every PPE may be assigned up to N 2nd Level Technicians. (We have here optionality because we can also have maybe that certain PPE are never used).
- ***PPE → Made_Of_Material → Material (1,N):*** Each PPE may be made up to N materials which determine its model and usage
- ***Material → Made_Of_Material → PPE (0, N):*** Each material can be used to made up to N PPEs.
- ***1st_Level_Technician → Supervision_1st_Tech → Manager (1,1):*** Every First Level Technician must be Supervised by one Manager (Account Manager, in particular).
- ***Manager → Supervision_1st_Tech → 1st_Level_Technician (0, N):*** Every Manager (Account Manager, in particular), may supervise up to N Technicians.
- ***2nd_Level_Technician → Supervision_2nd_Tech → Manager (1,1):*** Every Second Level Technician must be Supervised by one Manager (Account Manager, in particular).
- ***Manager → Supervision_2nd_Tech → 2nd_Level_Technician (0, N):*** Every Manager (Account Manager, in particular) may supervise up to N Second Level Technician

- ***2nd_Level_Technician → Drive_Vehicle → Vehicle (1, N):*** Every 2nd Level Technician must be able to Drive up to N Vehicles. (In order to work and offer its on-site service.)
- ***Vehicle → Drive_Vehicle → 2nd_Level_Technician (0, N)*****:** Every Vehicle may be driven by up to N 2nd Level Technicians. (The optionality is due to the fact that some Vehicles may be in the HelpDesk Database but are not already ready to be driven or maybe some other vehicles are too old.)
- ***Vehicle → Buying → Fleet_Company (1, 1):*** Every Vehicle must be bought from a Fleet Company
- ***Fleet_Company → Buying → Vehicle (0,N)*****:** The Fleet Company may sell up to N Vehicle to HelpDesk
- ***Product → Manufacture → External_Company (1, 1):*** Every Product must be Manufactured by one External Company. (The latter, we knew corresponds to the Company that has made the product the Customer has problems on, for instance, Apple, Samsung ecc…)
- ***External_Company → Manufacture → Product (0, N):*** Every External Company may manufacture up to N products. (The optionality is due to the fact that HelpDesk would want to save External Companies for which it has not already a Contract).
- ***Manager → Responsible→ External_Company (0,N):*** Every Manager (Business Manager, in particular) may manage up to N External Companies.
- ***External_Company → Responsible→ Manager (1,1)*****:** Every External Company must be managed by up a Manager (Business Manager, in particular).
- ***External_Company → Signature→ External_Contract (1,N):*** Every External Company must sign up to N External Contract with HelpDesk. (The External Company may want to sign a different Contract because maybe they are about different Products manufactured by them).
- ***External_Contract → Signature → External_Company (1,1):*** Every External Contract must be signed by an External Company.
- ***Cross_Functionals → Administration → External_Contract (0,N):*** Every Cross_Functionals Employee may administer up to N External Contracts. (Finance, in particular).They will indeed speak with the External Company about the Bureaucracy part (and Money)).
- ***External_Contract → Administration → Cross_Functionals(1,1):*** Every External Contract must be administered by a Cross_Functionals Employee (Finance, in particular).
- ***Cross_Functionals → Registration → Employee (0, N):*** Every Cross_Functionals Employee (Finance, in particular) may registrate up to N Employee in the HelpDesk's DB. (Indeed HR are the only one allowed to accomplish this task in the whole Company).
- ***Employee → Registration → Cross_Functionals (1,1):*** Every Employee must be Registered by a Cross_Functionals Employee (Human Resources, in particular).
- ***Cross_Functionals → Handler_Contract → Contract (0, N):*** Every Cross_Functional Employee may handle up to N Contracts on HelpDesk's Employees (Human Resources, in particular, are the only ones allowed to do so).

- ***Contract → Handler_Contract → Cross_Functionals (1, N):*** Every Contract must be handled by up to N Cross_Functionals Employees (Human Resource Employees, in particular).
- ***Employee → Sign → Contract (1, 1):*** Every Employee must Sign a Contract.
- ***Contract → Sign → Employee (1, 1):*** Every Contract must be Signed by an Employee.
- ***Employee → Achievement_Certification → Certification (0, N)*****:** Every Employee may achieve up to N Certifications.
- ***Certification → Achievement_Certification → Employee (0, N):*** Every Certification may be achieved by up to N Employees.
- ***Course_Company → Release → Certification (0, N):*** Every Course Company may release up to N Certifications.
- ***Certification → Release → Course_Company (1, 1):*** Every Certification must be released by a Course Company.
- ***Contract → Inclusion_Courses_Bonus → Courses_Bonus (1, N):*** Every Contract must include up to N Courses_Bonus.
- ***Courses_Bonus → Inclusion_Courses_Bonus → Contract (1, N):*** Every Courses_Bonus must be Included in up to N Contracts.
- ***Contract → Inclusion_IPAD_Benefit →IPAD_Benefit (0,1):*** Every Contract may include an Ipad_Benefit.
- ***IPAD_Benefit → Inclusion_IPAD_Benefit →Contract (0,N):*** Every IPAD may be Included in up to N Contracts.
- ***Contract → Inclusion_PC_Benefit →PC_Benefit (0,1):*** Every Contract may include a PC_Benefit.
- ***PC_Benefit → Inclusion_PC_Benefit →Contract (0, N):*** Every PC may be Included in up to N Contracts.
- ***Contract → Inclusion_Phone_Benefit →Phone_Benefit (0,1):*** Every Contract may include a Phone_Benefit.
- ***Phone_Benefit → Inclusion_Phone_Benefit →Contract (0, N):*** Every Phone may be Included in up to N Contracts.
- ***Contract → Inclusion_Car_Benefit → Car (0,1):*** Every Contract may include a Car.
- ***Car → Inclusion_Car_Benefit → Contract (0,N):*** Every Car may be Included in up to N Contracts.
- ***Contract → Inclusion_Book_Bonus → Book_Bonus (0,N):*** Every Contract may include up to N Book_Bonus.
- ***Book_Bonus → Inclusion_Book_Bonus →Contract (0,N):*** Every Book_Bonus may be Included in up to N Contracts.
- ***Contract → Inclusion_Food_Stamp →Food_Stamp (1,N)*** **:** Every Contract must include up to N Food_Stamp.
- ***Food_Stamp → Inclusion_Food_Stamp →Contract (1,N)*****:** Every Food_Stamp must be Included in up to N Contracts.
- ***Contract → Inclusion_Stock_Option →Stock_Option (0,N)*****:** Every Contract may include up to N Food_Stamp.

- ***Stock_Option → Inclusion_Stock_Options →Contract (0,N)*:** Every Stock_Option may be Included in up to N Contracts.
- ***Contract → Inclusion_Credit_Card →Credit_Card (0,1):*** Every Contract may include a Credit_Card.
- ***Credit_Card → Inclusion_Credit_Card →Contract (0, N):*** Every Credit_Card may be Included in up to N Contracts.
- ***Course_Company → Organization →Course (0, N):*** Every Course Company may Organize up to N Courses.
- ***Course → Organization → Course_Company (1,1):*** Every Course must be Organized by a Course Company.
- ***Course → Usage → Courses_Bonus (0,1):*** Every Course may use a Course Bonus.
- ***Courses_Bonus → Usage → Course (0,1):*** Every Course Bonus may be used in a Course.
- ***Techology_Company → Sale_IPAD → IPAD_Benefit (0, N):*** Every Technology Company may sell up to N IPAD. (Those will be used by 2nd Level Technicians.
- ***IPAD_Benefit → Sale_IPAD →Techology_Company (1,1)*:** Every IPAD must be sold by A Technology Company.
- ***Techology_Company → Sale_PC→ PC_Benefit (0, N):*** Every Technology company may sell up to N PC.
- ***PC_Benefit → Sale_PC → Techology_Company (1,1)*:** Every PC must be sold by A Technology Company.
- ***Techology_Company → Sale_TEL→ Phone_Benefit (0,N):*** Every Technology company may sell up to N Phones.
- ***Phone_Benefit → Sale_TEL → Techology_Company (1,1)*:** Every Phone must be sold by A Technology Company.
- ***Car_Benefit → Rent → Fleet_Company (1, 1):*** Every Car must be rented by a Fleet Company.
- ***Fleet_Company → Rent → Car_Benefit (0, N):*** Every Fleet Company may rent up to N Cars.
- ***Car_Benefit → Assignment → Parking (0,1):*** Every Car may have assigned parking to it.
- ***Parking → Assignment → Car_Benefit (0,1):*** Every Parking may be assigned to a Car_Benefit(Car assigned only to certain categories of Employees).
- ***Vehicle → Positioning → Parking (1,1):*** Every Vehicle must be positioned in one parking.
- ***Parking → Positioning → Vehicle (0,1)*:** Every Parking may be assigned with a Vehicle(Car assigned only to certain categories of Employees).
- ***Book_Bonus → Partnership →Store (1, N):*** Every Book Bonus must be in partnership with up to N Stores.
- ***Store → Partnership →Book_Bonus (0, N):*** Every Store may accept up to N Book Bonuses.
- **F*ood_Stamp → Affiliation_Food →Store (1, N):*** Every Food Stamp must be in affiliation with up to N stores.

- ***Store → Affiliation_Food →Food_Stamp (0, N):*** Every Store may accept up to N Food Stamps.
- ***Stock_Option → Exchanged → Bank (1,1):*** Every Stock Option must be Exchanged by a Bank.
- ***Bank → Exchanged → Stock_Option (0, N):*** Every Bank may Exchange up to N Stock Options.
- ***Credit_Card → Affiliation_CC →Bank (1,1):*** Every Credit Card must be affiliated with a Bank.
- ***Bank → Affiliation_CC → Credit_Card (0, N):*** Every Bank may have affiliated up to N Credit Cards.
- ***Employee → Leadership_Hierarchy → Employee (0, N):*** Every Employee may lead up to N Employee.
- ***Employee → Leadership_Hierarchy → Employee (1,1):*** Every Employee must be led by one Employee (belonging to the Leadership Team).
- ***Employee → Correspondence_1_st_Level_Tech → 1st_Level_Technician (0,1):*** Every Employee may correspond to a 1st_Level_Technician.
- ***1st_Level_Technician → Correspondence_1_st_Level_Tech → Employee (1,1):*** Every 1st_Level_Technician must correspond to an Employee.
- ***Employee → Correspondence_2_nd_Level_Tech → 2nd_Level_Technician (0,1):*** Every Employee may correspond to a 2nd_Level_Technician.
- ***2nd_Level_Technician → Correspondence_2_nd_Lev_Tech → Employee (1,1):*** Every 2nd_Level_Technician must correspond to an Employee.
- ***Employee → Correspondence_Manager → Manager (0,1):*** Every Employee may correspond to a Manager.
- ***Manager → Correspondence_Manager → Employee (1,1)*****:** Every Manager must correspond to an Employee.
- ***Employee → Correspondence_Cross_Functionals → Cross_Functionals (0,1):*** Every Employee may correspond to a Cross_Functionals Employee.
- ***Cross_Functionals → Correspondence_Cross_Functionals → Employee (1,1):*** Every Cross_Functionals Employee must correspond to an Employee.
- ***Employee → Living →Address_Employee (1, N):*** Each employee has one or more addresses linked to its account
- ***Address_Employee → Living → Employee (1,1):*** Each address is connected only to 1 Employee
- ***Employee →Communicate → Email_Employee(1, N)*****:** Each Employee must declare up to N of his/her email address
- ***Email_Employee → Communicate → Employee (1,1):*** Each email address is uniquely linked to one person
- ***Employee → Owning → Phone_Employee (0, N):*** Each employee may declare up to N different phone numbers
- ***Phone_Employee → Owning → Employee(1,1):*** each phone number is related only to one person
- ***Customer → Customer_Living → Address_Customer(1,N):*** Each Customer may either have a singular address or multiple addresses

- ***Address_Customer → Customer_Living → Customer (1, 1):*** Each address is related to only one Customer
- ***Customer → Cust_Owning → Email_Customer (1, N):*** Each Customer must have up to N email addresses
- ***Email_Costumer → Cust_Owning → Customer (1, 1):*** Each email address is connected to only one Customer
- ***Customer → Cust_Contact → Phone_Customer (0, N)*****:** Each customer may declare up to N phone numbers
- ***Phone_Customer → Cust_Contact → Customer (1, 1):*** Each phone number is related to only one person
- ***2nd_Level_Technician → Required → Driving_Licence (1, N):*** It is necessary that technicians have up to N driving licence to help the customer on-site
- ***Driving_Licence →Required→2nd_Level_Technician (1, 1):*** Each driving licence is uniquely owned by a technician.

ACCESS TABLE:

General:

- Select the Customer stars of the report referred to Tickets which have been closed on the same day on which they have been opened

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
Convalidation	R	1	R
Customer	E	1	R
Closure_Tickets	R	1	R
Closed_Tickets	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
Closure_Tickets	R	1	R
Tickets	E	1	R

- Find the number of Tickets having requests about products manufactured from a certain external company

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Ticket	E	1	R

Submission	R	1	R
Product	E	1	R
Customer	E	1	R
Record	R	1	R
Manufacture	R	1	R
External_ Company	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Ticket	E	1	R
Submission_ Ticket	R	1	R
Product	E	1	R
Customer	E	1	R
Record	R	1	R
Manufacture	R	1	R
External_ Company	E	1	R

- Select all the companies that hired HelpDesk that have at least a certain number of employees

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
External_Company	E	1	R

- Find the address of all the Customers where the day of start and end of the intervention is different

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Customer	E	1	R
Closure_Ticket	R	1	R
Closed_Ticket	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Customer	E	1	R
Customer_Living	R	1	R
Address_ Customer	E	1	R
Closure_Ticket	R	1	R
Ticket	E	1	R

- Insert the Customer Feedback of all the Customers where the day of start and end of the intervention is different

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	W
Convalidation	R	1	R
Customer	E	1	R
Closed_Ticket	E	1	R
Closure_Ticket	R	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	W
Convalidation	R	1	R
Customer	E	1	R
Ticket	E	1	R
Closure_Ticket	R	1	R

- Find the Company's names of the type of products that have requested more interventions

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Product	E	1	R
Manufacture	R	1	R
External_Com	E	1	R
Ticket	E	1	R

- Find all the Closed Tickets that in the Reports have defined that the Insurance on the product has expired

Concept	Element_Type	Access	Access_Type
Ticket	E	1	R
Closure_Ticket	R	1	R
Report	E	1	R

- Select the dates of the rent expiry on the HelpDesk cars used as benefit

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Car_Benefit	E	1	R

- Select the model of all the technological benefits that HelpDesk has offered, which are older than a certain "age"

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Contract	E	1	R
Inclusion	R	1	R
Car_Benefit	E	1	R
IPAD_Benefit	E	1	R
PC_Benefit	E	1	R
Phone_Benefit	E	1	R
Book_Bonus	E	1	R
Food_Stamp	E	1	R
Courses_Bonus	E	1	R
Stock_Options	E	1	R
Credit_Card	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Contract	E	1	R
Inclusion_Car_ Benefit	R	1	R
Car_Benefit	E	1	R
Inclusion_IPAD_ Benefit	R	1	R
IPAD_Benefit	E	1	R
Inclusion_PC_ Benefit	R	1	R
PC_Benefit	E	1	R
Inclusion_Phone_ Benefit	R	1	R
Phone_Benefit	E	1	R
Inclusion_Book_ Bonus	R	1	R
Book_Bonus	E	1	R
Inclusion_Food_ Stamp	R	1	R
Food_Stamp	E	1	R
Inclusion_ Courses_Bonus	R	1	R
Courses_Bonus	E	1	R
Inclusion_Stock_ Option	R	1	R
Stock_Options	E	1	R
Inclusion_Credit_ Card	R	1	R

Credit_Card	E	1	R

- Insert all the data of the client and the product needing assistance

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Customer	E	1	R
Ticket	E	1	W
Submission	R	1	R
Record	R	1	R
Product	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Customer	E	1	R
Ticket	E	1	W
Submission_Ticket	R	1	R
Record	R	1	R
Product	E	1	R

- Select all the Tickets with the highest level of priority still not closed

PRE-RESTRUTCURATION=POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Ticket	E	1	R

1st Level Technician:

- Assign a WorkStation to each 1st Level Technician

PRE-RESTRUTCURATION=POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
1st_Level_Tech	E	1	W

- Find how many times a certain 1st Level technician has viewed a Ticket

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
1st_Level_Tech	E	1	R
Viewing_Ticket	R	1	R
Ticket	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Employee	E	1	R
Correspondence_1st_Level_Tech	R	1	R
1st_Level_Tech	E	1	R
Viewing_Ticket	R	1	R
Ticket	E	1	R

- Find how many times a certain 1st Level technician has inspected a Report

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
1st_Level_Tech	E	1	R
Inspection	R	1	R
Report	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Employee	E	1	R
Correspondence_1st_Level_Tech	R	1	R
1st_Level_Tech	E	1	R
Inspection	R	1	R
Report	E	1	R

- Insert in the Report all the ID Customer and personal data of 1st Level

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
1st_Level_Tech	E	1	R
Compilation	R	1	R
Report	E	1	W
Customer	E	1	R
Convalidation	R	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
1st_Level_Tech	E	1	R
Compilation_1st_ Tech	R	1	R
Report	E	1	W
Customer	E	1	R
Convalidation_ Report	R	1	R

- Insert all the ID Customer and personal data of 1st Level Technicians that have closed a ticket without delegating to a 2nd Level Technicians.

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
1st_Level_Tech	E	1	W
Compilation	R	1	R

Ticket	E	1	R
Customer	E	1	R
Submission	R	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
1st_Level_Tech	E	1	W
Compilation_1st_ Tech	R	1	R
Ticket	E	1	R
Customer	E	1	R
Submission_Ticket	R	1	R

- Select the Course Company correspondent to the amount of certifications released to 1st Level Technicians

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course_Company	E	1	R
Realease	R	1	R
Certification	E	1	R
Achievement_Certif	R	1	R
Employee	E	1	R

2nd Level Technician:

- Find the personal data of all the 2nd Level Technicians who compiled a Report

PRE-RESTRUTCURATION=POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
2nd_Level_Tech	E	1	R

- Select all the PPE used by the 2nd Level Technicians to repair Products manufactured by a certain external company

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
PPE	E	1	R
PPE_Assigned	R	1	R
2nd_Level_Tech	E	1	R
Managemen	R	1	R
Ticket	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
PPE	E	1	R
PPE_Assigned	R	1	R
2nd_Level_Tech	E	1	R
Management_2nd_ Tech	R	1	R
Ticket	E	1	R

- Select all the Driving Licences Levels of the 2nd Level Technicians who have chosen a van

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
2nd_Level_Tech	E	1	R
Drive_Vehicle	R	1	R
Vehicle	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Drive_Licence	E	1	R
Required	R	1	R
2nd_Level_Tech	E	1	R
Drive_Vehicle	R	1	R
Vehicle	E	1	R

- Select the Courses Companies correspondent to the certifications released to 2nd Level Technicians

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course_Company	E	1	R
Realease	R	1	R
Certification	E	1	R
Achievement_Certif	R	1	R
Employee	E	1	R

- Find the Costumer Rate of a 2nd Level Technician who spent a certain amount of money using the Benefit Credit Card

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
2nd_Level_Technician	E	1	R
Compilation	R	1	R
Sign	R	1	R
Contract	E	1	R
Inclusion	R	1	R
Credit_Card	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
Compilation_2nd_Tech	R	1	R
2nd_Level_Technician	E	1	R
Correspondence_2nd_ Level	R	1	R
Employee	E	1	R
Sign	R	1	R
Contract	E	1	R
Inclusion_Credit_Card	R	1	R

Credit_Card	E	1	R

Account Manager:

- Select the Customer Star of the Technicians supervised by the Account Manager with the highest amount of Bonus

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
Compilation	R	1	R
Technician	E	1	R
Supervision	R	1	R
Account_Manager	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
Compilation_1_st_ Tech	R	1	R
Compilation_2nd_ Tech	R	1	R
1st_Level_Tech	E	1	R
2nd_Level_Tech	E	1	R
Supervised_1st_Tech	R	1	R
Supervised_2nd_Tech	R	1	R
Manager	E	1	R

- Select the amount of closed Tickets of all the Technicians under the responsibility of a certain Manager

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Ticket	E	1	R
Management	R	1	R
Technician	E	1	R
Supervision	R	1	R
Account_Manager	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Ticket	E	1	R
Management_1_st_ Tech	R	1	R
Management_2nd_ Tech	R	1	R
1st_Level_Tech	E	1	R
2nd_Level_Tech	E	1	R
Supervised_1st_Tech	R	1	R
Supervised_2nd_Tech	R	1	R
Manager	E	1	R

- Select the amount of Report compiled by all the 1st Level Technicians under the responsibility of a certain Manager

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
Compilation	R	1	R
1st_Level_Tech	E	1	R
Supervision	R	1	R
Account_Manager	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Report	E	1	R
Compilation_1_st_ Tech	R	1	R
1st_Level_Tech	E	1	R
Supervised_1st_Tech	R	1	R
Manager	E	1	R

- Select the Courses Companies correspondent to the certifications released to Account Managers

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course_Company	E	1	R
Release	R	1	R
Certification	E	1	R
Achievement_Certif	R	1	R
Employee	E	1	R

Business Manager:

- Select the amount of External Companies supervised by the Business Manager with the highest amount of bonuses

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
External_Companies	E	1	R
Responsible	R	1	R
External_Contract	E	1	R
Administration	R	1	R
Business_Manager	E	1	R
Sign			
Contract	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
External_Companies	E	1	R
Responsible	R	1	R
External_Contract	E	1	R
Administration	R	1	R
Manager	E	1	R
Sign			
Contract	E	1	R

- Insert the data of each Business manager who viewed a Ticket

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Business_Manager	E	1	R
Viewing_Ticket	R	1	W
Ticket	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Manager	E	1	R
Correspondence_ Manager	R	1	R
Employee	E	1	R
Viewing_Ticket	R	1	W
Ticket	E	1	R

- Find the number of the reserved parking space assigned to Business managers

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Business_Manager	E	1	R
Sign	R	1	R
Contract	E	1	R
Inclusion	E	1	R
Car_Benefit	E	1	R
Assigned	R	1	R

Parking	E	1	R
Reserved	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Manager	E	1	R
Sign	R	1	R
Contract	E	1	R
Inclusion_Car_Benefit	R	1	R
Car_Benefit	E	1	R
Assigned	R	1	R
Parking	E	1	R

- Select the Courses Company correspondent to the certifications released to Business Managers

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course_Company	E	1	R
Realease	R	1	R
Certification	E	1	R
Achievement	R	1	R
Employee	E	1	R

Finance:

- Insert all the info of the External_Company that will be used to fill out their Contract, which are administrated by a certain Finance Employess

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
External_Company	E	1	R
Signature	R	1	R
External_Contract	E	1	R
Administration	R	1	R
Finance	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
External_Company	E	1	R
Signature	R	1	R
External_Contract	E	1	R
Administration	R	1	R
Cross_Functional	E	1	R

- Select the model of the Car Benefit assigned to the Finance employee

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Car_Benefit	E	1	R
Inclusion	R	1	R
Contract	E	1	R
Sign	R	1	R
Employee	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Car_Benefit	E	1	R
Inclusion_Car_Benefit	R	1	R
Contract	E	1	R
Sign	R	1	R
Employee	E	1	R

- Select the Courses Company correspondent to the certifications released to Finance employee

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course_Company	E	1	R
Realease	R	1	R
Certification	E	1	R
Achievement	R	1	R
Employee	E	1	R

- Find the number of all the Courses Bonus spent by the Finance team

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course	E	1	R
Usage	R	1	R
Courses_Bonus	E	1	R
Inclusion	R	1	R
Contract	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course	E	1	R
Usage	R	1	R
Courses_Bonus	E	1	R
Inclusion_Courses_ Bonus	R	1	R
Contract	E	1	R

Human Resources(HR):

- Find the personal information of the new Employees hired from a certain employee of the Human Resources

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Employees	E	1	R
Registration	R	1	R
Human_Resources	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Living	R	1	R
Owning	R	1	R
Communicate	R	1	R
Address_Employee	E	1	R
Phone_Employee	E	1	R
Email_Employee	E	1	R
Employees	E	1	R
Registration	R	1	R
Cross_Functional	E	1	R

- Find the amount of the Contract handled by the HR employee with the highest number of Bonus

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Contract	E	1	R
Sign	R	1	R
Employee	E	1	R

Handler_Contract	R	1	R
Human_Resources	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Contract	E	1	R
Sign	R	1	R
Employee	E	1	R
Handler_Contract	R	1	R
Cross_Functional	E	1	R

- Select the model of the PC Benefit assigned to the Human Resources employee

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
PC_Benefit	E	1	R
Inclusion	R	1	R
Contract	E	1	R
Sign	R	1	R
Finance	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
PC_Benefit	E	1	R
Inclusion_PC_ Benefit	R	1	R
Contract	E	1	R
Sign	R	1	R
Finance	E	1	R

- Select the Courses Company correspondent to the certifications released to Human Resources employee

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Course_Company	E	1	R
Realease	R	1	R
Certification	E	1	R
Achievement	R	1	R
Employee	E	1	R

Leadership:

- Insert the data of all the Leadership Employee in HelpDesk company

PRE-RESTRUTCURATION=POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Employee	E	1	W

- Insert the credentials of the Credit Card given to Leadership employee

PRE-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Credit_Card	E	1	W
Inclusion	R	1	R
Contract	E	1	R
Sign	R	1	R
Employee	E	1	R

POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Credit_Card	E	1	W
Inclusion_Credit_Card	R	1	R
Contract	E	1	R
Sign	R	1	R
Employee	E	1	R

- Select the amount of Certification released to Leadership employees

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	Element_Type	Access	Access_Type
Certification	E	1	R
Achievement_ Certification	R	1	R

Employee	E	1	R

- Select the Course Company correspondent to the amount of certifications released to Leadership employees

PRE-RESTRUCTURATION = POST-RESTRUCTURATION

Concept	**Element_Type**	**Access**	**Access_Type**
Course_Company	E	1	R
Realease	R	1	R
Certification	E	1	R
Achievement	R	1	R
Employee	E	1	R

LOGIC SCHEMA:

- ***PHONE_CUSTOMER*** **(**Number_Customer, Country_Code_Customer, *ID_Customer*)
- ***EMAIL_CUSTOMER*** **(**Email_Address_Customer, *ID_Customer***)**
- ***ADDRESS_CUSTOMER*** **(**Address_Customer, Flat_Number_Customer, ZIP_Code_Customer, *ID_Customer*)
- ***CUSTOMER*** **(**ID_Customer, Level_Customer, Name_Customer, Surname_Customer, Username_Customer, Password_Customer)
- ***WATCHING_REPORT*** (*ID_Customer, ID_Report,* Date_Time_Watching)
- ***REPORT*** (ID_Report, Customer_Star, Customer_Feedback, Problem_Solved, Time_Intervent, Customer_Convalidation, Travel_Time, Date_Time_Convalidation, Date_Time_Compilation_Tech_1, Date_Time_Compilation_Tech_2, ID_Ticket, *ID_Customer*, *ID_Tech_1, ID_Techn_2*)
- ***INSPECTION*** (ID_Employee, ID_Report, Date_Time_Inspection)
- ***TICKET*** (ID_Ticket, Closed_Ticket, Product_Typology, Priority_Ticket, Company_Failure, Problem_Description, Name_Applicant, Date_Time_Submission, Date_Time_Closure, ID_Customer, ID_Product)
- ***VIEWING_TICKET*** (ID_Ticket, ID_Employee, Date_Time_Viewing_Ticket)
- ***MANAGEMENT_1ST_TECH*** (ID_Ticket, ID_Tech_1, Date_Time_Management_Tech_1)
- ***MANAGEMENT_2ND_TECH*** (ID_Ticket, ID_Tech_2, Date_Time_Management_Tech_2)
- ***REVISED_TICKET*** (*ID_Customer, ID_Ticket,* Date_Time_Revising)
- ***PRODUCT*** (ID_Product, Weight, Product_Level, Category, Model, Date_Buying, Insurance_Expiry, Data_Manifacture,Date_Time_Product_Recording, *ID_Customer, ID_External_Company*)
- ***EXTERNAL_COMPANY*** (ID_External_Company, External_Company_Name, External_Company_Level, *ID_Business_Manager*)
- ***EXTERNAL_CONTRACT*** (ID_External_Contract, Amount_Contract, Validity_Contract, Date_Time_Signature_External_Contract, *ID_Finance_Employee, ID_External_Company*)
- ***EMPLOYEE*** (ID_Employee, Level_Employee, Employee_Type, Name_Employee, Surname_Employee, Number_Bank_Account, Username_Employee, Password_Employee, Parking_Number, *ID_Human_Resources*)
- ***LEADERSHIP_HIERARCHY*** (*ID_Workforce_Employee, ID_Leadership_Employee*)
- ***ADDRESS_EMPLOYEE*** (ZIP_Code, Address_Empl, Flat_Employee, *ID_Employee)*
- ***PHONE_EMPLOYEE*** (Number_Employee, Country_Code_Employee,*ID_Employee)*
- ***EMAIL_EMPLOYEE*** (Email_Address_Employee, *ID_Employee)*
- ***1ST_LEVEL_TECHNICIAN*** (*ID_Tech_1,* Number_Workstation, *ID_Account_Manager*)

- ***DELEGATION_TECHNICIAN*** (*ID_Tech_1, ID_Tech_2,* Date_Time_Delegation_Technician, ID_Ticket)
- ***2ND_LEVEL_TECHNICIAN*** (*ID_Tech_2,* Shoes_Size, Helmet_Size, Clothing_Size, *ID_Account_Manager*)
- ***DRIVE_LICENSE*** (ID_Driving_License, Level_Driving_License, Expiry_Date_Driving_License, *ID_Tech_2*)
- ***PPE_ASSIGNED*** (*ID_Tech_2, ID_PPE*)
- ***PPE*** (ID_PPE, Size_PPE, Type_PPE, Visor, Brand_PPE)
- ***MADE_OF_MATERIAL***(*ID_PPE, ID_Material*)
- ***MATERIAL*** (ID_Material**,** Name_Material)
- ***VEHICLE*** (Plate_Number, Insurance_Validity_Up_To, Drive_License_Needed, Kilometers_Traveled, Model_Vehicle, Vehicle_Type, Brand_Vehicle, Price_Vehicle, Vehicle_Level_Old, Vehicle_Level_Price, Camera_Assist, Attention_Assist, Security_Clothing, GPS_Assigned, Weight_Limit, Engine_Cylinder_Volume_Vehicle, Horsepower_Vehicle, Number_Seats_Vehicle, *ID_Fleet_Company*)
- ***DRIVE_VEHICLE*** (*ID_Tech_2, ID_Vehicle,* Date_Time_Drive_Vehicle)
- ***PARKING*** (ID_Park, Type_Parking, ID_Employee_Reserved)
- ***CAR_BENEFIT*** (ID_Car_Benefit, Model_Car, Assigned_Up_To_Car, Price_Car, Brand_Car, Horsepower_Car, Engine_Cylinder_Volume_Car, Rent_Price_Car, Number_Seats_Car, Insurance_Validity_Up_To, Kilometers_Traveled, Rent_Expiry_Date, Car_Level_Price, *ID_Fleet_Company*)
- ***FLEET_COMPANY*** (ID_Fleet_Company, Fleet_Bank_Account, Address_Fleet_Company, Telephone_Fleet_Company, Name_Fleet_Company)
- ***CONTRACT*** (ID_Contract, Type_Contract, Contract_Expiration, Bonus, Insurance, Holidays_Left, Salary, Hiring_Date, ID_Employee, *ID_Car_Benefit, ID_Phone_Benefit, ID_Ipad_Benefit, ID_PC_Benefit, ID_Benefit_Credit_Card)*
- ***INCLUSION_CAR_BENEFIT*** (*ID_Car_Benefit, ID_Contract)*
- ***MANAGER*** (*ID_Manager,* Type_Manager)
- ***CROSS_FUNCTIONAL*** (*ID_Cross_Functional,* Type_Cross_Functional)
- ***HANDLER_CONTRACT*** (*ID_Contract, ID_Human_Resources*)
- ***CERTIFICATION*** (ID_Certification**,** Expiration_Date_Certification, *ID_Course_Company*)
- ***ACHIEVEMENT_CERTIFICATION*** (*ID_Certification, ID_Employee,* Date_Time_Certification)
- ***COURSE_COMPANY*** (ID_Course_Company, Address_Course_Company, Telephone_Course_Company, Email_Course_Company, Name_Course_Company)
- ***COURSE*** (ID_Course, Expiry_Date_Inscription, Category_Course, Price_Course, *ID_Course_Company*)
- ***COURSES_BONUS*** (ID_Benefit_Courses_Bonus, Value_Courses_Bonus, Expiry_Date_Bonus, Date_Time_Usage)
- ***INCLUSION_COURSES_BONUS*** (*ID_Benefit_Courses_Bonus, ID_Contract*)

- ***PHONE_BENEFIT*** (ID_Benefit_Phone_Benefit, Assigned_Up_To_Phone, OS_Phone, Brand_Phone, Width_phone, Height_Phone, Price_Phone, Inch_Screen_Phone,Model_Phone_Benefit, *ID_Company_Tech*)
- ***IPAD_BENEFIT*** (ID_Benefit_IPAD_Benefit, Assigned_Up_To_IPAD, OS_IPAD, Brand_IPAD, Width_IPAD, Height_IPAD, Price_IPAD, Inch_Screen_IPAD, Model_IPAD_Benefit, *ID_Company_Tech*)
- ***TECHNOLOGY_COMPANY*** (ID_Company_Tech, Category_Tech_Company, Address_Tech_Company, Name_Company_Tech)
- ***PC_BENEFIT*** (ID_Benefit_PC_Benefit, Assigned_Up_To_PC, OS_PC, Brand_Pc, Width_PC, Height_PC, Price_PC, HD_Dimension_PC, RAM_Dimension_PC, Inch_Screen_PC,Model_PC_Benefit, *ID_Company_Tech*)
- ***BOOK_BONUS*** (ID_Benefit_Book_Bonus, Expiry_Date_Book_Bonus, Value_Book_Bonus, Date_Time_Usage)
- ***INCLUSION_BOOK_BONUS*** (*ID_Contract, ID_Benefit_Book_Bonus*)
- ***FOOD_STAMP*** (ID_Benefit_Food_Stamp, Expiry_Date_Food_Stamp, Value_Food_Stamp, Date_Time_Usage)
- ***INCLUSION_FOOD_STAMP*** (*ID_Contract, ID_Benefit_Food_Stamp)*
- ***STORE*** (ID_Store, Store_Type, Address_Store, Store_Modality, Name_Store)
- ***PARTNERSHIP*** (*ID_Benefit_Book_Bonus, ID_Store,* Date_Time_Expiration_Partnership)
- ***AFFILIATION*** (*ID_Benefit_Food_Stamp, ID_Store,* Date_Time_Expiration_Affiliation)
- ***STOCK_OPTION*** (ID_Benefit_Stock_Option, Price_When_Assigned, Volume, *ID_Bank*)
- ***INCLUSION_STOCK_OPTION*** (*ID_Contract, ID_Benefit_Stock_Option*)
- ***CREDIT_CARD*** (ID_Benefit_Credit_Card, IBAN, Expiry_Date_Credit_Card, Credit_Limit, *ID_Bank*)
- ***BANK*** (ID_Bank, Reference_Email, Reference_Phone, Reference_Address, Bank_Type, Name_Store)

POSTGRESQL ADDED DOMAINS

TYPES:

Type	SQL Implementation
{XS, S, M, L, XL, XXL}	Create type Size_Clothing as enum (‘XS’, ‘S’, ‘M’, ‘L’, ‘XL’, ‘XXL’)
{Metal, Bronze, Silver, Gold, Platinum}	Create type Level as enum (‘Metal’, ‘Bronze’, ‘Silver’, ‘Gold’, ‘Platinum’)
{Android, I-Phone OS, Palm OS, Blackberry, Windows Mobile, Symbian, EMUI, ColorOS}	Create type Type_OS_Phone as enum ('Android', 'I-Phone OS', 'Palm OS', 'Blackberry', 'Windows Mobile', 'Symbian', 'EMUI', 'ColorOS')
{Account Manager, Business Manager}	Create type Type_Manager as enum (‘Account Manager’, ‘Business Manager’)
{Human Resources, Finance}	Create type Type_CrossFunctionals as enum (‘Human Resources’, ‘Financer’)
{Physical Store, Online Store}	Create type Modality_Store as enum (‘Physical Store’, ‘Online Store’)

{4 GB, 8 GB, 16 GB, 32 GB, 64 GB}	Create type Dimension_RAM_PC as enum (‘4 GB’, ‘8 GB’, ‘16 GB’, ‘32 GB’ ‘64 GB’)
{AM, A1, A2, A, B1, B, BE, B96, C1, C1E, C, CE}	Create type Type_DriveLicence as enum (‘AM’, ‘A1’, ‘A2’, ‘A’, ‘B1’, ‘B’, ‘BE’, ‘B96’, ‘C1’, ‘C1E’, ‘C’, ‘CE’)
{Helmet, Gloves, Safety Jacket, Safety Shoes}	Create type PPE as enum (‘Helmet’, ‘Gloves’, ‘Safety Jacket’, ‘Safety Shoes’)
{Fixed-Term, Permanent}	Create type Type_Contract as enum (‘Fixed-Term’,

	‘Permanent’)
{Domestic Appliance, Electronic Products, Elevators}	Create type Type_Product as enum (‘Domestic Appliance’, ‘Electronic Products’, ‘Elevators’)
{Leadership, Workforce, CEO}	Create type Type_Employee as enum (‘Leadership’, ‘Workforce’, ‘CEO’)
{Android, iPadOS, Windows, HUAWEI}	Create type Type_OS_IPAD as enum (‘Android’, ‘iPadOS’, ‘Windows’, 'HUAWEI')
{Linux, MacOS, Windows, Chrome OS}	Create type Type_OS_PC as enum (‘Linux’, ‘MacOS’, ‘Windows’ , 'Chrome OS')
{Free, Reserved}	Create type Type_Parking as enum (‘Free’, ‘Reserved’)
{Shop, Library}	Create type Store_Type as enum (‘Shop’, ‘Library’)
{Credit Company, Stock Company}	Create type Bank_Type as enum (‘Credit Company’,’ Stock Company’)
{500 GB, 750 GB, 1 Tr, 2 Tr, 3 Tr, 4	Create type HD_Dimension_PC as

Tr, 256 GB}	enum (‘500 GB’,’ 750 GB’,’ 1 Tr’,’ 2 Tr’, ‘3 Tr’, ‘4 Tr’,'256 GB')
{IDCT}	CREATE TYPE public.id_cert_cost AS ENUM ('IDCT')
Composite Type	CREATE TYPE public.id_certification AS (id_cert id_cert_cost, value_certification numeric(6,0));
Composite Type	CREATE TYPE public.email AS (utente text, chiocciola chiocciola, email_server text, dot dot, dominio text);
{@}	CREATE TYPE public.chiocciola AS ENUM ('@');
{.}	CREATE TYPE public.dot AS ENUM ('.');

Composite Type	CREATE TYPE public.id_bank AS (id_bank id_bank_cost, value_bank numeric(10,0));
{IDBK}	CREATE TYPE public.id_bank_cost AS ENUM ('IDBK');
Composite Type	CREATE TYPE public.id_foodbonus AS (id_food id_food_cost, value_food numeric(15,0));
{IDBFD}	CREATE TYPE public.id_food_cost AS ENUM ('IDBFD');
Composite Type	CREATE TYPE public.id_bookbonus AS (id_bookbonus id_bookbonus_cost, value_bookbonus numeric(15,0));
{IDBCR}	CREATE TYPE public.id_carben_cost AS ENUM

	('IDBCR');
Composite Type	CREATE TYPE public.id_carbenefit AS (id_carbenefit id_carben_cost, value_plate text);
Composite Type	CREATE TYPE public.id_contract AS (id_contr id_contract_cost, value_con numeric(10,0));
{IDCON}	CREATE TYPE public.id_contract_cost AS ENUM ('IDCON');
Composite Type	CREATE TYPE public.id_course AS (id_couse id_course_cost, value_course numeric(5,0));
{IDCRS}	CREATE TYPE public.id_course_cost AS ENUM ('IDCRS');
{IDBCB}	CREATE TYPE public.id_coursebon_cost AS ENUM

	('IDBCB');
Composite Type	CREATE TYPE public.id_coursebonus AS (id_cousebonus id_coursebon_cost, value_coursebonus numeric(10,0));
{IDCCP}	CREATE TYPE public.id_coursecom_cost AS ENUM ('IDCCP');
Composite Type	CREATE TYPE public.id_coursecompany AS (id_cousecompany id_coursecom_cost, value_ccp numeric(10,0));
{IDBCC}	CREATE TYPE public.id_credcard_cost AS ENUM ('IDBCC');
Composite Type	CREATE TYPE public.id_creditcard AS (id_creditcard id_credcard_cost, value_creditcard numeric(10,0));

{IDCU}	CREATE TYPE public.id_cust_cost AS ENUM ('IDCU');
Composite Type	CREATE TYPE public.id_customer AS (id_customer id_cust_cost, value_customer numeric(10,0));
Composite Type	CREATE TYPE public.id_drivelicence AS (id_dl1 character varying(2), value_numeric numeric(4,0), id_dl2 character varying(2));
{IDEXCP}	CREATE TYPE public.id_extcompany_cost AS ENUM ('IDEXCP');
{IDEXCT}	CREATE TYPE public.id_extcontract_cost AS ENUM ('IDEXCT');
Composite Type	CREATE TYPE

	public.id_externalcompany AS (id_extcom id_extcompany_cost, value_extcomp numeric(6,0));
Composite Type	CREATE TYPE public.id_externalcontract AS (id_extcontr id_extcontract_cost, value_extcon numeric(6,0));
{IDBFC}	CREATE TYPE public.id_fleetcom_cost AS ENUM ('IDBFC');
Composite Type	CREATE TYPE public.id_fleetcompany AS (id_fleetcompany id_fleetcom_cost, value_fleet numeric(10,0));
{IDBIP}	CREATE TYPE public.id_ipad_cost AS ENUM ('IDBIP');
Composite Type	CREATE TYPE public.id_ipadbenefit AS

	(id_ipad id_ipad_cost, value_ipad numeric(10,0));
Composite Type	CREATE TYPE public.id_material AS (code_material id_material_cost, value_material numeric(5,0));
{IDMAT}	CREATE TYPE public.id_material_cost AS ENUM ('IDMAT');
{IDPRK}	CREATE TYPE public.id_parking AS ENUM ('IDPRK');
Composite Type	CREATE TYPE public.id_pc AS (code_pc id_pc_const, value_pc numeric(10,0));
{IDBPC}	CREATE TYPE public.id_pc_const AS ENUM ('IDBPC');

Composite Type	CREATE TYPE public.id_phone AS (code_phone id_phone_const, value_phone numeric(10,0));
{IDBPH}	CREATE TYPE public.id_phone_const AS ENUM ('IDBPH');
Composite Type	CREATE TYPE public.id_ppe AS (code_ppe id_ppe_cost, value_ppe numeric(10,0));
{IDPPE}	CREATE TYPE public.id_ppe_cost AS ENUM ('IDPPE');
Composite Type	CREATE TYPE public.id_product AS (code_produtc id_product_const, value_product numeric(10,0));
{IDPRD}	CREATE TYPE public.id_product_const AS ENUM ('IDPRD');

Composite Type	CREATE TYPE public.id_report AS (id_rep id_report_cost, value_report numeric(10,0));
{IDRPT}	CREATE TYPE public.id_report_cost AS ENUM ('IDRPT');
{IDBSO}	CREATE TYPE public.id_stock_cost AS ENUM ('IDBSO');
Composite Type	CREATE TYPE public.id_stockoption AS (id_stock id_stock_cost, value_stock numeric(10,0));
Composite Type	CREATE TYPE public.id_store AS (id_store id_store_cost, value_store numeric(10,0));
{IDST}	CREATE TYPE public.id_store_cost AS ENUM ('IDST');

{IDTHC}	CREATE TYPE public.id_techcom_cost AS ENUM ('IDTHC');
Composite Type	CREATE TYPE public.id_techcompany AS (id_techcom id_techcom_cost, value_techcom numeric(10,0));
Composite Type	CREATE TYPE public.id_ticket AS (code_ticket id_ticket_cost, value_ticket numeric(10,0));
{IDTCK}	CREATE TYPE public.id_ticket_cost AS ENUM ('IDTCK');
Composite Type	CREATE TYPE public.id_vehicle AS (id_dl1 character varying(2), value_numeric numeric(3,0), id_dl2 character varying(2));
{IDWAM, IDWT1, IDWT2, IDWBM, IDWFIN, IDWHR, IDL}	CREATE TYPE public.id_workforce_general AS ENUM ('IDWAM', 'IDWT1', 'IDWT2',

	'IDWBM', 'IDWFIN', 'IDWHR', 'IDL');
{Safety_Jacket, Helmet, Knee_Pads, Reflective Jacket, Boots}	CREATE TYPE public.safety_clothing AS ENUM ('Safety_Jacket', 'Helmet', 'Knee_Pads', 'Reflective Jacket', 'Boots');

DOMAINS:

Domain	SQL Domain
[1,5]	Create domain Value_Customer_Star as integer check (value >= 1 and value <= 5)
[1,3]	Create domain Value_Priority as integer check (value >= 1 and value <= 3)
[35, 50]	Create domain Shoes_Size as integer check (value >= 35 and value <= 50)
[0,30]	Create domain Holidays_Left as integer check (value >= 0 and value <= 30)
[2,9]	Create domain Seats_Available as integer check (value >= 2 and value <= 9)
[18,30]	Create domain Dimension_Display_PC as integer check (value >= 18 and value <= 30)
[0, 13]	Create domain Level_Old_Vehicle as integer check (value >= 0 and value <= 13)
[0,30]	Create domain Number_WorkStation as integer check (value >= 0 and value <= 30)

[000, 9999999]	Create domain ZIP_Code as integer check (value >= 000 and value <= 9999999)
[160, 950]	Create domain public.weight_limit as integer check (value >= 160 andvalue<= 950)
[0, 200000]	Create domain Km_Travelled as integer check (value >= 0 and value <= 200000)

ENTITY ATTRIBUTES AND THEIR DATATYPE:

Entity	Attribute	DataType	Reference	Null?	Primary Key
1st_Tech	ID_Tech_1	id_employee	Employee (ID_ Employee)	NOT NULL	✓
1st_Tech	Number Workstation	number workstation		NOT NULL	X
1st_Tech	ID_Account_ Manager	id_employee	Manager (ID_ Manager)	NOT NULL	X
2nd_Tech	ID_Tech_2	id_employee	Employee (ID_ Employee)	NOT NULL	✓
2nd_Tech	Shoes_Size	shoes_size		Optional	X
2nd_Tech	Helmet_Size	size_clothing		Optional	X
2nd_Tech	Clothing_Size	size_clothing		Optional	X
2nd_Tech	ID_Account_ Manager	id_employee	Manager (ID_ Manager)	NOT NULL	X
Achievement _Certification	ID_ Certification	id_certification	Certification (ID_ Certification)	NOT NULL	✓
Achievement _Certification	ID_Employee	id_employee	Employee (ID_ Employee)	NOT NULL	✓
Achievement _Certification	Date_Time_ Certification	timestamp without time zone		NOT NULL	X

Address _ Customer	Address _ Customer	character varying		NOT NULL	✓
Address _ Customer	Flat_Number_ Customer	character varying		NOT NULL	✓
Address _ Customer	ZIP_Code_ Customer	zip_code		NOT NULL	✓
Address _ Customer	ID_Customer	id_customer	Customer(ID _Customer)	NOT NULL	X
Address_ Employee	ZIP_Code	zip_code		NOT NULL	✓
Address_ Employee	Address_ Empl	character varying		NOT NULL	✓
Address_ Employee	Flat_Employee	character varying		NOT NULL	✓
Address_ Employee	ID_Employee (UNIQUE)	id_employee	Employee (ID_ Employee)	NOT NULL	X
Affiliation	ID_Benfit_ Food_Stamp	id_ben_food	Benefit_Food _Stamp (ID_Benfit_ Food_Stamp)	NOT NULL	✓
Affiliation	ID_Store	id_store	Store (ID_Store)	NOT NULL	✓
Affiliation	Date_Time_ Expiration	date		Optional	X
Bank	ID_Bank	id_bank		NOT NULL	✓
Bank	Reference_	email		NOT NULL	X

	Email				
Bank	Reference_ Phone	numeric(10,0)		Optional	X
Bank	Reference_ Address	character varying		Optional	X
Bank	Bank_Type	bank_type		Optional	X
Bank	Name_Bank	text		NUT NULL	X
Book_Bonus	ID_Benefit_ Book_Bonus	id_bookbonus		NOT NULL	✓
Book_Bonus	Expiry_Date_ Book_Bonus	date		NOT NULL	X
Book_Bonus	Value_Book_ Bonus	integer		NOT NULL	X
Book_Bonus	Data_Time_ Usage	timestamp without time zone		Optional	X
Car_Benefit	ID_Car_ Benefit	id_carbenefit		NOT NULL	X
Car_Benefit	Model_Car	character varying		NOT NULL	X
Car_Benefit	Assigned_Up_ To_Car	date		NOT NULL	X
Car_Benefit	Price_Car	integer		NOT NULL	X
Car_Benefit	Brand_Car	character varying		NOT NULL	X
Car_Benefit	Horsepower_ Car	integer		NOT NULL	X
Car_Benefit	Engine_	integer		NOT NULL	X

	Cylinder_ Volume_ Vehicle				
Car_Benefit	Rent_Price_ Car	integer		NOT NULL	X
Car_Benefit	Number_ Seats_Car	seats_available		NOT NULL	X
Car_Benefit	Insurance_ Validity_Up_ To	date		NOT NULL	X
Car_Benefitv	Kilometers_ Traveled	km_travelled		NOT NULL	X
Car_Benefit	Rent_Expiry_ Date	date		NOT NULL	X
Car_Benefit	Car_Level_ Price	level_value		NOT NULL	X
Car_Benefit	ID_Fleet_ Company	id_ fleetcompany	Fleet_ Company (ID_Fleet_ Company)	NOT NULL	X
Certification	ID_ Certification	id_certification		NOT NULL	✓
Certification	ID_Course_ Company	id_ coursecompany	Company_ Course(ID_ Course_ Company)	NOT NULL	X
Certification	Expiration_ Date_ Certification	boolean		NOT NULL	X
Contract	ID_Contract	id_contract		NOT NULL	✓

Contract	Type_Contract	type_contract		NOT NULL	X
Contract	Contract_ Expiration	date		Optional	X
Contract	Bonus	integer		Optional	X
Contract	Insurance	text		NOT NULL	X
Contract	Holidays_Left	holidays_left		NOT NULL	X
Contract	Salary	integer		NOT NULL	X
Contract	Hiring_Date	date		NOT NULL	X
Contract	ID_Employee	id_employee	EMPLOYEE (ID_ Employee)	NOT NULL	X
Contract	ID_Car_ Benefit (UNIQUE)	id_carbenefit	CAR_ BENEFIT (ID_Car_ Benefit)	Optional	X
Contract	ID_Phone_ Benefit (UNIQUE)	id_phone	PHONE_ BENEFIT (ID_Benefit_ Phone_ Benefit)	Optional	X
Contract	ID_IPAD_ Benefit (UNIQUE)	id_ipadbenefit	IPAD_ BENEFIT (ID_Benefit_ IPAD_ Benefit)	Optional	X
Contract	ID_PC_ Benefit (UNIQUE)	id_pc	PC_ BENEFIT(ID _Benefit_PC _Benefit)	Optional	X
Contract	ID_Benefit_ Credit_Card (UNIQUE)	id_creditcard	CREDIT_ CARD (ID_Benefit_ Credit_Card)	Optional	X

Course	ID_Course	id_course		NOT NULL	✓
Course	Expiry_Date_ Inscription	date		NOT NULL	X
Course	Category_ Course	text		NOT NULL	X
Course	Price_Course	numeric		NOT NULL	X
Course	ID_Course_ Company	id_ coursecompany	COURSE_ COMPANY (ID_Course_ Company)	NOT NULL	X
Course_ Bonus	ID_Benefit_ Course_Bonus	id_cousebonus		NOT NULL	✓
Course_ Bonus	Value_Course_ Bonus	integer		NOT NULL	X
Course_ Bonus	Expiry_Date_ Bonus	date		NOT NULL	X
Course_ Bonus	Date_Time_ Usage	timestamp without time zone		Optional	X
Course_ Company	ID_ course_ company	id_ coursecompany		NOT NULL	✓
Course_ Company	Address_ Course_ Company	character varying		Optional	X
Course_ Company	Telephone_ Course_ Company	numeric (10,0)		Optional	X
Course_ Company	Email_ Course_	email		Optional	X

	Company				
Course_ Company	Name_Course _Company	text		NOT NULL	X
Credit_Card	ID_Benefit_ Credit_Card	id_creditcard		NOT NULL	✓
Credit_Card	IBAN (UNIQUE)	character varying		NOT NULL	X
Credit_Card	Expiry_Date_ Credit_Card	date		NOT NULL	X
Credit_Card	Credit_Limit	integer		NOT NULL	X
Credit_Card	ID_Bank	id_bank	Bank (ID_Bank)	NOT NULL	X
Cross_ Functional	ID_Cross_ Functional	id_employee	Employee (ID_ Employee)	NOT NULL	✓
Cross_ Functional	Type_Cross_ Functional	type_ crossfunctional	Employee (Employee_ Type)	NOT NULL	X
Customer	ID_Customer	id_customer		NOT NULL	✓
Customer	Name_ Customer	character varying		NOT NULL	X
Customer	Surname_ Customer	character varying		NOT NULL	X
Customer	Username_ Customer	character varying		NOT NULL	X
Customer	Password_ Customer	character varying		NOT NULL	X

Customer	Level_ Custemer	level_value		NOT NULL	X
Delegation_ Technician	ID_Tech_1	id_employee	1st_Level_ Technician (ID_ Tech_1)	NOT NULL	✓
Delegation_ Technician	ID_Tech_2	id_employee	2nd_Level_ Technician (ID_ Tech_2)	NOT NULL	✓
Delegation_ Technician	Date_Time_ Delegation_ Technician	timestamp without time zone		NOT NULL	X
Delegation_ Technician	ID_Ticket	id_ticket	TICKET(ID_ Ticket)	NOT NULL	X
Drive_ Licence	ID_Drive_ Licence	id_drivelicence		NOT NULL	✓
Drive_ Licence	Level_Driving _Licence	type_ drivelicence		NOT NULL	X
Drive_ Licence	Expiry_Date_ Driving_ Licence	date		NOT NULL	X
Drive_ Licence	ID_Tech_2 (UNIQUE)	id_employee	2nd_Level_ Technician (ID_ Tech_2)	NOT NULL	X
Drive_ Vehicle	ID_Tech_2	id_employee	2nd_Level_ Technician (ID_ Tech_2)	NOT NULL	✓
Drive_ Vehicle	ID_Vehicle	id_vehicle	Vehicle (ID_Vehicle)	NOT NULL	✓

Drive_ Vehicle	Date_Time_ Drive_Vehicle	timestamp without time zone		NOT NULL	X
Email_ Customer	Email_ Address_ Customer	email		NOT NULL	✓
Email_ Customer	ID_Customer (UNIQUE)	id_customer	Customer (ID_ Customer)	NOT NULL	X
Email_ Employee	Email_ Address_ Employee	email		NOT NULL	✓
Email_ Employee	ID_Employee (UNIQUE)	id_employee	Employee (ID_ Employee)	NOT NULL	✓
Employee	ID_Employee	id_employee		NOT NULL	✓
Employee	Level_ Employee	level_value		NOT NULL	X
Employee	Employee_ Type	type_employee		NOT NULL	X
Employee	Name_ Employee	character varying		NOT NULL	X
Employee	Surname_ Employee	character varying		NOT NULL	X
Employee	Number_Bank _Account (UNIQUE)	character varying		NOT NULL	X
Employee	Username_ Employee	character varying		NOT NULL	X

Employee	Password_ Employee	numeric (10,0)		NOT NULL	X
Employee	Parking_ Number	character varying		Optional	X
Employee	ID_Human_ Resources	id_employee	Cross_Functional(ID_Cross_Functional)	NOT NULL	X
External_ Company	ID_External_ Company	id_externalcompany		NOT NULL	✓
External_ Company	External_ Company_ Level	level_value		NOT NULL	X
External_ Company	ID_Business_ Manager	id_employee	Manager(ID_ Manager)	NOT NULL	X
External_ Company	External_ Company_ Name	text		NOT NULL	X
External_ Contract	ID_External_ Contract	id_externalcontract			✓
External_ Contract	Amount_ Contract	numeric		NOT NULL	X
External_ Contract	Validity_ Contract	date		NOT NULL	X
External_ Contract	Date_Time_ Signature_ External_ Contract	timestamp without time zone		NOT NULL	X
External_ Contract	ID_External_ Company	id_externalcompany	External_ Compant (ID_External_Company)	NOT NULL	X

External_ Contract	ID_Finance_ Employee	id_employee	Cross_ Functional (ID_Cross_ Functional)	NOT NULL	X
Fleet_ Company	ID_Fleet_ Company	id_ fleetcomapny		NOT NULL	✓
Fleet_ Company	Fleet_Bank_ Account (UNIQUE)	character varying		NOT NULL	X
Fleet_ Company	Address_Fleet _Company (UNIQUE)	character varying		NOT NULL	X
Fleet_ Company	Telephone_ Fleet_Compay (UNIQUE)	numeric (10,0)		NOT NULL	X
Fleet_ Company	Name_Fleet_ Company	text		NOT NULL	X
Food_Stamp	ID_Benefit_ Food_Stamp	id_foodbonus		NOT NULL	✓
Food_Stamp	Expiry_Date_ Food_Stamp	date		NOT NULL	X
Food_Stamp	Value_Food_ Stamp	integer		NOT NULL	X
Food_Stamp	Date_Time_ Usage	timestamp without time zone		Optional	X
Handler_ Contract	ID_Contract	id_contract	Contract (ID_ Contract)	NOT NULL	✓

Handler_ Contract	ID_Human_Re sources	id_employee	Cross_ Functional (ID_ Cross_ Functional)	NOT NULL	✓
Inclusion_ Book_Bonus	ID_Contract	id_contract		NOT NULL	✓
Inclusion_ Book_Bonus	ID_Benefit_ Book_Bonus	id_bookbonus	Benefit_ Book_Bonus (ID_Benefit_ Book_Bonus)	NOT NULL	✓
Inclusion_ Car_Benefit	ID_Contract	id_contract	Contract(ID_ Contract)	NOT NULL	✓
Inclusion_ Car_Benefit	ID_Car_ Benefit	id_carbenefit	Car_Benefit (ID_Car_ Benefit)	NOT NULL	✓
Inclusion_ Courses _Bonus	ID_Benefit_ Courses_ Bonus	id_coursebonus	Course_ Bonus (ID_Benefit_ Courses_ Bonus)	NOT NULL	✓
Inclusion_ Courses_ Bonus	ID_Contract	id_contract	Contract(ID_ Contract)	NOT NULL	✓
Inclusion_ Food_Stamp	ID_Contract	id_contract	Contract(ID_ Contract)	NOT NULL	✓
Inclusion_ Food_Stamp	ID_Benefit_ Food_Stamp	id_ben_food	Food_Stamp (ID_Benefit_ Food_Stamp)	NOT NULL	✓

Inclusion_ Stock_Option	ID_Contract	id_contract	Contract(ID_ Contract)	NOT NULL	✓
Inclusion_ Stock_Option	ID_Benefit_ Stock_Option	id_stockoption	Stock_Option (ID_Benefit_ Stock_ Option)	NOT NULL	✓
Inspection	ID_Employee	id_employee	Employee (ID_ Employee)	NOT NULL	✓
Inspection	ID_Report	id_report	Report (ID_Report)	NOT NULL	X
Inspection	Date_Time_ Inspection	timestamp without time zone		NOT NULL	X
IPAD_Benefit	ID_Benefit_ IPAD_Benefit	id_ipadbenefit		NOT NULL	✓
IPAD_Benefit	Assigned_Up_ To_IPAD	date		NOT NULL	X
IPAD_Benefit	OS_IPAD	type_os_ipad		NOT NULL	X
IPAD_Benefit	Brand_IPAD	character varying		NOT NULL	X
IPAD_Benefit	Width_IPAD	integer		NOT NULL	X
IPAD_Benefit	Height_IPAD	integer		NOT NULL	X
IPAD_Benefit	Price_IPAD	integer		NOT NULL	X
IPAD_Benefit	Inch_Screen_ IPAD	integer		NOT NULL	X
IPAD_Benefit	ID_Company_ Tech	id_ techcompany	Technology_ Company(ID_Company _tech)	NOT NULL	X

IPAD_Benefit	Model_Ipad_ Benefit	text		NOT NULL	X
Leadership_ Hierarchy	ID_Workforce _Employee	id_employee	Employee (ID_ Employee)	NOT NULL	✓
Leadership_ Hierarchy	ID_Leadership _Employee (UNIQUE)	id_employee	Employee (ID_ Employee)	NOT NULL	✓
Made_Of_ Material	ID_PPE	id_ppe	PPE (ID_PPE)	NOT NULL	✓
Made_Of_ Material	ID_Material	id_material	Material (ID_ Material)	NOT NULL	✓
Management_ 1st_Tech	ID_Ticket	id_ticket	Ticket (ID_Ticket)	NOT NULL	✓
Management_ 1st_Tech	ID_Tech_1	id_employee	1st_Level_ Technician (ID_ Tech_1)	NOT NULL	✓
Management_ 1st_Tech	Date_Time_ Management_ Tech_1st	timestamp without time zone		NOT NULL	✓
Management_ 2nd_Tech	ID_Ticket	id_ticket		NOT NULL	✓
Management_ 2nd_Tech	ID_Tech_2	id_employee	2nd_Level_ Technician (ID_ Tech_2)	NOT NULL	✓
Management_ 2nd_Tech	Date_Time_ Management_ Tech_2nd	timestamp without time zone		NOT NULL	X

Manager	ID_Employee	id_manager_ employee	Employee(ID _ Employee)	NOT NULL	✓
Manager	Type_Manager	type_manager	Employee (Employee_ Type)	NOT NULL	X
Material	ID_Material	id_material		NOT NULL	✓
Material	Name_ Material	character varying		NOT NULL	X
Parking	ID_Park	id_parking		NOT NULL	✓
Parking	Type_Parking	type_parking		NOT NULL	X
Parking	ID_Employee_ Reserved	id_ employee	EMPLOYEE (ID_ Employee)	NOT NULL	X
Partnership	ID_Benefit_ Book_Bonus	id_bookbonus	Book_Bonus (ID_Benefit_ Book_Bonus)	NOT NULL	✓
Partnership	ID_Store	id_store	Store(ID_ Store)	NOT NULL	✓
Partnership	Date_Time_ Expiration_ Partnership	date		Optional	X
PC_Benefit	ID_Benefit_ PC_Benefit	id_pc		NOT NULL	✓
PC_Benefit	Assigned_Up_ To_PC	date		NOT NULL	X

PC_Benefit	OS_PC	type_os_pc		NOT NULL	X
PC_Benefit	Brand_PC	character varying		NOT NULL	X
PC_Benefit	Width_PC	integer		NOT NULL	X
PC_Benefit	Height_PC	integer		NOT NULL	X
PC_Benefit	Price_PC	numeric		NOT NULL	X
PC_Benefit	HD_ Dimension_PC	hd_dimension_ pc		NOT NULL	X
PC_Benefit	RAM_ Dimension_PC	dimension_ram _pc		NOT NULL	X
PC_Benefit	Inch_Screen_ PC	integer		NOT NULL	X
PC_Benefit	ID_Company_ Tech	id_ techcomapny	Technology_ Company(ID_Company _tech)	NOT NULL	X
PC_Benefit	Model_PC_ Benefit	text		NOT NULL	X
Phone_ Benefit	ID_Benefit_ Phone_Benefit	id_phone		NOT NULL	✓
Phone_ Benefit	Assigned_Up_ To_Phone	data		NOT NULL	X
Phone_ Benefit	OS_Phone	type_os_phone		NOT NULL	X
Phone_ Benefit	Brand_Phone	character varying		NOT NULL	X
Phone_ Benefit	Width_Phone	integer		NOT NULL	X
Phone_	Height_Phone	integer		NOT NULL	X

Benefit					
Phone_ Benefit	Price_Phone	numeric		NOT NULL	X
Phone_ Benefit	Inch_Screen_ Phone	integer		NOT NULL	X
Phone_ Benefit	ID_Company_ Tech	id_ techcomapny	Technology_ Company(ID_Company _tech)	NOT NULL	X
Phone_ Benefit	Model_Phone_ Benefit	text		NOT NULL	X
Phone_ Customer	Number_ Customer	numeric (10,0)		NOT NULL	✓
Phone_ Customer	Country_Code _Customer	integer		NOT NULL	✓
Phone_ Customer	ID_Customer	id_customer	Customer (ID_ Customer)	NOT NULL	X
Phone_ Employee	Number_ Employee	numeric (10,0)		NOT NULL	✓
Phone_ Employee	Country_Code _Employee	integer		NOT NULL	✓
Phone_ Employee	ID_Employee	id_employee	Employee(ID _ Employee)	NOT NULL	✓
PPE	IP_PPE	id_ppe		NOT NULL	✓
PPE	Size_PPE	size_clothing		NOT NULL	X
PPE	Type_PPE	ppe		NOT NULL	X

PPE	Visor	boolean		NOT NULL	X
PPE	Brand_PPE	character varying		NOT NULL	X
PPE_ Assigned	ID_Tech_2	id_employee	2nd_Level_ Technician (ID_ Tech_2)	NOT NULL	✓
PPE_ Assigned	ID_PPE	id_ppe	PPE(ID_PPE	NOT NULL	X
Product	ID_Product	id_product		NOT NULL	✓
Product	Weight	integer		NOT NULL	X
Product	Product_Level	level_value		NOT NULL	X
Product	Category	type_product		NOT NULL	X
Product	Model	character varying		NOT NULL	X
Product	Date_Buying	date		NOT NULL	X
Product	Insurance_ Expiry	date		NOT NULL	X
Product	Date_ Manifacture	date		NOT NULL	X
Product	Date_Time_ Product_ Recording	timestamp without time zone		NOT NULL	X
Product	ID_Customer	id_customer	Customer(ID _Customer)	NOT NULL	X
Product	ID_Extrenal_ Company	id_ externalcompa ny	External_ Company (ID_External _Company)	NOT NULL	X

Report	ID_Report	id_report		NOT NULL	✓
Report	Customer_ Star	value_ customer_star		Optional	X
Report	Customer_ Feedback	character varying		Optional	X
Report	Problem_ Solved	boolean		NOT NULL	X
Report	Time_ Intervent	integer		Optional	X
Report	Customer_ Convalidation	boolean		NOT NULL	X
Report	Travel_Time	integer		NOT NULL	X
Report	Date_Time_ Convalidation	timestamp without time zone		Optional	X
Report	Date_Time_ Compilation_ Tech_1	timestamp without time zone		Optional	X
Report	Date_Time_ Compilation_ Tech_2	timestamp without time zone		Optional	X
Report	ID_Ticket	id_ticket	Ticket (ID_Ticket)	NOT NULL	X
Report	ID_Customer	id_customer	Customer (ID_ Customer)	NOT NULL	X
Report	ID_Tech_1	id_employee	1st_Level_ Technician (ID_ Tech_1)	Optional	X
Report	ID_Tech_2	id_employee	2nd_Level_ Technician	Optional	X

			(ID_ tech_2)		
Revised_ Ticket	ID_Customer	id_customer	Customer (ID_ Customer)	NOT NULL	✓
Revised_ Ticket	ID_Ticket	id_ticket	Ticket (ID_Ticket)	NOT NULL	X
Revised_ Ticket	Date_Time_ Revising	timestamp without time zone		NOT NULL	✓
Stock_Option	ID_Benefit_ Stock_Option	id_stockoption		NOT NULL	✓
Stock_Option	Proce_When_ Assigned	numeric		NOT NULL	X
Stock_Option	Volume	integer		NOT NULL	X
Stock_Option	ID_Bank	id_bank	Bank (ID_Bank)	NOT NULL	X
Store	ID_Store	id_store		NOT NULL	✓
Store	Store_Type	store_type		NOT NULL	X
Store	Address_Store	character varying		Optional	X
Store	Store_ Modality	modality_store		NOT NULL	X
Store	Name_Store	text		NOT NULL	X
Technology_ Company	ID_Company_ Tech	id_ techcompany		NOT NULL	✓
Technology_	Category_Tech	character		NOT NULL	X

Company	_Company	varying			
Technology_ Company	Address_Tech _Company	character varying		NOT NULL	X
Technology_ Company	Name_ Company_tech (UNIQUE)	text		NOT NULL	X
Ticket	ID_Ticket	id_ticket		NOT NULL	✓
Ticket	Closed_Ticket	boolean		NOT NULL	X
Ticket	Product_ Typology	type_product		NOT NULL	X
Ticket	Priority_Ticket	value_priority		NOT NULL	X
Ticket	Company_ Failure	character varying		NOT NULL	X
Ticket	Problem_ Description	character varying		NOT NULL	X
Ticket	Name_ Applicant	character varying	Customer (Name_ Customer)	NOT NULL	X
Ticket	Date_Time_ Submission	timestamp without time zone		NOT NULL	X
Ticket	Date_Time_ Closure	timestamp without time zone		Optional	X
Ticket	ID_Customer	id_customer	Customer (ID_ Customer)	NOT NULL	X
Ticket	ID_Product	id_product	Product (ID_Product)	NOT NULL	X

Vehicle	Plate_Number	id_vehicle		NOT NULL	✓
Vehicle	Insurance_ Validity_Up_ To	date		NOT NULL	X
Vehicle	Drive_Licence _Needed	type_ drivelicence		NOT NULL	X
Vehicle	Kilometers_ Travelled	km_travelled		NOT NULL	X
Vehicle	Model_Vehicle	character varying		NOT NULL	X
Vehicle	Vehicle_Type	type_vehicle		NOT NULL	X
Vehicle	Brand_Vehicle	character varying		NOT NULL	X
Vehicle	Price_Vehicle	numeric		NOT NULL	X
Vehicle	Vehicle_Level _Old	level_old_ vehicle		NOT NULL	X
Vehicle	Vehicle_Level _Price	level_value		NOT NULL	X
Vehicle	Camera_Assist	boolean		Optional	X
Vehicle	Attention_ Assist	boolean		Optional	X
Vehicle	Security_ Clothing	boolean		Optional	X
Vehicle	GPS_Assigned	boolean		Optional	X
Vehicle	Weight_Limit	weight_limit		Optional	X
Vehicle	Engine_ Cylinder_ Volume_ Vehicle	integer		NOT NULL	X
Vehicle	Hersepower_ Vehicle	integer		NOT NULL	X

Vehicle	Number_Seats_Vehicle	seats_available		NOT NULL	X
Vehicle	ID_Fleet_Company	id_fleetcompany	Fleet_Company (ID_Fleet_Company)	NOT NULL	X
Viewing_Ticket	ID_Ticket	id_ticket	Ticket(ID_Ticket)	NOT NULL	✓
Viewing_Ticket	ID_Employee	id_employee	Employee(ID_Employee)	NOT NULL	✓
Viewing_Ticket	Date_Time_Viewing_Ticket	timestamp without time zone		NOT NULL	✓
Watching_Report	ID_Customer	id_customer	Customer(ID_Customer)	NOT NULL	✓
Watching_Report	ID_Report	id_report	Report(ID_Report)	NOT NULL	✓
Watching_Report	Date_Time_Watching	timestamp without time zone		NOT NULL	✓

VIEWS IMPLEMENTATION:

- **View 1(1ST_LEVEL_TECHNICIAN_DATA):** It will be used for gruping all the data between 1ST_LEVEL_TECHNICIAN and EMPLOYEE.

```
CREATE VIEW "1ST_LEVEL_TECHNICIAN_DATA" AS
SELECT *
FROM "EMPLOYEE" JOIN "1ST_LEVEL_TECHNICIAN" ON "ID_Tech_1" =
"ID_Employee"
```

- **View 2(2ND_LEVEL_TECHNICIAN_DATA):** It will be used for grouping all the data between 2ND_LEVEL_TECHNICIAN and EMPLOYEE.

```
CREATE VIEW "2ND_LEVEL_TECHNICIAN_DATA" AS
SELECT *
FROM "EMPLOYEE" JOIN "2ND_LEVEL_TECHNICIAN" ON "ID_Tech_2" =
"ID_Employee"
```

- **View 3(ACCOUNT_MANAGER_DATA):** It will be used for grouping all the data between ACCOUNT_MANAGER and EMPLOYEE.

```
CREATE VIEW "ACCOUNT_MANAGER_DATA" AS
SELECT *
FROM "EMPLOYEE" JOIN "MANAGER" ON "ID_Manager" = "ID_Employee"
WHERE "Type_Manager" = 'Account Manager'
```

- **View 4(BUSINESS_MANAGER_DATA):** It will be used for grouping all the data between BUSINESS_MANAGER and EMPLOYEE.

```
CREATE VIEW "BUSINESS_MANAGER_DATA" AS
SELECT *
FROM "EMPLOYEE" JOIN "MANAGER" ON "ID_Manager" = "ID_Employee"
WHERE "Type_Manager" = 'Business Manager'
```

- **View 5(FINANCE_DATA):** It will be used for gruping all the data between FINANCE and EMPLOYEE.

```
CREATE VIEW "FINANCE_DATA" AS
SELECT *
FROM "EMPLOYEE" JOIN "CROSS_FUNCTIONAL" ON "ID_Cross_Functional"
= "ID_Employee"
WHERE "Type_Cross_Functional" = 'Finance'
```

- **View 6(HUMAN_RESOURCES_DATA):** It will be used for grouping all the data between HUMAN_RESOURCES and EMPLOYEE.

```
CREATE VIEW "HUMAN_RESOURCES_DATA" AS
SELECT *
FROM "EMPLOYEE" JOIN "CROSS_FUNCTIONAL" ON "ID_Cross_Functional"
= "ID_Employee"
WHERE "Type_Cross_Functional" = 'Human Resources'
```

- **View 7(LEADERSHIP_DATA):** It will be used for grouping all the data between LEADERSHIP and EMPLOYEE.

```
CREATE VIEW "LEADERSHIP_DATA" AS
SELECT *
FROM "EMPLOYEE"
WHERE "Employee_Type" = 'Leadership'
```

- **View 8(TICKET_REPORT_MERGING):** It will be used to merge in one table all the information about any Ticket and its corresponding Report

```
CREATE VIEW "TICKET_REPORT_MERGING" AS
SELECT *
FROM "REPORT" NATURAL JOIN "TICKET"
```

- **View 9(MANUFACTURING_DATA_IN_TICKET):** This view will give us the name of the External Company which produced the Product inserted in a Ticket

```
CREATE OR REPLACE VIEW public."MANUFACTURING_DATA_IN_TICKET"
 AS
 SELECT ticket."ID_Ticket",
    ticket."ID_Product",
    extcompany."External_Company_Name",
    product."ID_External_Company"
   FROM "TICKET" ticket
     JOIN "PRODUCT" product ON product."ID_Product" = ticket."ID_Product"
     JOIN "EXTERNAL_COMPANY" extcompany ON
product."ID_External_Company" = extcompany."ID_External_Company";
```

- **View 10(ADDRESS_CUSTOMER_IN_TICKET):** This will return us all the addresses of the customer and the date and time of their Ticket management

```
CREATE OR REPLACE VIEW public."ADDRESS_CUSTOMER_IN_TICKET"
 AS
 SELECT ticket."Date_Time_Submission",
    ticket."Date_Time_Closure",
    customer."ID_Customer",
    address."Address_Customer"
   FROM "TICKET" ticket
     JOIN "CUSTOMER" customer ON customer."ID_Customer" =
ticket."ID_Customer"
     JOIN "ADDRESS_CUSTOMER" address ON customer."ID_Customer" =
address."ID_Customer";
```

- **View 11(PRODUCT_TICKET_MERGING):** It will be used to merge in one table the information about closed ticket and the corresponding insurance expiry date

```
CREATE VIEW "PRODUCT_TICKET_MERGING" AS
SELECT

        ticket."ID_Ticket",
        ticket."Closed_Ticket",
        ticket."ID_Product",
        product."Insurance_Expiry"

FROM public."TICKET" as ticket
INNER JOIN public."PRODUCT" as product
   ON product."ID_Product" = ticket."ID_Product"
```

- **View 12(TICKET_REPORT):** It will be used for gruping all the data between TICKET and its correspectives REPORTS.

```
CREATE OR REPLACE VIEW "TICKET_REPORT" AS
SELECT ticket."ID_Ticket", ticket."Priority_Ticket", ticket."ID_Customer",
ticket."ID_Product", report."ID_Report",
report."Customer_Star",report."Time_Intervent",report."ID_Tech_1",
report."ID_Tech_2"
FROM "TICKET" ticket JOIN "REPORT" report ON ticket."ID_Ticket" =
report."ID_Ticket"
WHERE ticket."Closed_Ticket" = True
```

- **View 13(EXTERNAL_CO_BMANAGER):** It will be used for gruping all the data between TICKET and its correspectives REPORTS.

```
CREATE OR REPLACE VIEW "EXTERNAL_CO_BMANAGER" AS
```

```
SELECT *
FROM "EXTERNAL_COMPANY" ext JOIN "MANAGER" man ON
ext."ID_Business_Manager" = man."ID_Manager"
WHERE man."Type_Manager" = 'Business Manager'
```

QUERY IMPLEMENTATION:

Implementation of Some Very Basic Operation

- **Operation - General 1:**
  - **Text:** Select the Customer rate of the report referred to Tickets that have been closed on the same day on which they have been opened.
  - As we can notice we have used taken the datas from the table “TICKET_REPORT_MERGING” which is a view table created previously
  - **Implementation**

```
SELECT "Customer_Star"
from "TICKET_REPORT_MERGING"
where date("Date_Time_Submission") = date("Date_Time_Closure")
```

- **Operation - General2:**
  - **Text:** Select all the Tickets with the highest level(1st Level) of priority still not closed.
  - **Implementation:**

```
SELECT *
FROM "TICKET"
WHERE "Priority_Ticket" = 1 and "Closed_Ticket" = false
```

- **Operation - General 3:**
  - **Text:**Find the number of Tickets having requests about products manufactured from a certain external company.
  - A we can notice we have used taken the datas from the table “MANUFACTURING_DATA_IN_TICKET” which is a view table created previously
  - **Implementation:**

```
SELECT "External_Company_Name", COUNT ("ID_Ticket")
from public."MANUFACTURING_DATA_IN_TICKET"
GROUP BY   "External_Company_Name"
```

- **Operation - General 4**
  - **Text:**
  - **Implementation:**

```
SELECT
        "External_Company_Name",
        "External_Company_Level"
from public."EXTERNAL_COMPANY"
WHERE "External_Company_Level" = ‘Metal'
```

```
-----------------------------------------------------------

SELECT
        "External_Company_Name",
        "External_Company_Level"
from public."EXTERNAL_COMPANY"
WHERE "External_Company_Level" = 'Bronze'

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

SELECT
        "External_Company_Name",
        "External_Company_Level"
from public."EXTERNAL_COMPANY"
WHERE "External_Company_Level" = 'Silver'

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

SELECT
        "External_Company_Name",
        "External_Company_Level"
from public."EXTERNAL_COMPANY"
WHERE "External_Company_Level" = 'Gold'

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


SELECT
        "External_Company_Name",
        "External_Company_Level"
from public."EXTERNAL_COMPANY"
WHERE "External_Company_Level" = 'Platinum'
```

- **Operation - General 5:**
    - **Text:** Find the address of all the Customers where the day of start and end of the intervention is different

- A we can notice we have used taken the datas from the table “ADDRESS_CUSTOMER_IN_TICKET” which is a view table created previously
- **Implementation:**

```
SELECT
        "Address_Customer"
from public."ADDRESS_CUSTOMER_IN_TICKET"
WHERE date("Date_Time_Submission")!=date("Date_Time_Closure")
```

- **Operation - General 6**
  - **Text:** Find the Customer Feedback of all the Customers where the day of start and end of the intervention is different
  - A we can notice we have used taken the datas from the table “TICKET_REPORT_MERGING” which is a view table created previously
  - **Implementation:**

```
SELECT "Customer_Feedback"
from "PRODUCT_TICKET_MERGING"
where date("Date_Time_Submission") != date("Date_Time_Closure");
```

- **Operation - General 7**
  - **Text:**Find all the Closed Tickets that in the Products have defined that the Insurance on the product has expired
  - A we can notice we have used taken the datas from the table “TICKET_REPORT_MERGING” which is a view table created previously
  - **Implementation:**

```
SELECT "ID_Ticket"
from public."PRODUCT_TICKET_MERGING"
WHERE "Insurance_Expiry" < now() AND "Closed_Ticket" = true;
```

- **Operation - General 8**
  - **Text:** Select the dates of the rent expiry on the HelpDesk cars used as benefits.
  - **Implementation:**

```
SELECT "Rent_Expiry_Date"
from public."CAR_BENEFIT"
WHERE "Rent_Expiry_Date" < now()
```

1_ST_LEVEL_TECHNICIAN:

- ***QUERY 1:***
  - **Text:** Find the ID, the name, the bonus value, the average Customer star and the remaining holidays of the 1st level Technicians who ever managed a Ticket ordered in a Crescent manner basing on their Customer Star.
  - **Implementation:**

```
SELECT empl."ID_Employee", empl."Name_Employee", contract."Bonus",
AVG(report."Customer_Star") as average_customer_star,
contract."Holidays_Left", report."ID_Ticket"
FROM "EMPLOYEE" as empl INNER JOIN "CONTRACT" as contract ON
contract."ID_Employee" = empl."ID_Employee"
INNER JOIN public."REPORT" as report ON report."ID_Tech_1" =
empl."ID_Employee" OR report."ID_Tech_2" = empl."ID_Employee"
WHERE CAST(("empl"."ID_Employee").id_role as varchar) LIKE 'IDWT1'
GROUP BY empl."ID_Employee", empl."Name_Employee",
contract."Bonus", report."Customer_Star", contract."Holidays_Left",
report."ID_Ticket"
ORDER BY average_customer_star asc
```

- ***QUERY 2:***
  - **Text:** Select the ID of the Account Managers, of that 1st level Technicians who have closed a ticket. Count also the number of Tickets that have been closed under him/her responsibility.
  - **Implementation:**

```
SELECT tech1."ID_Account_Manager", empl."Name_Employee",
ticket."ID_Ticket", tech1."ID_Tech_1", COUNT(tech1."ID_Tech_1") as
"#Closed_Ticket"
FROM public."TICKET" as ticket INNER JOIN public."REPORT" as report
ON ticket."ID_Ticket" = report."ID_Ticket"
        INNER JOIN public."1ST_LEVEL_TECHNICIAN" as tech1 ON
        tech1."ID_Tech_1" = report."ID_Tech_1"
        INNER JOIN  public."EMPLOYEE" as empl ON
        tech1."ID_Account_Manager" =  empl."ID_Employee"
WHERE ticket."Closed_Ticket" = true
GROUP BY tech1."ID_Account_Manager", empl."Name_Employee",
ticket."Closed_Ticket", ticket."ID_Ticket", tech1."ID_Tech_1",
tech1."ID_Tech_1"
```

- ***QUERY 3:***
  - **Text:** Select the ID Customer and personal data of 1st Level Technicians that have closed a ticket without delegating to a 2nd Level Technicians.
  - **Implementation:**

```
SELECT report."ID_Customer", empl."Name_Employee",
 empl."Surname_Employee", report."ID_Tech_1",
phone."Number_Employee", address."Address_Empl",
email."Email_Address_Employee", ticket."Closed_Ticket"
FROM public."REPORT" as report
INNER JOIN public."EMPLOYEE"  as empl
        ON empl."ID_Employee" = report."ID_Tech_1"
INNER JOIN public."PHONE_EMPLOYEE" as phone
        ON phone."ID_Employee" = empl."ID_Employee"
INNER JOIN public."ADDRESS_EMPLOYEE" as address
        ON address."ID_Employee" = empl."ID_Employee"
INNER JOIN public."EMAIL_EMPLOYEE" as email
        ON email."ID_Employee" = empl."ID_Employee"
INNER JOIN public."TICKET" as ticket
        ON ticket."ID_Ticket" = report."ID_Ticket"
WHERE ticket."Closed_Ticket" = true
```

- ***QUERY 4:***
  - **Text:** Select all the Course Companies that have released a certification to a 1st Level Technicians in the last year, and the number of certification released to them.
  - **Implementation:**

```
SELECT DISTINCT cert."ID_Course_Company",
coursecom."Name_Course_Company", COUNT(cert."ID_Certification") as
"#Certification_Released"
FROM public."ACHIEVEMENT_CERTIFICATION" as  achcert INNER
JOIN public."EMPLOYEE" as empl ON empl."ID_Employee" =
achcert."ID_Employee"
        INNER JOIN public."CERTIFICATION" as cert ON
        cert."ID_Certification" = achcert."ID_Certification"
        INNER JOIN public."COURSE_COMPANY" as coursecom ON
        coursecom."ID_Course_Company" = cert."ID_Course_Company"
WHERE CAST((empl."ID_Employee").id_role as varchar) LIKE 'IDWT1'and
"Date_Time_Certification" > current_date - interval '1 year'
GROUP BY cert."ID_Course_Company",
coursecom."Name_Course_Company", cert."ID_Certification"
```

- ***QUERY 5:***
  - **Text:** Find the Customer Star, the Customer Feedback, the Level and the Salary about the 1st Level Technicians that earn more than 1200 euros a month and that are not Metal Level, who managed a Report without delegating.
  - **Implementation:**

```
SELECT report."Customer_Star", report."Customer_Feedback",
empl."Name_Employee", empl."Surname_Employee",
report."ID_Tech_1",contract."Salary"
FROM "REPORT" as report INNER JOIN public."EMPLOYEE" as empl ON
empl."ID_Employee" = report."ID_Tech_1"
        INNER JOIN public."CONTRACT" as contract ON
contract."ID_Employee" = empl."ID_Employee"
WHERE contract."Salary" > 1200 and "Level_Employee" != 'Metal'
```

2_ND_LEVEL_TECHNICIAN:

- ***QUERY 1:***
  - **Text:** Find the personal data of all the 2nd Level Technicians who compiled a Report.
  - **Implementation:**

```
SELECT
 phone."Number_Employee", address."Address_Empl",report."ID_Tech_2",
 empl."Name_Employee",
 empl."Surname_Employee", empl."Number_Bank_Account"
FROM public."REPORT" as report
INNER JOIN public."EMPLOYEE" as empl
        ON empl."ID_Employee" = report."ID_Tech_2"
INNER JOIN public."PHONE_EMPLOYEE" as phone
        ON phone."ID_Employee" = empl."ID_Employee"
INNER JOIN public."ADDRESS_EMPLOYEE" as address
        ON address."ID_Employee" = empl."ID_Employee"
```

- ***QUERY 2:***
  - **Text:** Find the ID, the name, the bonus value, the average Customer star and the remaining holidays of the 2nd level Technicians who ever managed a Ticket ordered in a Crescent manner basing on their Customer Star.
  - **Implementation:**

```
SELECT empl."ID_Employee", empl."Name_Employee", contract."Bonus",
AVG(report."Customer_Star") as “Average_Customer_Star“,
contract."Holidays_Left", report."ID_Ticket"
FROM public."EMPLOYEE" as empl INNER JOIN public."CONTRACT" as
contract ON contract."ID_Employee" = empl."ID_Employee"
        INNER JOIN public."REPORT" as report ON report."ID_Tech_1" =
empl."ID_Employee" OR report."ID_Tech_2" = empl."ID_Employee"
WHERE CAST(("empl"."ID_Employee").id_role as varchar) LIKE 'IDWT1'
OR CAST(("empl"."ID_Employee").id_role as varchar) LIKE 'IDWT2'
GROUP BY empl."ID_Employee", empl."Name_Employee",
contract."Bonus", report."Customer_Star", contract."Holidays_Left",
report."ID_Ticket"
ORDER BY “Average_Customer_Star“ASC
```

- ***QUERY 3:***
  - **Text:** Find the Model of vehicles used by the Technicians to solve the problems on site, and for each Technician assigned to a Van which have

traveled for at least 30000 km, find the amount of Current Expense, made with the Credit Card, average of his/her Customer Star, the bonus received and his/her salary.

- **Implementation:**

```
SELECT fleetcom."ID_Fleet_Company", vehicle."Plate_Number",
vehicle."Model_Vehicle", cc."Current_Expense",
vehicle."Kilometers_Travelled" as "Kilometers_Traveled", contract."Bonus",
contract."Salary", tech2."ID_Tech_2"
FROM "FLEET_COMPANY" AS fleetcom INNER JOIN public."VEHICLE"
as vehicle ON vehicle."ID_Fleet_Company" = fleetcom."ID_Fleet_Company"
        INNER JOIN public."DRIVE_VEHICLE" as drivevehicle ON
        drivevehicle."ID_Vehicle" = vehicle."Plate_Number"
        INNER JOIN public."2ND_LEVEL_TECHNICIAN" as tech2 ON
        tech2."ID_Tech_2" = drivevehicle."ID_Tech_2"
        INNER JOIN "EMPLOYEE" as empl ON empl."ID_Employee" =
        tech2."ID_Tech_2"
        INNER JOIN "CONTRACT" as contract ON contract."ID_Employee"
        = empl."ID_Employee"
        INNER JOIN "CREDIT_CARD" as cc ON
        cc."ID_Benefit_Credit_Card" = contract."ID_Benefit_Credit_Card"
WHERE vehicle."Vehicle_Type" = 'Van' and vehicle."Kilometers_Travelled" > 30000
```

- ***QUERY 4:***
  - **Text:** Find all the PPE used by the 2nd Level Technicians who used a vehicle of the Fleet Company Citroën.
  - **Implementation:**

```
SELECT ppe."Type_PPE", drveh."ID_Vehicle", ppeass."ID_PPE",
 vehicle."Model_Vehicle", vehicle."ID_Fleet_Company",
 vehicle."Brand_Vehicle"
FROM  public."PPE" as ppe
INNER JOIN public."PPE_ASSIGNED" as ppeass
        ON ppeass."ID_PPE" = ppe."ID_PPE"
INNER JOIN public."DRIVE_VEHICLE" as drveh
        ON drveh."ID_Tech_2" = ppeass."ID_Tech_2"
INNER JOIN public."VEHICLE" as vehicle
        ON vehicle."Plate_Number" = drveh."ID_Vehicle"
WHERE vehicle."Brand_Vehicle" = 'Citroën'
```

- ***QUERY 5:***
  - **Text:** For each 2nd Level technician find out how many money HelpDesk has spent for his/her technological benefits, and express the model of these benefit and their technology company.

- **Implementation:**
  SELECT contract."ID_Employee", ipadben."Model_Ipad_Benefit",
  ipadben."Price_IPAD", ipadben."Brand_IPAD"
  FROM public."CONTRACT" as contract INNER JOIN
  public."IPAD_BENEFIT" as ipadben ON
  ipadben."ID_Benefit_IPAD_Benefit" = contract."ID_IPAD_Benefit"
  WHERE CAST((contract."ID_Employee").id_role as varchar) LIKE 'IDWT2';

HUMAN RESOURCES:

- ***QUERY 1:***

    - **Text:** Select the name, the email and the phone number of the HR who watched the Tickets and if the Ticket has been closed, check in how much time the issue has been solved basing on the report information.
    - **Implementation:**

```
SELECT
empl."Name_Employee", emailempl."Email_Address_Employee",
empl."ID_Employee",
phoneempl."Number_Employee",
viewticket."ID_Ticket",
ticket."Priority_Ticket", ticket."Closed_Ticket",
CASE WHEN ticket."Closed_Ticket"
 THEN report."Time_Intervent" ELSE NULL END as closed
FROM public."EMPLOYEE" as empl
INNER JOIN public."PHONE_EMPLOYEE" as phoneempl
   ON phoneempl."ID_Employee" = empl."ID_Employee"
INNER JOIN public."EMAIL_EMPLOYEE" as emailempl
   ON emailempl."ID_Employee" = empl."ID_Employee"
INNER JOIN public."VIEWING_TICKET" as viewticket
        on viewticket."ID_Employee" = empl."ID_Employee"
INNER JOIN public."TICKET" as ticket
        on viewticket."ID_Ticket" = ticket."ID_Ticket"
INNER JOIN public."REPORT" as report
        on report."ID_Ticket" = ticket."ID_Ticket"
WHERE  CAST(("empl"."ID_Employee").id_role as varchar)
 LIKE 'IDWHR'
```

- ***QUERY 2:***
    - **Text:** Find the amount of the Contract handled by the HR employee that have received the highest number of Bonus.
    - **Implementation:**

```
SELECT *
from (
        SELECT
         MAX(contract."Bonus") as max_bonus,
        empl."ID_Human_Resources",
        COUNT(empl."ID_Employee") as count_contract
        FROM public."EMPLOYEE" as empl
```

```
INNER JOIN "CONTRACT" as contract
ON empl."ID_Human_Resources" = contract."ID_Employee"
WHERE contract."Bonus"  IS NOT NULL
GROUP BY empl."ID_Human_Resources"
ORDER BY max_bonus DESC) as temp1
where temp1."max_bonus" = (
SELECT temp2."max_bonus" from (
        SELECT MAX(contract."Bonus") as max_bonus,
        empl."ID_Human_Resources"
        FROM public."EMPLOYEE" as empl
        INNER JOIN "CONTRACT" as contract
        ON
         empl."ID_Human_Resources" = contract."ID_Employee"
        WHERE contract."Bonus"  IS NOT NULL
        GROUP BY empl."ID_Human_Resources"
        ORDER BY max_bonus DESC  LIMIT 1 ) as temp2
)
```

- ***QUERY 3:***
  - **Text:** Select in a descendent order the first 15 name of the employees and the ID of the HR employee who hired them, who still have more than 13 days of holidays left depending on the number of days of holidays left.
  - **Implementation:**

```
SELECT contract."Holidays_Left", empl."ID_Employee",
empl."Name_Employee", empl."Surname_Employee",
contract."Salary"*14 as "Annual_Salary", empl."ID_Human_Resources"
FROM public."EMPLOYEE" as empl INNER JOIN
public."CONTRACT" as contract ON contract."ID_Employee" =
empl."ID_Employee"
WHERE contract."Holidays_Left" > 13 AND
empl."ID_Human_Resources" IS NOT NULL
ORDER BY contract."Holidays_Left" DESC
LIMIT 15
```

- ***QUERY 4:***
  - **Text:** Find all the email, name and surname HR who have a phone benefit of the brand "Samsung" and their annual salary is beyond the 25 k treshold. Order the result by descendent Annual Salary.
  - **Implementation:**

```
SELECT phone."Model_Phone_Benefit", contract."ID_Phone_Benefit",
email."Email_Address_Employee",  empl."ID_Employee",
empl."Name_Employee", (contract."Salary")*14 as "Annual_Income"
FROM public."PHONE_BENEFIT" as phone INNER JOIN
public."CONTRACT" as contract ON contract."ID_Phone_Benefit" =
phone."ID_Benefit_Phone_Benefit"
        INNER JOIN public."EMPLOYEE" as empl ON
empl."ID_Employee" = contract."ID_Employee"
        INNER JOIN public."EMAIL_EMPLOYEE" as email ON
email."ID_Employee" = empl."ID_Employee"
WHERE phone."Brand_Phone" = 'Samsung' and (contract."Salary")*14
> 25000 and and CAST((“contract"."ID_Employee").id_role as varchar)
LIKE 'IDW'
ORDER BY (contract."Salary")*14 DESC
```

- ***QUERY 5:***
  - **Text:** Find the ranking of the HR basing on how many employee they have hired specifying their name and surname, ordering in descending order w.r.t. the number of employee hired.
  - **Implementation:**

```
SELECT temp1."ID_Human_Resources", temp1.count_employee_hired
as "#Employee_Hired", employee."Name_Employee",
employee."Surname_Employee", employee."Level_Employee"
FROM (
                SELECT empl."ID_Human_Resources",
                COUNT(empl."ID_Human_Resources") as
                count_employee_hired
                FROM public."EMPLOYEE" as empl
                GROUP BY empl."ID_Human_Resources"
        ) as temp1
```

```
        INNER JOIN public."EMPLOYEE" as employee ON
        temp1."ID_Human_Resources" = employee."ID_Employee"
ORDER BY temp1.count_employee_hired DESC
```

A*CCOUNT MANAGER*:

- ***QUERY 1:***
  - **Text:** Select the average Customer Star got in the Reports in general, by 1_st_Level_Technicians and by 2_nd_Level_Technicians.
  - **Implementation:**
    - **Implementation with View(Used View 12, "Customer_Star"):**

```
SELECT tot."Average_Customer_Star",
st."Average_Customer_Star_1_st_Level_Technician",
nd."Average_Customer_Star_2_nd_Level_Technician"
FROM(
        SELECT AVG("Customer_Star") as
       "Average_Customer_Star_1_st_Level_Technician"
        FROM "TICKET_REPORT"
       WHERE "ID_Tech_1" IS NOT NULL AND "ID_Tech_2" IS
       NULL
       ) as st,
       (
        SELECT AVG("Customer_Star") as
       "Average_Customer_Star_2_nd_Level_Technician"
        FROM "TICKET_REPORT"
       WHERE "ID_Tech_2" IS NOT NULL AND "ID_Tech_1" IS
       NULL
       ) as nd,
       (
       SELECT AVG("Customer_Star") as "Average_Customer_Star"
        FROM "TICKET_REPORT"
       )as tot
```

    - **Implementation without View:**

```
SELECT tot."Average_Customer_Star",
st."Average_Customer_Star_1_st_Level_Technician",
nd."Average_Customer_Star_2_nd_Level_Technician"
FROM(
        SELECT AVG(r."Customer_Star") as
        "Average_Customer_Star_1_st_Level_Technician"
        FROM "TICKET" ti NATURAL JOIN "REPORT" r
        WHERE r."ID_Tech_1" IS NOT NULL AND r."ID_Tech_2"
        IS NULL AND ti."Closed_Ticket" = True
        )  as st,
        (
        SELECT AVG(r."Customer_Star") as
        "Average_Customer_Star_2_nd_Level_Technician"
```

```
FROM "TICKET" ti NATURAL JOIN "REPORT" r
WHERE r."ID_Tech_2" IS NOT NULL AND r."ID_Tech_1"
IS NULL AND ti."Closed_Ticket" = True
) as nd,
(
SELECT AVG("Customer_Star") as "Average_Customer_Star"
FROM "TICKET" NATURAL JOIN "REPORT"
WHERE "Closed_Ticket" = True
)as tot
```

- ***QUERY 2:***
  - **Text:** Compose a Ranking of the Technician with the highest average Customer Star got in the Reports, grouped in a Descendent Manner.

    First, we find the same thing for 1_st_Level_Technician and then for 2_nd_Level_Technician. Then unify the results throught the Union Set Operation.
  - **Implementation:**
    - **Implementation of the Query for 1_st_Level_Technician:**

      ```
      SELECT AVG("Customer_Star") as "Customer_Star_Avg",
      "ID_Tech_1" as "ID_Tech"
      FROM "TICKET" NATURAL JOIN "REPORT"
      WHERE "Closed_Ticket" = True and "ID_Tech_1" IS NOT NULL
      GROUP BY "ID_Tech_1"
      ```

    - **Implementation of the Query for 2_nd_Level_Technician:**

      ```
      SELECT AVG("Customer_Star") as "Customer_Star_Avg",
      "ID_Tech_2" as "ID_Tech"
      FROM "TICKET" NATURAL JOIN "REPORT"
      WHERE "Closed_Ticket" = True and  "ID_Tech_2" IS NOT NULL
      GROUP BY "ID_Tech_2"
      ```

    - **Implementation of the Query Complete:**

      ```
      SELECT *
      FROM (
          SELECT AVG("Customer_Star") as "Customer_Star_Avg",
          "ID_Tech_2" as "ID_Tech"
          FROM "TICKET" NATURAL JOIN "REPORT"
      ```

```
        WHERE "Closed_Ticket" = True and  "ID_Tech_2" IS NOT
        NULL
        GROUP BY "ID_Tech_2"

        UNION

        SELECT AVG("Customer_Star") as "Customer_Star_Avg",
        "ID_Tech_1" as "ID_Tech"
        FROM "TICKET" NATURAL JOIN "REPORT"
        WHERE "Closed_Ticket" = True and "ID_Tech_1" IS NOT
        NULL
        GROUP BY "ID_Tech_1"
        ) as fo
ORDER BY "Customer_Star_Avg" DESC
```

- ***QUERY 3:***
  - **Text:** Find the Average Time Intervent for an High-Priority Tickets, Medium-Priority Tickets and Low-Priority Tickets.
  - **Implementation:**

```
SELECT *
FROM(
        SELECT ROUND(AVG(report."Time_Intervent")) as
        "Avg_Time_Intervent_High_Priority"
        FROM public."TICKET" as ticket
        INNER JOIN public."REPORT" as report
        ON report."ID_Ticket" = ticket."ID_Ticket"
        WHERE ticket."Priority_Ticket" = 1) as high,

        (SELECT ROUND(AVG(report."Time_Intervent")) as
        "Avg_Time_Intervent_Medium_Priority"
        FROM public."TICKET" as ticket
        INNER JOIN public."REPORT" as report
        ON report."ID_Ticket" = ticket."ID_Ticket"
        WHERE ticket."Priority_Ticket" = 2) as medium,

        (SELECT ROUND(AVG(report."Time_Intervent")) as
        "Avg_Time_Intervent_Low_Priority"
        FROM public."TICKET" as ticket
        INNER JOIN public."REPORT" as report
        ON report."ID_Ticket" = ticket."ID_Ticket"
        WHERE ticket."Priority_Ticket" = 3) as low
```

- ***QUERY 4:***
    - **Text:** Select the ID, Name, Surname, Level, and Annual Income of the first 5 Account Managers of HelpDesk whose Brand Car is Volkswagen, ordered by annual income. Furthermore find the km travelled with the Vehicle, the ID of the Fleet company and the Name of the Car.
    - **Implementation:**

```
SELECT empl."ID_Employee", empl."Name_Employee", empl."Surname_Employee", empl."Level_Employee", contract."Salary"*14 as "Annual_Income", fleetcom."ID_Fleet_Company", carben."Model_Car", carben."Kilometers_Traveled"
FROM public."FLEET_COMPANY" as fleetcom INNER JOIN public."CAR_BENEFIT" as carben ON carben."ID_Fleet_Company" = fleetcom."ID_Fleet_Company"
        INNER JOIN public."CONTRACT" as contract ON contract."ID_Car_Benefit" = carben."ID_Car_Benefit"
        INNER JOIN public."EMPLOYEE" as empl ON empl."ID_Employee" = contract."ID_Employee"
        INNER JOIN public."MANAGER" as manager ON manager."ID_Manager" = empl."ID_Employee"
WHERE fleetcom."Name_Fleet_Company" = 'Volkswagen' AND manager."Type_Manager" = 'Account Manager'
ORDER BY contract."Salary"*14 DESC
LIMIT 5
```

- ***QUERY 5:***
    - **Text:** Find the total number of Technicians (both of 1st and 2nd level) managed by each Account Manager having more than 7 employee under its control. Order the result in a descendent order.
    - **Implementation:**

```
SELECT tech2_count."ID_Manager", SUM(tech2_count.count_employee + tech1_count.count_employee) as "#Technicians_Controlled"

FROM(
```

```
            SELECT manager."ID_Manager",
            count(manager."ID_Manager") as count_employee
            FROM public."1ST_LEVEL_TECHNICIAN" as tech1
            INNER JOIN public."MANAGER" as manager ON
            tech1."ID_Account_Manager"  = manager."ID_Manager"
            GROUP BY manager."ID_Manager") as tech1_count
        INNER JOIN (
            SELECT manager."ID_Manager",
            count(manager."ID_Manager") as count_employee
            FROM public."2ND_LEVEL_TECHNICIAN" as tech2
            INNER JOIN public."MANAGER" as manager ON
            tech2."ID_Account_Manager"  = manager."ID_Manager"
            GROUP BY manager."ID_Manager") as tech2_count
        ON tech2_count."ID_Manager" = tech1_count."ID_Manager"
GROUP BY tech2_count."ID_Manager"
HAVING SUM(tech2_count.count_employee + tech1_count.count_employee)
> 7
ORDER BY SUM(tech2_count.count_employee +
tech1_count.count_employee) DESC
```

BUSINESS MANAGER:

- ***QUERY 1:***
  - **Text:** Select the Business Managers ID, Name, Surname, Annual and Monthly Salary that handles a Platinum External Company that has signed a Contract with HelpDesk of an amount greater than 500k.
  - **Implementation:**

```
SELECT DISTINCT empl."ID_Employee", empl."Name_Employee",
empl."Surname_Employee", contract."Salary", (contract."Salary" * 14) as
“Annual_Income”, extcomp."External_Company_Name",
extcontract."Amount_Contract"
FROM "EXTERNAL_COMPANY" as extcomp INNER JOIN
"EXTERNAL_CONTRACT" as extcontract ON
extcontract."ID_External_Company" = extcomp."ID_External_Company"
                    INNER JOIN "EMPLOYEE" as empl ON
empl."ID_Employee" = extcomp."ID_Business_Manager"
                    INNER JOIN "CONTRACT" as contract ON
contract."ID_Employee" = empl."ID_Employee"
WHERE extcomp."External_Company_Level" = 'Platinum' AND
extcontract."Amount_Contract" > 500000
```

- ***QUERY 2:***
  - **Text:** Select Business Managers that have either a Gold or Platinum Level, ordered by their current expenses of their Credit Card assigned. Select also some information about the Car they have assigned to.
  - **Implementation:**

```
SELECT empl."ID_Employee", empl."Name_Employee",
empl."Surname_Employee", cc."Current_Expense" as
"Current_Expense_Credit_Card",fleetcom."Name_Fleet_Company",
carben."Model_Car", carben."Kilometers_Traveled",
empl."Level_Employee", carben."Rent_Expiry_Date"
FROM public."FLEET_COMPANY" as fleetcom INNER JOIN
public."CAR_BENEFIT" as carben ON carben."ID_Fleet_Company" =
fleetcom."ID_Fleet_Company"
        INNER JOIN public."CONTRACT" as contract ON
        contract."ID_Car_Benefit" = carben."ID_Car_Benefit"
        INNER JOIN public."EMPLOYEE" as empl ON empl."ID_Employee"
        = contract."ID_Employee"
        INNER JOIN public."MANAGER" as manager ON
        manager."ID_Manager" = empl."ID_Employee"
        INNER JOIN "CREDIT_CARD" as cc ON
        cc."ID_Benefit_Credit_Card" = contract."ID_Benefit_Credit_Card"
```

```
WHERE carben."Rent_Expiry_Date" >= now() AND
CAST(("empl"."ID_Employee").id_role as varchar) LIKE 'IDWBM' AND
(empl."Level_Employee" = 'Platinum' or empl."Level_Employee" = 'Gold')
ORDER BY cc."Current_Expense" DESC
```

- ***QUERY 3:***
  - **Text:** Select the Business Manager with the highest amount of bonuses and count how many External Companies are supervised by him.
  - **Implementation:**

```
SELECT *
from (
        SELECT MAX(contract."Bonus") as max_bonus,
        extcom."ID_Business_Manager",
        COUNT(extcom."ID_External_Company") as count_company
        FROM public."EXTERNAL_COMPANY" as extcom
        INNER JOIN "CONTRACT" as contract
        ON extcom."ID_Business_Manager" = contract."ID_Employee"
        WHERE contract."Bonus"  IS NOT NULL
        GROUP BY extcom."ID_Business_Manager"
        ORDER BY max_bonus DESC) as temp1
        where temp1.max_bonus = (
        SELECT temp2.max_bonus from (
                SELECT MAX(contract."Bonus") as max_bonus,
                extcom."ID_Business_Manager"
                FROM public."EXTERNAL_COMPANY" as extcom
                LEFT JOIN "CONTRACT" as contract
                ON extcom."ID_Business_Manager" =
                contract."ID_Employee"
                WHERE contract."Bonus"  IS NOT NULL
                GROUP BY extcom."ID_Business_Manager"
                ORDER BY max_bonus DESC  LIMIT 1 ) as temp2
        )
```

- ***QUERY 4:***
  - **Text:** Select the ID, Name and Surname of the Business Manager who spent the most with their Credit Card BENEFIT assigned.
  - **Implementation:**

```
SELECT *
FROM (
        SELECT cc."Current_Expense", contract."ID_Employee",
        empl."Name_Employee", empl."Surname_Employee"
```

```
FROM public."EMPLOYEE" as empl INNER JOIN
public."CONTRACT" as contract ON contract."ID_Employee" =
empl."ID_Employee"
INNER JOIN public."CREDIT_CARD" as cc ON
cc."ID_Benefit_Credit_Card" = contract."ID_Benefit_Credit_Card"
WHERE CAST((contract."ID_Employee").id_role as varchar) LIKE
'IDWBM'
ORDER BY cc."Current_Expense" DESC) as temp1
WHERE temp1."Current_Expense" = (
        SELECT temp2."Current_Expense"
        FROM (
                SELECT cc."Current_Expense",
                contract."ID_Employee"
                FROM public."EMPLOYEE" as empl INNER JOIN
                public."CONTRACT" as contract ON
                contract."ID_Employee" = empl."ID_Employee"
                INNER JOIN public."CREDIT_CARD" as cc ON
                cc."ID_Benefit_Credit_Card" =
                contract."ID_Benefit_Credit_Card"
                WHERE CAST((contract."ID_Employee").id_role as
        varchar) LIKE 'IDWBM'
ORDER BY cc."Current_Expense" DESC  LIMIT 1 ) as temp2
)
```

- **QUERY 5:**
    - **Text:** Find the store where the Business Manager may use their foodstamp.
    - **Implementation:**

```
SELECT contract."ID_Employee", inclfood."ID_Benefit_Food_Stamp",
affiliation."ID_Store", store."Name_Store"
FROM public."CONTRACT" as contract INNER JOIN
public."INCLUSION_FOOD_STAMP" as inclfood ON
inclfood."ID_Contract" = contract."ID_Contract"
        INNER JOIN public."AFFILIATION" as affiliation ON
        affiliation."ID_Benefit_Food_Stamp" =
        inclfood."ID_Benefit_Food_Stamp"
        INNER JOIN public."STORE" as store ON store."ID_Store" =
        affiliation."ID_Store"
WHERE CAST((contract."ID_Employee").id_role as varchar) LIKE
'IDWBM'
```

FINANCE:

- ***QUERY 1:***

  Text: Select the Total Amount that a particular External Company, cumulating the External Company Contract, selecting only those External Company that bring a Cumulative Income greater than 500k.

  Implementation:

```
SELECT SUM(con."Amount_Contract") as "Cumulative_Income",
exco."ID_External_Company", exco."External_Company_Name"
FROM "EXTERNAL_COMPANY" exco JOIN "EXTERNAL_CONTRACT" con
ON exco."ID_External_Company" = con."ID_External_Company"
GROUP BY exco."ID_External_Company"
HAVING SUM(con."Amount_Contract")> 500000
ORDER BY SUM(con."Amount_Contract") DESC
```

- ***QUERY 2:***

  Text: Select the ID of the Finance Employee who defined the details of the oldest Contract signed by an External Contract.

  Implementation:

```
SELECT "ID_Finance_Employee"
FROM "EXTERNAL_CONTRACT"
WHERE "ID_External_Contract" = (
        SELECT ec."ID_External_Contract"
        FROM "EXTERNAL_CONTRACT" as ec
        WHERE ec."Date_Time_Signature_External_Contract" = (
                SELECT MIN( "Date_Time_Signature_External_Contract" )
                FROM "EXTERNAL_CONTRACT"
                )
        )
```

- ***QUERY 3:***
  - **Text:** Select the number of Tickets opened by Customers related to a product that belongs to the Biggest External Company Client(Platinum Level).
  - **Implementation:**

```
SELECT COUNT(ticket."ID_Ticket") as Count_Ticket,
extcom."ID_External_Company", extcom."External_Company_Name",
extcom."External_Company_Level"
```

```
FROM public."TICKET" as ticket INNER JOIN "PRODUCT" as product ON
product."ID_Product" = ticket."ID_Product"
                INNER JOIN "EXTERNAL_COMPANY" as extcom ON
        extcom."ID_External_Company" = product."ID_External_Company"
WHERE extcom."External_Company_Level" = 'Platinum'
GROUP BY extcom."ID_External_Company"
HAVING COUNT(ticket."ID_Ticket") > 2
```

- ***QUERY 4:***
  - **Text:** For each Finance employee find out how many money HelpDesk has to spend for his/her technological benefits, both singularly both as a whole, and express the model of these benefit and their technology company.
  - **Implementation:**

```
SELECT contract."ID_Employee", phoneben."Model_Phone_Benefit",
phoneben."Price_Phone", phoneben."Brand_Phone",
pcben."Model_PC_Benefit",pcben."Price_PC",
pcben."Brand_PC",(phoneben."Price_Phone" + pcben."Price_PC") as
"Total_Expenses"
FROM public."CONTRACT" as contract INNER JOIN
public."PHONE_BENEFIT" as phoneben ON
phoneben."ID_Benefit_Phone_Benefit" = contract."ID_Phone_Benefit"
INNER JOIN public."PC_BENEFIT" as pcben ON
pcben."ID_Benefit_PC_Benefit" = contract."ID_PC_Benefit"
WHERE CAST(("contract"."ID_Employee").id_role as varchar) LIKE
'IDWFIN';
```

- ***QUERY 5:***
  - **Text:** Select the Financer ID, Name, Surname, Annual and Monthly Salary that have closed a deal with a Platinum External Company that has signed a Contract with HelpDesk of an amount greater than 500k.
  - **Implementation:**

```
SELECT e."ID_Employee", e."Name_Employee", e."Surname_Employee",
contract."Salary", (contract."Salary"* 14) as "Annual_Income",
eco."External_Company_Name", ec."Amount_Contract"
FROM "EXTERNAL_CONTRACT" as ec JOIN "EMPLOYEE" as e ON
ec."ID_Finance_Employee" = e."ID_Employee"
JOIN "EXTERNAL_COMPANY" as eco ON ec."ID_External_Company" =
eco."ID_External_Company"
JOIN "CONTRACT" contract ON e."ID_Employee" =
contract."ID_Employee"
WHERE eco."External_Company_Level" = 'Platinum' and
ec."Amount_Contract" > 500000
```

LEADERSHIP:

- ***QUERY 1:***

  Text: Select the total number of Closed and Opened Tickets. In addition to this, display the percentantage of Total Ticket over Closed Ticket at this moment.

  Implementation:

```
SELECT t."Total_Tickets", c."Closed_Tickets", o."Open_Tickets",
ROUND(CAST((c."Closed_Tickets" * 100.0 / t."Total_Tickets") AS FLOAT)) as
"%Closed", ROUND(CAST((o."Open_Tickets"* 100.0 / t."Total_Tickets") AS
FLOAT))as "%Opened"   FROM
        (SELECT COUNT(*) as "Total_Tickets"
         FROM "TICKET") as t,
        (SELECT COUNT(*)as "Closed_Tickets"
         FROM "TICKET"
        WHERE "Closed_Ticket" = true) as c,
        (SELECT COUNT(*)as "Open_Tickets"
        FROM "TICKET"
        WHERE "Closed_Ticket" = false) as o
```

- ***QUERY 2:***

  Text: Select the Difference in how much HelpDesk gains (sum of external companies contract amount minus how much spend at year (sum of salaries + bonuses). Technically, it is called “Operating Profit”.

  Implementation:

```
SELECT (e."Total_Income")-(s."Total_Salary_Expense" +
b."Total_Bonus_Expense"+ car."Total_Car_Expense"+ book."Total_Book_Expense"+
course."Total_Course_Expense"+food."Total_Food_Expense"+credit."Total_Credit_C
ard_Expense") as "Operating_Profit",e."Total_Income", s."Total_Salary_Expense",
b."Total_Bonus_Expense", car."Total_Car_Expense", book."Total_Book_Expense",
course."Total_Course_Expense", food."Total_Food_Expense",
credit."Total_Credit_Card_Expense"
FROM
        (SELECT SUM("Bonus") as "Total_Bonus_Expense"
         FROM "CONTRACT"
        WHERE "Contract_Expiration" > (SELECT CURRENT_DATE) or
        "Contract_Expiration" ISNULL) as b,

        (SELECT SUM("Salary") as "Total_Salary_Expense"
         FROM "CONTRACT"
         WHERE "Contract_Expiration" > (SELECT CURRENT_DATE) or
        "Contract_Expiration" ISNULL) as s,
```

```
(SELECT SUM("Amount_Contract") as "Total_Income"
 FROM "EXTERNAL_CONTRACT"
 WHERE "Validity_Contract" > (SELECT CURRENT_DATE) or
"Validity_Contract" ISNULL) as e,

 (SELECT (SUM("Rent_Price_Car")*12) as "Total_Car_Expense"
 FROM "CAR_BENEFIT"
 WHERE "Assigned_Up_To_Car" > (SELECT CURRENT_DATE) or
"Assigned_Up_To_Car" ISNULL) as car,

(SELECT SUM("Value_Book_Bonus") as "Total_Book_Expense"
 FROM "BOOK_BONUS"
 WHERE "Expiry_Date_Book_Bonus" > (SELECT CURRENT_DATE) or
"Expiry_Date_Book_Bonus" ISNULL) as book,

 (SELECT SUM("Value_Courses_Bonus") as "Total_Course_Expense"
 FROM "COURSES_BONUS"
 WHERE "Expiry_Date_Bonus" > (SELECT CURRENT_DATE) or
"Expiry_Date_Bonus" ISNULL) as course,

 (SELECT SUM("Value_Food_Stamp") as "Total_Food_Expense"
 FROM "FOOD_STAMP"
 WHERE "Expiry_Date_Food_Stamp" > (SELECT CURRENT_DATE) or
"Expiry_Date_Food_Stamp" ISNULL) as food,

 (SELECT SUM("Current_Expense") as "Total_Credit_Card_Expense"
 FROM "CREDIT_CARD"
 WHERE "Expiry_Date_Credit_Card" > (SELECT CURRENT_DATE) or
"Expiry_Date_Credit_Card" ISNULL) as credit
```

- ***QUERY 3:***
  Text: Select the Sum of the Salary, The sum of Bonuses, the Sum of Employee, and the Average Annual Salary depending on their Employee_Type(Workforce, Leadership and CEO).

Implementation:

```
SELECT (SUM(co."Salary"))*14 as "Sum_Salary", SUM(co."Bonus") as "Sum_Bonus", COUNT(*) as "Number_Employee", "Employee_Type", ROUND(AVG("Salary")*14) as "Average_Salary_Type_Employee_at_Year", ROUND(AVG("Salary")) as "Average_Salary_Type_Employee_at_Month"
FROM "EMPLOYEE" e JOIN "CONTRACT" co ON e."ID_Employee" = co."ID_Employee"
GROUP BY e."Employee_Type"
ORDER BY ROUND(AVG("Salary")) DESC
```

- ***QUERY 4:***

  Text: Select the ID, the Employee Typology, the Name and Surname and their annual salary of the Employees present in the DB. Then select the TOP 10 by taking into account the Annual Salary as method of judgement.

  Implementation:

```
SELECT e."ID_Employee", e."Employee_Type", e."Name_Employee", e."Surname_Employee", co."Salary"*14
FROM "EMPLOYEE" e JOIN "CONTRACT" co ON e."ID_Employee" = co."ID_Employee"
ORDER BY co."Salary" DESC
LIMIT 10
```

- ***QUERY 5:***

  Text: Select the Average number of Certification that HelpDesk's Employee have got during the year, depending on their type.

  Implementation:

```
SELECT cert."Number_Certification_Tot", cert."Employee_Type", tot."Tot_Employee",ROUND((cert."Number_Certification_Tot"/tot."Tot_Employee")*100.) as "%Certification_per_Employee", y."Number_Certification_Year" as "Number_Certification_Year",ROUND((y."Number_Certification_Year"/tot."Tot_Employee")*100.) as "%Certification_per_Employee_this_Year"
FROM (
    SELECT COUNT(*) as "Number_Certification_Tot", e."Employee_Type"
    FROM "EMPLOYEE" e JOIN "ACHIEVEMENT_CERTIFICATION" as ac
    ON e."ID_Employee" = ac."ID_Employee" JOIN "CERTIFICATION" as ce
    ON ac."ID_Certification"= ce."ID_Certification"
    GROUP BY e."Employee_Type"
    ) as cert
    JOIN
    (
```

```
SELECT COUNT(*) as "Tot_Employee", "Employee_Type" as "Type_Tot_Employee"
 FROM "EMPLOYEE"
GROUP BY "Employee_Type"
) as tot
ON cert."Employee_Type"= tot."Type_Tot_Employee"
JOIN
(
SELECT COUNT(*) as "Number_Certification_Year", e."Employee_Type"
FROM "EMPLOYEE" e JOIN "ACHIEVEMENT_CERTIFICATION" as ac ON e."ID_Employee" = ac."ID_Employee" JOIN "CERTIFICATION" as ce ON ac."ID_Certification"= ce."ID_Certification"
WHERE "Date_Time_Certification" > current_date - interval '1 year'
GROUP BY e."Employee_Type"
) as y
ON cert."Employee_Type"= y."Employee_Type"
```

- ***QUERY 6:***

  Text: Select the Average Annual Salary of HelpDesk's Employees have got depending on their Category.

  - **Implementation Number 1:** We could either make use of simple joins, also between more table, like here. The query below can be used and sobstituted(with clearly some changes in the name of tables), to the second implementation.

    ```
    SELECT ROUND(AVG(c."Salary")*14) as "Annual_Salary_1_st_Tech"
    FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee" = c."ID_Employee" JOIN "1ST_LEVEL_TECHNICIAN" st ON e."ID_Employee" = st."ID_Tech_1"
     WHERE e."Employee_Type" = 'Workforce'
    ```

  - **Implementation Number 2:**

    ```
    SELECT ceo."Annual_Salary_CEO", l."Annual_Salary_Leadership", tech1."Annual_Salary_1_st_Tech", tech2."Annual_Salary_2_nd_Tech", hr."Annual_Salary_Human_Resources", f."Annual_Salary_Finance", am."Annual_Salary_Account_Manager",
    bm."Annual_Salary_Business_Manager"
    FROM
        (SELECT ROUND(AVG(c."Salary")*14) as "Annual_Salary_CEO", ROUND(AVG(c."Bonus")) as "Annual_Bonus_CEO"
    ```

```
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
WHERE e."Employee_Type" = 'CEO') as ceo,

(SELECT ROUND(AVG(c."Salary")*14) as
"Annual_Salary_Leadership"
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
WHERE e."Employee_Type" = 'Leadership') as l,

(SELECT ROUND(AVG(c."Salary")*14) as
"Annual_Salary_1_st_Tech"
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
WHERE e."Employee_Type" = 'Workforce' and
CAST(("e"."ID_Employee").id_role as varchar) LIKE 'IDWT1') as
tech1,

(SELECT ROUND(AVG(c."Salary")*14) as
"Annual_Salary_2_nd_Tech"
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
WHERE e."Employee_Type" = 'Workforce' and
CAST(("e"."ID_Employee").id_role as varchar) LIKE 'IDWT2') as
tech2,

(SELECT ROUND(AVG(c."Salary")*14) as
"Annual_Salary_Human_Resources"
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
WHERE e."Employee_Type" = 'Workforce' and
CAST(("e"."ID_Employee").id_role as varchar) LIKE 'IDWHR') as hr,

(SELECT ROUND(AVG(c."Salary")*14) as "Annual_Salary_Finance"
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
WHERE e."Employee_Type" = 'Workforce' and
CAST(("e"."ID_Employee").id_role as varchar) LIKE 'IDWFIN') as f,

(SELECT ROUND(AVG(c."Salary")*14) as
"Annual_Salary_Account_Manager"
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
```

```
WHERE e."Employee_Type" = 'Workforce' and
CAST(("e"."ID_Employee").id_role as varchar) LIKE 'IDWAM') as
am,

(SELECT ROUND(AVG(c."Salary")*14) as
"Annual_Salary_Business_Manager"
FROM "EMPLOYEE" e JOIN "CONTRACT" c ON e."ID_Employee"
= c."ID_Employee"
WHERE e."Employee_Type" = 'Workforce' and
CAST(("e"."ID_Employee").id_role as varchar) LIKE 'IDWBM') as
bm
```

- ***QUERY 7*:**

  Text: Select the ID_Employee, its ID_Contract, its Annual and Monthly Salary, and the model and Rent_Price of its car of an employee that have some Stocks of the Company. Moreover, its car must have more than 200 Horsepower and less than 200000 kilometers traveled. He must also have a Credit Card included in the Contract.

  Implementation:

```
SELECT e."ID_Employee", co."ID_Contract", co."Salary"*14 as "Annual_Salary",
co."Salary" as "Monthly_Salary", cb."Model_Car", cb."Rent_Price_Car"
FROM "INCLUSION_STOCK_OPTION" iso JOIN "CONTRACT" co ON
iso."ID_Contract" = co."ID_Contract"
JOIN "EMPLOYEE" e ON co."ID_Employee" = e."ID_Employee"
JOIN "INCLUSION_CAR_BENEFIT" icb ON co."ID_Contract" = icb."ID_Contract"
JOIN "CAR_BENEFIT" as cb ON icb."ID_Car_Benefit" = cb."ID_Car_Benefit"
WHERE cb."Horsepower_Car" > 200 and cb."Kilometers_Traveled" < 200000 and
co."ID_Benefit_Credit_Card" IS NOT NULL
```

- ***QUERY 8:***

Text: Select the Number of Employee's Cars for every Fleet Company, the sum of the rent price per Fleet Company, its Id, Name and Address. Return only thosse Fleet Company that have more than 5 cars rented to HelpDesk's Employees.
Implementation:

```
SELECT COUNT(fc."ID_Fleet_Company") as "#Car_per_Company",
SUM(cb."Rent_Price_Car") as "Sum_Rent_Price", fc."ID_Fleet_Company",
fc."Name_Fleet_Company", fc."Address_Fleet_Company"
FROM "FLEET_COMPANY" fc JOIN "CAR_BENEFIT" cb ON
fc."ID_Fleet_Company" = cb."ID_Fleet_Company"
GROUP BY fc."ID_Fleet_Company"
HAVING COUNT(fc."ID_Fleet_Company") > 5
```

TRIGGERS IMPLEMENTATION:

- **Trigger 1:**This trigger automatically checks the of the amount of ticket opened by a certain customer, it checks the range in which the level should be (considering the value of the amount of ticket opened by the same customer), and in the case in which the amount does not reflects the leve,. it automatically updates the value of the level of the Customer

```
CREATE OR REPLACE FUNCTION public.check_cus_lev() RETURNS trigger
AS
$$
	DECLARE count_customer int;

	BEGIN
		SELECT temp1."ticket_count" INTO count_customer
		FROM (
			SELECT ticket."ID_Customer", COUNT(ticket."ID_Customer")
			 as ticket_count
			FROM public."TICKET" as ticket
			WHERE ticket."ID_Customer" = new."ID_Customer"
			GROUP BY ticket."ID_Customer") as temp1;

			RAISE NOTICE 'count_customer: %', count_customer;

			IF  count_customer BETWEEN 0 AND 5 THEN
			UPDATE public."CUSTOMER"
			SET "Level_Customer" = 'Metal'
			WHERE "ID_Customer" = new."ID_Customer";
			END IF;
			IF  count_customer BETWEEN 6 AND 10 THEN
			UPDATE  public."CUSTOMER"
			SET "Level_Customer" = 'Bronze'
			WHERE "ID_Customer" = new."ID_Customer";
			END IF;
			IF  count_customer BETWEEN 11 AND 15 THEN
			UPDATE  public."CUSTOMER"
			SET "Level_Customer" = 'Silver'
			WHERE "ID_Customer" = new."ID_Customer";
			END IF;
			IF count_customer BETWEEN 16 AND 20 THEN
			UPDATE  public."CUSTOMER"
			SET "Level_Customer" = 'Gold'
```

```
                    WHERE "ID_Customer" = new."ID_Customer";
                    END IF;
                    IF  count_customer > 20          THEN
                    UPDATE public."CUSTOMER"
                    SET "Level_Customer" = 'Platinum'
                    WHERE "ID_Customer" = new."ID_Customer";
                    END IF;

        return new;
        END;
$$
LANGUAGE plpgsql VOLATILE;
```

```
CREATE TRIGGER check_customer_level AFTER INSERT OR UPDATE ON public."TICKET"
        FOR EACH ROW EXECUTE PROCEDURE public.check_cus_lev()
```

- **Check 1:**

  ```
  INSERT INTO public."TICKET"
  ("ID_Ticket","Closed_Ticket","Product_Typology","Priority_Ticket","Company_Failure","Problem_Description","Name_Applicant","Date_Time_Submission","Date_Time_Closure","ID_Customer","ID_Product")
  VALUES
  (('IDTCK',1525849632), true,'Electronic Products', 1, 'Apple', 'Cannot activate Bluetooth ','Angelo',
  '2022-02-16 12:15:35',
   '2022-03-15 16:56:54',('IDCU',4646522109),('IDPRD',2547896523));
  ```

- **Trigger 2:** This trigger automatically checks the sum of the values of the contracts made for a certain external company, it checks the range in which the level should be (considering the value of the sum of Amount_Contrant), and in the case in which the amount does not reflects the level, it automatically updates the value of the level of the External Company

```
CREATE OR REPLACE FUNCTION public.check_ext_company_lev() RETURNS
trigger
AS
$$
	DECLARE
	count_contract_value int;

	BEGIN
		SELECT temp1."amount_contract" INTO count_contract_value
		FROM (
			SELECT
			extcontract."ID_External_Company",
			SUM(extcontract."Amount_Contract") as amount_contract
			FROM public."EXTERNAL_CONTRACT" as extcontract
			WHERE
			 extcontract."ID_External_Company" =
			 new."ID_External_Company"
			GROUP BY extcontract."ID_External_Company") as temp1;

		RAISE NOTICE 'count_contract_value: %', count_contract_value;

		IF  count_contract_value BETWEEN 0 AND 250000 THEN
			UPDATE public."EXTERNAL_COMPANY"
			SET "External_Company_Level" = 'Metal'
			WHERE
			 "External_Company_Level" = new."ID_External_Company";
		END IF;
		IF  count_contract_value BETWEEN 250001 AND 500000 THEN
			UPDATE  public."EXTERNAL_COMPANY"
			SET "External_Company_Level" = 'Bronze'
			WHERE
			 "ID_External_Company" = new."ID_External_Company";
		END IF;
		IF  count_contract_value BETWEEN 500001 AND 750000 THEN
			UPDATE  public."EXTERNAL_COMPANY"
			SET "External_Company_Level" = 'Silver'
			WHERE
			"ID_External_Company" = new."ID_External_Company";
		END IF;
		IF count_contract_value BETWEEN 750001 AND 1250000 THEN
			UPDATE  public."EXTERNAL_COMPANY"
			SET "External_Company_Level" = 'Gold'
			WHERE
			 "ID_External_Company" = new."ID_External_Company";
```

```
                END IF;
                IF  count_contract_value > 1250001 THEN
                        UPDATE public."EXTERNAL_COMPANY"
                        SET "External_Company_Level" = 'Platinum'
                        WHERE
                         "ID_External_Company" = new."ID_External_Company";
                END IF;

        return new;
        END;
$$
LANGUAGE plpgsql VOLATILE;


CREATE TRIGGER check_extcom_level AFTER INSERT OR UPDATE ON
public."EXTERNAL_CONTRACT"
FOR EACH ROW
 EXECUTE PROCEDURE public.check_ext_company_lev()
```

- **Check 2:**

```
INSERT INTO public."EXTERNAL_CONTRACT"
("ID_External_Contract","Amount_Contract","Validity_Contract","Date_Time_Signa
ture_External_Contract","ID_Finance_Employee","ID_External_Company")
VALUES
(('IDEXCT',152699),  100000,
'2022-12-31','2019-04-15 09:28:20',('IDWFIN',65311),('IDEXCP',098765));
```

- **Trigger 3:** It checks that the price level of the vehicle used by the 2nd level technicians actually reflects the price range at which the vehicle is sold. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE FUNCTION public.check_vehicle_price_lev()
RETURNS trigger
AS
$$
        DECLARE
        price_vehicle numeric;
```

```
BEGIN

        IF (new."Price_Vehicle" BETWEEN 0 AND 5000)
        AND NOT new."Vehicle_Level_Price" = 'Metal' THEN
        RAISE EXCEPTION 'The Vehicle_Level_Price
        is NOT correct, please choose Metal';
        END IF;

        IF  (new."Price_Vehicle" BETWEEN 5001 AND 15000)
         AND NOT new."Vehicle_Level_Price" = 'Bronze' THEN
        RAISE EXCEPTION 'The Vehicle_Level_Price
        is NOT correct, please choose Bronze';
        END IF;

        IF  (new."Price_Vehicle" BETWEEN 15001 AND 25000)
        AND NOT new."Vehicle_Level_Price" = 'Silver' THEN
        RAISE EXCEPTION 'The Vehicle_Level_Price
         is NOT correct, please choose Silver';
        END IF;

        IF (new."Price_Vehicle" BETWEEN 25001 AND 35000)
        AND NOT new."Vehicle_Level_Price" = 'Gold' THEN
        RAISE EXCEPTION 'The Vehicle_Level_Price
        is NOT correct, please choose Gold';
        END IF;

        IF  new."Price_Vehicle" > 35001
        AND NOT new."Vehicle_Level_Price" = 'Platinum' THEN
        RAISE EXCEPTION 'The Vehicle_Level_Price
        is NOT correct, please choose Platinum';
        END IF;

return new;
END;
$$
```

```
LANGUAGE plpgsql VOLATILE;


CREATE TRIGGER check_vehicle_level BEFORE INSERT OR
UPDATE ON public."VEHICLE"
        FOR EACH ROW EXECUTE PROCEDURE
public.check_vehicle_price_lev()
```

- **Check 3:**

```
INSERT INTO public."VEHICLE"
("Plate_Number","Insurance_Validity_Up_To","Drive_License_Needed",
"Kilometers_Travelled","Model_Vehicle","Vehicle_Type","Brand_Vehicl
e","Price_Vehicle","Vehicle_Level_Old","Vehicle_Level_Price","Camera
_Assist","Attention_Assist","Security_Clothing","GPS_Assigned","Weig
ht_Limit","Engine_Cylinder_Volume_Vehicle","Horsepower_Vehicle","
Number_Seats_Vehicle","ID_Fleet_Company")
VALUES
(('KO',166,'LK'),'2022-12-31','A', 10000,' TMAX TECH
MAX','Motorcycle','Yamaha',15119,1,'Bronze',false,false,true,true,160,2,4
7,2, ('IDBFC',1587496532));
```

- **Trigger 4:** It checks that the price level of the vehicle actually reflects the price range at which the car benefit is sold. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE
 FUNCTION public.check_car_benefit_price_lev() RETURNS trigger
AS
$$
        DECLARE
        BEGIN
```

```
        IF (new."Price_Car" BETWEEN 0 AND 20000)
        AND NOT new."Car_Level_Price" = 'Metal' THEN
        RAISE EXCEPTION 'The Car_Level_Price
        is NOT correct, please choose Metal';
        END IF;

        IF  (new."Price_Car" BETWEEN 21000 AND 30000)
        AND NOT new."Car_Level_Price" = 'Bronze' THEN
        RAISE EXCEPTION 'The Car_Level_Price
        is NOT correct, please choose Bronze';
        END IF;

        IF  (new."Price_Car" BETWEEN 30001 AND 35000)
        AND NOT new."Car_Level_Price" = 'Silver' THEN
        RAISE EXCEPTION 'The Car_Level_Price
        is NOT correct, please choose Silver';
        END IF;

        IF (new."Price_Car" BETWEEN 35001 AND 43000)
        AND NOT new."Car_Level_Price" = 'Gold' THEN
        RAISE EXCEPTION 'The Car_Level_Price
        is NOT correct, please choose Gold';
        END IF;

        IF  new."Price_Car" > 43001
        AND NOT new."Car_Level_Price" = 'Platinum' THEN
        RAISE EXCEPTION 'The Car_Level_Price
        is NOT correct, please choose Platinum';
        END IF;

    return new;
    END;
$$
LANGUAGE plpgsql VOLATILE;
```

```
CREATE TRIGGER check_carben_level BEFORE INSERT OR
UPDATE ON public."CAR_BENEFIT"
        FOR EACH ROW
        EXECUTE PROCEDURE public.check_car_benefit_price_lev()
```

- **Check 4:**

```
INSERT INTO public."CAR_BENEFIT"
("ID_Car_Benefit","Model_Car","Assigned_Up_To_Car","Price_Car","B
rand_Car","Horsepower_Car","Engine_Cylinder_Volume_Vehicle","Rent
_Price_Car","Number_Seats_Car","Insurance_Validity_Up_To","Kilomet
ers_Traveled","Rent_Expiry_Date","Car_Level_Price","ID_Fleet_Compa
ny")
VALUES
(('IDBCR','BH555VQ'),'BMW Serie 6 Gran Turismo (G32)'
,'2022-12-31',
67770,'BMW',340,6,622,5,'2021-12-31',75000,'2023-06-31','Bronze',('ID
BFC',5559874563));
```

- **Trigger 5:**This trigger checks that the inserted type of the cross functional member correctly reflects its ID. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE FUNCTION
public.check_manager_type() RETURNS trigger
AS
$$
DECLARE
BEGIN


    IF CAST((new."ID_Manager").id_role as varchar) =
     'IDWAM'
    AND NOT new."Type_Manager" = 'Account Manager'
    THEN
```

```
            RAISE EXCEPTION 'The Type_Manager
            is NOT correct,please choose Account_Manager';
        END IF;
        IF  CAST((new."ID_Manager").id_role as varchar) =
        'IDWBM'
        AND NOT new."Type_Manager" = 'Business Manager'
        THEN RAISE EXCEPTION 'The Type_Manager
         is NOT correct, please choose Business_Manager';
            END IF;
        return new;
        END;
$$
LANGUAGE plpgsql VOLATILE;


CREATE TRIGGER check_man_type BEFORE INSERT OR
UPDATE ON public."MANAGER"
        FOR EACH ROW
         EXECUTE PROCEDURE public.check_manager_type()
```

- **Check 5:**

```
INSERT INTO public."MANAGER"
( "ID_Manager","Type_Manager")
VALUES
(('IDWBM',44120),'Account Manager' );
```

- **Trigger 6:** This trigger checks that the inserted type of the manager correctly reflects its ID. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE
 FUNCTION public.check_crossfunctional_type() RETURNS trigger
AS
$$
DECLARE
BEGIN
```

```
        IF CAST((new."ID_Cross_Functional").id_role as varchar) =
        'IDWFIN' AND NOT new."Type_Cross_Functional" = 'Finance'
        THEN
        RAISE EXCEPTION
         'The Type_Cross_Functional is NOT correct, please choose
        Finance';
        END IF;

        IF  CAST((new."ID_Cross_Functional").id_role as varchar) =
        'IDWHR' AND NOT new."Type_Cross_Functional" = 'Human
        Resources'
         THEN
        RAISE EXCEPTION 'The Type_Cross_Functional is NOT correct,
        please choose Human Resources';
        END IF;


    return new;
    END;
    $$
    LANGUAGE plpgsql VOLATILE;
    CREATE TRIGGER check_cf_type BEFORE INSERT OR UPDATE ON
    public."CROSS_FUNCTIONAL"
            FOR EACH
             ROW EXECUTE
             PROCEDURE public.check_crossfunctional_type()
```

- **Check 6:**

```
INSERT INTO public."CROSS_FUNCTIONAL"
( "ID_Cross_Functional","Type_Cross_Functional")
VALUES
(('IDWHR',24115),'Finance' );
```

- **Trigger 7:** This trigger checks that the date of the submission of a ticket is always “lower” than the date of closure of it, because the insert have to

be coherent with the reality. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE FUNCTION
public.check_ordering_date() RETURNS trigger
AS
$$
	DECLARE
	BEGIN

		IF
		new."Date_Time_Submission" > new."Date_Time_Closure"
		THEN
			RAISE EXCEPTION
			'Be carefoul! You cannot submit the ticket
			before opening it';
		END IF;

	return new;
	END;
$$
LANGUAGE plpgsql VOLATILE;
CREATE TRIGGER check_ticket_date_order BEFORE INSERT OR
UPDATE ON public."TICKET"
	FOR EACH ROW
	EXECUTE PROCEDURE public.check_ordering_date()
```

- **Check 7:**

```
INSERT INTO public."TICKET"
("ID_Ticket","Closed_Ticket","Product_Typology","Priority_Ticket","Co
mpany_Failure","Problem_Description","Name_Applicant","Date_Time_
Submission","Date_Time_Closure","ID_Customer","ID_Product")
VALUES
(('IDTCK',5587496632), true,'Electronic Products', 1, 'Apple', 'Cannot
activate Bluetooth ','Iole', '2023-02-16 12:15:35', '2022-03-15
16:56:54',('IDCU',6429735520),('IDPRD',2547896523));
```

- **Trigger 8:** This trigger checks that the OS chosen reflects the Brand of the PC chosen. Because we know that the choice is not random, but there must be some reasoning behind it. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE FUNCTION public.type_os_pc() RETURNS
trigger
AS
$$
DECLARE
BEGIN

        IF new."Brand_PC" = 'Samsung'
        AND NOT
        (new."OS_PC" = 'Windows'  OR new."OS_PC" = 'Chrome
        OS')
        THEN
                RAISE EXCEPTION 'Wrong OS
                 for the brand Samsung';
        END IF;

        IF new."Brand_PC" = 'Apple'
        AND NOT new."OS_PC" = 'MacOS'
        THEN
                RAISE EXCEPTION 'Wrong OS for the brand Apple';
        END IF;

        IF new."Brand_PC" = 'HP'
        AND NOT (new."OS_PC" = 'Windows'
        OR new."OS_PC" = 'Chrome OS')
        THEN
                RAISE EXCEPTION 'Wrong OS for the brand HP';
        END IF;

        IF new."Brand_PC" = 'Asus'
        AND NOT new."OS_PC" = 'Windows'
```

```
            THEN
                RAISE EXCEPTION 'Wrong OS for the brand Asus';
            END IF;

            IF new."Brand_PC" = 'Microsoft'
             AND NOT new."OS_PC" = 'Windows'
            THEN
                RAISE EXCEPTION
                 'Wrong OS for the brand Microsoft';
            END IF;

        return new;
        END;
$$
LANGUAGE plpgsql VOLATILE;
CREATE TRIGGER check_type_os_pc BEFORE INSERT OR UPDATE
ON public."PC_BENEFIT"
        FOR EACH ROW EXECUTE PROCEDURE public.type_os_pc()
```

- **Check 8:**

```
INSERT INTO public."PC_BENEFIT"
("ID_Benefit_PC_Benefit","Assigned_Up_To_PC","OS_PC","Brand_PC
","Width_PC","Height_PC","Price_PC","HD_Dimension_PC","RAM_Di
mension_PC","Inch_Screen_PC","ID_Company_Tech",
"Model_PC_Benefit")
VALUES
(('IDBPC',55874699),  '2023-12-31' , 'Windows', 'Apple', 304, 156,
1479,'256 GB','8 GB',13,('IDTHC',6429735520), 'MacBook Pro 13');
```

- **Trigger 9:** This trigger checks that the OS chosen reflects the Brand of the phone chosen. Because we know that the choice is not random, but there must be some reasoning behind it. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE FUNCTION public.type_os_phone()
RETURNS trigger
```

```
AS
$$
DECLARE
BEGIN

		IF new."Brand_Phone" = 'Samsung'
		AND NOT new."OS_Phone" = 'Android'
		THEN
			RAISE EXCEPTION
			 'Wrong OS for the brand Samsung';
		END IF;

		IF new."Brand_Phone" = 'Apple'
		 AND NOT new."OS_Phone" = 'I-Phone OS'
		THEN
			RAISE EXCEPTION
			 'Wrong OS for the brand Apple';
		END IF;

		IF new."Brand_Phone" = 'Huawei'
		AND NOT new."OS_Phone" = 'EMUI'
		THEN
			RAISE EXCEPTION
			'Wrong OS for the brand Huawei';
		END IF;

		IF new."Brand_Phone" = 'Oppo'
		 AND NOT new."OS_Phone" = 'ColorOS'
		THEN
			RAISE EXCEPTION 'Wrong OS for the brand Oppo';
		END IF;

	return new;
	END;
$$
LANGUAGE plpgsql VOLATILE;
```

```
CREATE TRIGGER check_type_os_phone BEFORE INSERT OR
UPDATE ON public."PHONE_BENEFIT"
        FOR EACH ROW
        EXECUTE PROCEDURE public.type_os_phone()
```

- **Check 9:**

```
INSERT INTO public."PHONE_BENEFIT"
("ID_Benefit_Phone_Benefit","Assigned_Up_To_Phone","OS_Phone","
Brand_Phone","Width_Phone","Height_Phone","Price_Phone","Inch_Scr
een_Phone","ID_Company_Tech", "Model_Phone_Benefit")
VALUES
(('IDBPH',5587459633), '2025-12-31', 'Android', 'Oppo',
73,160,300,6,('IDTHC',7551036521),'Oppo Reno 7');
```

- **Trigger 10:** This trigger checks that the OS chosen reflects the Brand of the IPAD chosen. Because we know that the choice is not random, but there must be some reasoning behind it. It raises an exception message in case of mistake and disables the insertion of the data.

```
CREATE OR REPLACE FUNCTION public.type_os_ipad() RETURNS
trigger
AS
$$
        DECLARE
        BEGIN

                IF new."Brand_IPAD" = 'Samsung'
                 AND NOT new."OS_IPAD" = 'Android'
                THEN
                        RAISE EXCEPTION
                         'Wrong OS for the brand Samsung';
                END IF;
```

```
        IF new."Brand_IPAD" = 'Apple'
         AND NOT new."OS_IPAD" = 'iPadOS'
        THEN
            RAISE EXCEPTION 'Wrong OS for the brand Apple';
        END IF;

        IF new."Brand_IPAD" = 'Huawei'
         AND NOT new."OS_IPAD" = 'HUAWEI'
        THEN
            RAISE EXCEPTION
             'Wrong OS for the brand Huawei';
        END IF;

        IF new."Brand_IPAD" = 'Lenovo'
         AND NOT new."OS_IPAD" = 'Android'
        THEN
            RAISE EXCEPTION
             'Wrong OS for the brand Lenovo';
        END IF;

    return new;
    END;
$$
LANGUAGE plpgsql VOLATILE;
CREATE TRIGGER check_type_os_ipad BEFORE INSERT OR
UPDATE ON public."IPAD_BENEFIT"
    FOR EACH ROW EXECUTE PROCEDURE
public.type_os_ipad()
```

- **Check 10:**

```
INSERT INTO public."IPAD_BENEFIT"
("ID_Benefit_IPAD_Benefit","Assigned_Up_To_IPAD","OS_IPAD","Br
and_IPAD","Width_IPAD","Height_IPAD","Price_IPAD","Inch_Screen_
IPAD","ID_Company_Tech", "Model_Ipad_Benefit")
VALUES
```

(('IDBIP',5663214895),2023-12-31', 'HUAWEI',
'Apple', 178,247,900,11,('IDTHC',6429735520),'iPad Pro 11');

PROBLEMS ENCOUNTERED:

- **DURING THE JOINS OF TWO TABLES**: One of the first problem we learn of was encountered when we performed a JOIN of two tables and PostgreSQL does not allowed us to define a JOIN on two attributes on which we're going to perform the Join on, that have the same Name.
    - **Example**:

      ```
      SELECT *
      FROM "EXTERNAL_CO_BMANAGER" exco
      JOIN "EXTERNAL_CONTRACT" con ON
      exco."ID_External_Company" = con."ID_External_Company"
      ```

    - **Solution:**

      We needed to select all the columns we were interested in, and exclude one of the two columns "ID_External_Company". This is because the Join was going to create a new table with two Columns that were equal. LESSON LEARNED ✓

      ```
      SELECT exco.”Name”, exco.”Surname”, con.”Amount”,
      exco.”ID_External_Company"
      FROM "EXTERNAL_CO_BMANAGER" exco
      JOIN "EXTERNAL_CONTRACT" con ON
      exco."ID_External_Company" = con."ID_External_Company"
      ```

- **CASTING:** We encountered many problems to manage the composite types, we have used them as the ID value to identify almost any entity, so it was necessary for us to understand how to manage them, because it seemed that PostgresSQL was not able to read them for what they were. For example we have seen this problem while managing queries
    - **Example:**

      ```
      SELECT
      contract."ID_Employee",
      foodbonus."ID_Food_Bonus",
       affiliation."ID_Store", store."Name_Store"
      FROM public."CONTRACT" as contract
      INNER JOIN public."INCLUSION_FOOD_STAMP" as inclfood
              ON inclfood."ID_Contract" = contract."ID_Contract"
      INNER JOIN public."AFFILIATION" as affiliation
              ON affiliation."ID_Benefit_Food_Stamp" =
              inclfood."ID_Benefit_Food_Stamp"
      ```

```
INNER JOIN public."STORE" as store
        ON store."ID_Store" = afffiliation."ID_Store"
WHERE CAST((contract."ID_Employee").id_role as varchar)
 LIKE 'IDWBM'
```

- **Solution:** As we can see above we had to use the CAST keyword to specify what we were looking for, we have applied this procedure in all the cases in which we had to select some Employees belonging to a certain role. LESSON LEARNED ✓

- **UPDATE/DELETE COMPOSITE TYPES:** We have encountered other problems with the composite types, such as updating or deleting their values in a row, either or in the same row of composite types.
  - **Example:**

```
DELETE
FROM public."VEHICLE"
WHERE "Plate_Number" = cast(('KO',166,'LK') as id_vehicle);
```

```
UPDATE  public."CONTRACT"
SET "ID_PC_Benefit" = cast(('IDBPC',4848596231) as id_pc)
WHERE "ID_Contract"=cast(('IDCON',7895624545) as id_contract);
```

  - **Solution:** As it is underlined above, we had to cast the type also in this case, in order to define explicitly that the type used was one of the composite ones. LESSON LEARNED ✓

- **ALTER TYPE OF COMPOSITE TYPES:** One of the problems we have encountered is the alteration of composite types into other types, because PostgresSQL could not move directly from two types that we have created.
  - **Example:**

```
ALTER TABLE public."HANDLER_CONTRACT"
ALTER COLUMN "ID_Human_Resources"  TYPE id_cross_functional
USING "ID_Human_Resources"::text::id_cross_functional;
```

  - **Solution:** As we can see we had to use a primitive type as an intermediary between the two types. LESSON LEARNED ✓

APPLICATION DEVELOPMENT

For supporting our Database we developed two Applications: A Basic Version in Python, able to create and solve all the Query we have developed, and an Advanced Version in PHP, with the aim to represent how the different points of views of our DB (Customer Point of view(POV), Technicians POV, Managers(POV)). We also decided to develop the two parts in two different ways, namely the former running on PostegresSQL, whilst the latter running on MYSQL, since our aim was to learn more things. In this way, we learned how to interact with both DBMS.

BASIC VERSION: PYTHON

In this version, as precedently mentioned, we represented in a direct way the interaction with the DBMS, with a very Basic GUI, represented by the Terminal of Visual Studio Code, with the goal of visualizing query results (done with the help of PANDAS). In the next part of the Documentation, we will show how to use this application, as a sort of Manual.

INTRODUCTION: ROLES

In the Basic Version, we make use of the following 7 “Toy Roles” that will make our life easier during the interaction with the GUI. However, any other Username/Password present in the DB will be valid!

- **1_st_Level_Tech:**
  - **Usr:** tech1
  - **Psw:** tech1
- **2_st_Level_Tech:**
  - **Usr:** tech2
  - **Psw:** tech2
- **Account_Manager:**
  - **Usr:** managera
  - **Psw:** managera
- **Business_Manager:**
  - **Usr:** managerb
  - **Psw:** managerb
- **Finance:**
  - **Usr:** finance
  - **Psw:** finance
- **Human Resources:**
  - **Usr:** hr
  - **Psw:** hr
- **Leadership_Team:**

- **Usr:** lead
- **Psw:** lead

SUBMENUS:

- 1_ST_LEVEL_TECHNICIAN:

```
1: 'Find the ID, the name, the bonus value, the average Customer star and
the remaining holidays of the 1st level Technicians who ever managed a
Ticket ordered in a Crescent manner basing on their Customer      Star',
2: 'Select the ID of the Account Managers, of that 1st level Technicians who
have closed a ticket. Count also the number of Tickets that have been closed
under him/her responsibility.',
3: 'Insert in the Report all the ID Customer and personal data of 1st Level
Technicians that have closed a ticket without delegating to a 2nd Level
Technicians.',
4: 'Select all the Course Companies that have released a certification to a
1st Level Technicians in the last year, and the number of certification
released to them. ',
5: 'Find the Customer Star, the Customer Feedback, the Level and the Salary
about the 1st Level Technicians that earn more than 1200 euros at month and
that are not Metal Level, who managed a Report without     delegating.',

'Q': 'Exit',
'H': 'Help',
'B': 'Back',
'L:': 'Go Back to LEADESHIP Working Area (WARNING: Only if you have
PERMISSIONS!!!) '
```

- 2_ND_LEVEL_TECHNICIAN:

```
1: 'Find the personal data of all the 2nd Level Technicians who compiled a
Report.',
2: 'Find the ID, the name, the bonus value, the average Customer star and
the remaining holidays of the 2nd level Technicians who ever managed a
Ticket ordered in a Crescent manner basing on their Customer      Star.',
3: 'Find the Model of vehicles used by the Technicians to solve the problems
on site, and for each Technician assigned to a Van which have traveled for
at least 30000 km, find the amount of Current Expense,     made with the
Credit Card, average of his/her Customer Star, the bonus received and
his/her salary.',
4: 'Find all the PPE used by the 2nd Level Technicians who used a vehicle of
the Fleet Company Citroën',
5: 'For each 2nd Level technician find out how many money HelpDesk has spent
for his/her technological benefits, and express the model of these benefit
and their technology company.',

'Q': 'Exit',
```

```
'H': 'Help',
'B': 'Back',
'L:': 'Go Back to LEADESHIP Working Area (WARNING: Only if you have
PERMISSIONS!!!) '
```

- FINANCE:

```
1: 'Select the Total Amount that a particular External Company, cumulating
the External Company Contract, selecting only those External Company that
bring a Cumulative Income greater than 500k.',
2: 'Select the ID of the Finance Employee who defined the details of the
oldest Contract signed by an External Contract.',
3: 'Select the number of Tickets(that need to be more than two) opened by
Customers related to a product that belongs to the Biggest External Company
Client(Platinum Level).',
4: 'For each Finance employee find out how many money HelpDesk has to spend
for his/her technological benefits, both singularly both as a whole, and
express the model of these benefit and their technology       company.',
5: 'Select the Financer ID, Name, Surname, Annual and Monthly Salary that
have closed a deal with a Platinum External Company that has signed a
Contract with HelpDesk of an amount greater than 500k.',

'Q': 'Exit',
'H': 'Help' ,
'B': 'Back',
'L:': 'Go Back to LEADESHIP Working Area (WARNING: Only if you have
PERMISSIONS!!!) '
```

- HUMAN_RESOURCES:

```
1: 'Select the name, the email and the phone number of the HR who watched
the Tickets and if the Ticket has been closed, check in how much time the
issue has been solved basing on the report information.',
2: 'Find the amount of the Contract handled by the HR employee that have
received the highest number of Bonus.',
3: 'Select in a descendent order the first 15 name of the employees and the
ID of the HR employee who hired them, who still have more than 13 days of
holidays left depending on the number of days of holidays left.',
4: 'Find all the email, name and surname HR who have a phone benefit of the
brand "Samsung" and their annual salary is beyond the 25 k treshold. Order
the result by descendent Annual Salary.',
5: 'Find the ranking of the HR basing on how many employee they have hired
specifying their name and surname, ordering in descending order w.r.t. the
number of employee hired.',

'Q': 'Exit',
'H': 'Help',
```

```
'B': 'Back',
'L:': 'Go Back to LEADESHIP Working Area (WARNING: Only if you have
PERMISSIONS!!!) '
```

- BUSINESS_MANAGER:

```
1: 'Select the Business Managers ID, Name, Surname, Annual and Monthly
Salary that handles a Platinum External Company that has signed a Contract
with HelpDesk of an amount greater than 500k.',
2: 'Select Business Managers that have either a Gold or Platinum Level,
ordered by their current expenses of their Credit Card assigned. Select also
some information about the Car they have assigned to.',
3: 'Select the Business Manager with the highest amount of bonuses and count
how many External Companies are supervised by him.',
4: 'Select the ID, Name and Surname of the Business Manager who spent the
most with their Credit Card BENEFIT assigned.',
5: 'Find the store where the Business Manager may use their foodstamp.',

'Q': 'Exit',
'H': 'Help',
'B': 'Back',
'L:': 'Go Back to LEADESHIP Working Area (WARNING: Only if you have
PERMISSIONS!!!) '
```

- ACCOUNT_MANAGER:

```
1: 'Select the average Customer Star got in the Reports in general, by
1_st_Level_Technicians and by 2_nd_Level_Technicians.',
2: 'Compose a Ranking of the Technician with the highest average Customer
Star got in the Reports, grouped in a Descendent Manner.',
3: 'Find the Average Time Intervent for an High-Priority Tickets,
Medium-Priority Tickets and Low-Priority Tickets.',
4: 'Select the ID, Name, Surname, Level, and Annual Income of the first 5
Account Managers of HelpDesk whose Brand Car is Volkswagen, ordered by
annual income. Furthermore find the km travelled with the         Vehicle,
the ID of the Fleet company and the Name of the Car.',
5: 'Find the total number of Technicians (both of 1st and 2nd level) managed
by each Account Manager having more than 7 employee under its control. Order
the result in a descendent order.',

'Q': 'Exit',
'H': 'Help',
'B': 'Back',
'L:': 'Go Back to LEADESHIP Working Area (WARNING: Only if you have
PERMISSIONS!!!) '
```

- LEADERSHIP:

```
1: '-------------------Select the total number of Closed and Opened Tickets.
In addition to this, display the percentantage of Total Ticket over Closed
Ticket at this moment.',
2: '-------------------Select the Difference in how much HelpDesk gains (sum
of external companies contract amount minus how much spend at year (sum of
salaries + bonuses). Technically, it is called the
"Operating Profit"',
3: '-------------------Select the Sum of the Salary, The sum of Bonuses, the
Sum of Employee, and the Average Annual Salary depending on their
Employee_Type(Workforce, Leadership and CEO).',
4: '-------------------Select the ID, the Employee Typology, the Name and
Surname and their annual salary of the Employees present in the DB. Then
select the TOP 10 by taking into account the Annual Salary as
method of judgement.',
5: '-------------------Select the Average number of Certification that
HelpDesk's Employee have got during the year, depending on their type.',
6: '-------------------Select the Average Annual Salary of HelpDesk's
Employees have got depending on their Category.',
7: '-------------------Select the ID_Employee, its ID_Contract, its Annual
and Monthly Salary, and the model and Rent_Price of its car of an employee
that have some Stocks of the Company. Moreover, its car
must have more than 200 Horsepower and less than 200000 kilometers traveled.
He must also have a Credit Card included in the Contract.',
8: '-------------------Select the Number of Employee's Cars for every Fleet
Company, the sum of the rent price per  Fleet Company, its Id, Name and
Address. Return only thosse Fleet Company that have more than
5 cars rented to HelpDesk's Employees. ',

'Tech1:': '--------------Enter 1_st_Level_Technician Working Area.',
'Tech2:': '--------------Enter 2_nd_Level_Technician Working Area.',
'HR:': '-----------------Enter Human Resources Working Area.',
'Account Manager:': '----Enter Account Manager Working Area.',
'Business Manager:': '---Enter Business Manager Working Area.',
'Finance:': '------------Enter Finance Working Area.',


'Q:': 'Exit',
'H:': 'Help',
'B:': 'Back',
'L:': 'Go Back to LEADESHIP Working Area (WARNING: Only if you have
PERMISSIONS!!!) '
```

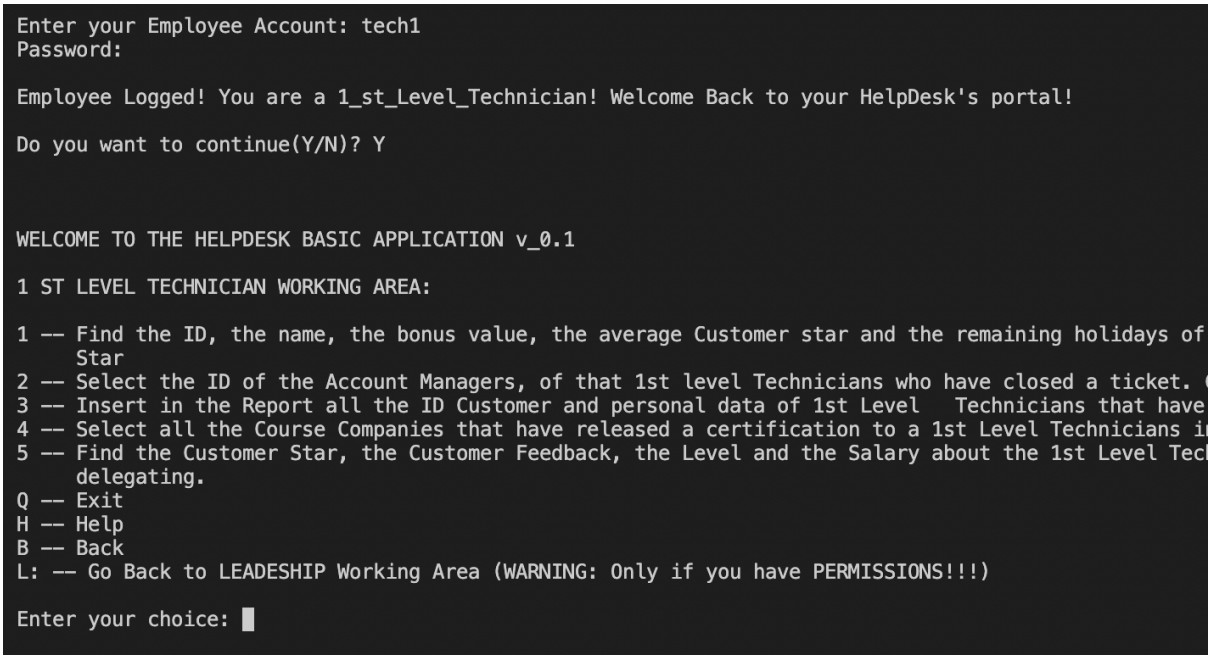
FIRST APPROACH WITH THE GUI:

The first approach with the Python GUI, is a simple Login Interface, in which we are asked to provide Username and Password (Note that the insertion of the password is hidden from the view of the user. Indeed, when we will insert the password it will appear as if we are not inserting anything. This is just the standard solution for passwords in our application).
The program will then, once we inserted the two requested data, query the DB and will recognize the "Category" to which the Employee belongs to. Consequently, the Employee, will be directed to the Submenu dedicated to it, with some option (Query) that it is allowed to perform. Note that in the submenus, containing a total of 38 Queries, there are some "special options" like:

- 'Q', that will be used to quit the framework,
- 'H' that will be used to ask for help,
- 'B' that will be used to go back to the initial screen in which we are asked for inserting usr/psw, and
- 'L' that is a very special command allowing to go back to Leadership. (More in the next section…)

DESCRIPTION: HOW IT WORKS

For instance: let us suppose that we are a 1_ST_LEVEL_TECHNICIAN Employee, we will enter the combination "tech1", "tech1" in usrn and psw. The Program will then query the DB, and, if it recognizes us, it will direct us to a submenu dedicated to us. (Here below there is the picture of the first approach to the DB. Note that the photo is cut for allowing a better visualization. Indeed, in the real-time version, we will see the entire text of the Query we are allowed to perform). Note that, at every step, we are asked the confirmation whether we want to continue or not.



Once we are here, we can enter one of the options indicated in the Menu. By selecting one of the options, if the option that has been selected is one of the numbers, a query will be performed, returning the value of the query. Instead, if we press one of the "special characters", one of the options described before will happen.

For instance, let's press the choice for the Query number.

```
Enter your choice: 1
RESULT OF THE 1 ST LEVEL TECHNICIAN QUERY 1:
TASK: Find the ID, the name, the bonus value, the average Customer star and the remaining holidays of t
 Star

     ID_Employee Name_Employee  Bonus average_customer_star  Holidays_Left          ID_Ticket
0  (IDWT1,31408)        Gloria    250    4.0000000000000000             20  (IDTCK,7896541203)
1  (IDWT1,36725)         Paola    500    5.0000000000000000              3  (IDTCK,7896542301)

Do you want to continue(Y/N)?
```

The result will be displayed and also the test of the Query will be repeated. After the result, we are asked whether we want to continue or not. If we press the positive option, we will come back to our manù and we will be able to perform the other task present in the DB.

The same thing happens for all the other "Categories" of employees. However, something changes for the Leadership ones.

LEADERSHIP OMNISCIENCE:

Let's suppose that now, going back to our menu, after having performed query # 1 for 1_ST LEVEL_TECHNICIAN, we press the option 'B'. In this way, we go back to the initial screen of login.

```
Do you want to continue(Y/N)? y



WELCOME TO THE HELPDESK BASIC APPLICATION v_0.1

1 ST LEVEL TECHNICIAN WORKING AREA:

1 -- Find the ID, the name, the bonus value, the average Customer star and the remaining holidays of the
     Star
2 -- Select the ID of the Account Managers, of that 1st level Technicians who have closed a ticket. Cour
3 -- Insert in the Report all the ID Customer and personal data of 1st Level   Technicians that have clo
4 -- Select all the Course Companies that have released a certification to a 1st Level Technicians in th
5 -- Find the Customer Star, the Customer Feedback, the Level and the Salary about the 1st Level Technic
     delegating.
Q -- Exit
H -- Help
B -- Back
L: -- Go Back to LEADESHIP Working Area (WARNING: Only if you have PERMISSIONS!!!)

Enter your choice: b
Going back to the Login Part

Enter your Employee Account:
```

We now opt for entering in the Leadership Working Area, that in our Application acts as a sort of ADMIN User, able to enter in whatever Working Area it wants without asking for permissions. It has, in addition to the option of performing all the queries of the others, also a dedicated “Special Query” that will act on the most important information of the HelpDesk, such as Operating Profit, etc.

We, therefore, enter the Leadership Working Area. Arrived at this point, we can perform one of the 8 queries dedicated to it or Enter in one of the submenus of the other categories of employees.

```
Enter your Employee Account: lead
Password:

Employee Logged! You are a Leadership_Employee! Welcome Back to your HelpDesk's portal!
Invalid Option. Please enter a number between 1 and 8.

Do you want to continue(Y/N)? y



WELCOME TO THE HELPDESK BASIC APPLICATION v_0.1

LEADERSHIP WORKING AREA:

1 --------------------Select the total number of Closed and Opened Tickets. In addition to this, display
2 --------------------Select the Difference in how much HelpDesk gains (sum of external companies contrac
                      “Operating Profit”
3 --------------------Select the Sum of the Salary, The sum of Bonuses, the Sum of Employee, and the Aver
4 --------------------Select the ID, the Employee Typology, the Name and Surname and their annual salary
                      method of judgement.
5 --------------------Select the Average number of Certification that HelpDesk’s Employee have got during
6 --------------------Select the Average Annual Salary of HelpDesk’s Employees have got depending on thei
7 --------------------Select the ID_Employee, its ID_Contract, its Annual and Monthly Salary, and the mod
                      must have more than 200 Horsepower and less than 200000 kilometers traveled. He mus
8 --------------------Select the Number of Employee’s Cars for every Fleet Company, the sum of the rent p
                      5 cars rented to HelpDesk’s Employees.
Tech1: ---------------Enter 1_st_Level_Technician Working Area.
Tech2: ---------------Enter 2_nd_Level_Technician Working Area.
HR: ------------------Enter Human Resources Working Area.
Account Manager: -----Enter Account Manager Working Area.
Business Manager: ----Enter Business Manager Working Area.
Finance: -------------Enter Finance Working Area.
Q: Exit
H: Help
B: Back
L: Go Back to LEADESHIP Working Area (WARNING: Only if you have PERMISSIONS!!!)

Enter your choice:
```

We press for the “Query #2”, and we will receive in output the result of the query performed on the DB.

```
Tech2: ---------------Enter 2_nd_Level_Technician Working Area.
HR: ------------------Enter Human Resources Working Area.
Account Manager: -----Enter Account Manager Working Area.
Business Manager: ----Enter Business Manager Working Area.
Finance: -------------Enter Finance Working Area.
Q: Exit
H: Help
B: Back
L: Go Back to LEADESHIP Working Area (WARNING: Only if you have PERMISSIONS!!!)

Enter your choice: 2
RESULT OF THE LEADERSHIP QUERY 8:
TASK: Select the Difference in how much HelpDesk gains (sum of external companies contract amount minus how much spend at year (sum of salaries + bonuses). Technically, it is called the “Operating Profit”
   Operating_Profit Total_Income  Total_Salary_Expense  Total_Bonus_Expense  Total_Car_Expense  Total_Book_Expense  Total_Course_Expense  Total_Food_Expense  Total_Credit_Card_Expense
0           8374246      9250000                138150               194170             293292                 380                  2480                 260                     247022
```

We now go back to the Menu and we show the so-called “Omniscience” of Leadership. Indeed, by pressing, for instance, ‘f’, we are able to go to the Finance Working Area without asking for any permission

```
Enter your choice: f

Do you want to continue(Y/N)? y



WELCOME TO THE HELPDESK BASIC APPLICATION v_0.1

FINANCE WORKING AREA:

1 -- Select the Total Amount that a particular External Company, cumulating the External Company Contract, s
2 -- Select the ID of the Finance Employee who defined the details of the oldest Contract signed by an Exter
3 -- Select the number of Tickets(that need to be more than two) opened by Customers related to a product th
4 -- For each Finance employee find out how many money HelpDesk has to spend for his/her technological benef
     company.
5 -- Select the Financer ID, Name, Surname, Annual and Monthly Salary that have closed a deal with a Platinu
Q -- Exit
H -- Help
B -- Back
L: -- Go Back to LEADESHIP Working Area (WARNING: Only if you have PERMISSIONS!!!)

Enter your choice:
```

Now, we can perform all the queries allowed to a Financer. For instance, we can perform Query# 4.

```
Enter your choice: 4
RESULT OF THE FINANCE QUERY 4:
TASK: For each Finance employee find out how many money HelpDesk has to spend for his/her technological benefits, both singu
company.

      ID_Employee Model_Phone_Benefit Price_Phone Brand_Phone           Model_PC_Benefit Price_PC Brand_PC Total_Expenses
0  (IDWFIN,45606)          iPhone SE         529       Apple      ASUS Vivo AIO 24 V241     1099     Asus           1628
1  (IDWFIN,65210)   Huawei P50 Pocket        1500      Huawei  HP Chromebase 22-aa0003nl      699       HP           2199
2  (IDWFIN,32613)      iPhone 13 mini         839       Apple  HP Chromebase 22-aa0003nl      699       HP           1538
3  (IDWFIN,65819)          iPhone SE         529       Apple      ASUS Vivo AIO 24 V241     1099     Asus           1628
4  (IDWFIN,45818)          iPhone SE         529       Apple      ASUS Vivo AIO 24 V241     1099     Asus           1628
5  (IDWFIN,54515)          iPhone SE         529       Apple      ASUS Vivo AIO 24 V241     1099     Asus           1628
6  (IDWFIN,53516)          iPhone SE         529       Apple      ASUS Vivo AIO 24 V241     1099     Asus           1628
7  (IDWFIN,52517)          iPhone SE         529       Apple      ASUS Vivo AIO 24 V241     1099     Asus           1628
```

Now, suppose that we want to go back to our Leadership Menu, we need only to select ‘l’, and we will go back.

```
Enter your choice: l
Going back to the Leadership Working Area
Invalid Option. Please enter a number between 1 and 8.

Do you want to continue(Y/N)? y



WELCOME TO THE HELPDESK BASIC APPLICATION v_0.1

LEADERSHIP WORKING AREA:

1 --------------------Select the total number of Closed and Opened Tickets. In addition to this, display the percentantage o
2 --------------------Select the Difference in how much HelpDesk gains (sum of external companies contract amount minus how
                     “Operating Profit”
3 --------------------Select the Sum of the Salary, The sum of Bonuses, the Sum of Employee, and the Average Annual Salary o
4 --------------------Select the ID, the Employee Typology, the Name and Surname and their annual salary of the Employees pr
                     method of judgement.
5 --------------------Select the Average number of Certification that HelpDesk’s Employee have got during the year, dependin
6 --------------------Select the Average Annual Salary of HelpDesk’s Employees have got depending on their Category.
7 --------------------Select the ID_Employee, its ID_Contract, its Annual and Monthly Salary, and the model and Rent_Price o
                     must have more than 200 Horsepower and less than 200000 kilometers traveled. He must also have a Credi
8 --------------------Select the Number of Employee’s Cars for every Fleet Company, the sum of the rent price per  Fleet Com
                     5 cars rented to HelpDesk’s Employees.
Tech1: ---------------Enter 1_st_Level_Technician Working Area.
Tech2: ---------------Enter 2_nd_Level_Technician Working Area.
HR: ------------------Enter Human Resources Working Area.
Account Manager: -----Enter Account Manager Working Area.
Business Manager: ----Enter Business Manager Working Area.
Finance: -------------Enter Finance Working Area.
Q: Exit
H: Help
B: Back
L: Go Back to LEADESHIP Working Area (WARNING: Only if you have PERMISSIONS!!!)

Enter your choice:
```

We can now enter another Working Area. As we can notice option 'L' is still available in the menu. This means that ONLY IF we are Leadership Employees we could access the Leadership Queries.
For showing this fact. Let's suppose to enter in the DB with the "Business Manager" Account. We will see something like this.

```
Enter your Employee Account: managerb
Password:

Employee Logged! You are a Business_Manager! Welcome Back to your HelpDesk's portal!
Invalid Option. Please enter a number between 1 and 5.

Do you want to continue(Y/N)? y



WELCOME TO THE HELPDESK BASIC APPLICATION v_0.1

BUSINESS MANAGER WORKING AREA:

1 -- Select the Business Managers ID, Name, Surname, Annual and Monthly Salary that handles a Platinum External Company that
2 -- Select Business Managers that have either a Gold or Platinum Level, ordered by their current expenses of their Credit C
3 -- Select the Business Manager with the highest amount of bonuses and count how many External Companies are supervised by
4 -- Select the ID, Name and Surname of the Business Manager who spent the most with their Credit Card BENEFIT assigned.
5 -- Find the store where the Business Manager may use their foodstamp.
Q -- Exit
H -- Help
B -- Back
L: -- Go Back to LEADESHIP Working Area (WARNING: Only if you have PERMISSIONS!!!)

Enter your choice:
```

Now, we can perform one of the queries that we are allowed to select, for instance, Query #5. We will see the result as:

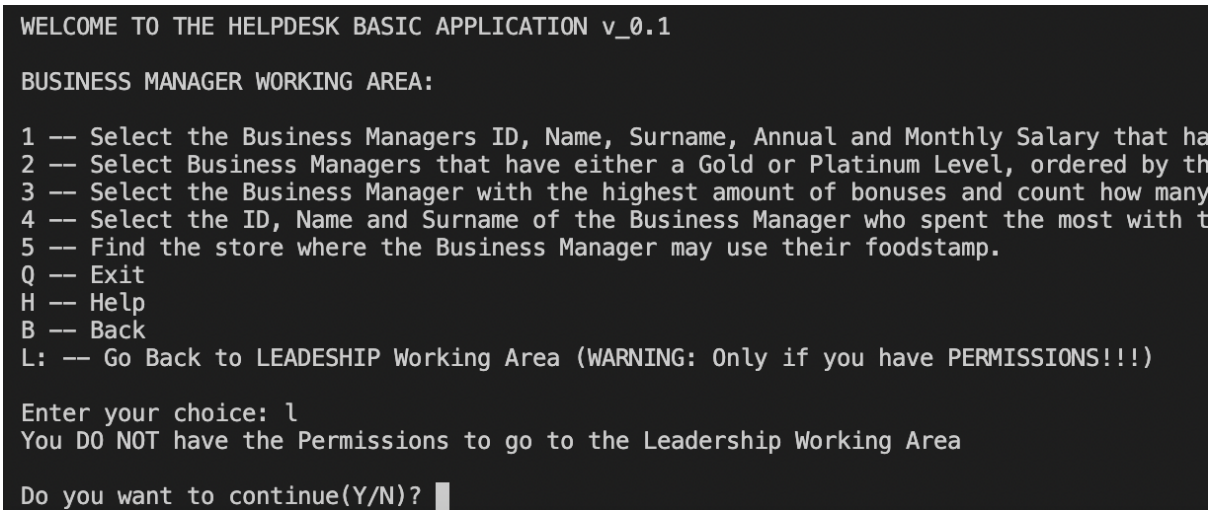
```
Enter your choice: 5
RESULT OF THE BUSINESS MANAGER QUERY 5:
TASK: Find the store where the Business Manager may use their foodstamp.

      ID_Employee    ID_Benefit_Food_Stamp          ID_Store Name_Store
0   (IDWBM,98507)  (IDBFD,111784520011579)  (IDST,1248703652)  Carrefour
1   (IDWBM,87500)  (IDBFD,555846325100193)  (IDST,1114487520)      Todis
2   (IDWBM,87500)  (IDBFD,555846325100193)  (IDST,1248703652)  Carrefour
3   (IDWBM,87500)  (IDBFD,555846325100193)  (IDST,4489226984)       Coop
4   (IDWBM,58501)  (IDBFD,589745126377765)  (IDST,1248703652)  Carrefour
5   (IDWBM,68602)  (IDBFD,445872631024898)  (IDST,1248703652)  Carrefour
6   (IDWBM,22404)  (IDBFD,445872631012483)  (IDST,5521896351)      Conad
7   (IDWBM,35311)  (IDBFD,445872631272742)  (IDST,1114487520)      Todis
8   (IDWBM,35311)  (IDBFD,445872631272742)  (IDST,5521896351)      Conad
9   (IDWBM,98507)  (IDBFD,445872631010000)  (IDST,4489226984)       Coop
10  (IDWBM,52512)  (IDBFD,445872631014477)  (IDST,4489226984)       Coop

Do you want to continue(Y/N)? y
```

Now, by going back, to the Business Manager menu, suppose we go press the 'l' option. Since we are not a Leadership Employee, and therefore we are not in a situation like the previous one, in which a Leadership Employee was entering another employee's working

area, we are rejected from doing this action.



ADVANCED VERSION: PHP SITE

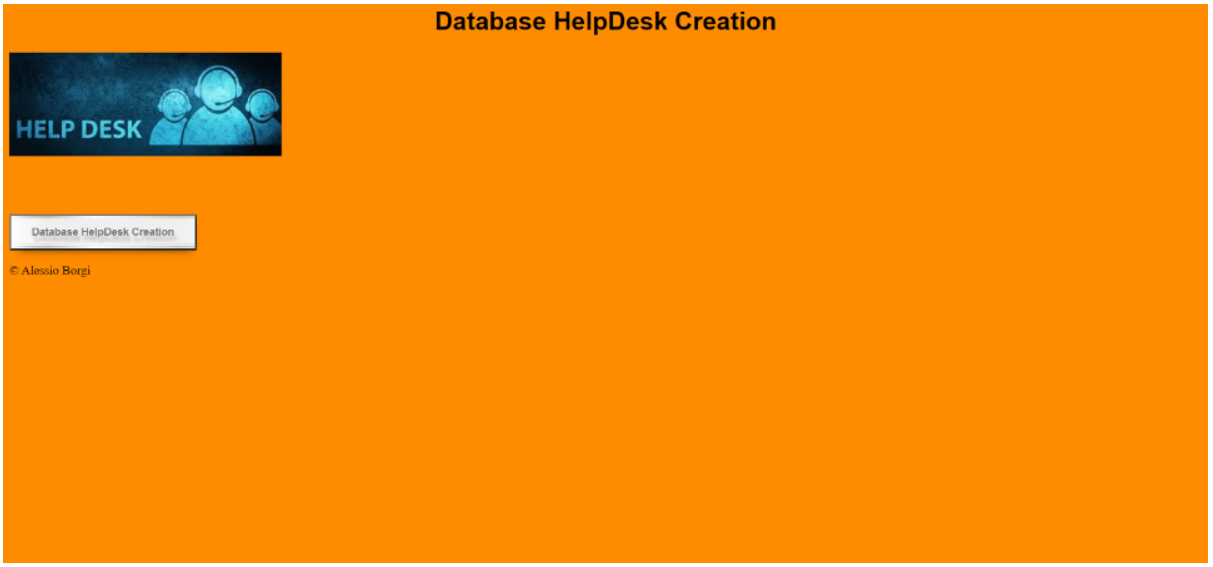
This version, as previously explained, has been developed only for the purpose of showing how the various interfaces to the DB should have been. They are really basic, and they do not involve all the Queries, but only a small amount of them. However, one of the Future Projects is to merge the two Versions. Note that the implementation of the site is very basic, which is another thing that we will have to increment in future implementations. Note also that this application is based on a simpler version of our DB (principal entities are always present. However there are some lacks, due to simplifying the architecture for our site).

APPLICATION INTRODUCTION:

The implementation of the Help Desk site, is implemented in compliance with the requirements provided above and in line with the growth objectives of the company that intends to expand on a national and international basis.
From an aesthetic point of view, we chose the use of the color ORANGE, since, in the "Marketing" field, it corresponds to Kindness, Happiness, and Trust and is therefore completely in line with the company's will to devote maximum attention to its customers. Since the web portal is the main contact tool between customers and the company, the aesthetic component has a fundamental value, and its simple access and use is a guarantee of satisfaction for Electroservice customers.
The site consists of 3 different interfaces, one for customers, one for technicians, and one for the managers. Below, we will explain the features of the aforementioned site step by step.

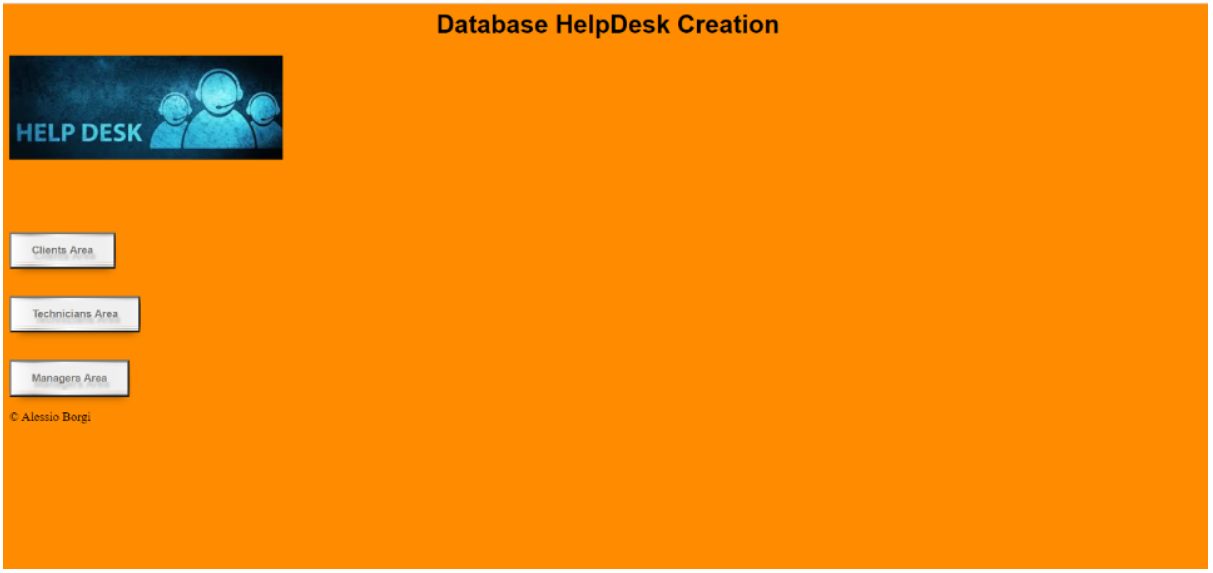
HELPDESK DATABASE CREATION:

This initial phase, which occurs only once and is prior to the creation of the site, corresponds to the creation of the database and is divided into two screens.



After pressing the “Database HelpDesk Creation” button, the database will be created and its tables will be allocated accordingly. On the next screen, after pressing one of the buttons, you

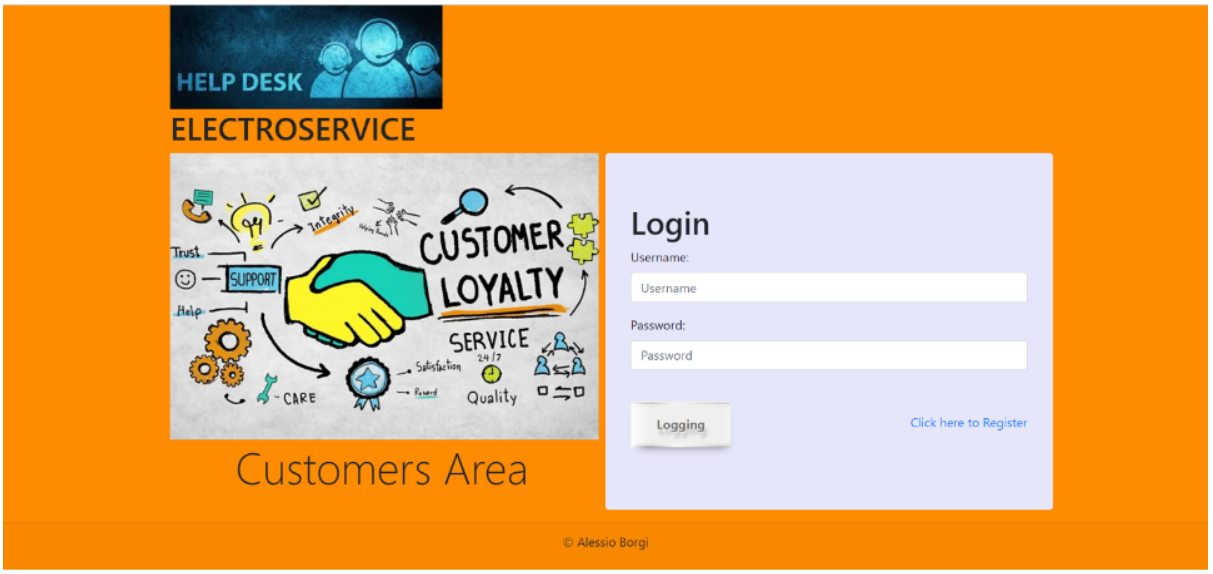
will have the option to go to one of the three interfaces set up for Customers, Technicians, and Managers.



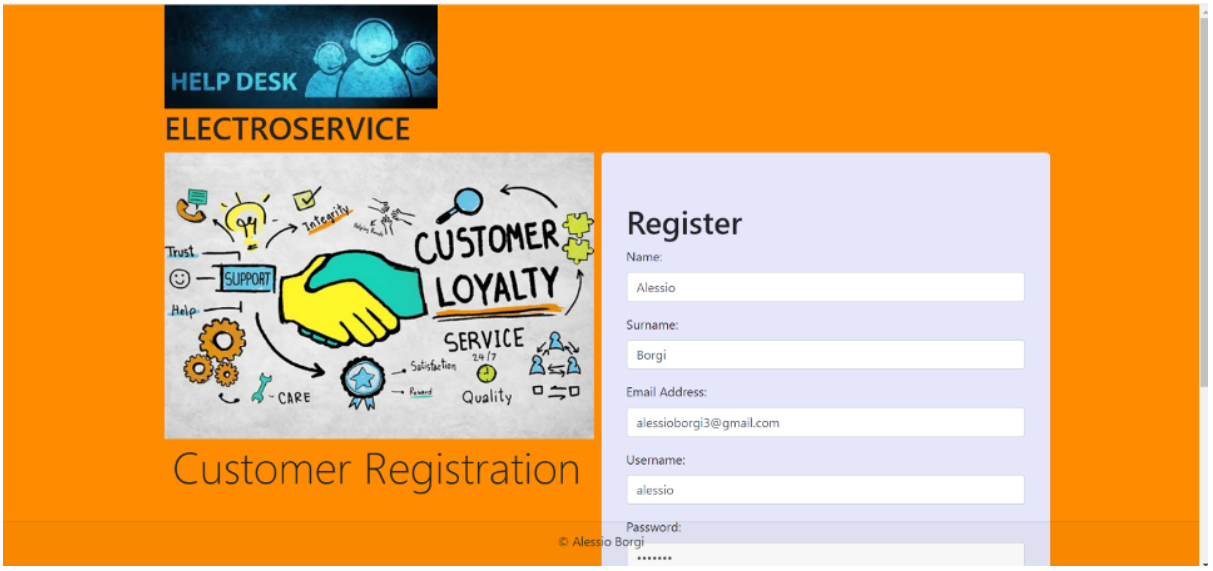
After pressing one of the buttons, you can go to the interfaces. Clearly, every Client, every Employee, and every Manager, will never see this initial phase. Their access platform, for each of the three categories, will be represented by the following interfaces which will be listed in the following order: Customers, Technicians and Managers.

CUSTOMER INTERFACE:

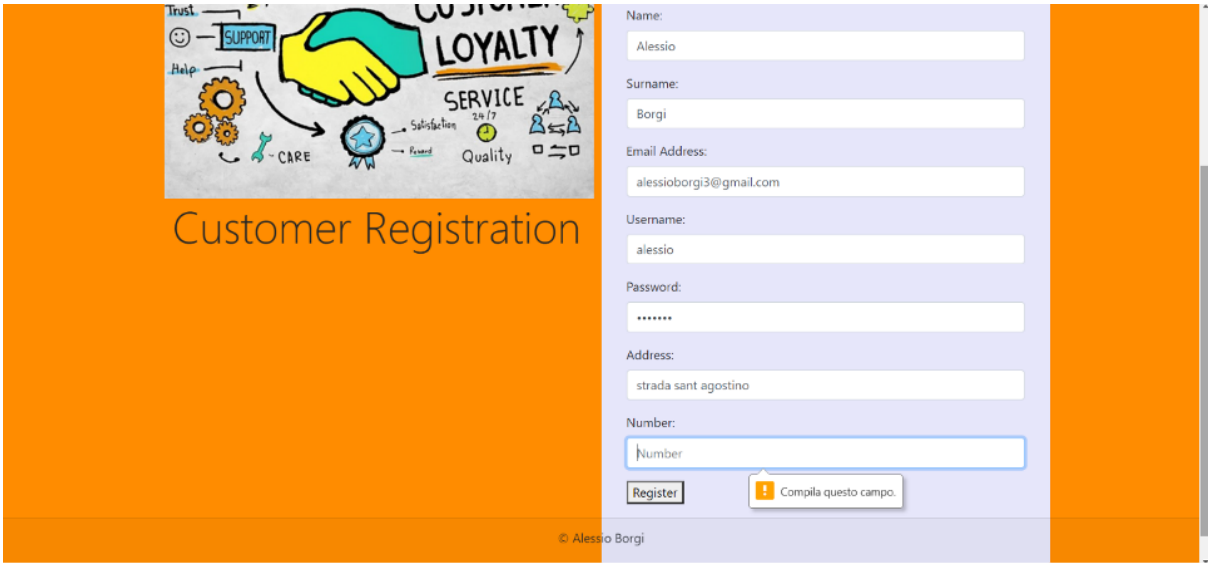
After pressing the "Clients Area" button, the following screen will be displayed. This screen, as mentioned above, will be the initial login screen that every customer of the Electroservice company will see.



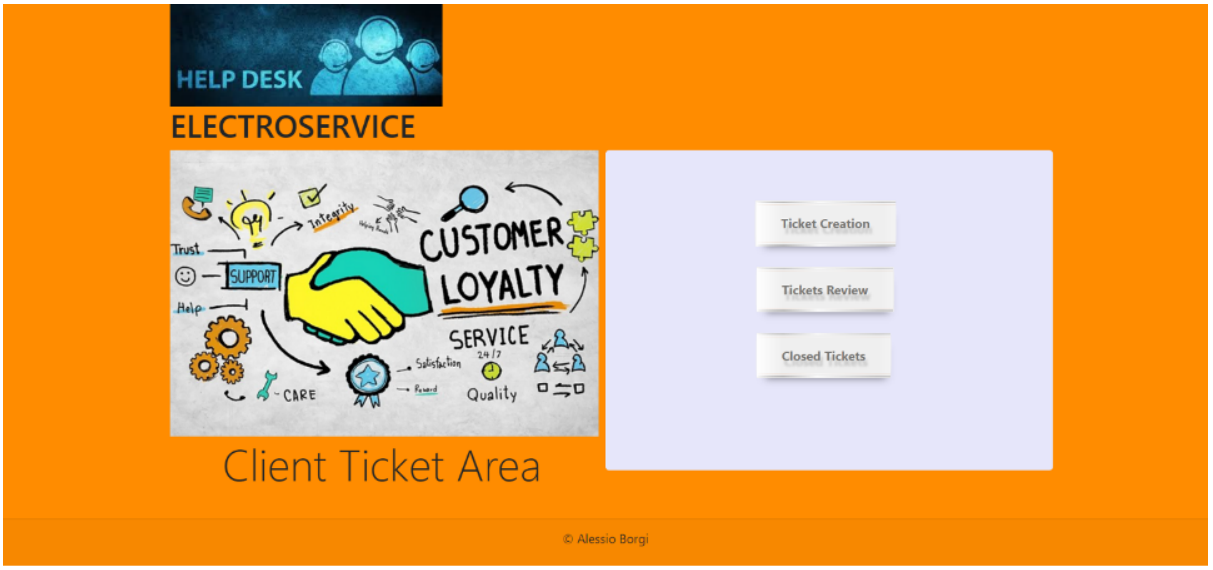
From the screen above, the Customer can perform two actions: access the reserved area of the customers (if already registered) or proceed with registration. Following registration, the customer will still be directed to this screen again. The screens relating to the Registrations will be displayed below, reserved for a new Customer who wishes to use the Electroservice services.



Once on this screen, the Customer will have to register. Filling in all fields is required to confirm successful registration, as displayed on the following screen.

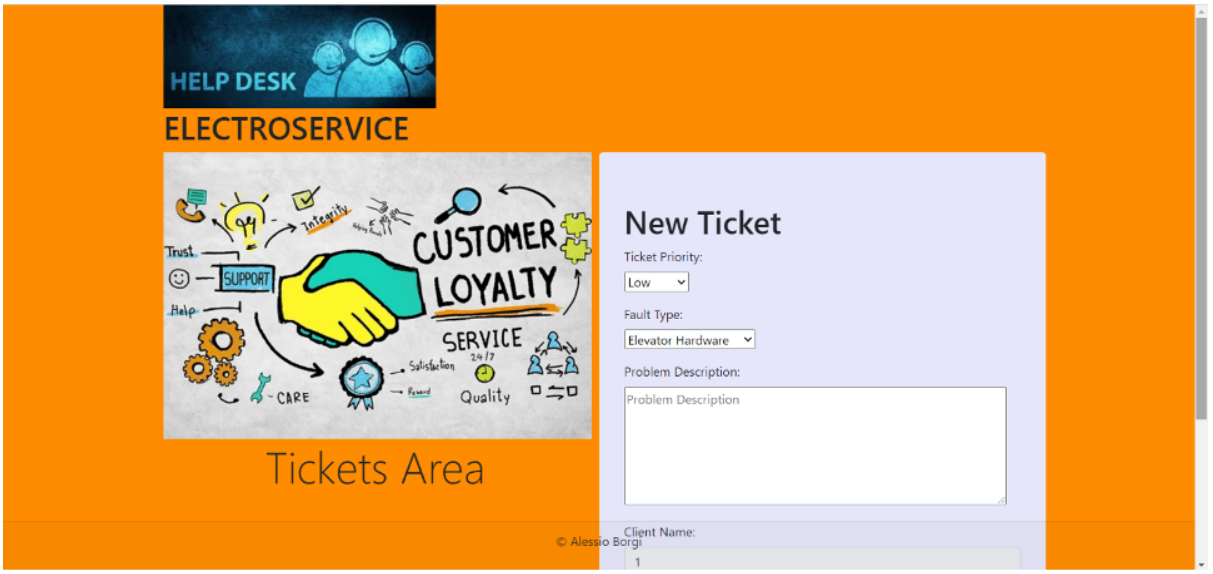


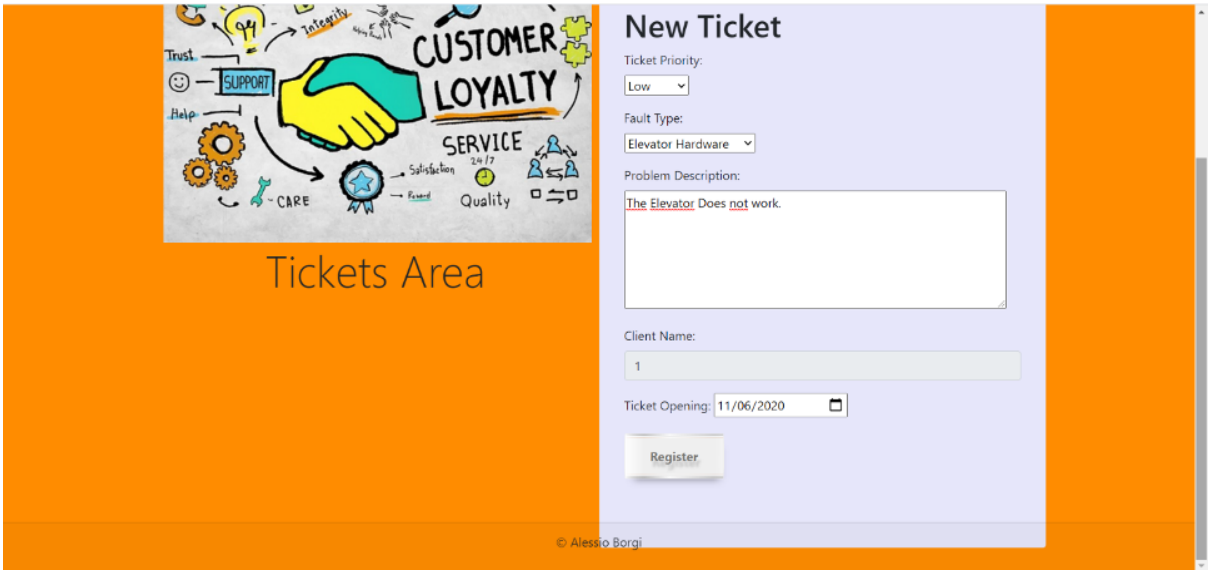
As explained above, the customer will be directed to the Login page. Once the credentials (Username and Password) have been entered, the customer will see the following screen.



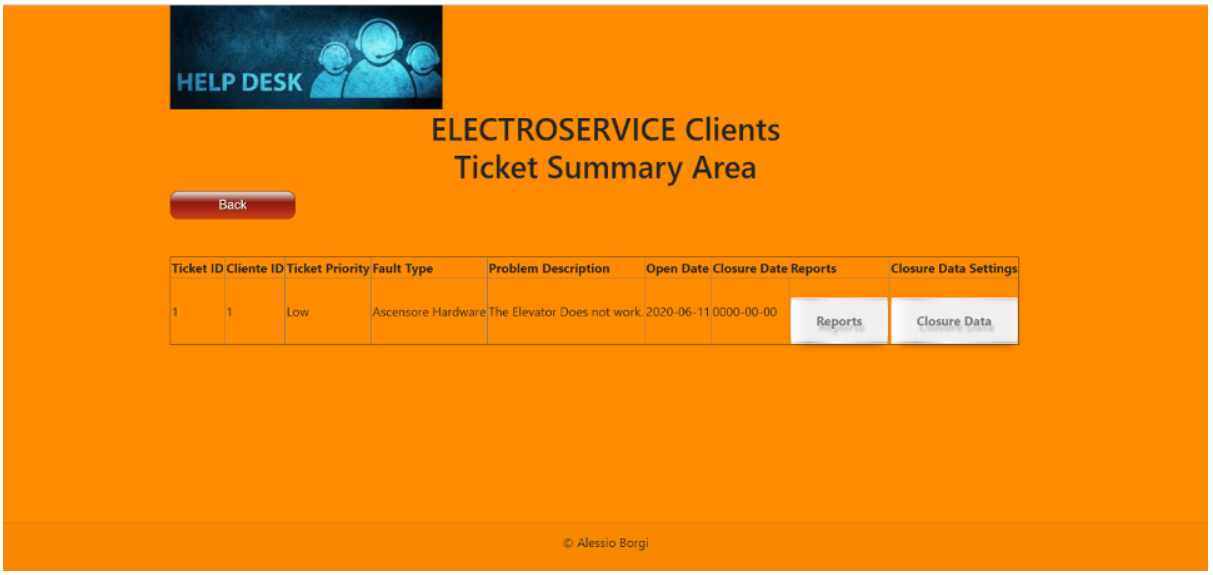
Once at this screen, the Customer will have three Options: Submit a new Ticket, and then request intervention from the Electroservice technical staff, View the Tickets submitted by him still open, and see the Tickets submitted by him closed. I will explain the following features in the following order: Creating Tickets, Viewing Open Tickets, and Viewing Closed Tickets.

Let's start with the first Function that is activated by pressing the “Ticket Creation” button.



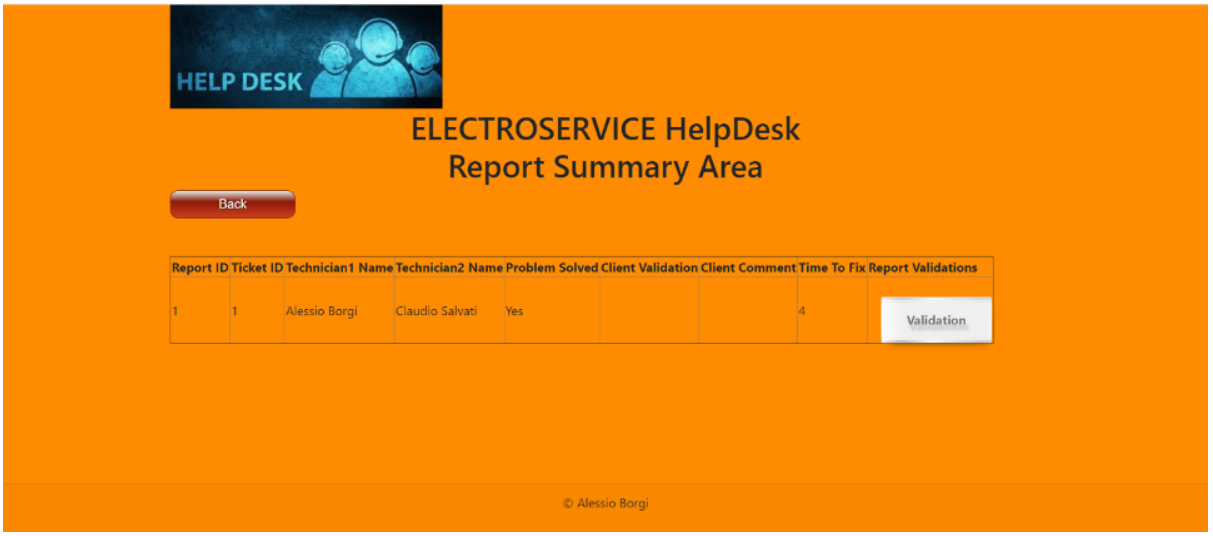


In this section, the Customer reports information relating to the problem encountered. He will be able to choose the priority of request for intervention, the type of fault, he will be able to provide the details of the problem to the extent of his knowledge and competence, the alleged fault, and enter the date of the ticket, which from the moment in which the Customer submits it, will represent the official opening of the Ticket. The Customer ID will also be automatically displayed in the compilation form.
Once the Ticket has been submitted, the Customer will be directed directly to the login screen.
Proceeding in the description of the functions that are reserved for the customer, the next one corresponds to the Display of the Open Tickets relating to the customer and which is activated by pressing the "Tickets Review" button.

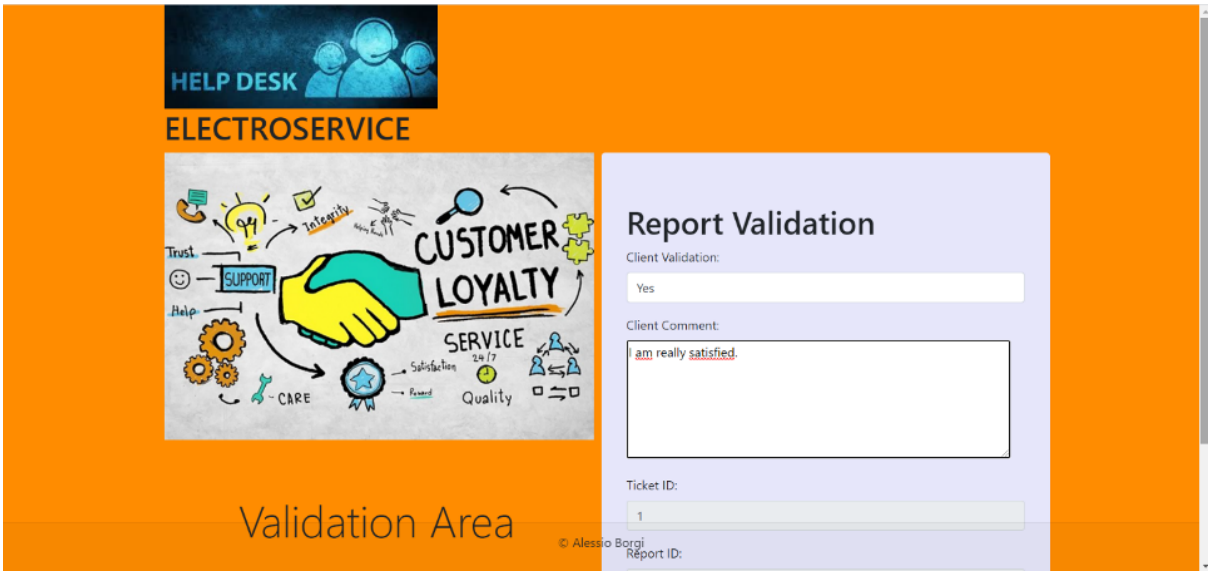


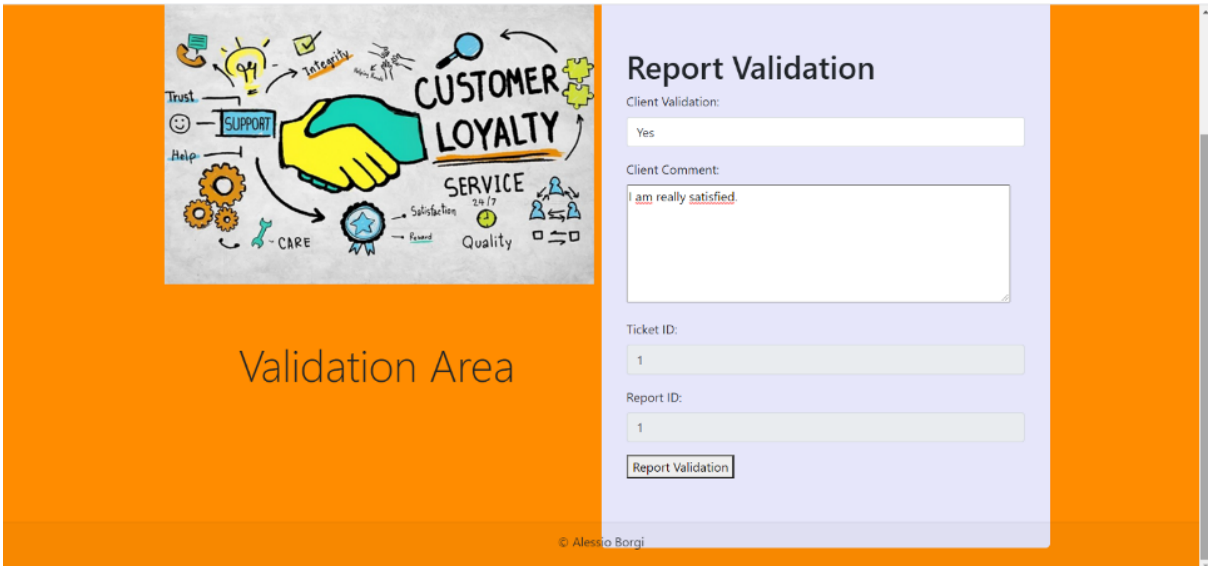
Once on this screen, the Customer will be able to view his Tickets that are still open. In this section, you can also go to two additional screens. By clicking on the "Reports" button, you will be able to view all the Reports that the Technicians have compiled, whether they are from the HelpDesk or the On-Site Team. By pressing the "Closure Date" button instead, the

customer, if satisfied and if he believes that the intervention or the various interventions carried out by the technicians have been satisfactory and have solved the problem, can start the process of closing the Ticket. Furthermore, from this screen, the customer can return to the previous screen by pressing the "Back" button. The screens of the signed Reports and the Ticket Closure will follow. On the next screen, there will be a report to explain and view the intervention of the technician. How the Report will be submitted will be explained in the Technical section.

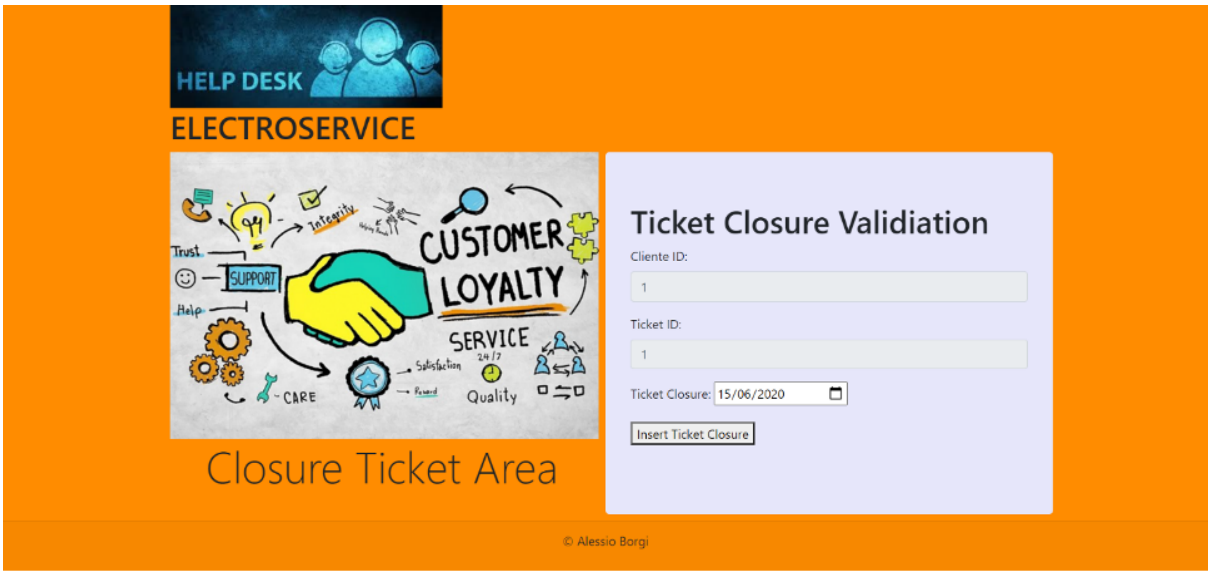


On this screen, it is possible to view all the reports that the technicians have subscribed to for the respective ticket. It is also possible, by pressing the "Validation" button, to validate the report, and also to express an opinion on the intervention carried out by the technician. This part will be very important for the managers, who will have direct feedback and will have to manage the CUSTOMER's SATISFACTION. Next, is the report validation screen.

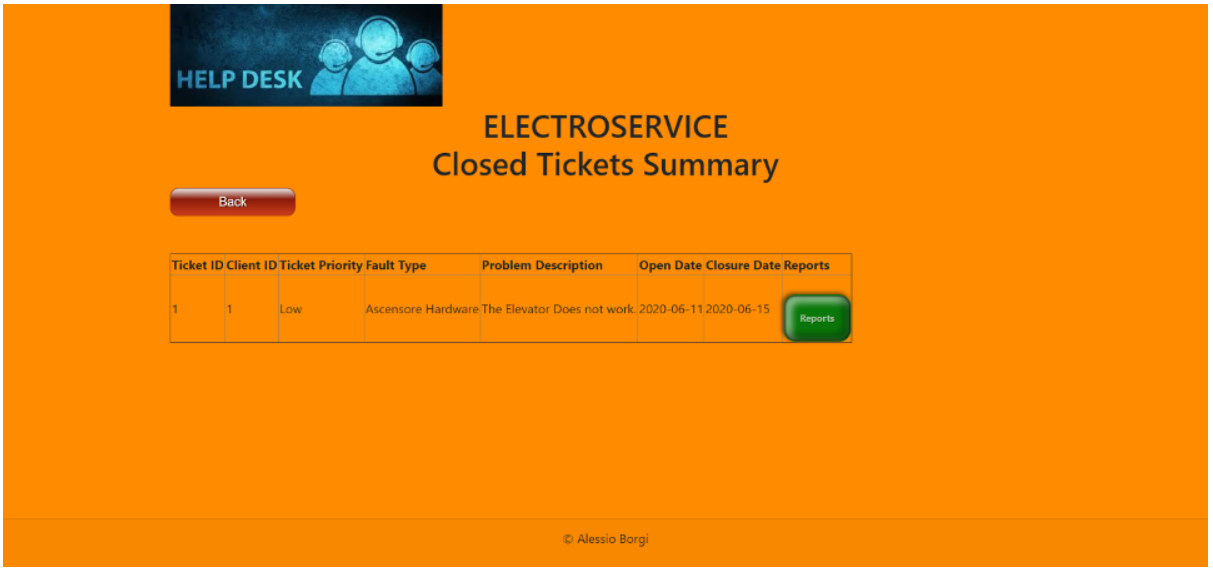




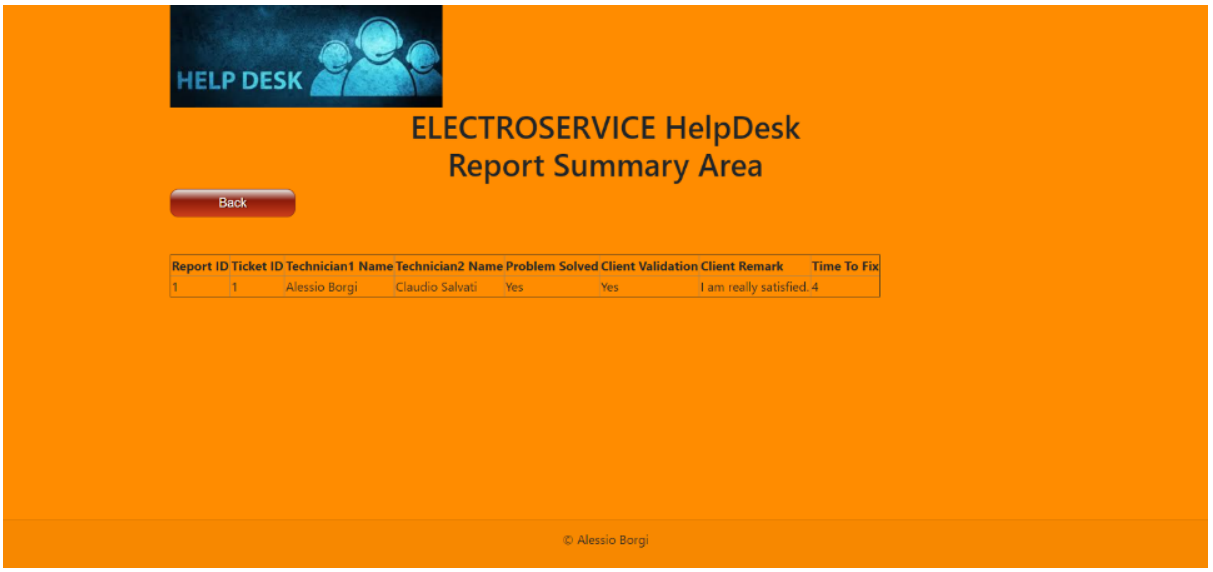
Returning to the open ticket display screen, you can also press the "Closure Date" button. On this screen, it is possible to permanently close the Ticket. In fact, this action, as well as the opening of the ticket, is reserved for the customer himself. In fact, it will be possible to close the Ticket by entering the closing date.



Once the closing date of the Ticket is entered, it will only be possible to view it in the "Closed Tickets" section and no longer in the "Ticket Review" section. By accessing the aforementioned section, the customer will therefore find the following screen.



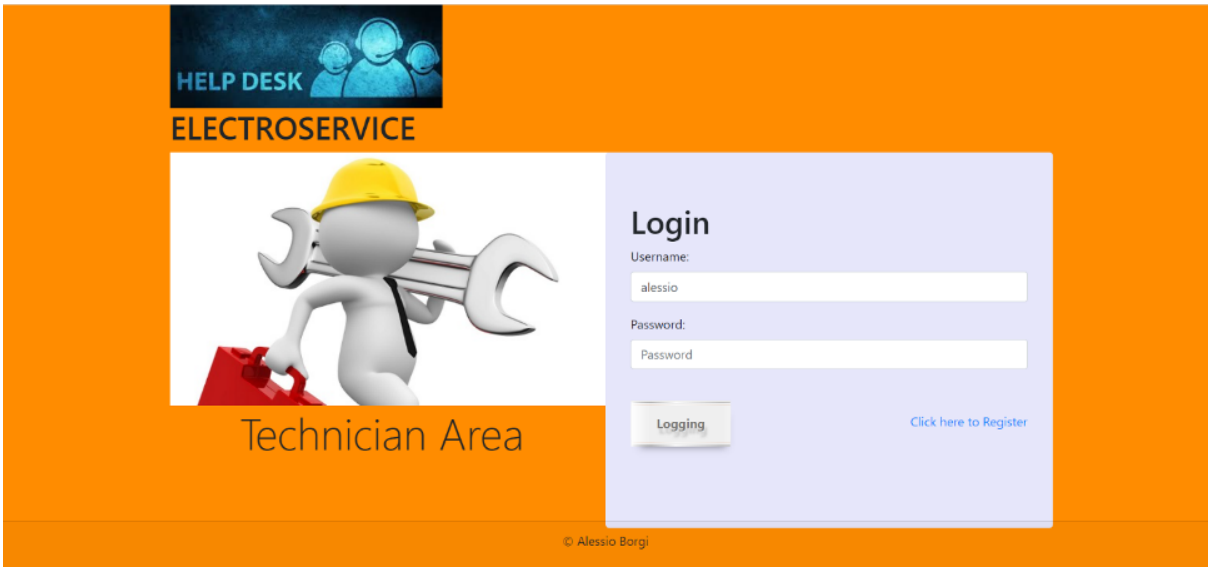
From this screen, the customer can view the list of submitted and already closed tickets. In addition, he can also view the reports related to each of them. The next screen shows the list of reports related to the closed ticket on the previous screen.



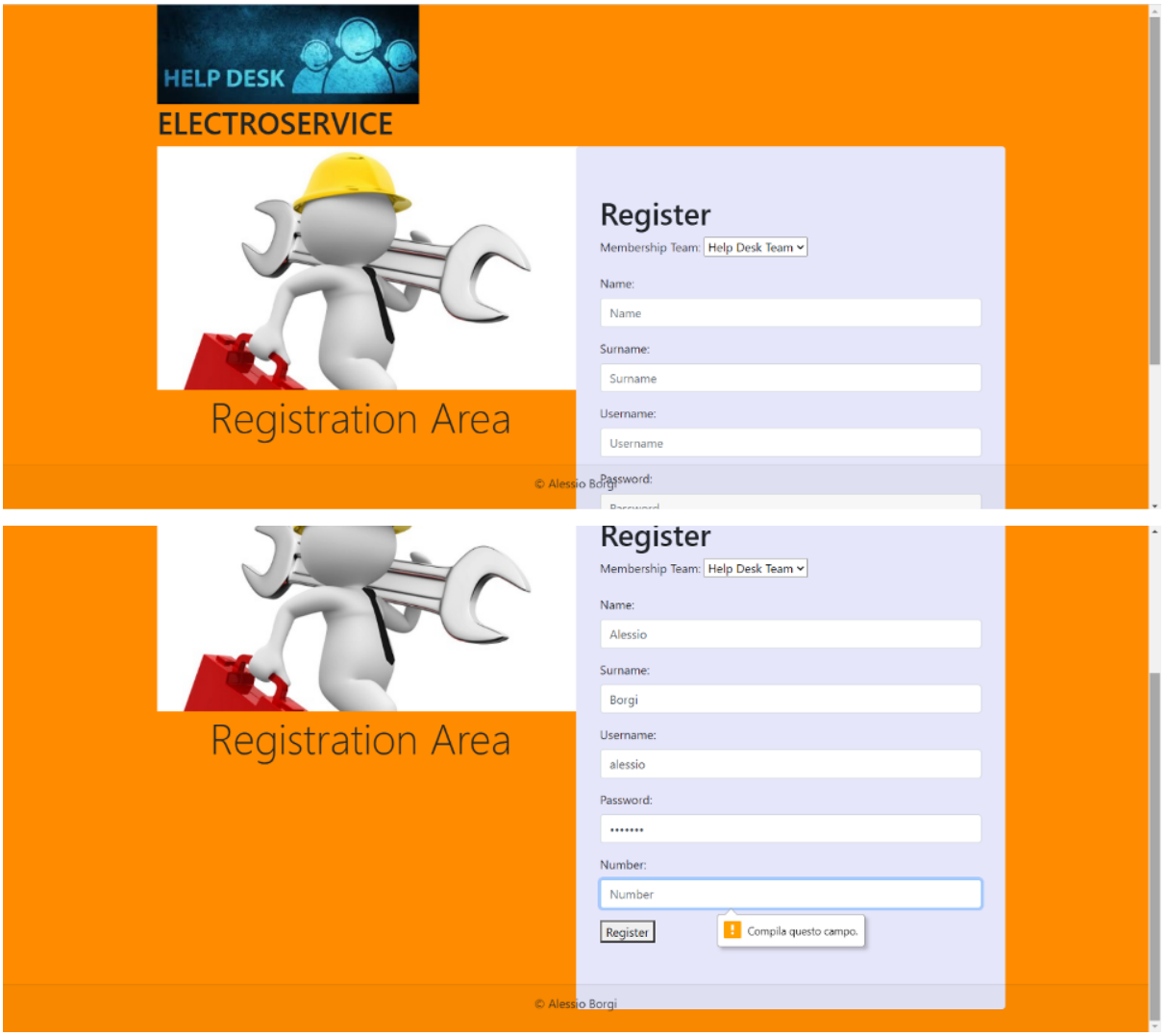
On this screen, you can see all validated reports.

TECHNICIAN INTERFACE:

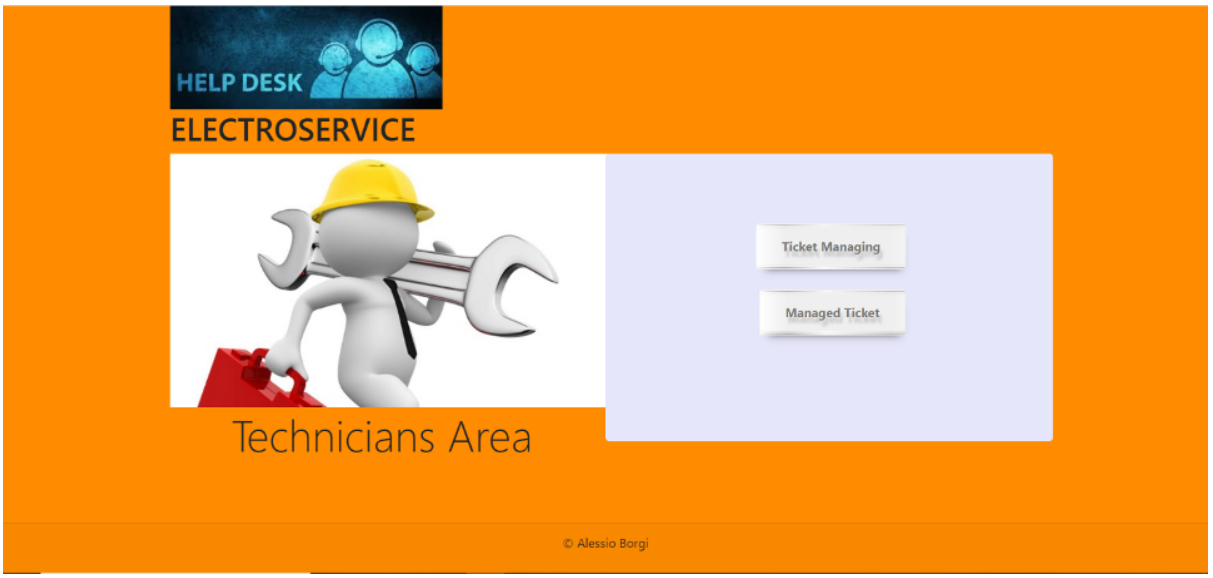
After pressing the "Technicians Area" button, the following screen will be displayed reserved for initial access for the technicians of the Electroservice company.



From this screen, as in the case of the Customers section, you can directly access the technical area or register. By clicking on "Click Here to Register", the following screen will be displayed.

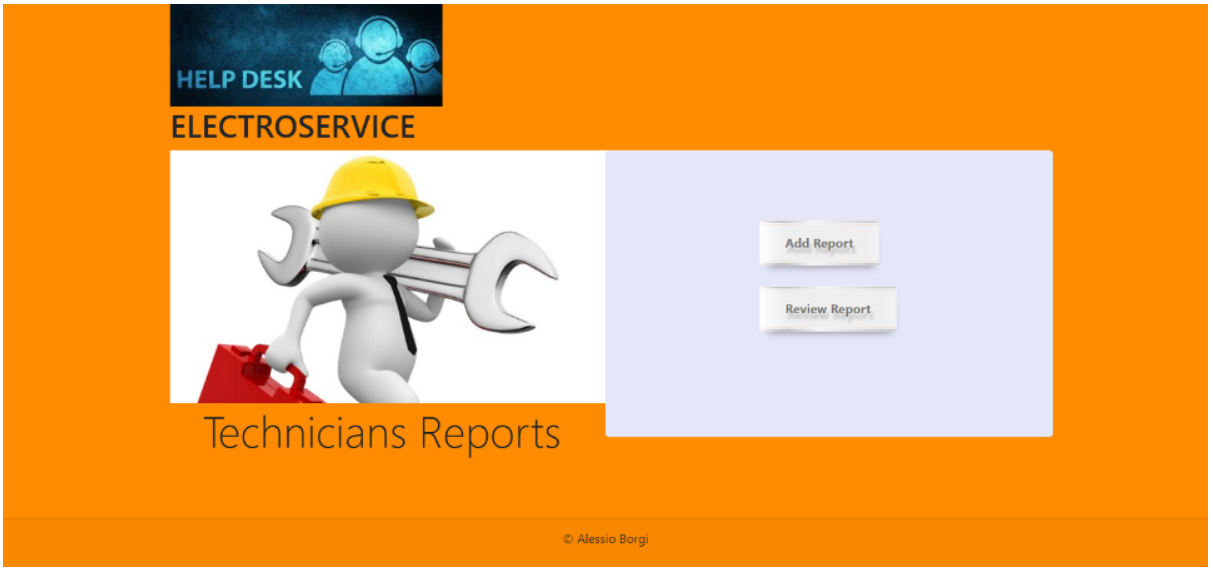


Also in this case it will be necessary to enter all the fields to register. Once logged in, you will be directed to the login screen which can be accessed using your credentials.
After logging in, the technicians will view this screen.



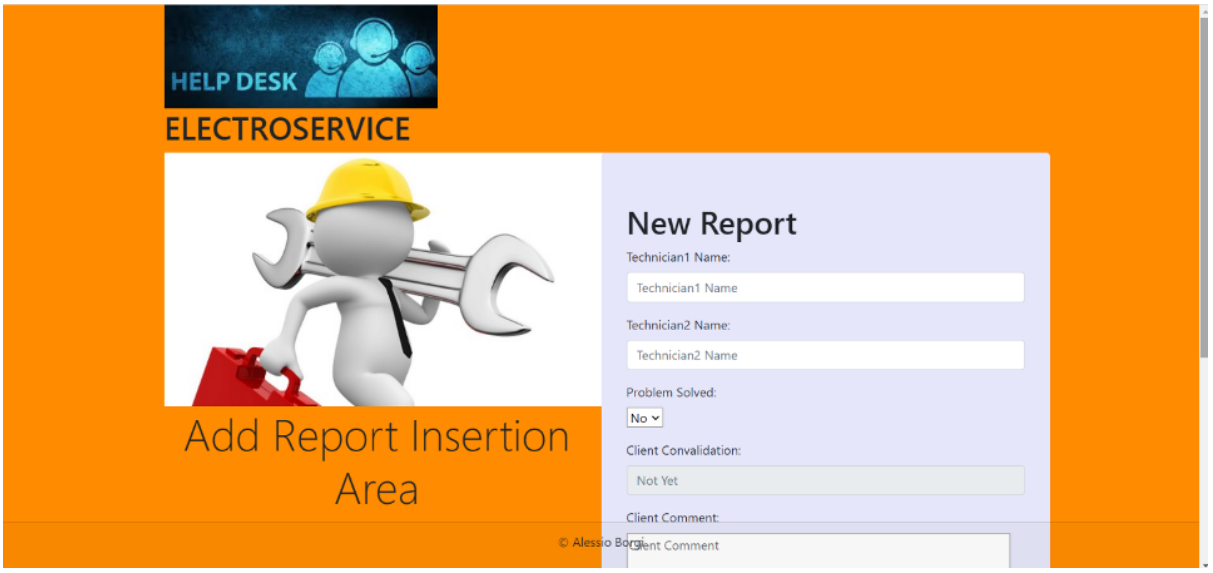
The Technician, as soon as he enters the reserved area, can either start managing a Ticket or view the tickets he has managed and which are closed. We will first explain the case in which We want to manage new Tickets and then we will deal with the other case. We emphasize that now you will find a second ticket just submitted by the customer, as the ticket dealt with previously in the section of the customer's screen has already been closed.

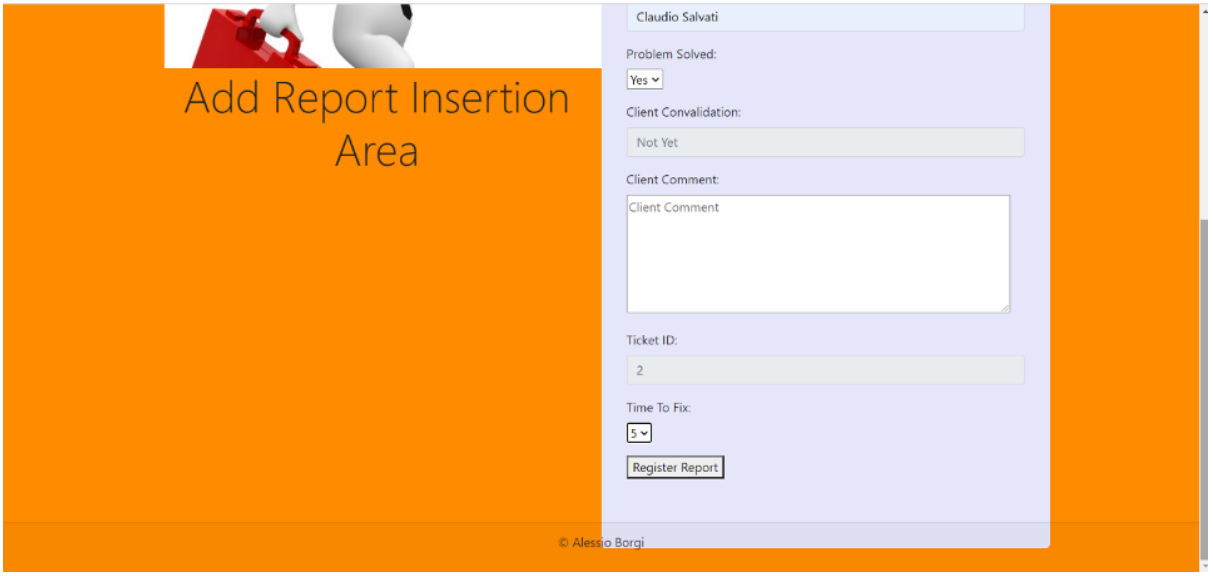
In the Open Tickets management screen, technicians will be able to read the problems encountered by customers. By deciding to take charge of a ticket, they will intervene promptly and then have to access "Reports". At this point, you will find the following screen.



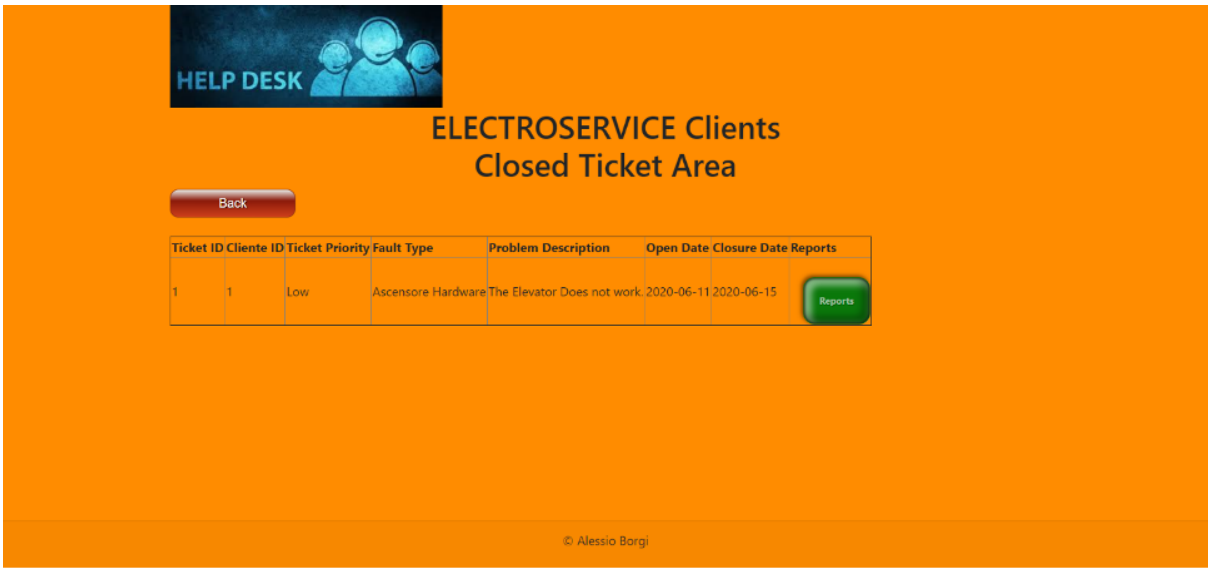
From this screen, you can either view the Reports of interventions already carried out and then see at what point the situation of the fault/problem is, through the "Review Report" button or submit a new Report.

By accessing the "Add Report" screen, the technician will find the following:

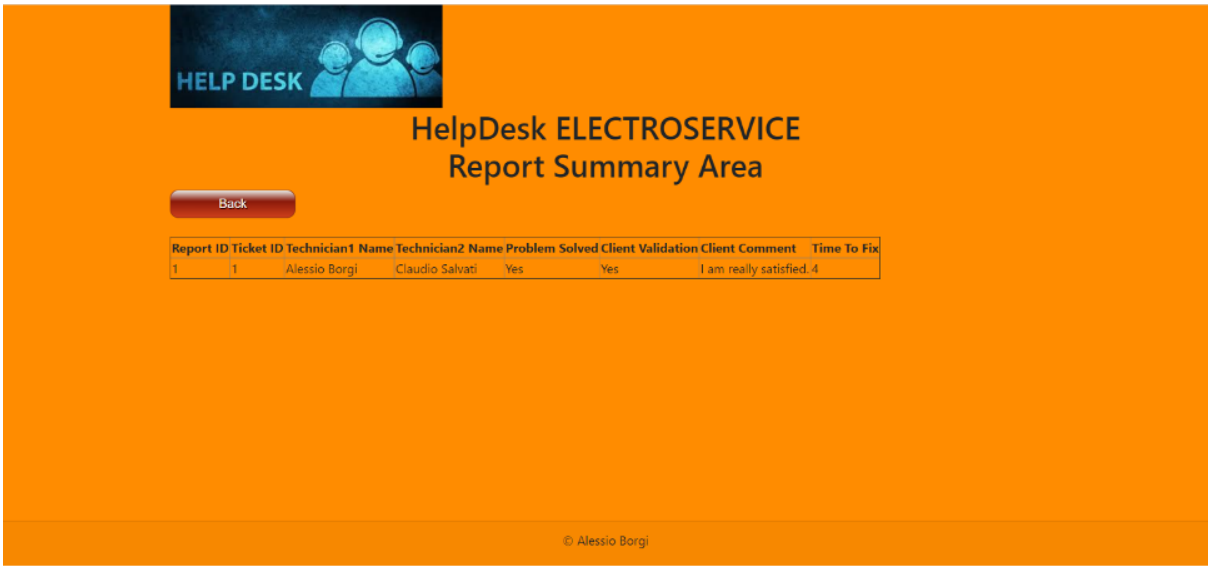




In this way, it will be possible for the technician to register an intervention form, the so-called Report. The "Client Validation" and "Client Comment" fields cannot be edited as it will be up to the Client to do so in the already explained section of Report Validation. Once the Report has been registered, the technician will be directed to the login area.
Returning to the initial section and clicking on the "Managed Ticket" button you will instead arrive at the following screen which will display the complete list of all tickets already managed and closed.



By clicking on the “Reports” button, it will be possible to view all the Reports of the interventions carried out related to that Ticket.



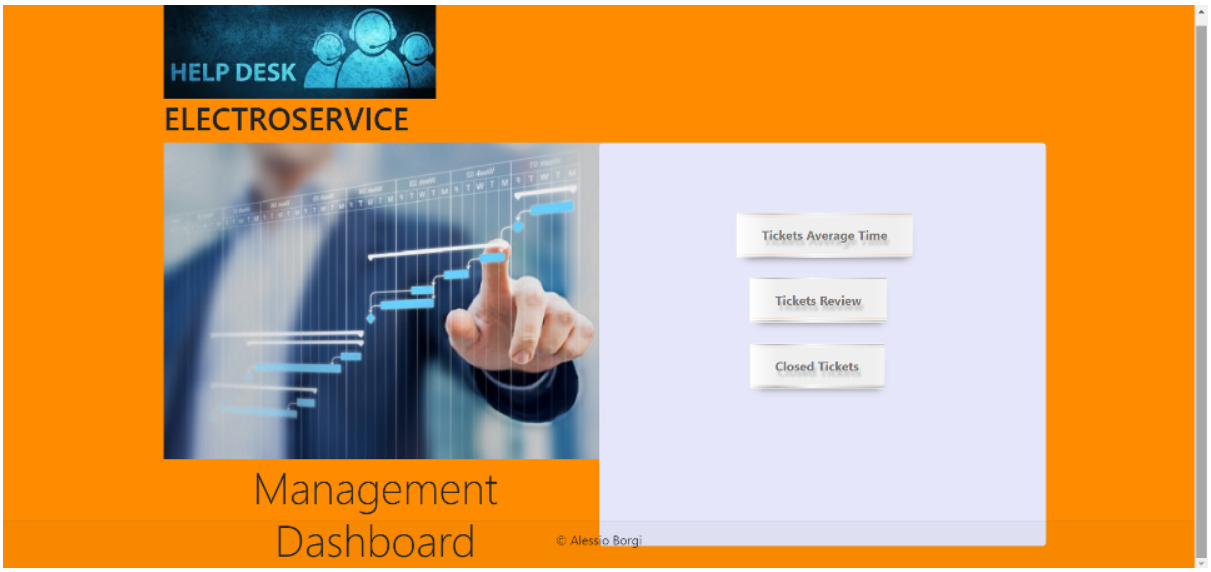
MANAGER INTERFACE:

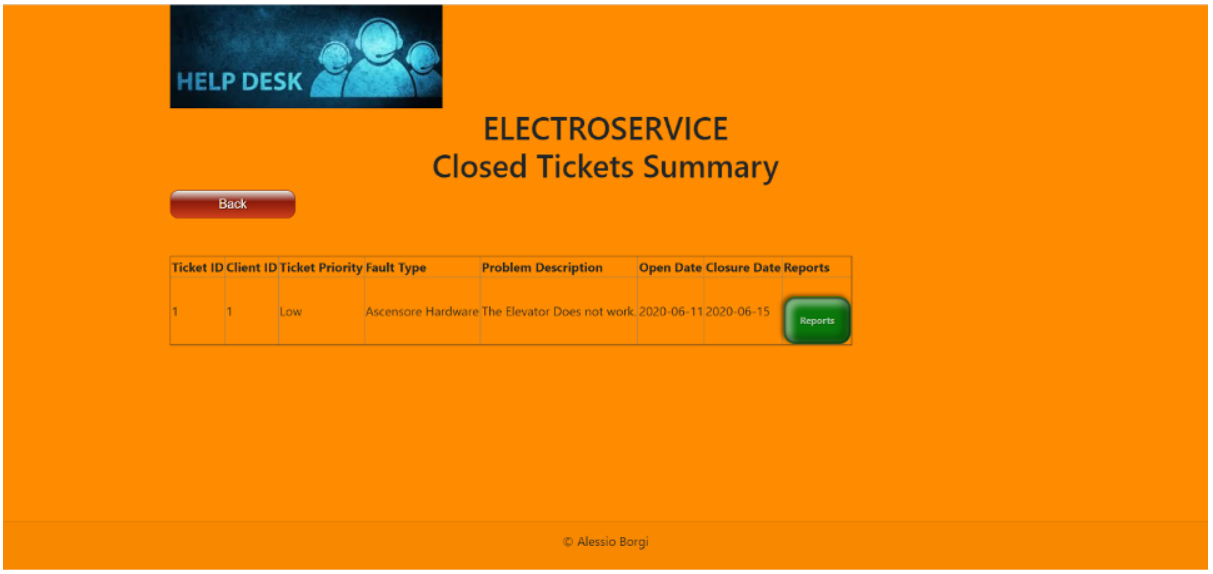
As the last interface, we developed the Management area. The latter we created specifically to display the queries that were highlighted in the text. The Managers of Electroservice, in fact, must have a general overview both of the work of the technicians, on the CUSTOMER SATISFACTION, and on the average ticket closing time, by entering a date as a search range. The management interface, from the database creation page, is accessible by pressing the "Managers Area" button. The interface that will follow will be the home page that managers will encounter when they have to supervise the work of the teams.

The Executives login area will allow, as in the other interfaces, to log in to the restricted area or to register. If the executive needs to register, this screen will be displayed.

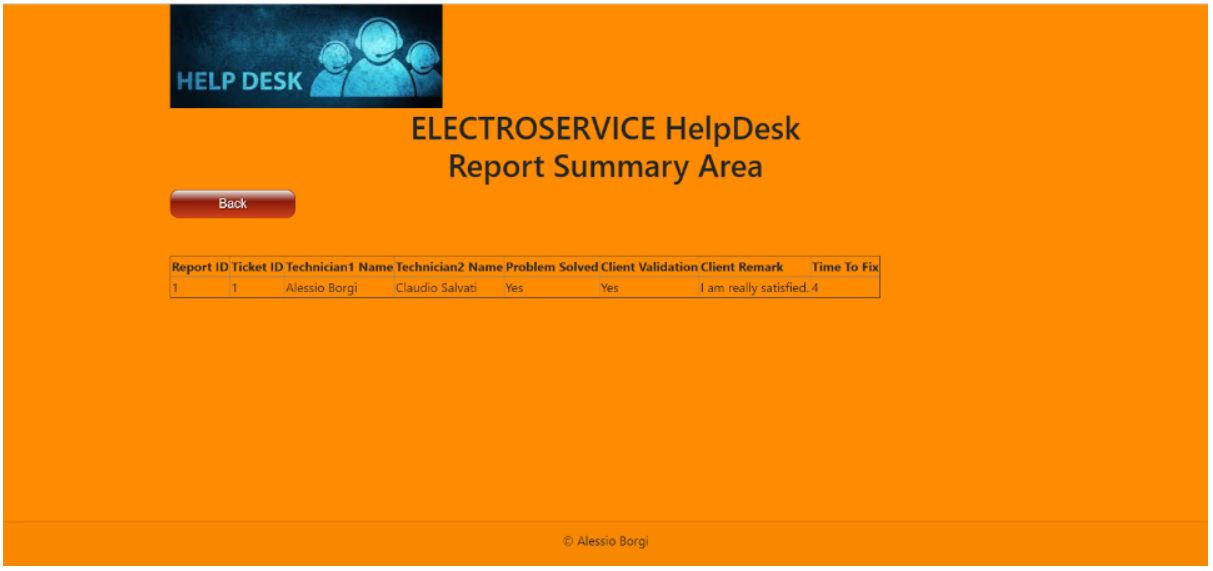
The Manager will be able to carry out three actions: View all closed tickets and related tickets, view all open tickets and reports made, and, by entering a search range, calculate the average ticket closing time in the required time frame.
The Manager, by accessing the closed Tickets, using the "Closed Tickets" button will display the following screen.

Just like the other interfaces, all data must be entered for registration to be successful. Once registered, the managers will be directed to the login area. Once logged in, managers will be able to access the supervision menu which will have the following screen.

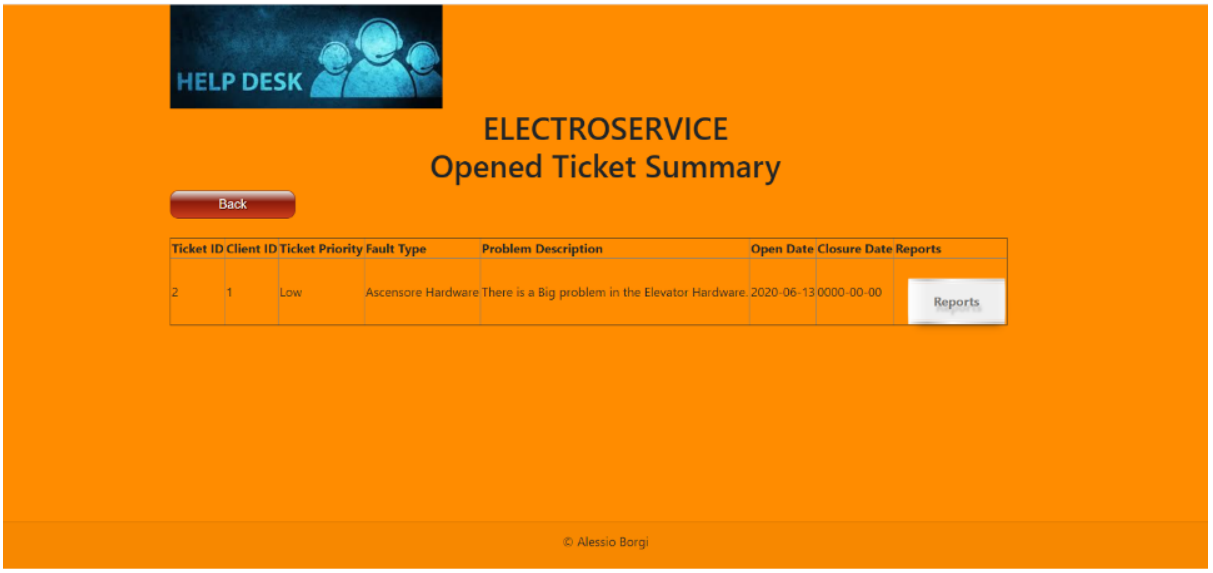




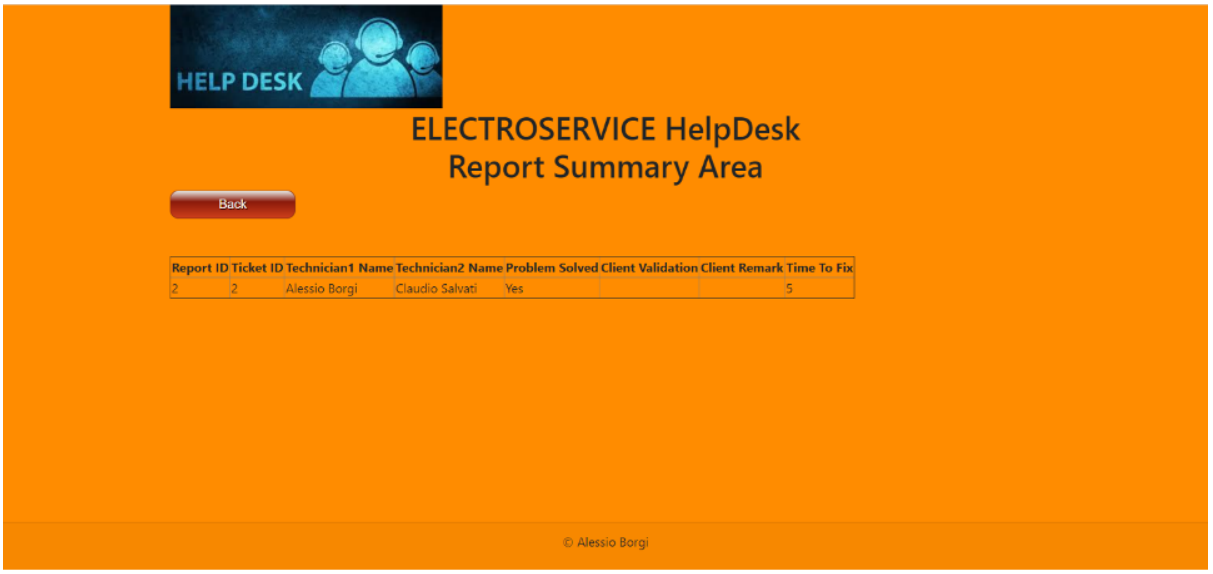
This screen will offer him the opportunity to view the complete list of closed tickets. By clicking on "Reports", you can also view the related reports.



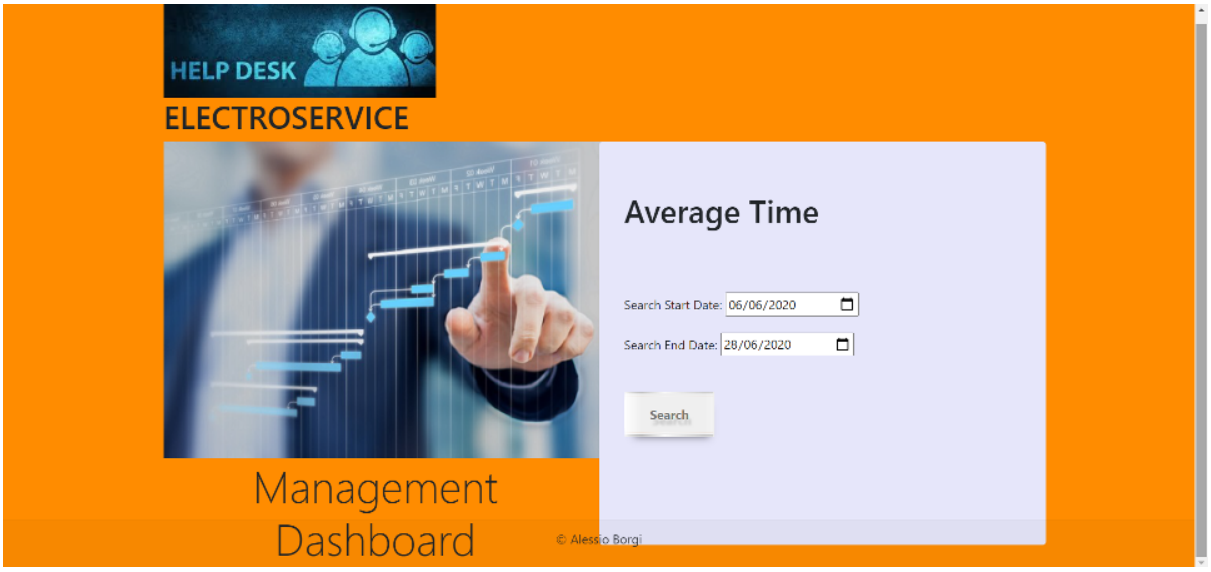
Returning to the executive management menu, the Executive, by accessing the open Tickets, using the "Tickets Review" button, will display the following screen. This screen displays what was required by the trace, the first Query.



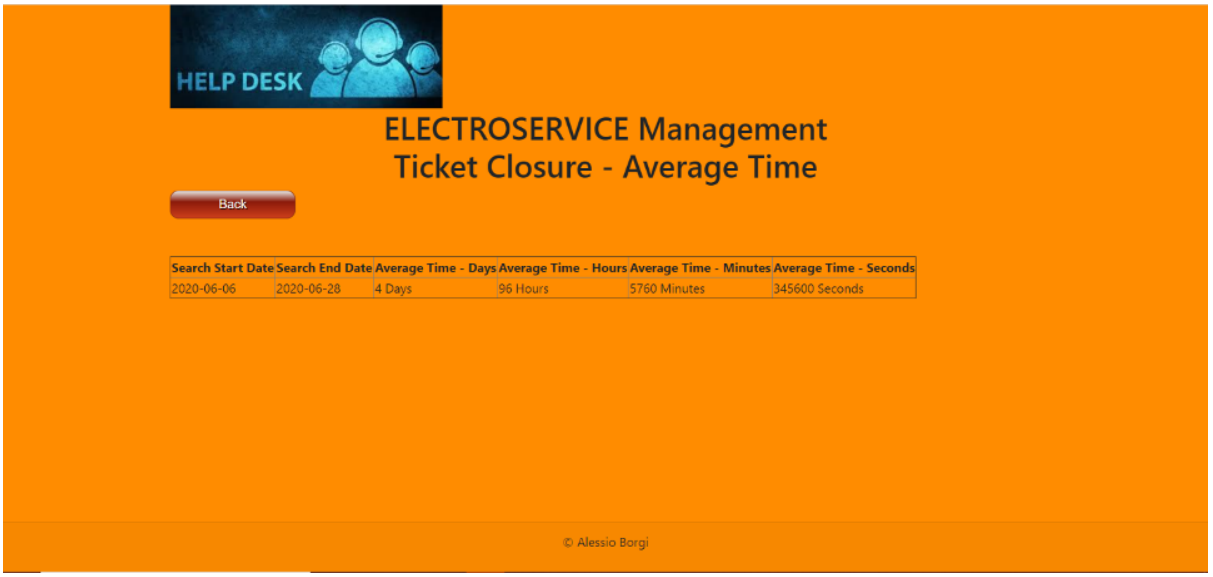
Also in this case you will be able to view the reports of the interventions carried out up to then.



Returning to the executives management menu, the Executive, by accessing the section that calculates the average closing of a ticket, will display the following screen using the "Tickets Average Time" button. This screen will display the request for the second trace query.



In this screen, the range of consideration of the open and closed tickets in the aforementioned range will be chosen. The result is displayed on the next screen.

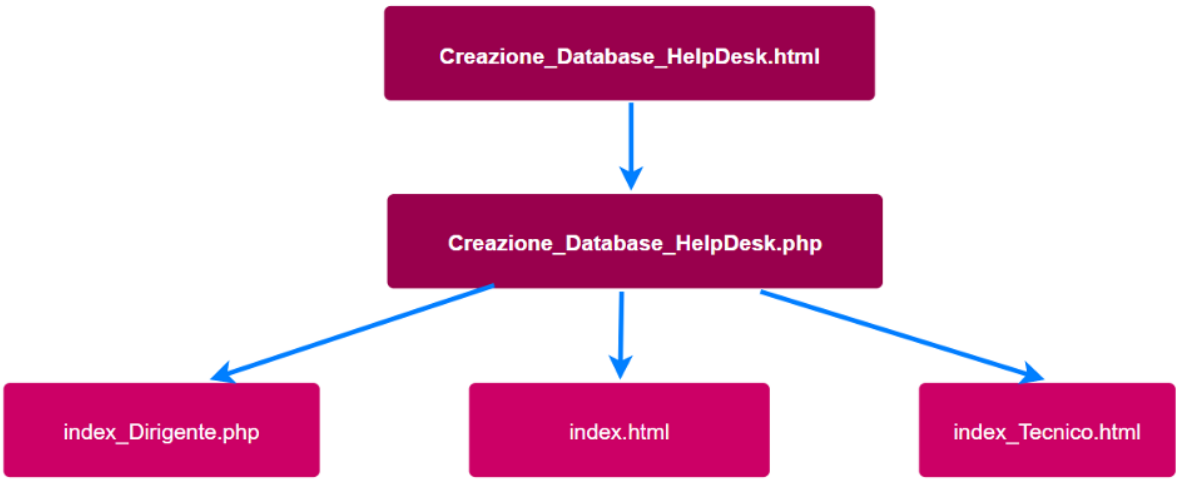


The result of the research is reported in particular as an average in days, hours, minutes and seconds, in such a way as to allow the Executives to be able to carry out an in-depth analysis. This was the last feature of the site that we've implemented.

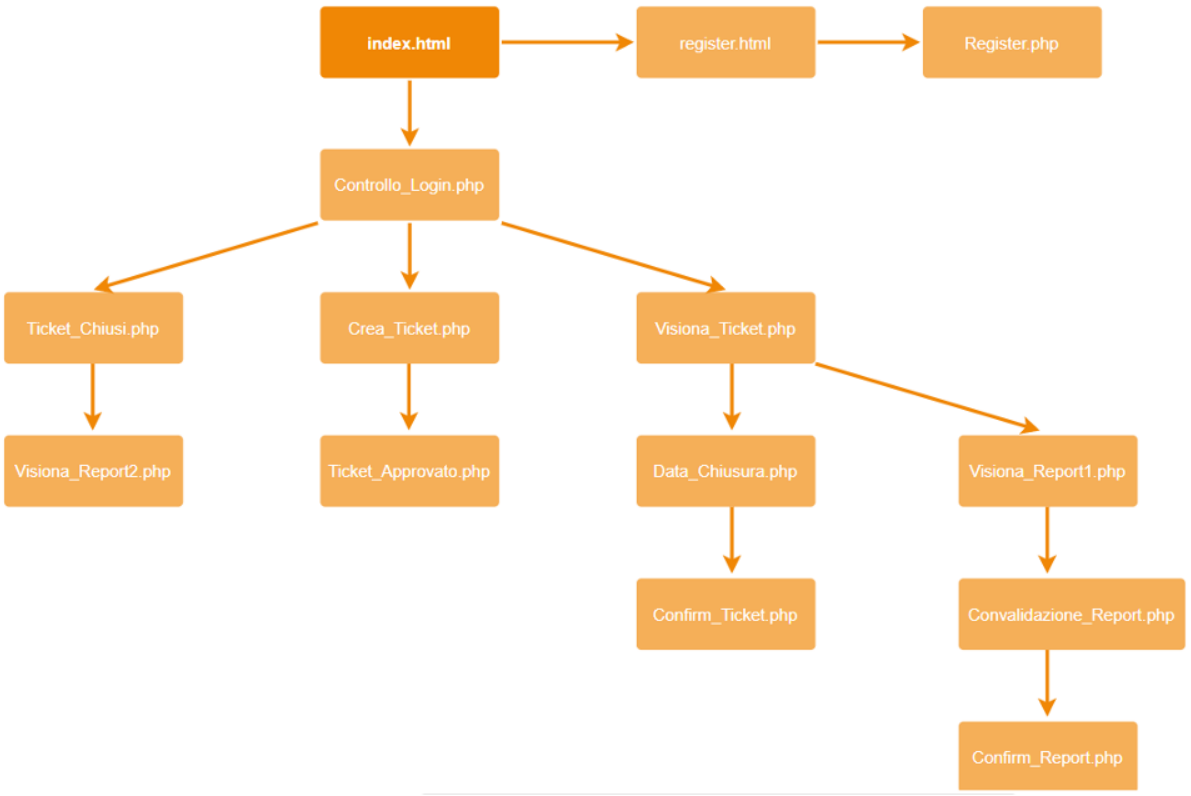
SITE HIERARCHY:

In order to ensure a better understanding of the zip file that we will attach to the report, these are the various hierarchies of the site's files.

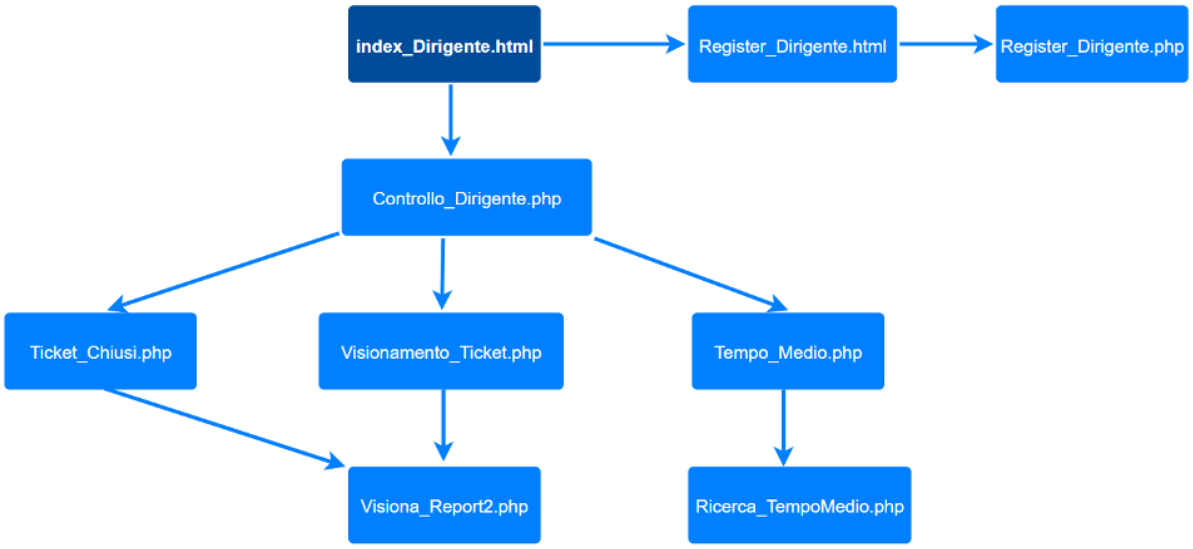
STARTING INTERFACE



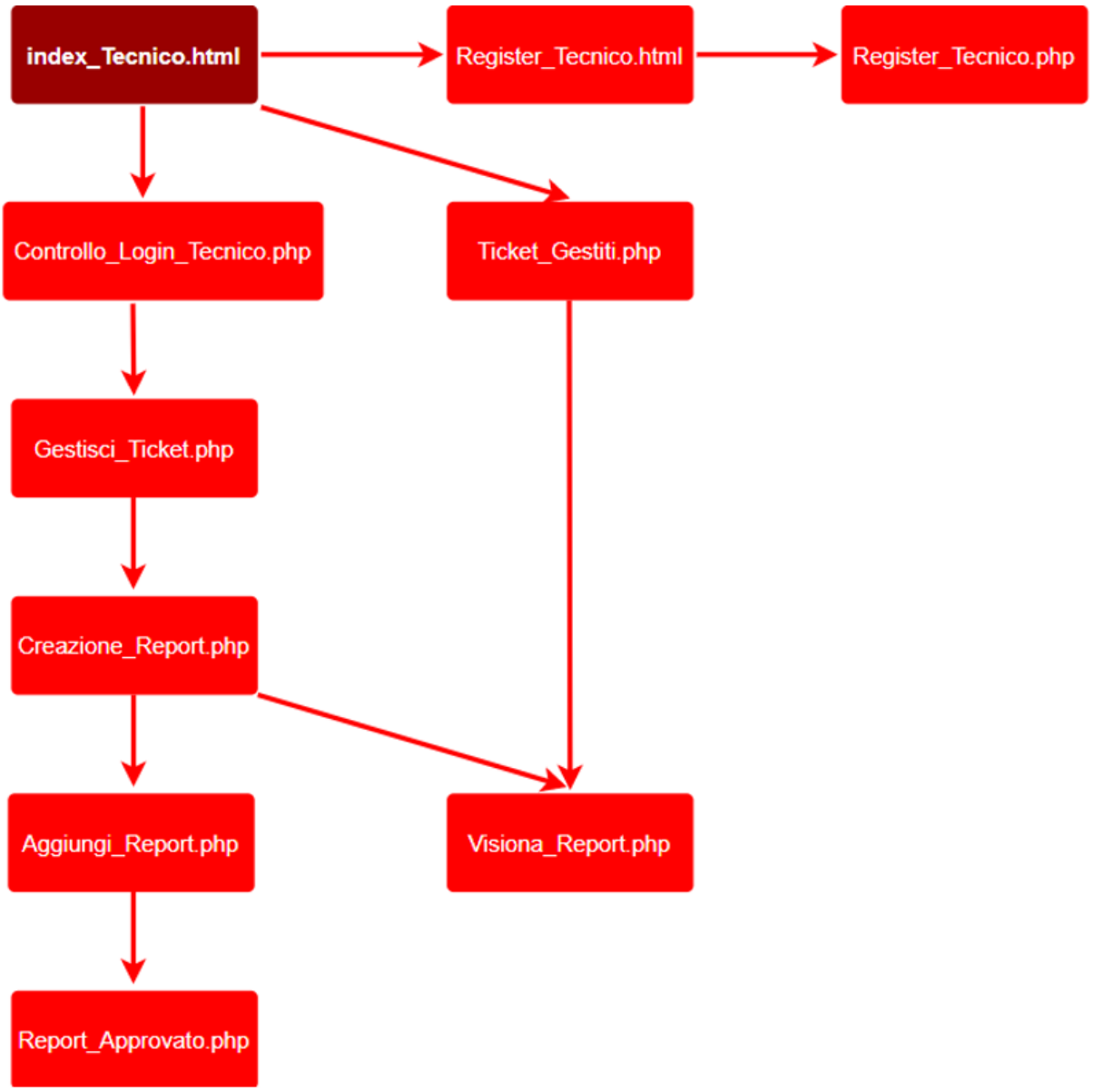
CLIENT INTERFACE



MANAGER INTERFACE



TECHNICIAN INTERFACE


index_Tecnico.html
Register_Tecnico.html
Register_Tecnico.php
Controllo_Login_Tecnico.php
Ticket_Gestiti.php
Gestisci_Ticket.php
Creazione_Report.php
Aggiungi_Report.php
Visiona_Report.php
Report_Approvato.php

FUTURE: NEXT VERSIONS DEVELOPMENTS

We have really had great fun in developing this project. As always, all is improvable!!! We expect that there will be some development in both the Application, and a merging of the two. We excpect to increase the quality of the site, and for sure insert all the functionalities that the Python site has.

THANK YOU FOR YOUR ATTENTION!!!

ALESSIO BORGI & ELENA MUIA