Marmara University

Department of Computer Science and Engineering

CSE3055 Database Systems

Technical Service Database Report

150117013 – Mehmet Ali Yüksel

150117023 – Anıl Şenay

150117072 – Bilgehan Geçici

Due Date: 30/12/2020

Report

Project Description

Database project that we implemented is about managing a technical service of mobile devices and storing the repairing process information of them. This database will be used by the workers of the technical service.

A complete repairment process of one device is as following:

Step-1: A customer arrives to service and contacts a Smart Service Employee. Then customer is asked to give their personal and device's information by the employee. Registration is completed, the customer gives their device and leaves the service.

Step-2: Since device's information is registered, warranty information is also known on the system. According to warranty information device is handed in to a related technician (In Warranty Technician: only repairs the devices with valid warranty conditions and date. Out of Warranty Technician: only repairs the devices with invalid warranty conditions and date.)

Step-3: Technician detects the problem and changes the damaged parts of the device with a new one. To do this replacement first technician needs to check storage if the needed parts are available. If yes, the technician repairs the device and fills the repairment field of this specific case with the actions he/she took. Such as; changed parts, end time of the repairment, repairment degree, remark about the actions taken in details. If no, the technician creates an order and skips the current repairment until the ordered part comes.

Step-4: Repairment is completed and device is tested. The device is ready to deliver. The customer comes to take their device, contacts to accountant. Accountant states the total cost to the customer and payment (if needed) is done via several payment methods.

This is an example scenario which covers most of the steps, different scenarios also happen.

Our database handles storing information of repairing processes mentioned above. Information of each actor, device and process status are kept. Different actions are taken in different events related to this information.

Scope

What is included?

- Personal information of the actors and their performances (employees, customers) is stored.
- Since a device is our main object in this business, we keep detailed information about devices.
- Categorized case description is stored to have detailed data about different cases and to understand the problem.
- Repairment is the business, so we keep detailed information about this process. Technicians update the repairment information during the process, according to their actions.
- Available parts (including spare devices) to use is kept and these parts are stored in a physical storage and we also keep the information of physical place of the parts(boxNumbers).
- Orders of needed parts are kept to track orders.
- Payment information is stored along with the process information to inform the customers during payment.
- Inserting new actors, devices, creating new repairments, making orders and updating this information is handled.

What is excluded?

- Salary of the employees is not included because it is included in a different database along with insurance information and is not a requirement in this project.
- Queuing of customers inside the services is not included because it is handled in a different system and is not a requirement in this project. Categorized case description is stored to have detailed data about different cases and to understand the problem.
- Tracking of spare devices, which are given to customers after they leave their device in the service, is not included because it is handled on papers, physically and is not a requirement in this project. Available parts (including spare devices) to use are kept and these parts are stored in a physical storage and we also keep the information of physical place of the parts(boxNumbers).
- Device logs are not covered in details because logs of devices are taken via special application in this service and very technically detailed also is not a requirement in this project. There's only a placeholder of this feature in our project, represented as deviceLogs in Customer Device table. Payment information is stored along with the process information to inform the customers during payment.
- Inserting new actors, devices, creating new repairments, making orders and updating this information is handled.
- Testing process of the devices after repairment is not included because it is also handled in detail via device log application and not a requirement of this project.

Data and Requirement Analysis

We indicated data types, defaults, unique variables, identities, indexes and computed variables. We usually use identities at ID values for simplicity. For the default values we usually use bit data types for that. We will cover them in store procedures. For the computed values, we usually use our date variable types. We also add unique values for our table variables. We usually use unique for the data types that also can be primary key. We indexed the most frequent selected foreign keys to have a speed efficiency in queries.

We indicated our requirements based on efficiency and functionality, we used also some check constrains that also optimize the database and prevent some functionality errors. Here is the list of the necessary requirements.

- The Database project can be used by multiple employees.
- The Database project must save the information taken from the customers.
- Each employee must have a unique account on system.
- Only admins can update or change saved information after they've been saved.
- No multiple employees can serve the same customer device simultaneously in vice versa.
- If a customer has to wait more than needed time for their device to be repaired, they are provided with a spare device, excluding tablets and smart watches. This case may occur when a part needs to be changed and there aren't any available new parts in the storage.
- If an in-warranty technician detects any out-of-warranty condition on the device he/she is working on, then the technician must assign the device to a technician who repairs only out-of-warranty devices.
- Customers can demand for change of their devices unless the device meets the requirements of changing procedure.
- An employee must be over 18 years old.
- An employee's username's length must be greater than 3 and must be unique.
- Each employee has a unique e-mail.
- Repairment degree can only be 0, 1, 2 or 3. This degree represents the level of labor done on repairment and cost is determined accordingly.
- In storage quantity and boxNumber should be greater than or equal 0.

Business Process and Their Definitions

Goals

The purpose of the process is to provide a wide range database system to a technical service who needs the store information about their customers, employee, repairment process and the parts being physically stored in their building etc. This business is created in order to simplify the information storage and keep the operation of repairment work effective and quick. We can understand our success from our customer's satisfaction from the system we created.

Planning and Mapping Process

Our first strategy to achieve our goal is to understand our customer's requirements. Then split the work parts between team members and execute. After that we keep our customer informed about our process, ask them their opinions and make improvements guided by our customer if needed. Repeat this process until the work is done.

Actions and Stakeholders

All the team members have participated in creating tables, procedures, triggers and views in SQL server and testing their functionality. We applied extreme programming method.

Our customer Destek Bilişim (Kadıköy Şubesi) provided necessary information about their business process and their requirements. Also, they provided overviews and feedback to guide us in our progress.

Business Rules

- A Manager manages multiple employees.
- An employee must be over 18 years old.
- An employee's username's length must be greater than 3 and must be unique
- Each employee has a unique e-mail
- An employee may request multiple orders.
- An order has to have one type of ordered part vice versa.
- Multiple parts may be ordered.
- Exactly one storage has at least one part.
- Multiple parts may be needed to change in at most one repairment.
- At most one accountant may manage multiple payments.
- Exactly one customer may have multiple payments.
- Exactly one customer has at least one customer device.
- Exactly one customer has at least one repairment process.
- Exactly one customer device has exactly one case.
- Exactly one customer device has at least one repairment process.
- At least one employee may control multiple repairments.
- Exactly one employee provides exactly one spare device to exactly one customer.
- Exactly one repairment requires at most one payment.
- Exactly one case type has at least one case category
- Exactly one case category has at least one case specification
- An employee may have many employee availability
- A device may have many device logs
- A payment must have exactly one payment method
- Repairment degree can only be 0, 1, 2 or 3. This degree represents the level of labor done on repairment and cost is determined accordingly.
- In storage quantity and boxNumber should be greater than or equal 0.
- Exactly one case type has at least one case category.
- Exactly one case category has at least one case specification.
- An employee may have many employee availabilities.
- A device may have many device logs.
- A payment must have exactly one payment method.

Tables

We implemented 24 tables for our Technical Service (Destek Bilişim). We demonstrate the main parts of the business process of our technical service below, including all necessary tables listed with their definitions.

EMPLOYEE								
Definition: This table stores personal and business-related information of employees.								
Check Constraints: All the employees must be over 18 - username's length value must be greater than 3.								
Triggers: -								
PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☑	☐	☐	☐	☐
☐	firstName	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	lastName	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	phoneName	nvarchar(20)	☐	☐	☐	☐	☐	☐
☐	username	nvarchar(30)	☐	☐	☐	☑	☐	☐
☐	password	nvarchar(32)	☐	☐	☐	☐	☐	☐
☐	email	nvarchar(100)	☐	☐	☐	☑	☐	☐
☐	address	nvarchar(150)	☐	☐	☐	☐	☐	☐
☐	dateOfBirth	date	☐	☐	☐	☐	☐	☐
☐	startDate	date	☐	☐	☐	☐	☐	☐
☐	isManager	bit	☐	☐	☑	☐	☐	☐
☐	isSmartService	bit	☐	☐	☑	☐	☐	☐
☐	isTechician	bit	☐	☐	☑	☐	☐	☐
☐	isStorageMan	bit	☐	☐	☑	☐	☐	☐
☐	isTester	bit	☐	☐	☑	☐	☐	☐
☐	isAccountant	bit	☐	☐	☑	☐	☐	☐

EMPLOYEE_AVAILABILITY

Definition: This table stores the employee's available days and their start and end work hours

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	smallint	☐	☑	☐	☐	☐	☐
☐	employeeID	tinyint	☑	☐	☐	☐	☐	☐
☐	monday	bit	☐	☐	☑	☐	☐	☐
☐	tuesday	bit	☐	☐	☑	☐	☐	☐
☐	wednesday	bit	☐	☐	☑	☐	☐	☐
☐	thursday	bit	☐	☐	☑	☐	☐	☐
☐	friday	bit	☐	☐	☑	☐	☐	☐
☐	saturday	bit	☐	☐	☑	☐	☐	☐
☐	startHour	time(0)	☐	☐	☑	☐	☐	☐
☐	endHour	time(0)	☐	☐	☑	☐	☐	☐

DEVICE_LOG

Definition: This table stores detailed information about customer's device on its parts.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☑	☐	☐	☐	☐
☐	deviceID	bigint	☑	☐	☐	☐	☐	☐
☐	logMessage	tinyint	☐	☐	☐	☐	☐	☐
☐	logDate		☐	☐	☐	☐	☐	☑

REPAIRMENT

Definition: This table represents the repairment process (during and after process). Stores the information gathered from devices, customers and the employees.

Check Constraints: -

Triggers: It updates the **numOfLateRepairment** value by one when the repairmentDuration exceeds 1 hour.

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	int	☐	☑	☐	☐	☐	☐
☐	deviceID	bigint	☑	☐	☐	☐	☑	☐
☐	caseID	int	☑	☐	☐	☐	☐	☐
☐	customerID	int	☑	☐	☐	☐	☑	☐
☐	employeeID	tinyint	☑	☐	☐	☐	☑	☐
☐	repairmentStartDate	datetime	☐	☐	☐	☐	☐	☐
☐	repairmentEndDate	datetime	☐	☐	☐	☐	☐	☐
☐	isPartWaited	bit	☐	☐	☑	☐	☐	☐
☐	isInWarranty	bit	☐	☐	☑	☐	☐	☐
☐	remark	nvarchar(100)	☐	☐	☐	☐	☐	☐
☐	repairmentDuration		☐	☐	☐	☐	☐	☑

CASE

Definition: This table keeps descriptive information about customer's complaint taken from themselves.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	int	☐	☑	☐	☐	☐	☐
☐	caseType	tinyint	☑	☐	☐	☐	☐	☐
☐	caseCategory	tinyint	☑	☐	☐	☐	☐	☐
☐	caseSpecification	tinyint	☑	☐	☐	☐	☐	☐
☐	caseDescription	nvarchar(100)	☐	☐	☐	☐	☐	☐

CASE_TYPE

Definition: This table stores the case type.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☑	☐	☐	☐	☐
☐	type	nvarchar(30)	☐	☐	☐	☑	☐	☐

CASE_CATEGORY

Definition: This table stores the case categories.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☑	☐	☐	☐	☐
☐	category	nvarchar(30)	☐	☐	☐	☑	☐	☐
☐	caseType	tinyint	☑	☐	☐	☐	☐	☐

CASE_SPECIFICATION

Definition: This table stores the case specification.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☑	☐	☐	☐	☐
☐	specification	nvarchar(50)	☐	☐	☐	☑	☐	☐
☐	caseCategory	tinyint	☑	☐	☐	☐	☐	☐

CUSTOMER

Definition: This table stores personal information about customers.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☑	☐	☐	☐	☐
☐	firstName	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	lastName	nvarchar(20)	☐	☐	☐	☐	☐	☐
☐	phoneNumber	nvarchar(20)	☐	☐	☐	☐	☐	☐

CUSTOMER_DEVICE

Definition: This table stores information taken from the customers themselves about their devices, in addition to specifications and condition of their devices.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	deviceID	bigint	☐	☐	☐	☐	☐	☐
☐	customerID	int	☑	☐	☐	☐	☐	☐
☐	model	nvarchar(20)	☐	☐	☐	☐	☐	☐
☐	colorCode	nvarchar(10)	☐	☐	☐	☐	☐	☐
☐	serialCode	nvarchar(15)	☐	☐	☐	☑	☐	☐
☐	warrantyDueDate	date	☐	☐	☐	☐	☐	☐
☐	warranty		☐	☐	☑	☐	☐	☑
☐	physicalCondition	nvarchar(100)	☐	☐	☐	☐	☐	☐
☐	proofOfPurchase	bit	☐	☐	☑	☐	☐	☐

ADDRESS

Definition: This table stores the customer's detailed address.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	int	☐	☑	☐	☐	☐	☐
☐	customerID	int	☑	☐	☐	☐	☐	☐
☐	streetName	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	streetNumber	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	city	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	country	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	zipcode	nvarchar(10)	☐	☐	☐	☐	☐	☐

MANAGER

Definition: This table stores title of managers, for instance "Branch Manager" or "Smart Service Leader".

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☐	☐	☐	☐	☐
☐	title	nvarchar(30)	☐	☐	☐	☐	☐	☐

SMART_SERVICE

Definition: This table stores availability and business-related information of workers in days and hours.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☐	☐	☐	☐	☐
☐	customerHappiness	tinyint	☐	☐	☑	☐	☐	☐
☐	RRR	decimal(5,2)	☐	☐	☑	☐	☐	☐
☐	numOfLateRepairment	tinyint	☐	☐	☑	☐	☐	☐

TECHNICIAN

Definition: This table stores types of technicians and business-related information. There are 2 main type of technicians, these are “In-Warranty Repairer” and “Out-of-Warranty Repairer”.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☐	☐	☐	☐	☐
☐	isInWarranty	tinyint	☐	☐	☑	☐	☐	☐
☐	RRR	decimal(5,2)	☐	☐	☑	☐	☐	☐
☐	numOfLateRepairment	tinyint	☐	☐	☑	☐	☐	☐

ORDER

Definition: This table stores keeps track of the orders.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	orderID	int	☐	☑	☐	☐	☐	☐
☐	totalCost	smallmoney	☑	☐	☑	☐	☐	☐
☐	employeeID	tinyint	☐	☐	☐	☐	☐	☐
☐	orderDate		☐	☐	☐	☐	☐	☑
☐	isConfirmed	bit	☐	☐	☑	☐	☐	☐

ORDERED_PART

Definition: This table stores quantity of ordered parts.

Check Constraints: quantity value must be greater than 0.

Triggers: It updates order’s price when the technician makes the order.

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	orderID	int	☑	☐	☐	☐	☐	☐
☑	partID	bigint	☑	☐	☐	☐	☐	☐
☐	quantity	smallint	☐	☐	☑	☐	☐	☐

PART

Definition: This table stores detailed information of parts.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	bigint	☐	☐	☐	☐	☐	☐
☐	partName	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	partModel	nvarchar(50)	☐	☐	☐	☐	☐	☐
☐	partColor	nvarchar(10)	☐	☐	☐	☐	☐	☐
☐	price	smallmoney	☐	☐	☑	☐	☐	☐

PARTS_NEED_CHANGE

Definition: This table connects the PARTS table and REPAIRMENT table.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	repairmentID	int	☑	☐	☐	☐	☐	☐
☑	partID	bigint	☑	☐	☐	☐	☐	☐

PAYMENT

Definition: This table stores information very similar to a receipt. In addition, it has information about the repairment process to inform the customers.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	int	☐	☐	☐	☐	☐	☐
☐	repairmentID	int	☑	☐	☐	☐	☐	☐
☐	accountantID	tinyint	☑	☐	☐	☐	☐	☐
☐	totalCost	smallmoney	☐	☐	☑	☐	☐	☐
☐	paymentMethod	tinyint	☑	☐	☐	☐	☐	☐
☐	date		☐	☐	☐	☐	☐	☑

PAYMENT_METHOD

Definition: This table stores all the payment methods.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	int	☐	☑	☐	☐	☐	☐
☐	paymentMethod	nvarchar(50)	☐	☐	☐	☑	☐	☐

SMART_SERVICE_REPAIRMENT

Definition: This table represents the repairments which can be done quickly and without a need of technical assistance.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	int	☐	☐	☐	☐	☐	☐
☐	solutionType	tinyint	☑	☐	☐	☐	☐	☐

TECHNICAL_REPAIRMENT

Definition: This table represents the repairments which can only be done via technical operation.

Check Constraints: repairmentDegree value should be 0, 1, 2 or 3.

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	int	☐	☐	☐	☐	☐	☐
☐	repairmentDegree	tinyint	☐	☐	☐	☐	☐	☐

SOLUTION_TYPE

Definition: This table stores the SMART_SERVICE_REPAIRMENT's all solution types.

Check Constraints: -

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	ID	tinyint	☐	☐	☐	☐	☐	☐
☐	solutionType	nvarchar(50)	☐	☐	☐	☑	☐	☐

STORAGE

Definition: This table represents a real storage exists in the technical service building. Keeps the information about stored parts.

Check Constraints: quantity and boxNumber should be greater than or equal 0.

Triggers: -

PK	Column Name	Data Type	FK	Identity	Default Value	Unique	Index	Computed Columns
☑	partID	bigint	☑	☐	☐	☐	☐	☐
☐	quantity	smallint	☐	☐	☑	☐	☐	☐
☐	boxNumber	smallint	☐	☐	☐	☐	☐	☐

VIEWS

We decided to implement 5 views about the tables. We tried to create different views to reach specified data. Here is the list of the views.

1-) employeesWithRepairmentCountsThisMonth

Definition: It selects all the repairments of the technicians in current month interval.

```sql
CREATE VIEW employeesWithRepairmentCountsThisMonth as
SELECT e.ID AS [Employee ID], e.firstName + ' ' + e.lastName AS Employee, empWithCount.repairmentCount AS Repairment
FROM dbo.EMPLOYEE AS e
    INNER JOIN (SELECT r.employeeID, COUNT(r.employeeID) AS repairmentCount FROM dbo.REPAIRMENT AS r
                INNER JOIN dbo.EMPLOYEE AS e ON e.ID = r.employeeID WHERE (DATEDIFF(day, r.repairmentStartDate, GETDATE()) <= 31) AND
                                                                          (MONTH(r.repairmentStartDate) = MONTH(GETDATE()))
                GROUP BY r.employeeID) AS empWithCount ON e.ID = empWithCount.employeeID
```

110 %

Results | Messages

	Employee ID	Employee	Repairment
1	5	Cemal Aksoy	2
2	9	Burak Öztürk	2
3	10	Gizem Güneş	1
4	17	Arda Dolu	1
5	20	Selin Katar	2

(Figure 1.1 Result and Query of the view)

2-) getAvgDurationPerEmployee

Definition: It lists all the employee's average repairment time.

```sql
CREATE VIEW getAvgDurationPerEmployee AS
SELECT e1.ID, e1.firstName + ' ' + e1.lastName AS FullName, averageTable.averageDuration
FROM (SELECT r.employeeID, AVG(r.repairmentDuration) AS averageDuration
      FROM dbo.EMPLOYEE AS e
           INNER JOIN dbo.REPAIRMENT AS r ON e.ID = r.employeeID
                      WHERE (e.isTechnician = 1) OR (e.isSmartService = 1)
                      GROUP BY r.employeeID) AS averageTable
                               INNER JOIN dbo.EMPLOYEE AS e1 ON averageTable.employeeID = e1.ID
```

110 %

Results | Messages

	ID	FullName	averageDuration
1	4	İsmail Özhan	NULL
2	5	Cemal Aksoy	71100
3	6	Meral Gül	NULL
4	8	Bahar Koç	NULL
5	9	Burak Öztürk	NULL
6	10	Gizem Güneş	NULL
7	13	Muhammet Çetin Şen	NULL
8	14	Yeşim Yıldırım	NULL
9	17	Arda Dolu	NULL
10	18	Selim Atar	NULL
11	19	Elif Sever	NULL
12	20	Selin Katar	NULL
13	24	Songül Aslan	NULL

(Figure 1.2 Result and Query of the view)

3-) getMostFreqRepairedModel

Definition: It list most frequent repaired model.

```sql
CREATE VIEW getMostFreqRepairedModel AS
SELECT model, COUNT(*) AS numOfTotalRepairments
FROM dbo.REPAIRMENT AS r
     INNER JOIN dbo.CUSTOMER_DEVICE AS cd ON r.deviceID = cd.deviceID
GROUP BY model
HAVING (COUNT(*) = (SELECT MAX(modelCount) AS Expr1
                    FROM (SELECT cd1.model, COUNT(*) AS modelCount
                          FROM dbo.REPAIRMENT AS r
                               INNER JOIN  dbo.CUSTOMER_DEVICE AS cd1 ON r.deviceID = cd1.deviceID
                          GROUP BY cd1.model) AS repairedTable))
```

110 %

Results | Messages

	model	numOfTotalRepairments
1	SM-T290	4

(Figure 1.3 Result and Query of the view)

4-) numberOfChangedPartsDistinctModel

Definition: It list number of changed parts based on distinct model (Given number 3 as an example).

```sql
CREATE VIEW numberOfChangedPartsDistinctModel AS
SELECT e.ID AS [Employee ID], e.firstName + ' ' + e.lastName AS Employee, p.partName, p.partModel, COUNT(p.partName) AS numberOfChange
FROM dbo.REPAIRMENT AS r
     INNER JOIN dbo.PARTS_NEED_CHANGE AS pnc ON r.ID = pnc.repairmentID
     INNER JOIN dbo.PART AS p ON pnc.partID = p.ID
     INNER JOIN dbo.TECHNICIAN AS t ON r.employeeID = t.ID
     INNER JOIN dbo.EMPLOYEE AS e ON e.ID = t.ID
GROUP BY e.ID, p.partName, e.firstName, e.lastName, p.partModel
HAVING (COUNT(p.partName) >= 3)
```

110 %

Results | Messages

	Employee ID	Employee	partName	partModel	numberOfChange
1	17	Arda Dolu	Mikrafon	SM-S8+	3

(Figure 1.4 Result and Query of the view)

5-) sameDeviceRepairmentsMoreThanThree

Definition: It list all related repairment information that devices which are repaired more than 3 times.

```sql
CREATE VIEW sameDeviceRepairmentsMoreThanThree AS
SELECT r.ID, e.firstName + ' ' + e.lastName AS employee, cu.firstName + ' ' + cu.lastName AS customer,
       r.deviceID, cd.model, cd.colorCode, r.remark, ct.type, cc.category, cs.specification, c.caseDescription
FROM dbo.REPAIRMENT AS r
     INNER JOIN dbo.[CASE] AS c ON r.caseID = c.ID
     INNER JOIN dbo.CUSTOMER_DEVICE AS cd ON r.deviceID = cd.deviceID
     INNER JOIN dbo.CASE_TYPE AS ct ON c.caseType = ct.ID
     INNER JOIN dbo.CASE_CATEGORY AS cc ON c.caseCategory = cc.ID
     INNER JOIN dbo.CASE_SPECIFICATION AS cs ON c.caseSpecification = cs.ID
     INNER JOIN dbo.EMPLOYEE AS e ON r.employeeID = e.ID
     INNER JOIN dbo.CUSTOMER AS cu ON r.customerID = cu.ID
WHERE (cd.deviceID IN (SELECT cd.deviceID
                       FROM dbo.REPAIRMENT AS r
                            INNER JOIN dbo.[CASE] AS c ON r.caseID = c.ID
                            INNER JOIN dbo.CUSTOMER_DEVICE AS cd ON r.deviceID = cd.deviceID
                       GROUP BY cd.deviceID, c.caseType, c.caseCategory, c.caseDescription
                       HAVING (COUNT(cd.deviceID) > 3)))
```

100 %

Results | Messages

	ID	employee	customer	deviceID	model	colorCode	remark	type	category	specification	caseDescription
1	30	Selin Katar	Sefa Taş	357544046582628	SM-T290	GLD	Saatin anakartı değiştirildi, yeni seri no: R4AK...	Aksesuar	Akıllı Saat	Akıllı Saatin Sensörünün Çalışmaması	Akıllı saatin nabız ölçer sensörü çalışmıyor
2	32	Selin Katar	Sefa Taş	357544046582628	SM-T290	GLD	Saatin anakartı tekrar değiştirildi.	Aksesuar	Akıllı Saat	Akıllı Saatin Sensörünün Çalışmaması	Akıllı saatin nabız ölçer sensörü çalışmıyor
3	34	Selin Katar	Sefa Taş	357544046582628	SM-T290	GLD	Saatin anakartı tekrar değiştirildi.	Aksesuar	Akıllı Saat	Akıllı Saatin Sensörünün Çalışmaması	Akıllı saatin nabız ölçer sensörü çalışmıyor
4	35	Songül Aslan	Sefa Taş	357544046582628	SM-T290	GLD	Saatin anakartı tekrar değiştirildi.	Aksesuar	Akıllı Saat	Akıllı Saatin Sensörünün Çalışmaması	Akıllı saatin nabız ölçer sensörü çalışmıyor

(Figure 1.5 Result and Query of the view)

TRIGGERS

We implemented 2 triggers for our REPAIRMENT and ORDERED_PARTS table. For the REPAIRMENT table we plan to update the **numOfLateRepairment** value by one when the repairmentDuration exceeds 1 hour. For the ORDERED_PARTS table we plan to update order's price when the technician makes the order. By the triggers we keep our database up-to-date when new insertions made. Here is the list of triggers.

1-) trg_employee_numOfLateRepairment_Update

```sql
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER TRIGGER [dbo].[trg_employee_numOfLateRepairment_Update]
ON [dbo].[REPAIRMENT]
AFTER UPDATE, INSERT
AS
BEGIN
declare @startDate datetime
declare @endDate datetime
declare @duration int
declare @employeeID int
declare @isPartWaited bit
SELECT @startDate = repairmentStartDate, @endDate = repairmentEndDate, @duration=repairmentDuration, @employeeID=employeeID, @isPartWaited = isPartWaited FROM inserted

declare @isTechnician bit
declare @isSmart bit
SELECT @isTechnician=isTechnician, @isSmart=isSmartService FROM EMPLOYEE e WHERE e.ID=@employeeID

IF( @endDate is not NULL and @isPartWaited = 0)
    BEGIN
    IF(@duration > 3600 and @isTechnician=1)
        BEGIN
            UPDATE dbo.[TECHNICIAN]
            SET numOfLateRepairment = numOfLateRepairment+1
            WHERE ID=@employeeID
        END

    IF(@duration > 3600 and @isSmart=1)
        BEGIN
            UPDATE dbo.[SMART_SERVICE]
            SET numOfLateRepairment = numOfLateRepairment+1
            WHERE ID=@employeeID
        END


    END
```

(Figure 2.1 Trigger of the REPAIRMENT table)

How It Works?

```
SELECT TOP (1000) [ID]
      ,[isInWarranty]
      ,[RRR]
      ,[numOfLateRepairment]
  FROM [TechnicalServiceDatabase].[dbo].[TECHNICIAN]
```

100 %

Results | Messages

	ID	isInWarranty	RRR	numOfLateRepairment
1	5	0	0.00	5
2	9	0	0.00	1
3	14	0	0.00	1
4	17	1	0.00	1
5	20	1	0.00	1
6	24	1	0.00	1
7	33	0	0.00	0

(Figure 2.1.1 List all the Technicians with numOfLateRepairment)

4	354797045519678	4	4	9	2020-12-25 15:50:00.000	2020-12-25 17:50:00.000	False	False	Arka kap

(Figure 2.1.2 Update the employee's repairmentEndDate with 2-hour delay)

100 %

Results | Messages

	ID	isInWarranty	RRR	numOfLateRepairment
1	5	0	0.00	9
2	9	0	0.00	2
3	14	0	0.00	1
4	17	1	0.00	1
5	20	1	0.00	1
6	24	1	0.00	1
7	33	0	0.00	0

(Figure 2.1.3 See the changed numberOfLateRepairment value)

2-) TrgUpdateTotalCost

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  Trigger [dbo].[TrgUpdateTotalCost]    Script Date: 27/12/2020 14:03:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER TRIGGER [dbo].[TrgUpdateTotalCost] ON [dbo].[ORDERED_PART]
    AFTER INSERT, UPDATE
AS
BEGIN
   declare @orderID int
   declare @partID bigint
   declare @quantity smallint

   SELECT @orderID=orderID, @partID=partID, @quantity=quantity FROM inserted

   declare @partCost smallmoney
   SELECT @partCost=price FROM dbo.[PART] p WHERE p.ID=@partID;

   --print before update
   SELECT * FROM dbo.[ORDER] o where o.orderID=@orderID;

   UPDATE dbo.[ORDER]
   SET totalCost = totalCost + @partCost * @quantity
   WHERE dbo.[ORDER].orderID=@orderID

    --print after update
   SELECT * FROM dbo.[ORDER] o where o.orderID=@orderID;
END
```

(Figure 2.2 Trigger of the ORDERED_PART table)

How It Works?

```
SELECT TOP (1000) [ID]
      ,[partName]
      ,[partModel]
      ,[partColor]
      ,[price]
  FROM [TechnicalServiceDatabase].[dbo].[PART]
```

00 %

Results | Messages

	ID	partName	partModel	partColor	price
1	1	Dokunmatik OLED Ekran	SM-730	NULL	100.00
2	2	Dokunmatik OLED Ekran	SM-M31S	NULL	100.00
3	3	Dokunmatik OLED Ekran	SM-M51	NULL	100.00
4	4	Dokunmatik Sensör	SM-M10	NULL	100.00
5	5	Dokunmatik Sensör	SM-A9100	NULL	100.00
6	6	Dokunmatik Sensör	SM-A700	NULL	100.00

(Figure 2.2.1 List all the parts from PART table (Selected ID=1))

```
Insert into dbo.[ORDER](employeeID) values(5)
```

177 %

Messages

```
(1 row affected)

Completion time: 2020-12-27T01:24:41.3215302+03:00
```

(Figure 2.2.2 Create an order with employeID=5)

```
Insert into dbo.[ORDERED_PART](partID,orderID,quantity) values(1,27,50)
```

177 %

Results | Messages

	orderID	totalCost	employeeID	orderDate	isConfirmed
1	27	0.00	5	2020-12-27 01:34:31.300	0

	orderID	totalCost	employeeID	orderDate	isConfirmed
1	27	5000.00	5	2020-12-27 01:34:31.300	0

(Figure 2.2.3 See the total cost when a new ordered part created with selected partID, created orderID and quantity value)

Stored Procedures

We implemented 22 stored procedures to handle multiple different jobs. We tried to select most essential jobs to handle with stored procedures. Our stored procedures contain select, update and delete operations. Here is the list of the stored procedures.

1-) sp_getAvgRepairmentDurationPerEmployee

Definition: Get the average repairment duration time per employee.

```sql
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getCustomerRepairments]    Script Date: 27/12/2020 01:54:50 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_getCustomerRepairments]
    -- Add the parameters for the stored procedure here
    @firstName nvarchar(50),
    @lastName nvarchar(50),
    @phoneNumber nvarchar(20)
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT r.ID,e.firstName+' '+e.lastName employee, cu.firstName+' '+cu.lastName customer, r.deviceID, cd.model,cd.colorCode, r.remark, ct.type, cc.category, cs.specification, c.caseDescription
    FROM REPAIRMENT r inner join [CASE] c on r.caseID=c.ID
        inner join CUSTOMER_DEVICE cd on r.deviceID=cd.deviceID
        inner join CASE_TYPE ct on c.caseType=ct.ID
        inner join CASE_CATEGORY cc on c.caseCategory=cc.ID
        inner join CASE_SPECIFICATION cs on c.caseSpecification=cs.ID
        inner join EMPLOYEE e on r.employeeID=e.ID
        inner join CUSTOMER cu on r.customerID=cu.ID
    WHERE (cu.firstName=@firstName and cu.lastName=@lastName) or (cu.phoneNumber=@phoneNumber)
END
```

83 %

Results | Messages

	ID	FullName	averageDuration
1	4	İsmail Özhan	NULL
2	5	Cemal Aksoy	71100
3	6	Meral Gül	NULL
4	8	Bahar Koç	NULL
5	9	Burak Öztürk	7200
6	10	Gizem Güneş	NULL
7	13	Muhammet Çetin Şen	NULL
8	14	Yeşim Yildirim	NULL
9	17	Arda Dolu	NULL
10	18	Selim Atar	NULL
11	19	Elif Sever	NULL
12	20	Selin Katar	NULL
13	24	Songül Aslan	NULL

(Figure 3.1 Result and Query of the stored procedure)

2-) sp_getCustomerRepairments

Definition: It lists all the customer's repairments with given customer firstName, lastName and phoneNumber variables.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getCustomerRepairments]    Script Date: 27/12/2020 02:07:00 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_getCustomerRepairments]
    -- Add the parameters for the stored procedure here
    @firstName nvarchar(50),
    @lastName nvarchar(50),
    @phoneNumber nvarchar(20)
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT r.ID,e.firstName+' '+e.lastName employee, cu.firstName+' '+cu.lastName customer, r.deviceID, cd.model,cd.colorCode, r.remark, ct.type, cc.category, cs.specification, c.caseDescription
    FROM REPAIRMENT r inner join [CASE] c on r.caseID=c.ID
        inner join CUSTOMER_DEVICE cd on r.deviceID=cd.deviceID
        inner join CASE_TYPE ct on c.caseType=ct.ID
        inner join CASE_CATEGORY cc on c.caseCategory=cc.ID
        inner join CASE_SPECIFICATION cs on c.caseSpecification=cs.ID
        inner join EMPLOYEE e on r.employeeID=e.ID
        inner join CUSTOMER cu on r.customerID=cu.ID
    WHERE (cu.firstName=@firstName and cu.lastName=@lastName) or (cu.phoneNumber=@phoneNumber)
END

exec sp_getCustomerRepairments 'Kemal', 'Yıldız ', '+908513778542'
```

85 %

Results | Messages

	ID	employee	customer	deviceID	model	colorCode	remark	type	category	specification	caseDescription
1	24	Yeşim Yıldırım	Kemal Yıldız	354528012882628	SM-TABS7	BLK	Batarya ücretli değiştirildi	Güç	Şarj	Bataryanın Dolmaması	Şarj %20'ye düştüğünde cihaz kapanıyor

(Figure 3.2 Result and Query of the stored procedure)

3-) sp_getDeviceRepairments

Definition: It lists all the device's repairments with given deviceID variables.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getDeviceRepairments]    Script Date: 27/12/2020 02:15:50 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_getDeviceRepairments]
    -- Add the parameters for the stored procedure here
    @id bigint
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT r.ID,e.firstName+' '+e.lastName employee, cu.firstName+' '+cu.lastName customer, r.deviceID, cd.model,cd.colorCode, r.remark, ct.type, cc.category, cs.specification, c.caseDescription
    FROM REPAIRMENT r inner join [CASE] c on r.caseID=c.ID
        inner join CUSTOMER_DEVICE cd on r.deviceID=cd.deviceID
        inner join CASE_TYPE ct on c.caseType=ct.ID
        inner join CASE_CATEGORY cc on c.caseCategory=cc.ID
        inner join CASE_SPECIFICATION cs on c.caseSpecification=cs.ID
        inner join EMPLOYEE e on r.employeeID=e.ID
        inner join CUSTOMER cu on r.customerID=cu.ID
    WHERE r.deviceID=@id
END

exec sp_getDeviceRepairments 350003487570893
```

91 %

Results | Messages

	ID	employee	customer	deviceID	model	colorCode	remark	type	category	specification	caseDescription
1	3	Selin Katar	Hüseyin Kaya	350003487570893	SM-M51	BLU	Ekran değişti, yazılım güncellendi	Fiziki Hasar	Ekran	Ekranda Çizgiler	Ekranda beyaz çizgi mevcut

(Figure 3.3 Result and Query of the stored procedure)

4-) sp_getEmployeeAvailability

Definition: It lists the employee's available days and hours with given employeeID, firstName and lastName.

```sql
ALTER PROCEDURE [dbo].[sp_getEmployeeAvailability]
    -- Add the parameters for the stored procedure here
    @id int,
    @fName nvarchar(50),
    @lName nvarchar(50)

AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here

    SELECT e.ID, e.firstName, e.lastName ,'Monday' as dayname, ea.startHour, ea.endHour
    FROM EMPLOYEE e inner join EMPLOYEE_AVAILABILITY ea on e.ID=ea.employeeID
    WHERE ((e.ID=@id) OR (e.firstName=@fName and e.lastName=@lName)) and ea.monday=1
    UNION
        SELECT e.ID, e.firstName, e.lastName ,'Tuesday' as dayname, ea.startHour, ea.endHour
        FROM EMPLOYEE e inner join EMPLOYEE_AVAILABILITY ea on e.ID=ea.employeeID
        WHERE ((e.ID=@id) OR (e.firstName=@fName and e.lastName=@lName)) and ea.tuesday=1
    UNION
        SELECT e.ID, e.firstName, e.lastName ,'Wednesday' as dayname, ea.startHour, ea.endHour
        FROM EMPLOYEE e inner join EMPLOYEE_AVAILABILITY ea on e.ID=ea.employeeID
        WHERE ((e.ID=@id) OR (e.firstName=@fName and e.lastName=@lName)) and ea.wednesday=1
    UNION
        SELECT e.ID, e.firstName, e.lastName ,'Thursday' as dayname, ea.startHour, ea.endHour
        FROM EMPLOYEE e inner join EMPLOYEE_AVAILABILITY ea on e.ID=ea.employeeID
        WHERE ((e.ID=@id) OR (e.firstName=@fName and e.lastName=@lName)) and ea.thursday=1
    UNION
        SELECT e.ID, e.firstName, e.lastName ,'Friday' as dayname, ea.startHour, ea.endHour
        FROM EMPLOYEE e inner join EMPLOYEE_AVAILABILITY ea on e.ID=ea.employeeID
        WHERE ((e.ID=@id) OR (e.firstName=@fName and e.lastName=@lName)) and ea.friday=1
    UNION
        SELECT e.ID, e.firstName, e.lastName ,'Saturday' as dayname, ea.startHour, ea.endHour
        FROM EMPLOYEE e inner join EMPLOYEE_AVAILABILITY ea on e.ID=ea.employeeID
        WHERE ((e.ID=@id) OR (e.firstName=@fName and e.lastName=@lName)) and ea.saturday=1

END

exec sp_getEmployeeAvailability 1, 'Yunus', 'Taş'
```

4 %

Results | Messages

	ID	firstName	lastName	dayname	startHour	endHour
1	1	Yunus	Taş	Friday	13:00:00	17:00:00
2	1	Yunus	Taş	Monday	09:00:00	13:00:00
3	1	Yunus	Taş	Saturday	09:00:00	13:00:00
4	1	Yunus	Taş	Thursday	09:00:00	13:00:00
5	1	Yunus	Taş	Tuesday	13:00:00	17:00:00
6	1	Yunus	Taş	Wednesday	09:00:00	13:00:00

(Figure 3.4 Result and Query of the stored procedure)

5-) sp_getEmployeeRepairments

Definition: It list all repairments made by the employee with given employeeID, firstName and lastName.

```
ALTER PROCEDURE [dbo].[sp_getEmployeeRepairments]
    -- Add the parameters for the stored procedure here
    @id tinyint,
    @firstName nvarchar(50),
    @lastName nvarchar(50)

AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT r.ID,e.firstName+' '+e.lastName employee, cu.firstName+' '+cu.lastName customer, r.deviceID, cd.model,cd.colorCode, r.remark, ct.type, cc.category, cs.specification, c.caseDescription
    FROM REPAIRMENT r inner join [CASE] c on r.caseID=c.ID
        inner join CUSTOMER_DEVICE cd on r.deviceID=cd.deviceID
        inner join CASE_TYPE ct on c.caseType=ct.ID
        inner join CASE_CATEGORY cc on c.caseCategory=cc.ID
        inner join CASE_SPECIFICATION cs on c.caseSpecification=cs.ID
        inner join EMPLOYEE e on r.employeeID=e.ID
        inner join CUSTOMER cu on r.customerID=cu.ID
    WHERE (e.firstName=@firstName and e.lastName=@lastName) or (e.ID=@id)
END

exec sp_getEmployeeRepairments 5, 'Cemal', 'Aksoy'
```

90 %

Results | Messages

	ID	employee	customer	deviceID	model	colorCode	remark	type	category	specification	caseDescription
1	1	Cemal Aksoy	Asya Şenay	357546012882628	SM-730	BLK	Ekran ücretli şekilde değişti	Fiziki Hasar	Ekran	Kırık Ekran	Ekranın alt bölümünde kırık mevcut
2	6	Cemal Aksoy	Fatma Özer	359546012882628	SM-S10+	PNK	Kamera camı ücretli şekilde değişti	Fiziki Hasar	Kamera	Kırık Kamera Camı	Kamera camı kırık, çalışıyor
3	19	Cemal Aksoy	Burcu Aslan	357546011252628	SM-A50	YLW	Mikrofon ücretli değişti	Görüşme Kalitesi/Ses	Arama Sesi	Sesin Karşıya Gitmemesi	Karşı tarafa ses gitmiyor
4	25	Cemal Aksoy	Berna Keskin	357598528852628	SM-BUDS	BLK	Anakart ücretli değiştirildi yeni IMEI no: 35464...	Güç	Açılma/Kapanma	Cihazın Açılmaması	Cihaz hiç açılmıyor

(Figure 3.5 Result and Query of the stored procedure)

6-) sp_getMostChangedPart

Definition: It lists the part that most changed.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getMostChangedParts]    Script Date: 27/12/2020 11:51:02 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_getMostChangedParts]
AS
SELECT p.partName, count(*) as partCount
FROM PART p inner join PARTS_NEED_CHANGE pt on p.ID=pt.partID
GROUP BY partName
HAVING Count(*) = (SELECT max(partName) from (SELECT count(*) as partName
                    FROM PART p inner join PARTS_NEED_CHANGE pnc on p.ID=pnc.partID
                    GROUP BY partName) tableParts)


exec sp_getMostChangedParts
```

90 %

Results | Messages

	partName	partCount
1	Mikrafon	5

(Figure 3.6 Result and Query of the stored procedure)

7-) sp_getMostFrequentRepairedModel

Definition: It lists most frequent repaired model.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getMostFrequentRepairedModel]    Script Date: 27/12/2020 11:53:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_getMostFrequentRepairedModel]
AS
SELECT cd.model, COUNT(*) as numOfTotalRepairments
FROM REPAIRMENT r inner join CUSTOMER_DEVICE cd on r.deviceID=cd.deviceID
GROUP BY model
HAVING Count(*) = (SELECT max(repairedTable.modelCount)
                   FROM (SELECT cd1.model, COUNT(*) as modelCount
                         FROM REPAIRMENT r inner join CUSTOMER_DEVICE cd1 on r.deviceID=cd1.deviceID
                         GROUP BY cd1.model) repairedTable)



exec sp_getMostFrequentRepairedModel
```

90 %

Results | Messages

	model	numOfTotalRepairments
1	SM-T290	4

(Figure 3.7 Result and Query of the stored procedure)

8-) sp_getOrderListForPart

Definition: It lists order list for that part with given partID.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getOrderListForPart]    Script Date: 27/12/2020 11:56:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_getOrderListForPart] @ID bigint
AS
SELECT op.partID, op.quantity, o.* FROM ORDERED_PART op inner join dbo.[ORDER] o on o.orderID=op.orderID WHERE op.partID=@ID

exec sp_getOrderListForPart 3
```

90 %

Results | Messages

	partID	quantity	orderID	totalCost	employeeID	orderDate	isConfirmed
1	3	5	15	0.00	17	2020-12-27 11:57:24.747	0

(Figure 3.8 Result and Query of the stored procedure)

9-) sp_getPaidRepairments

Definition: It lists order list for that part with given partID.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getPaidRepairments]    Script Date: 27/12/2020 12:00:31 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[sp_getPaidRepairments]
AS
SELECT r.ID, r.deviceID, r.caseID, r.customerID, r.employeeID, r.repairmentStartDate, r. repairmentEndDate, r.isInWarranty, r. remark, r.repairmentDuration
FROM REPAIRMENT r inner join TECHNICIAN t on r.employeeID=t.ID WHERE t.isInWarranty=0

exec sp_getPaidRepairments
```

90 %

Results | Messages

	ID	deviceID	caseID	customerID	employeeID	repairmentStartDate	repairmentEndDate	isInWarranty	remark	repairmentDuration
1	1	357546012882628	1	1	5	2020-12-25 14:15:00.000	2020-12-25 19:15:00.000	0	Ekran ücretli şekilde değişti	18000
2	6	359546012882628	6	6	5	2020-11-12 15:45:00.000	2020-11-12 17:45:00.000	0	Kamera camı ücretli şekilde değişti	7200
3	19	357546011252628	19	19	5	2020-11-23 10:30:00.000	2020-11-24 10:30:00.000	0	Mikrofon ücretli değişti	86400
4	25	357598528852628	25	25	5	2020-12-25 11:15:00.000	2020-12-27 11:15:00.000	0	Anakart ücretli değiştirildi yeni IMEI no: 35464...	172800
5	4	354797045519678	4	4	9	2020-12-25 15:50:00.000	2020-12-25 17:50:00.000	0	Arka kapak ücretli şekilde değişti	7200
6	21	357586062882628	21	21	9	2020-12-05 10:30:00.000	NULL	0	Şarj modülü ücretli değiştirildi	NULL
7	5	358546062882628	5	5	14	2020-11-12 14:30:00.000	NULL	0	Kamera ücretli şekilde değişti	NULL
8	7	359272037361617	7	7	14	2020-10-24 15:30:00.000	NULL	0	Güç tuşu ücretli şekilde değişti	NULL
9	24	354528012882628	24	24	14	2020-10-11 10:30:00.000	NULL	0	Batarya ücretli değiştirildi	NULL

(Figure 3.9 Result and Query of the store procedure)

10-) sp_getPartsLessThanInStorage

Definition: It lists the parts which has less than given quantity parameter in storage.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getPartsLessThanInStorage]    Script Date: 27/12/2020 12:25:19 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[sp_getPartsLessThanInStorage]
@lessThan int AS

SELECT s.partID, p.partName, p.partModel,p.partColor,s.quantity,s.boxNumber,p.price
FROM STORAGE s inner join PART p on s.partID=p.ID
WHERE s.quantity < @lessThan

exec sp_getPartsLessThanInStorage 100
```

120 %

Results | Messages

	partID	partName	partModel	partColor	quantity	boxNumber	price
1	8	Sim Okuyucu	SM-A810	NULL	50	3	100.00
2	14	Şarj ve Mikrofon Filmi	SM-A710	NULL	50	4	100.00

(Figure 3.10 Result and Query of the stored procedure)

11-) sp_getRepairmentCost

Definition: It lists repairment cost with given repairmentID.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getRepairmentCost]    Script Date: 27/12/2020 13:02:11 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_getRepairmentCost]
    -- Add the parameters for the stored procedure here
    @id int

AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT p.totalCost
    FROM REPAIRMENT r inner join PAYMENT p on r.ID=p.repairmentID
    WHERE r.ID=@id

END

exec sp_getRepairmentCost 1
```

120 %

Results | Messages

	totalCost
1	100.00

(Figure 3.11 Result and Query of the stored procedure)

12-) sp_getRepairmentInfo

Definition: It lists all related repairment info with given repairmentID.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_getRepairmentInfo]    Script Date: 27/12/2020 13:06:37 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_getRepairmentInfo]
    -- Add the parameters for the stored procedure here
    @id int
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    SELECT r.ID,e.firstName+' '+e.lastName employee, cu.firstName+' '+cu.lastName customer, r.deviceID, cd.model,cd.colorCode, r.remark, ct.type, cc.category, cs.specification, c.caseDescription
    FROM REPAIRMENT r inner join [CASE] c on r.caseID=c.ID
        inner join CUSTOMER_DEVICE cd on r.deviceID=cd.deviceID
        inner join CASE_TYPE ct on c.caseType=ct.ID
        inner join CASE_CATEGORY cc on c.caseCategory=cc.ID
        inner join CASE_SPECIFICATION cs on c.caseSpecification=cs.ID
        inner join EMPLOYEE e on r.employeeID=e.ID
        inner join CUSTOMER cu on r.customerID=cu.ID
    WHERE r.ID=@id
END

exec sp_getRepairmentInfo 1
```

99 %

Results | Messages

	ID	employee	customer	deviceID	model	colorCode	remark	type	category	specification	caseDescription
1	1	Cemal Aksoy	Asya Şenay	357546012882628	SM-730	BLK	Ekran ücretli şekilde değişti	Fiziki Hasar	Ekran	Kırık Ekran	Ekranın alt bölümünde kırık mevcut

(Figure 3.12 Result and Query of the stored procedure)

13-) sp_insertNewCase

Definition: It inserts new case to the CASE table with given caseType, caseCategory, caseSpec and caseDesc variables.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_insertNewCase]    Script Date: 27/12/2020 13:09:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_insertNewCase]
    -- Add the parameters for the stored procedure here
    @caseType tinyint,
    @caseCategory tinyint,
    @caseSpec tinyint,
    @caseDesc nvarchar(100)

AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    INSERT INTO dbo.[CASE](caseType, caseCategory, caseSpecification, caseDescription) VALUES
        (@caseType, @caseCategory, @caseSpec, @caseDesc)


END
```

(Figure 3.13 Query of the stored procedure)

How It Works?

	ID	caseType	caseCategory	caseSpecificati...	caseDescription
▸	1	1	1	2	Ekranın alt bölü...
	2	1	1	3	Ekranın sol üst ...
	3	1	1	5	Ekranda beyaz ...
	4	1	2	6	Arka kapakta kı...
	5	1	3	7	Kamera kırık, ça...
	6	1	3	8	Kamera camı kı...
	7	1	4	10	Güç tuşu kırık, ...
	8	2	5	12	Facebook uygu...
	9	2	5	13	Facebook uygu...
	10	2	6	14	Telefon ağır çalı...
	11	2	7	15	Cihaz kendiliği...
	12	2	5	16	Cihaz kendi ken...
	13	3	8	17	Hoparlörden se...
	14	3	8	18	Hoparlörden m...
	15	3	9	19	Mikrofon çalış...
	16	3	10	20	Ses karşıya boğ...
	17	3	10	21	Karşı tarafın ses...
	18	3	10	22	Konuşma sırası...
	19	3	10	23	Karşı tarafa ses ...
	20	4	11	24	Şarj çok hızlı bit...
	21	4	12	25	Telefon şarj olm...
	22	4	12	26	Cihaz 4 saatte, ...
	23	4	12	27	Hızlı şarj özelliğ...
	24	4	12	28	Şarj %20'ye düş...
	25	4	13	29	Cihaz hiç açılmı...
	26	4	13	30	Cihaz Samsung...
	27	5	14	31	Dokunmatik ek...
	28	5	15	36	Sol kulaklıktan ...
	29	5	15	37	Kablosuz kulakl...
	30	5	16	40	Akıllı saatin nab...
•	NULL	NULL	NULL	NULL	NULL

(Figure 3.13.1 Before the new case insertion)

```sql
exec sp_insertNewCase 1, 1, 2, 'Here a description for adding a new case'

Select * from dbo.[CASE]
```

32 %

Results | Messages

	ID	caseType	caseCategory	caseSpecification	caseDescription
1	1	1	1	2	Ekranın alt bölümünde kırık mevcut
2	2	1	1	3	Ekranın sol üst köşesinde, leke mevcut
3	3	1	1	5	Ekranda beyaz çizgi mevcut
4	4	1	2	6	Arka kapakta kırık mevcut
5	5	1	3	7	Kamera kırık, çalışmıyor
6	6	1	3	8	Kamera camı kırık, çalışıyor
7	7	1	4	10	Güç tuşu kırık, çalışmıyor
8	8	2	5	12	Facebook uygulaması açılmıyor
9	9	2	5	13	Facebook uygulaması güncellenmiyor
10	10	2	6	14	Telefon ağır çalışıyor, donuyor
11	11	2	7	15	Cihaz kendiliğinden kapanıyor
12	12	2	5	16	Cihaz kendi kendine reset atıyor
13	13	3	8	17	Hoparlörden ses çıkmıyor
14	14	3	8	18	Hoparlörden müzik dinlerken cızırtılı ses çıkıyor
15	15	3	9	19	Mikrofon çalışmıyor
16	16	3	10	20	Ses karşıya boğuk, cızırtılı gidiyor
17	17	3	10	21	Karşı tarafın sesi çok az geliyor veya hiç gelmi...
18	18	3	10	22	Konuşma sırasında ses kesikli geliyor
19	19	3	10	23	Karşı tarafa ses gitmiyor
20	20	4	11	24	Şarj çok hızlı bitiyor
21	21	4	12	25	Telefon şarj olmuyor
22	22	4	12	26	Cihaz 4 saatte, çok yavaş şarj oluyor
23	23	4	12	27	Hızlı şarj özelliği çalışmıyor
24	24	4	12	28	Şarj %20'ye düştüğünde cihaz kapanıyor
25	25	4	13	29	Cihaz hiç açılmıyor
26	26	4	13	30	Cihaz Samsung logosunda takılıyor, açılmıyor
27	27	5	14	31	Dokunmatik ekran kalemi algılamıyor
28	28	5	15	36	Sol kulaklıktan çok az ses geliyor
29	29	5	15	37	Kablosuz kulaklık telefona bağlanmıyor
30	30	5	16	40	Akıllı saatin nabız ölçer sensörü çalışmıyor
31	33	1	1	2	Here a description for adding a new case

(Figure 3.13.2 After the new case insertion)

14-) sp_insertNewCustomer

Definition: It inserts new customer to the CUSTOMER table with given firstName, lastName, phoneNumber, streetName, streetNumber, city, country and zipcode variables. At the end customer's address information will send to ADDRESS table.

```sql
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_insertNewCustomer]    Script Date: 27/12/2020 13:37:11 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_insertNewCustomer]
    -- Add the parameters for the stored procedure here
    @firstName nvarchar(50),
    @lastName nvarchar(50),
    @phoneNumber nvarchar(20),
    @streetName nvarchar(50),
    @streetNumber nvarchar(50),
    @city nvarchar(50),
    @county nvarchar(50),
    @zipcode nvarchar(10)
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    INSERT INTO dbo.[CUSTOMER](firstName, lastName, phoneNumber) VALUES
        (@firstName, @lastName, @phoneNumber)

    DECLARE @id int
    SELECT @id=c.ID FROM CUSTOMER c WHERE c.firstName=@firstName and c.lastName=@lastName and c.phoneNumber=@phoneNumber

    INSERT INTO dbo.[ADDRESS](customerID, streetName, streetNumber, city, country, zipcode) VALUES
        (@id, @streetName, @streetNumber, @city, @county, @zipcode)

END
```

(Figure 3.14 Query of the stored procedure)

How It Works?

	ID	firstName	lastName	phoneNumber
	1	Asya	Şenay	+908513449005
	2	Hilal	Şimşek	+908513568659
	3	Hüseyin	Kaya	+908512479358
	4	İsmail	Koç	+908515183821
	5	Ayşe	Aktaş	+908511156543
	6	Fatma	Özer	+908513443305
	7	Turgut	Sarı	+908513559005
	8	Ali	Doğan	+908213449005
	9	Berk	Yalçın	+905513549634
	10	Ahmet	Can	+905513449234
	11	Mehmet	Çakır	+908512351623
	12	Aslı	Aksoy	+908517818225
	13	Ayfer	Tekin	+908512832757
	14	Fuat	Güler	+908511397104
	15	Ali	Özcan	+908518345372
	16	Anıl	Şenay	+908516690684
	17	Çağrı	Bozkurt	+908515733352
	18	Veli	Yılmaz	+908514333616
	19	Burcu	Aslan	+908516470151
	20	Güneş	Çetin	+908511725530
	21	Meral	Kılıç	+908511928378
	22	Kadir	Acun	+908512851992
	23	Furkan	Aydın	+908516270387
	24	Kemal	Yıldız	+908513778542
	25	Berna	Keskin	+908516863830
	26	Betül	Kaplan	+908511483226
	27	Ecem	Gül	+908516945789
	28	Tufan	Özkan	+908511376048
	29	Arda	Özdemir	+908511373942
	30	Sefa	Taş	+908517044027

(Figure 3.14.1 Before the new customer insertion)

```
exec sp_insertNewCustomer 'Ali', 'Çevik', '+905364456622', 'Feneryolu Sokak', 'NO:322', 'Istanbul', 'Türkiye', '34100'

Select * from CUSTOMER
```

120 %

Results | Messages

	ID	firstName	lastName	phoneNumber
1	1	Asya	Şenay	+908513449005
2	2	Hilal	Şimşek	+908513568659
3	3	Hüseyin	Kaya	+908512479358
4	4	İsmail	Koç	+908515183821
5	5	Ayşe	Aktaş	+908511156543
6	6	Fatma	Özer	+908513443305
7	7	Turgut	Sarı	+908513559005
8	8	Ali	Doğan	+908213449005
9	9	Berk	Yalçın	+905513549634
10	10	Ahmet	Can	+905513449234
11	11	Mehmet	Çakır	+908512351623
12	12	Aslı	Aksoy	+908517818225
13	13	Ayfer	Tekin	+908512832757
14	14	Fuat	Güler	+908511397104
15	15	Ali	Özcan	+908518345372
16	16	Anıl	Şenay	+908516690684
17	17	Çağrı	Bozkurt	+908515733352
18	18	Veli	Yılmaz	+908514333616
19	19	Burcu	Aslan	+908516470151
20	20	Güneş	Çetin	+908511725530
21	21	Meral	Kılıç	+908511928378
22	22	Kadir	Acun	+908512851992
23	23	Furkan	Aydın	+908516270387
24	24	Kemal	Yıldız	+908513778542
25	25	Berna	Keskin	+908516863830
26	26	Betül	Kaplan	+908511483226
27	27	Ecem	Gül	+908516945789
28	28	Tufan	Özkan	+908511376048
29	29	Arda	Özdemir	+908511373942
30	30	Sefa	Taş	+908517044027
31	31	Ali	Çevik	+905364456622

(Figure 3.14.2 After the new case insertion)

15-) sp_insertNewEmployee

Definition: It inserts new order with given employeeID. It can also add maximum 10 part with their quantities to that order.

```sql
CREATE PROCEDURE [dbo].[sp_insertNewOrder]
    -- Add the parameters for the stored procedure here
    @employeeID tinyint,
    @partID_1 bigint = NULL,
    @partID_1_quantity smallint = NULL,
    @partID_2 bigint = NULL,
    @partID_2_quantity smallint = NULL,
    @partID_3 bigint = NULL,
    @partID_3_quantity smallint = NULL,
    @partID_4 bigint = NULL,
    @partID_4_quantity smallint = NULL,
    @partID_5 bigint = NULL,
    @partID_5_quantity smallint = NULL,
    @partID_6 bigint = NULL,
    @partID_6_quantity smallint = NULL,
    @partID_7 bigint = NULL,
    @partID_7_quantity smallint = NULL,
    @partID_8 bigint = NULL,
    @partID_8_quantity smallint = NULL,
    @partID_9 bigint = NULL,
    @partID_9_quantity smallint = NULL,
    @partID_10 bigint = NULL,
    @partID_10_quantity smallint = NULL
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    INSERT INTO dbo.[ORDER](employeeID) VALUES
        (@employeeID)

    DECLARE @orderID int
    SELECT @orderID = (SELECT SCOPE_IDENTITY())
    SELECT CONCAT('Order with id ', @orderID) + ' is added.';

    IF(@partID_1 is not NULL and @partID_1_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_1, @partID_1_quantity)
    END

    IF(@partID_2 is not NULL and @partID_2_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_2, @partID_2_quantity)
    END

    IF(@partID_3 is not NULL and @partID_3_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_3, @partID_3_quantity)
    END

    IF(@partID_4 is not NULL and @partID_4_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_4, @partID_4_quantity)
    END

    IF(@partID_5 is not NULL and @partID_5_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_5, @partID_5_quantity)
    END

    IF(@partID_6 is not NULL and @partID_6_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_6, @partID_6_quantity)
    END

    IF(@partID_7 is not NULL and @partID_7_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_7, @partID_7_quantity)
    END

    IF(@partID_8 is not NULL and @partID_8_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_8, @partID_8_quantity)
    END

    IF(@partID_9 is not NULL and @partID_9_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_9, @partID_9_quantity)
    END

    IF(@partID_10 is not NULL and @partID_10_quantity is not NULL)
    BEGIN
        INSERT INTO dbo.[ORDERED_PART](orderID, partID, quantity) VALUES (@orderID, @partID_10, @partID_10_quantity)
    END

END
```

(Figure 3.15 Query of the stored procedure)

How It Works?

```
Select * from dbo.[ORDER]

Select * from dbo.[ORDERED_PART]
```

120 %

Results | Messages

17	17	0.00	20	2020-12-27 14:57:57.217	0
18	18	0.00	20	2020-12-27 14:57:57.217	0
19	19	0.00	20	2020-12-27 14:57:57.217	0
20	20	0.00	24	2020-12-27 14:57:57.217	0
21	21	0.00	24	2020-12-27 14:57:57.217	0
22	22	0.00	24	2020-12-27 14:57:57.217	0
23	23	0.00	14	2020-12-27 14:57:57.217	0
24	24	0.00	14	2020-12-27 14:57:57.217	0
25	25	0.00	17	2020-12-27 14:57:57.217	0
26	27	5000.00	5	2020-12-27 14:57:57.217	0

	orderID	partID	quantity
1	12	1	5
2	27	1	50
3	13	2	3
4	15	3	5
5	4	4	5
6	10	5	3
7	17	7	2
8	11	8	6
9	14	9	2
10	1	10	2
11	18	12	1
12	16	13	6
13	5	15	7
14	6	16	23
15	7	17	1
16	19	18	4
17	20	19	1
18	23	20	1
19	2	21	4
20	21	22	2
21	22	23	3
22	24	27	2
23	8	28	4
24	25	28	3
25	3	31	1
26	9	39	5

(Figure 3.15.1 Before the insert operation)

```sql
exec sp_insertNewOrder 5, 1, 199, 2, 3, 4, 39

Select * from dbo.[ORDER]

Select * from dbo.[ORDERED_PART]
ORDER BY ORDERED_PART.orderID DESC
```

120 %

Results | Messages

	orderID	totalCost	employeeID	orderDate	isConfirmed
12	12	500.00	17	2020-12-27 15:01:20.447	0
13	13	0.00	17	2020-12-27 15:01:20.447	0
14	14	0.00	17	2020-12-27 15:01:20.447	0
15	15	0.00	17	2020-12-27 15:01:20.447	0
16	16	0.00	20	2020-12-27 15:01:20.447	0
17	17	0.00	20	2020-12-27 15:01:20.447	0
18	18	0.00	20	2020-12-27 15:01:20.447	0
19	19	0.00	20	2020-12-27 15:01:20.447	0
20	20	0.00	24	2020-12-27 15:01:20.447	0
21	21	0.00	24	2020-12-27 15:01:20.447	0
22	22	0.00	24	2020-12-27 15:01:20.447	0
23	23	0.00	14	2020-12-27 15:01:20.447	0
24	24	0.00	14	2020-12-27 15:01:20.447	0
25	25	0.00	17	2020-12-27 15:01:20.447	0
26	27	5000.00	5	2020-12-27 15:01:20.447	0
27	28	24100.00	5	2020-12-27 15:01:20.447	0

	orderID	partID	quantity
1	28	1	199
2	28	2	3
3	28	4	39
4	27	1	50
5	25	28	3
6	24	27	2
7	23	20	1
8	22	23	3
9	21	22	2
10	20	19	1
11	19	18	4
12	18	12	1
13	17	7	2
14	16	13	6
15	15	3	5
16	14	9	2

(Figure 3.15.2 After the insert operation (orderID=28))

16-) sp_insertNewEmployee

Definition: It inserts new employee with given, firstName, lastName, phoneNumber, username, password, email, address, dateOfBirth, startDate and type variables. At the end with type variable, the employee's branch and manager will be identified.

```sql
ALTER PROCEDURE [dbo].[sp_insertNewEmployee]
    -- Add the parameters for the stored procedure here
    @firstName nvarchar(50),
    @lastName nvarchar(50),
    @phoneNumber nvarchar(20),
    @username nvarchar(30),
    @password nvarchar(32),
    @email nvarchar(100),
    @address nvarchar(150),
    @dateOfBirth date,
    @startDate date,
    @type nvarchar(32)

AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    DECLARE @isManager bit
    DECLARE @isSmartService bit
    DECLARE @isTechnician bit
    DECLARE @isStorageMan bit
    DECLARE @isTester bit
    DECLARE @isAccountant bit

    SET @isManager = CASE WHEN LOWER(@type) like '%manager%' THEN 1 ELSE 0 END
    SET @isSmartService = CASE WHEN LOWER(@type) Like '%smart%' THEN 1 ELSE 0 END
    SET @isTechnician = CASE WHEN LOWER(@type) like '%tech%' THEN 1 ELSE 0 END
    SET @isStorageMan = CASE WHEN LOWER(@type) like '%storage%' THEN 1 ELSE 0 END
    SET @isTester = CASE WHEN LOWER(@type) like '%test%' THEN 1 ELSE 0 END
    SET @isAccountant = CASE WHEN LOWER(@type) like '%account%' THEN 1 ELSE 0 END

    -- Insert statements for procedure here
    INSERT INTO dbo.[EMPLOYEE](firstName, lastName, phoneNumber, username, password, email, address, dateOfBirth, startDate, isManager, isSmartService, isTechnician, isStorageMan, isTester, isAccountant) VALUES
        (@firstName, @lastName, @phoneNumber, @username, @password, @email, @address, @dateOfBirth, @startDate, @isManager, @isSmartService, @isTechnician, @isStorageMan, @isTester, @isAccountant)

    DECLARE @id int
    SELECT @id=ID FROM EMPLOYEE e WHERE e.firstName=@firstName and e.lastName=@lastName and e.email=@email

    IF (@isManager=1)
    BEGIN
        DECLARE @title nvarchar(30)

        SET @title = CASE WHEN @isSmartService=1 THEN 'Smart Manager' ELSE '' END
        SET @title = CASE WHEN @isTechnician=1 THEN 'Service Manager' ELSE '' END

        INSERT INTO dbo.[MANAGER](ID, title) VALUES
        (@id, @title)
    END

    IF (@isSmartService=1)
    BEGIN
        INSERT INTO dbo.[SMART_SERVICE](ID) VALUES
        (@id)
    END

        IF (@isTechnician=1)
    BEGIN
        INSERT INTO dbo.[TECHNICIAN](ID) VALUES
        (@id)
    END

END
```

(Figure 3.16 Query of the stored procedure)

How It Works?

ID	firstName	lastName	phoneNumber	username	password	email	address	dateOfBirth	startDate	isManager	isSmartService	isTechnician	isStorageMan
1	Yunus	Taş	+908514587265	yunustas	7815696ecbf1c9...	yunustas@gma...	Kumru Sokağı ...	1997-03-16	2019-03-16	False	False	True	False
2	Mahmut	Korkmaz	+908518232248	mahmutkorkmaz	e10adc3949ba5...	mahmutkorkm...	Ziyabey Sokağı ...	1991-07-14	2018-05-15	False	False	False	False
3	Adem	Işık	+908519495100	ademisik	e10adc3949ba5...	ademisik@gma...	Nadir Sokağı N...	1995-09-23	2017-04-21	False	False	False	True
4	İsmail	Özhan	+908518384567	ismailozhan	e10adc3949ba5...	ismailozkan122...	Taşlıçay Sokağı ...	1996-04-12	2018-02-12	False	True	False	False
5	Cemal	Aksoy	+908513719222	cemalaksoy	e10adc3949ba5...	cemalaksoy34...	Ziverbey Sokağı...	1997-06-24	2019-04-21	False	False	True	False
6	Meral	Gül	+908518784106	meralgul	e10adc3949ba5...	meral_gul@gm...	Bahariye Sokağı...	1997-02-12	2015-03-23	False	True	False	False
7	Nurcan	Işık	+908511321777	nurcanisik	e10adc3949ba5...	nurcanisik@gm...	Kent Caddesi N...	1993-08-18	2018-10-10	False	False	False	True
8	Bahar	Koç	+908518431181	baharkoc	e10adc3949ba5...	baharkoc35@g...	Rumeli Sokağı ...	1995-03-12	2015-07-15	False	True	False	False
9	Burak	Öztürk	+908513574740	burak_ozturk	e10adc3949ba5...	burakozturk23...	Haydarpaşa So...	1998-08-12	2018-01-23	False	False	True	False
10	Gizem	Güneş	+908517934116	gizemgunes	e10adc3949ba5...	gizemgunes52...	Veysel Sokağı N...	1998-06-12	2019-06-30	True	True	False	False
11	Recep	Özcan	+908516977295	recepozcan	e10adc3949ba5...	recepozcan1@...	Saimbeyli Soka...	1993-04-26	2014-06-12	False	False	False	False
12	Burak	Aslan	+908516237340	burakaslan	e10adc3949ba5...	burakaslan@g...	Kaya Sokağı No...	1996-02-20	2017-03-11	False	True	False	False
13	Muhammet Çe...	Şen	+908519854386	cetinsen	e10adc3949ba5...	cetin_sen@gm...	Yakup Sokağı N...	1999-07-14	2017-02-12	False	True	False	False
14	Yeşim	Yıldırım	+908517237264	yesimyildirim	e10adc3949ba5...	yesimyildirim21...	Lale Sokağı No:...	1997-03-12	2020-02-12	False	False	True	False
15	Tuğba	Küçüküstün	+905367896541	tugbakucukust...	e10adc3949ba5...	tugbakucukust...	Dolgun Sokağı ...	1988-04-23	2020-05-12	True	True	False	False
16	Muhtar	Bayram	+905367896542	muhtarbayram	e10adc3949ba5...	muhtarbayram...	Muhtar Sokağı ...	1985-01-21	2020-04-22	False	True	False	False
17	Arda	Dolu	+905367896532	ardadolu	e10adc3949ba5...	ardadolu@gma...	Turuncu Sokağı...	1998-02-03	2019-05-02	True	False	True	False
18	Selim	Atar	+905388896541	selimatar	e10adc3949ba5...	selimatar@gma...	Orta Sokağı No:...	1995-12-26	2020-01-30	False	True	False	False
19	Elif	Sever	+905367896333	elifsever	e10adc3949ba5...	elifsever@gmai...	Sakinler Sokağı ...	1999-07-21	2020-08-08	False	True	False	False
20	Selin	Katar	+905367897632	selinkatar	e10adc3949ba5...	selinkatar@gm...	Beyaz Sokağı N...	1995-04-22	2018-11-11	False	False	True	False
21	Ceren	Tok	+905367896547	cerentok	e10adc3949ba5...	cerentok@gma...	Dolu Sokağı No...	1997-08-27	2018-05-18	False	True	False	False
22	Enes	Yılmaz	+908513481289	enesyilmaz	e10adc3949ba5...	enesyilmaz@g...	Kaldırım Sokağı...	1993-08-11	2016-02-11	False	False	False	False
23	Nazlı	Kaya	+908518232248	nazlikaya	e10adc3949ba5...	nazlikaya@gma...	Aslanağzı Soka...	1996-01-23	2018-01-15	False	True	False	False
24	Songül	Aslan	+908518232248	songülkaya	e10adc3949ba5...	songulaslan@g...	Melen Sokağı N...	1991-11-12	2019-11-12	False	False	True	False
25	Kemal	Demir	+908513798434	kemaldemir	e10adc3949ba5...	kemaldemir@g...	Denizhan Soka...	1991-12-30	2017-03-12	False	False	False	True
33	Anıl	Şenay	+555555555	anilsenay	asdasasdasdasa	anil@anil.com	Zonguldak	1998-04-27	2020-12-12	True	True	True	False
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(Figure 3.16.1 Before the new employee insertion)

```
exec sp_insertNewEmployee 'Veli', 'Kılıç', '+905632114566', 'velikilic34', 'passwordhere',
                          'velikilic@gmail.com', 'Yokuş Sokak, Kadıköy',
                          '1990-02-02', '2020-01-12', 'Smart Service'

Select * from EMPLOYEE
```

120 %

Results | Messages

	ID	firstName	lastName	phoneNumber	username	password	email	address	dateOfBirth	startDate	isManager	isSmartService	isTechnician	isStorageMan
1	1	Yunus	Taş	+908514587265	yunustas	7815696ecbf1c96e6894b779456d330e	yunustas@gmail.com	Kumru Sokağı No:356 Kadıköy, Istanbul	1997-03-16	2019-03-16	0	0	1	0
2	2	Mahmut	Korkmaz	+908518232248	mahmutkorkmaz	e10adc3949ba59abbe56e057f20f883e	mahmutkorkmaz@gmail.com	Ziyabey Sokağı No:356 Kadıköy, Istanbul	1991-07-14	2018-05-15	0	0	0	0
3	3	Adem	Işık	+908519495100	ademisik	e10adc3949ba59abbe56e057f20f883e	ademisik@gmail.com	Nadir Sokağı No:234 Kadıköy, Istanbul	1995-09-23	2017-04-21	0	0	0	1
4	4	İsmail	Özhan	+908518384567	ismailozhan	e10adc3949ba59abbe56e057f20f883e	ismailozkan122@gmail.com	Taşlıçay Sokağı No:12 Kadıköy, Istanbul	1996-04-12	2018-02-12	0	1	0	0
5	5	Cemal	Aksoy	+908513719222	cemalaksoy	e10adc3949ba59abbe56e057f20f883e	cemalaksoy34@gmail.com	Ziverbey Sokağı No:45 Kadıköy, Istanbul	1997-06-24	2019-04-21	0	0	1	0
6	6	Meral	Gül	+908518784106	meralgul	e10adc3949ba59abbe56e057f20f883e	meral_gul@gmail.com	Bahariye Sokağı No:127 Kadıköy, Istanbul	1997-02-12	2015-03-23	0	1	0	0
7	7	Nurcan	Işık	+908511321777	nurcanisik	e10adc3949ba59abbe56e057f20f883e	nurcanisik@gmail.com	Kent Caddesi No:125 Kadıköy, Istanbul	1993-08-18	2018-10-10	0	0	0	1
8	8	Bahar	Koç	+908518431181	baharkoc	e10adc3949ba59abbe56e057f20f883e	baharkoc35@gmail.com	Rumeli Sokağı No:78 Kadıköy, Istanbul	1995-03-12	2015-07-15	0	1	0	0
9	9	Burak	Öztürk	+908513574740	burak_ozturk	e10adc3949ba59abbe56e057f20f883e	burakozturk23@gmail.com	Haydarpaşa Sokağı No:356 Kadıköy, Istanbul	1998-08-12	2018-01-23	0	0	1	0
10	10	Gizem	Güneş	+908517934116	gizemgunes	e10adc3949ba59abbe56e057f20f883e	gizemgunes52@gmail.com	Veysel Sokağı No:56 Kadıköy, Istanbul	1998-06-12	2019-06-30	1	1	0	0
11	11	Recep	Özcan	+908516977295	recepozcan	e10adc3949ba59abbe56e057f20f883e	recepozcan1@gmail.com	Saimbeyli Sokağı No:23 Kadıköy, Istanbul	1993-04-26	2014-06-12	0	0	0	0
12	12	Burak	Aslan	+908516237340	burakaslan	e10adc3949ba59abbe56e057f20f883e	burakaslan@gmail.com	Kaya Sokağı No:567 Kadıköy, Istanbul	1996-02-20	2017-03-11	0	1	0	0
13	13	Muhammet Çetin	Şen	+908519854386	cetinsen	e10adc3949ba59abbe56e057f20f883e	cetin_sen@gmail.com	Yakup Sokağı No:128 Kadıköy, Istanbul	1999-07-14	2017-02-12	0	1	0	0
14	14	Yeşim	Yıldırım	+908517237264	yesimyildirim	e10adc3949ba59abbe56e057f20f883e	yesimyildirim21@gmail.com	Lale Sokağı No:145 Kadıköy, Istanbul	1997-03-12	2020-02-12	0	0	1	0
15	15	Tuğba	Küçüküstün	+905367896541	tugbakucukustun	e10adc3949ba59abbe56e057f20f883e	tugbakucukustun@gmail.com	Dolgun Sokağı No:878 Altıntaş, Kadıköy	1988-04-23	2020-05-12	1	1	0	0
16	16	Muhtar	Bayram	+905367896542	muhtarbayram	e10adc3949ba59abbe56e057f20f883e	muhtarbayram@gmail.com	Muhtar Sokağı No:78 Kadıköy, Istanbul	1985-01-21	2020-04-22	0	1	0	0
17	17	Arda	Dolu	+905367896532	ardadolu	e10adc3949ba59abbe56e057f20f883e	ardadolu@gmail.com	Turuncu Sokağı No:878 Kadıköy, Istanbul	1998-02-03	2019-05-02	1	0	1	0
18	18	Selim	Atar	+905388896541	selimatar	e10adc3949ba59abbe56e057f20f883e	selimatar@gmail.com	Orta Sokağı No:88 Kadıköy, Istanbul	1995-12-26	2020-01-30	0	1	0	0
19	19	Elif	Sever	+905367896333	elifsever	e10adc3949ba59abbe56e057f20f883e	elifsever@gmail.com	Sakinler Sokağı No:178 Kadıköy, Istanbul	1999-07-21	2020-08-08	0	1	0	0
20	20	Selin	Katar	+905367897632	selinkatar	e10adc3949ba59abbe56e057f20f883e	selinkatar@gmail.com	Beyaz Sokağı No:578 Kadıköy, Istanbul	1995-04-22	2018-11-11	0	0	1	0
21	21	Ceren	Tok	+905367896547	cerentok	e10adc3949ba59abbe56e057f20f883e	cerentok@gmail.com	Dolu Sokağı No:53 Kadıköy, Istanbul	1997-08-27	2018-05-18	0	1	0	0
22	22	Enes	Yılmaz	+908513481289	enesyilmaz	e10adc3949ba59abbe56e057f20f883e	enesyilmaz@gmail.com	Kaldırım Sokağı No:45 Kadıköy, Istanbul	1993-08-11	2016-02-11	0	0	0	0
23	23	Nazlı	Kaya	+908518232248	nazlikaya	e10adc3949ba59abbe56e057f20f883e	nazlikaya@gmail.com	Aslanağzı Sokağı No:376 Kadıköy, Istanbul	1996-01-23	2018-01-15	0	1	0	0
24	24	Songül	Aslan	+908518232248	songülkaya	e10adc3949ba59abbe56e057f20f883e	songulaslan@gmail.com	Melen Sokağı No:56 Kadıköy, Istanbul	1991-11-12	2019-11-12	0	0	1	0
25	25	Kemal	Demir	+908513798434	kemaldemir	e10adc3949ba59abbe56e057f20f883e	kemaldemir@gmail.com	Denizhan Sokağı No:32 Kadıköy, Istanbul	1991-12-30	2017-03-12	0	0	0	1
26	33	Anıl	Şenay	+555555555	anilsenay	asdasasdasdasa	anil@anil.com	Zonguldak	1998-04-27	2020-12-12	1	1	1	0
27	34	Veli	Kılıç	+905632114566	velikilic34	passwordhere	velikilic@gmail.com	Yokuş Sokak, Kadıköy	1990-02-02	2020-01-12	0	1	0	0

(Figure 3.16.2 After the new employee insertion)

```sql
SELECT TOP (1000) [ID]
      ,[customerHappiness]
      ,[RRR]
      ,[numOfLateRepairment]
  FROM [TechnicalServiceDatabase].[dbo].[SMART_SERVICE]
```

120 %

Results | Messages

	ID	customerHappiness	RRR	numOfLateRepairment
1	4	0	0.00	0
2	6	0	0.00	0
3	8	0	0.00	0
4	10	0	0.00	0
5	12	0	0.00	0
6	13	0	0.00	0
7	15	0	0.00	0
8	16	0	0.00	0
9	18	0	0.00	0
10	19	0	0.00	0
11	33	0	0.00	0
12	34	0	0.00	0

(Figure 3.16.3 After the new employee insertion (Smart Service Insertion))

17-) sp_insertNewRepairment

Definition: It inserts new repairment with deviceID, caseID, customerID, employeeID, isInWarranty, remark, and value. Based on value and employee type it also inserts SMART_SERVICE and TECHNICAL_REPAIRMENT table.

```sql
ALTER PROCEDURE [dbo].[sp_insertNewRepairment]
    -- Add the parameters for the stored procedure here
    @deviceID bigint,
    @caseID int,
    @customerID int,
    @employeeID tinyint,
    @isInWarranty bit,
    @remark nvarchar(100),
    @value tinyint
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    INSERT INTO dbo.[REPAIRMENT](deviceID, caseID, customerID, employeeID, isInWarranty, remark, repairmentStartDate) VALUES
        (@deviceID, @caseID, @customerID, @employeeID, @isInWarranty, @remark, GETDATE())

    DECLARE @repairmentID int
    SELECT @repairmentID=r.ID FROM REPAIRMENT r WHERE r.deviceID=@deviceID and r.caseID=@caseID and r.customerID=@customerID and r.employeeID=@employeeID

    DECLARE @isTech bit
    DECLARE @isSmart bit
    SELECT @isTech=isTechnician, @isSmart=isSmartService FROM EMPLOYEE e WHERE e.ID=@employeeID

    IF(@isTech=1 and @value in (0,1,2,3))
    BEGIN
        INSERT INTO dbo.[TECHINCAL_REPAIRMENT](ID,repairmentDegree) VALUES (@repairmentID, @value)
    END

    IF(@isSmart=1 and @value in(1,2))
    BEGIN
        INSERT INTO dbo.[SMART_SERVICE_REPAIRMENT](ID, solutionType) VALUES (@repairmentID, @value)
    END

END
```

(Figure 3.17 Query of the stored procedure)

How It Works?

ID	deviceID	caseID	customerID	employeeID	repairmentStar...	repairmentEnd...	isPartWaited	isInWarranty	remark	repairmentDur...
1	357546012882628	1	1	5	2020-12-25 14:1...	2020-12-25 19:1...	False	False	Ekran ücretli şe...	18000
2	355802036131159	2	2	17	2020-12-25 16:3...	NULL	False	False	Ekran değişti, y...	NULL
3	350003487570893	3	3	20	2020-12-23 10:2...	NULL	False	False	Ekran değişti, y...	NULL
4	354797045519678	4	4	9	2020-12-25 15:5...	2020-12-25 17:5...	False	False	Arka kapak ücr...	7200
5	358546062882628	5	5	14	2020-11-12 14:3...	NULL	False	False	Kamera ücretli ...	NULL
6	359546012882628	6	6	5	2020-11-12 15:4...	2020-11-12 17:4...	False	False	Kamera camı ü...	7200
7	359272037361617	7	7	14	2020-10-24 15:3...	NULL	False	False	Güç tuşu ücretli...	NULL
8	354167045519678	8	8	4	2020-09-10 10:2...	NULL	False	False	Yazılım güncell...	NULL
9	352545642882628	9	9	10	2020-12-25 15:2...	NULL	False	False	Facebook uygu...	NULL
10	354646012882628	10	10	10	2020-10-21 13:5...	NULL	False	False	Yazılım güncell...	NULL
11	356546012882628	11	11	6	2020-10-21 14:3...	NULL	False	False	Yazılım güncell...	NULL
12	354654602882628	12	12	8	2020-10-21 15:2...	NULL	False	False	Yazılım güncell...	NULL
13	358546012652628	13	13	24	2020-08-24 15:3...	NULL	False	False	Hoparlör değişt...	NULL
14	357545012452628	14	14	20	2020-10-28 14:3...	NULL	False	False	Hoparlör değişt...	NULL
15	357845465482628	15	15	17	2020-10-28 16:3...	NULL	False	False	Mikrofon değiş...	NULL
16	357554611882628	16	16	17	2020-08-14 16:3...	NULL	False	False	Mikrofon değiş...	NULL
17	354646012982628	17	17	20	2020-09-20 17:3...	NULL	False	False	Ahize değişti, y...	NULL
18	359546017858262	18	18	24	2020-11-23 15:2...	NULL	False	False	Şebeke modülü...	NULL
19	357546011252628	19	19	5	2020-11-23 10:3...	2020-11-24 10:3...	False	False	Mikrofon ücretl...	86400
20	357896012886548	20	20	17	2020-11-23 10:2...	NULL	False	False	Batarya ve şarj ...	NULL
21	357586062882628	21	21	9	2020-12-05 10:3...	NULL	False	False	Şarj modülü üc...	NULL
22	357544512882628	22	22	24	2020-08-13 15:2...	NULL	False	False	Şarj modülü ve ...	NULL
23	357578012882628	23	23	20	2020-12-24 11:3...	NULL	False	False	Şarj modülü ve ...	NULL
24	354528012882628	24	24	14	2020-10-11 10:3...	NULL	False	False	Batarya ücretli ...	NULL
25	357598528852628	25	25	5	2020-12-25 11:1...	2020-12-27 11:1...	False	False	Anakart ücretli ...	172800
26	357546444212628	26	26	13	2020-10-23 12:3...	NULL	False	False	Yazılım güncell...	NULL
27	358546012452628	27	27	18	2020-07-24 14:2...	NULL	False	False	Telefonun ekra...	NULL
28	355644525482628	28	28	17	2020-06-12 14:2...	NULL	False	False	Sol kulaklık değ...	NULL
29	354846456882628	29	29	19	2020-06-12 16:3...	NULL	False	False	Telefonun yazılı...	NULL
30	357544046582628	30	30	20	2020-06-12 17:3...	NULL	False	False	Saatin anakartı ...	NULL
32	357544046582628	30	30	20	2013-06-20 13:0...	NULL	False	False	Saatin anakartı ...	NULL
34	357544046582628	30	30	20	2013-06-20 12:0...	NULL	False	False	Saatin anakartı ...	NULL
35	357544046582628	30	30	24	2013-06-30 00:0...	NULL	False	False	Saatin anakartı ...	NULL
37	357845465482628	15	15	17	2020-10-30 00:0...	NULL	False	False	Mikrofon değişti	NULL
38	357845465482628	15	15	17	2020-11-23 00:0...	NULL	False	False	Mikrofon değişti	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(Figure 3.17.1 Before the new repairment insertion)

```
exec sp_insertNewRepairment 350003487570893, 11, 21, 17, 0, 'remark', 1

Select * from REPAIRMENT
```

90 %

Results | Messages

	ID	deviceID	caseID	customerID	employeeID	repairmentStartDate	repairmentEndDate	isPartWaited	isInWarranty	remark	repairmentDuration
1	1	357546012882628	1	1	5	2020-12-25 14:15:00.000	2020-12-25 19:15:00.000	0	0	Ekran ücretli şekilde değişti	18000
2	2	355802036131159	2	2	17	2020-12-25 16:30:00.000	NULL	0	0	Ekran değişti, yazılım güncellendi	NULL
3	3	350003487570893	3	3	20	2020-12-23 10:25:00.000	NULL	0	0	Ekran değişti, yazılım güncellendi	NULL
4	4	354797045519678	4	4	9	2020-12-25 15:50:00.000	2020-12-25 17:50:00.000	0	0	Arka kapak ücretli şekilde değişti	7200
5	5	358546062882628	5	5	14	2020-11-12 14:30:00.000	NULL	0	0	Kamera ücretli şekilde değişti	NULL
6	6	359546012882628	6	6	5	2020-11-12 15:45:00.000	2020-11-12 17:45:00.000	0	0	Kamera camı ücretli şekilde değişti	7200
7	7	359272037361617	7	7	14	2020-10-24 15:30:00.000	NULL	0	0	Güç tuşu ücretli şekilde değişti	NULL
8	8	354167045519678	8	8	4	2020-09-10 10:25:00.000	NULL	0	0	Yazılım güncellendi	NULL
9	9	352545642882628	9	9	10	2020-12-25 15:25:00.000	NULL	0	0	Facebook uygulaması silinip yeniden yüklendi	NULL
10	10	354646012882628	10	10	10	2020-10-21 13:50:00.000	NULL	0	0	Yazılım güncellendi	NULL
11	11	356546012882628	11	11	6	2020-10-21 14:30:00.000	NULL	0	0	Yazılım güncellendi	NULL
12	12	354654602882628	12	12	8	2020-10-21 15:25:00.000	NULL	0	0	Yazılım güncellendi	NULL
13	13	358546012652628	13	13	24	2020-08-24 15:30:00.000	NULL	0	0	Hoparlör değişti, yazılım güncellendi	NULL
14	14	357545012452628	14	14	20	2020-10-28 14:30:00.000	NULL	0	0	Hoparlör değişti, yazılım güncellendi	NULL
15	15	357845465482628	15	15	17	2020-10-28 16:30:00.000	NULL	0	0	Mikrofon değişti, yazılım güncellendi	NULL
16	16	357554611882628	16	16	17	2020-08-14 16:30:00.000	NULL	0	0	Mikrofon değişti, yazılım güncellendi	NULL
17	17	354646012982628	17	17	20	2020-09-20 17:30:00.000	NULL	0	0	Ahize değişti, yazılım güncellendi	NULL
18	18	359546017858262	18	18	24	2020-11-23 15:20:00.000	NULL	0	0	Şebeke modülü değişti, yazılım güncellendi	NULL
19	19	357546011252628	19	19	5	2020-11-23 10:30:00.000	2020-11-24 10:30:00.000	0	0	Mikrofon ücretli değişti	86400
20	20	357896012886548	20	20	17	2020-11-23 10:25:00.000	NULL	0	0	Batarya ve şarj aleti değiştirildi, yazılım günc...	NULL
21	21	357586062882628	21	21	9	2020-12-05 10:30:00.000	NULL	0	0	Şarj modülü ücretli değiştirildi	NULL
22	22	357544512882628	22	22	24	2020-08-13 15:20:00.000	NULL	0	0	Şarj modülü ve data kablosu değiştirildi, ya...	NULL
23	23	357578012882628	23	23	20	2020-12-24 11:30:00.000	NULL	0	0	Şarj modülü ve data kablosu değiştirildi, ya...	NULL
24	24	354528012882628	24	24	14	2020-10-11 10:30:00.000	NULL	0	0	Batarya ücretli değiştirildi	NULL
25	25	357598528852628	25	25	5	2020-12-25 11:15:00.000	2020-12-27 11:15:00.000	0	0	Anakart ücretli değiştirildi yeni IMEI no: 354...	172800
26	26	357546444212628	26	26	13	2020-10-23 12:30:00.000	NULL	0	0	Yazılım güncellendi	NULL
27	27	358546012452628	27	27	18	2020-07-24 14:20:00.000	NULL	0	0	Telefonun ekran koruyucu kaldırıldı sorun çö...	NULL
28	28	355644525482628	28	28	17	2020-06-12 14:25:00.000	NULL	0	0	Sol kulaklık değiştirildi	NULL
29	29	354846456882628	29	29	19	2020-06-12 16:30:00.000	NULL	0	0	Telefonun yazılımı güncellendi	NULL
30	30	357544046582628	30	30	20	2020-06-12 17:30:00.000	NULL	0	0	Saatin anakartı değiştirildi, yeni seri no: R4...	NULL
31	32	357544046582628	30	30	20	2013-06-20 13:00:00.000	NULL	0	0	Saatin anakartı tekrar değiştirildi.	NULL
32	34	357544046582628	30	30	20	2013-06-20 12:00:00.000	NULL	0	0	Saatin anakartı tekrar değiştirildi.	NULL
33	35	357544046582628	30	30	24	2013-06-30 00:00:00.000	NULL	0	0	Saatin anakartı tekrar değiştirildi.	NULL
34	37	357845465482628	15	15	17	2020-10-30 00:00:00.000	NULL	0	0	Mikrofon değişti	NULL
35	38	357845465482628	15	15	17	2020-11-23 00:00:00.000	NULL	0	0	Mikrofon değişti	NULL
36	40	350003487570893	11	21	17	2020-12-27 14:22:02.597	NULL	0	0	remark	NULL

(Figure 3.17.2 After the new case insertion)

```
Select * from TECHINCAL_REPAIRMENT
```

90 %

Results | Messages

	ID	repairmentDegree
1	1	2
2	2	0
3	3	0
4	4	1
5	5	1
6	6	1
7	7	1
8	13	0
9	14	0
10	15	0
11	16	0
12	17	0
13	18	0
14	19	1
15	20	0
16	21	1
17	22	0
18	23	0
19	24	2
20	25	3
21	28	0
22	30	0
23	40	1

(Figure 3.17.3 After the new repairment insertion (Technical Repairment))

18-) sp_updateCustomer

Definition: It updates the customer's information based on customerID variable. If firstName, lastName and phoneNumber values are empty, it will update with old variables.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_updateCustomer]    Script Date: 27/12/2020 14:26:38 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_updateCustomer]
    -- Add the parameters for the stored procedure here
    @id int,
    @firstName nvarchar(50),
    @lastName nvarchar(50),
    @phoneNumber nvarchar(20)
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    DECLARE @fName nvarchar(50)
    DECLARE @lName nvarchar(50)
    DECLARE @pNumber nvarchar(20)

    SELECT @fName=firstName, @lName=lastName, @pNumber=phoneNumber FROM CUSTOMER c WHERE c.ID=@id

    SET @firstName = CASE WHEN @firstName='' THEN @fName ELSE @firstName END
    SET @lastName = CASE WHEN @lastName='' THEN @lName ELSE @lastName END
    SET @phoneNumber = CASE WHEN @phoneNumber='' THEN @pNumber ELSE @phoneNumber END

    -- Insert statements for procedure here
    UPDATE CUSTOMER SET firstName=@firstName, lastName=@lastName, phoneNumber=@phoneNumber WHERE ID=@id

END
```

(Figure 3.18 Query of the stored procedure)

How It Works?

ID	firstName	lastName	phoneNumber
1	Asya	Şenay	+908513449005
2	Hilal	Şimşek	+908513568659
3	Hüseyin	Kaya	+908512479358
4	İsmail	Koç	+908515183821
5	Ayşe	Aktaş	+908511156543
6	Fatma	Özer	+908513443305
7	Turgut	Sarı	+908513559005
8	Ali	Doğan	+908213449005
9	Berk	Yalçın	+905513549634
10	Ahmet	Can	+905513449234
11	Mehmet	Çakır	+908512351623
12	Aslı	Aksoy	+908517818225
13	Ayfer	Tekin	+908512832757
14	Fuat	Güler	+908511397104
15	Ali	Özcan	+908518345372
16	Anıl	Şenay	+908516690684
17	Çağrı	Bozkurt	+908515733352
18	Veli	Yılmaz	+908514333616
19	Burcu	Aslan	+908516470151
20	Güneş	Çetin	+908511725530
21	Meral	Kılıç	+908511928378
22	Kadir	Acun	+908512851992
23	Furkan	Aydın	+908516270387
24	Kemal	Yıldız	+908513778542
25	Berna	Keskin	+908516863830
26	Betül	Kaplan	+908511483226
27	Ecem	Gül	+908516945789
28	Tufan	Özkan	+908511376048
29	Arda	Özdemir	+908511373942
30	Sefa	Taş	+908517044027
31	Ali	Çevik	+905364456622
NULL	NULL	NULL	NULL

(Figure 3.18.1 Before the update operation (Selected ID=1))

ID	firstName	lastName	phoneNumber
1	Yeliz	Şentürk	+90531226484
2	Hilal	Şimşek	+908513568659
3	Hüseyin	Kaya	+908512479358
4	İsmail	Koç	+908515183821
5	Ayşe	Aktaş	+908511156543
6	Fatma	Özer	+908513443305
7	Turgut	Sarı	+908513559005
8	Ali	Doğan	+908213449005
9	Berk	Yalçın	+905513549634
10	Ahmet	Can	+905513449234
11	Mehmet	Çakır	+908512351623
12	Aslı	Aksoy	+908517818225
13	Ayfer	Tekin	+908512832757
14	Fuat	Güler	+908511397104
15	Ali	Özcan	+908518345372
16	Anıl	Şenay	+908516690684
17	Çağrı	Bozkurt	+908515733352
18	Veli	Yılmaz	+908514333616
19	Burcu	Aslan	+908516470151
20	Güneş	Çetin	+908511725530
21	Meral	Kılıç	+908511928378
22	Kadir	Acun	+908512851992
23	Furkan	Aydın	+908516270387
24	Kemal	Yıldız	+908513778542
25	Berna	Keskin	+908516863830
26	Betül	Kaplan	+908511483226
27	Ecem	Gül	+908516945789
28	Tufan	Özkan	+908511376048
29	Arda	Özdemir	+908511373942
30	Sefa	Taş	+908517044027
31	Ali	Çevik	+905364456622
NULL	NULL	NULL	NULL

(Figure 3.18.2 After the update operation)

19-) sp_updatePassword

Definition: It updates the employee's password with given username oldPassword and newPassword variables. It also handles basic password creation procedure.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_updatePassword]    Script Date: 27/12/2020 14:36:10 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_updatePassword]
    -- Add the parameters for the stored procedure here
    @username nvarchar(30),
    @oldPassword nvarchar(32),
    @newPassword nvarchar(32)
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    DECLARE @pass nvarchar(32)
    SELECT @pass=e.password FROM EMPLOYEE e WHERE e.username=@username
    IF(@pass=@oldPassword)
    BEGIN
        IF(@pass=@newPassword)
        BEGIN
            SELECT 'New password can not be the same as old!'
        END
        ELSE
        BEGIN
            UPDATE EMPLOYEE SET password=@newPassword WHERE username=@username
        END
    END
    ELSE
    BEGIN
        SELECT 'Wrong Password!'
    END

END
```

(Figure 3.19 Query of the stored procedure)

How It Works?

ID	firstName	lastName	phoneNumber	username	password	email	address	dateOfBirth	startDate	isManager	isSmartService	isTechnician	isStorageMan	isTeste
1	Yunus	Taş	+908514587265	yunustas	7815696ecbf1c9...	yunustas@gma...	Kumru Sokağı ...	1997-03-16	2019-03-16	False	False	True	False	False
2	Mahmut	Korkmaz	+908518232248	mahmut...	e10adc3949ba5...	mahmutkorkm...	Ziyabey Sokağı ...	1991-07-14	2018-05-15	False	False	False	False	True
3	Adem	Işık	+908519495100	ademisik	e10adc3949ba5...	ademisik@gma...	Nadir Sokağı N...	1995-09-23	2017-04-21	False	False	False	True	False
4	İsmail	Özhan	+908518384567	ismailozh...	e10adc3949ba5...	ismailozkan122...	Taşlıçay Sokağı ...	1996-04-12	2018-02-12	False	True	False	False	False
5	Cemal	Aksoy	+908513719222	cemalaks...	e10adc3949ba5...	cemalaksoy34...	Ziverbey Sokağı...	1997-06-24	2019-04-21	False	False	True	False	False
6	Meral	Gül	+908518784106	meralgul	e10adc3949ba5...	meral_gul@gm...	Bahariye Sokağı...	1997-02-12	2015-03-23	False	True	False	False	False
7	Nurcan	Işık	+908511321777	nurcanisik	e10adc3949ba5...	nurcanisik@gm...	Kent Caddesi N...	1993-08-18	2018-10-10	False	False	False	True	False
8	Bahar	Koç	+908518431181	baharkoc	e10adc3949ba5...	baharkoc35@g...	Rumeli Sokağı ...	1995-03-12	2015-07-15	False	True	False	False	False
9	Burak	Öztürk	+908513574740	burak_ozt...	e10adc3949ba5...	burakozturk23...	Haydarpaşa So...	1998-08-12	2018-01-23	False	False	True	False	False
10	Gizem	Güneş	+908517934116	gizemgu...	e10adc3949ba5...	gizemgunes52...	Veysel Sokağı N...	1998-06-12	2019-06-30	True	True	False	False	False
11	Recep	Özcan	+908516977295	recepozcan	e10adc3949ba5...	recepozcan1@...	Saimbeyli Soka...	1993-04-26	2014-06-12	False	False	False	False	True
12	Burak	Aslan	+908516237340	burakaslan	e10adc3949ba5...	burakaslan@g...	Kaya Sokağı No...	1996-02-20	2017-03-11	False	True	False	False	False
13	Muhammet Ç...	Şen	+908519854386	cetinsen	e10adc3949ba5...	cetin_sen@gm...	Yakup Sokağı N...	1999-07-14	2017-02-12	False	True	False	False	False
14	Yeşim	Yıldırım	+908517237264	yesimyild...	e10adc3949ba5...	yesimyildirim21...	Lale Sokağı No:...	1997-03-12	2020-02-12	False	False	True	False	False
15	Tuğba	Küçüküstün	+905367896541	tugbakuc...	e10adc3949ba5...	tugbakucukust...	Dolgun Sokağı ...	1988-04-23	2020-05-12	True	True	False	False	False
16	Muhtar	Bayram	+905367896542	muhtarb...	e10adc3949ba5...	muhtarbayram...	Muhtar Sokağı ...	1985-01-21	2020-04-22	False	True	False	False	False
17	Arda	Dolu	+905367896532	ardadolu	e10adc3949ba5...	ardadolu@gma...	Turuncu Sokağı...	1998-02-03	2019-05-02	True	False	True	False	False
18	Selim	Atar	+905388896541	selimatar	e10adc3949ba5...	selimatar@gma...	Orta Sokağı No:...	1995-12-26	2020-01-30	False	True	False	False	False
19	Elif	Sever	+905367896333	elifsever	e10adc3949ba5...	elifsever@gmai...	Sakinler Sokağı ...	1999-07-21	2020-08-08	False	True	False	False	False
20	Selin	Katar	+905367897632	selinkatar	e10adc3949ba5...	selinkatar@gm...	Beyaz Sokağı N...	1995-04-22	2018-11-11	False	False	True	False	False
21	Ceren	Tok	+905367896547	cerentok	e10adc3949ba5...	cerentok@gma...	Dolu Sokağı No...	1997-08-27	2018-05-18	False	True	False	False	False
22	Enes	Yılmaz	+908513481289	enesyilmaz	e10adc3949ba5...	enesyilmaz@g...	Kaldırım Sokağı...	1993-08-11	2016-02-11	False	False	False	False	True
23	Nazlı	Kaya	+908518232248	nazlikaya	e10adc3949ba5...	nazlikaya@gma...	Aslanağzı Soka...	1996-01-23	2018-01-15	False	True	False	False	False
24	Songül	Aslan	+908518232248	songülkaya	e10adc3949ba5...	songulaslan@g...	Melen Sokağı N...	1991-11-12	2019-11-12	False	False	True	False	False
25	Kemal	Demir	+908513798434	kemalde...	e10adc3949ba5...	kemaldemir@g...	Denizhan Soka...	1991-12-30	2017-03-12	False	False	False	True	False
33	Anıl	Şenay	+555555555	anilsenay	ıbe56e057f20f883e	anil@anil.com	Zonguldak	1998-04-27	2020-12-12	True	True	True	False	False
34	Veli	Kılıç	+905632114566	velikilic34	e10adc3949ba5...	velikilic@gmail....	Yokuş Sokak, K...	1990-02-02	2020-01-12	False	True	False	False	False
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(Figure 3.19.1 Before the update operation (Selected ID=34))

```
exec sp_updatePassword 'velikilic34', 'e10adc3949ba59abbe56e057f20f883e', '65e8800b5c6800aad896f888b2a62afc'

Select * from EMPLOYEE
```

109 %

Results | Messages

	ID	firstName	lastName	phoneNumber	username	password	email	address	dateOfBirth	startDate	isManager	isSmartService	isTechnician	isStorageMan
1	1	Yunus	Taş	+908514587265	yunustas	7815696ecbf1c96e6894b779456d330e	yunustas@gmail.com	Kumru Sokağı No:356 Kadıköy, Istanbul	1997-03-16	2019-03-16	0	0	1	0
2	2	Mahmut	Korkmaz	+908518232248	mahmutkorkmaz	e10adc3949ba59abbe56e057f20f883e	mahmutkorkmaz@gmail.com	Ziyabey Sokağı No:356 Kadıköy, Istanbul	1991-07-14	2018-05-15	0	0	0	0
3	3	Adem	Işık	+908519495100	ademisik	e10adc3949ba59abbe56e057f20f883e	ademisik@gmail.com	Nadir Sokağı No:234 Kadıköy, Istanbul	1995-09-23	2017-04-21	0	0	0	1
4	4	İsmail	Özhan	+908518384567	ismailozhan	e10adc3949ba59abbe56e057f20f883e	ismailozkan122@gmail.com	Taşlıçay Sokağı No:12 Kadıköy, Istanbul	1996-04-12	2018-02-12	0	1	0	0
5	5	Cemal	Aksoy	+908513719222	cemalaksoy	e10adc3949ba59abbe56e057f20f883e	cemalaksoy34@gmail.com	Ziverbey Sokağı No:45 Kadıköy, Istanbul	1997-06-24	2019-04-21	0	0	1	0
6	6	Meral	Gül	+908518784106	meralgul	e10adc3949ba59abbe56e057f20f883e	meral_gul@gmail.com	Bahariye Sokağı No:127 Kadıköy, Istanbul	1997-02-12	2015-03-23	0	1	0	0
7	7	Nurcan	Işık	+908511321777	nurcanisik	e10adc3949ba59abbe56e057f20f883e	nurcanisik@gmail.com	Kent Caddesi No:125 Kadıköy, Istanbul	1993-08-18	2018-10-10	0	0	0	1
8	8	Bahar	Koç	+908518431181	baharkoc	e10adc3949ba59abbe56e057f20f883e	baharkoc35@gmail.com	Rumeli Sokağı No:78 Kadıköy, Istanbul	1995-03-12	2015-07-15	0	1	0	0
9	9	Burak	Öztürk	+908513574740	burak_ozturk	e10adc3949ba59abbe56e057f20f883e	burakozturk23@gmail.com	Haydarpaşa Sokağı No:356 Kadıköy, Istanbul	1998-08-12	2018-01-23	0	0	1	0
10	10	Gizem	Güneş	+908517934116	gizemgunes	e10adc3949ba59abbe56e057f20f883e	gizemgunes52@gmail.com	Veysel Sokağı No:56 Kadıköy, Istanbul	1998-06-12	2019-06-30	1	1	0	0
11	11	Recep	Özcan	+908516977295	recepozcan	e10adc3949ba59abbe56e057f20f883e	recepozcan1@gmail.com	Saimbeyli Sokağı No:23 Kadıköy, Istanbul	1993-04-26	2014-06-12	0	0	0	0
12	12	Burak	Aslan	+908516237340	burakaslan	e10adc3949ba59abbe56e057f20f883e	burakaslan@gmail.com	Kaya Sokağı No:567 Kadıköy, Istanbul	1996-02-20	2017-03-11	0	1	0	0
13	13	Muhammet Çetin	Şen	+908519854386	cetinsen	e10adc3949ba59abbe56e057f20f883e	cetin_sen@gmail.com	Yakup Sokağı No:128 Kadıköy, Istanbul	1999-07-14	2017-02-12	0	1	0	0
14	14	Yeşim	Yıldırım	+908517237264	yesimyildirim	e10adc3949ba59abbe56e057f20f883e	yesimyildirim21@gmail.com	Lale Sokağı No:145 Kadıköy, Istanbul	1997-03-12	2020-02-12	0	0	1	0
15	15	Tuğba	Küçüküstün	+905367896541	tugbakucukustun	e10adc3949ba59abbe56e057f20f883e	tugbakucukustun@gmail.com	Dolgun Sokağı No:878 Altıntaş, Kadıköy	1988-04-23	2020-05-12	1	1	0	0
16	16	Muhtar	Bayram	+905367896542	muhtarbayram	e10adc3949ba59abbe56e057f20f883e	muhtarbayram@gmail.com	Muhtar Sokağı No:78 Kadıköy, Istanbul	1985-01-21	2020-04-22	0	1	0	0
17	17	Arda	Dolu	+905367896532	ardadolu	e10adc3949ba59abbe56e057f20f883e	ardadolu@gmail.com	Turuncu Sokağı No:878 Kadıköy, Istanbul	1998-02-03	2019-05-02	1	0	1	0
18	18	Selim	Atar	+905388896541	selimatar	e10adc3949ba59abbe56e057f20f883e	selimatar@gmail.com	Orta Sokağı No:88 Kadıköy, Istanbul	1995-12-26	2020-01-30	0	1	0	0
19	19	Elif	Sever	+905367896333	elifsever	e10adc3949ba59abbe56e057f20f883e	elifsever@gmail.com	Sakinler Sokağı No:178 Kadıköy, Istanbul	1999-07-21	2020-08-08	0	1	0	0
20	20	Selin	Katar	+905367897632	selinkatar	e10adc3949ba59abbe56e057f20f883e	selinkatar@gmail.com	Beyaz Sokağı No:578 Kadıköy, Istanbul	1995-04-22	2018-11-11	0	0	1	0
21	21	Ceren	Tok	+905367896547	cerentok	e10adc3949ba59abbe56e057f20f883e	cerentok@gmail.com	Dolu Sokağı No:53 Kadıköy, Istanbul	1997-08-27	2018-05-18	0	1	0	0
22	22	Enes	Yılmaz	+908513481289	enesyilmaz	e10adc3949ba59abbe56e057f20f883e	enesyilmaz@gmail.com	Kaldırım Sokağı No:45 Kadıköy, Istanbul	1993-08-11	2016-02-11	0	0	0	0
23	23	Nazlı	Kaya	+908518232248	nazlikaya	e10adc3949ba59abbe56e057f20f883e	nazlikaya@gmail.com	Aslanağzı Sokağı No:376 Kadıköy, Istanbul	1996-01-23	2018-01-15	0	1	0	0
24	24	Songül	Aslan	+908518232248	songülkaya	e10adc3949ba59abbe56e057f20f883e	songulaslan@gmail.com	Melen Sokağı No:56 Kadıköy, Istanbul	1991-11-12	2019-11-12	0	0	1	0
25	25	Kemal	Demir	+908513798434	kemaldemir	e10adc3949ba59abbe56e057f20f883e	kemaldemir@gmail.com	Denizhan Sokağı No:32 Kadıköy, Istanbul	1991-12-30	2017-03-12	0	0	0	1
26	33	Anıl	Şenay	+555555555	anilsenay	asdasasdasdasa	anil@anil.com	Zonguldak	1998-04-27	2020-12-12	1	1	1	0
27	34	Veli	Kılıç	+905632114566	velikilic34	65e8800b5c6800aad896f888b2a62afc	velikilic@gmail.com	Yokuş Sokak, Kadıköy	1990-02-02	2020-01-12	0	1	0	0

(Figure 3.19.3 After the update operation)

20-) sp_deleteAddress

Definition: It deletes the customer's address based on id variable.

```sql
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_deleteAddress]    Script Date: 27/12/2020 15:15:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_deleteAddress]
    -- Add the parameters for the stored procedure here
    @id int
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    DELETE FROM dbo.[ADDRESS] WHERE ID=@id

END
```

(Figure 3.20 Query of the stored procedure)

How It Works?

120 %

Results | Messages

	ID	customerID	streetName	streetNumber	city	country	zipcode
1	1	1	Akpınar Bulvarı	NO: 356	İstanbul	Türkiye	12345
2	2	2	23 Nisan Sokak	NO: 36	İstanbul	Türkiye	16515
3	3	3	Akdeniz Sokak	NO: 6	İstanbul	Türkiye	15615
4	4	4	Atatürk Caddesi	NO: 56	İstanbul	Türkiye	21212
5	5	5	Bayar Caddesi	NO: 5	İstanbul	Türkiye	12122
6	6	6	Begonya Sokak	NO: 36	İstanbul	Türkiye	12445
7	7	7	23 Nisan Sokak	NO: 3	İstanbul	Türkiye	54481
8	8	8	Bayar Sokak	NO: 16	İstanbul	Türkiye	23233
9	9	9	Akpınar Bulvarı	NO: 26	İstanbul	Türkiye	33424
10	10	10	Hilmipaşa Sokak	NO: 556	İstanbul	Türkiye	55352
11	11	11	Güneşli Sokak	NO: 76	İstanbul	Türkiye	23423
12	12	12	Gülay Sokak	NO: 46	İstanbul	Türkiye	12111
13	13	13	İnce Çıkmazı Sokak	NO: 56	İstanbul	Türkiye	53532
14	14	14	Kadıpaşa Sokak	NO: 6	İstanbul	Türkiye	52521
15	15	15	İnönü Caddesi	NO: 56	İstanbul	Türkiye	34223
16	16	16	Kavisli Sokak	NO: 3	İstanbul	Türkiye	35225
17	17	17	Kürkçü Sokak	NO: 351	İstanbul	Türkiye	66622
18	18	18	İnce Çıkmazı Sokak	NO: 316	İstanbul	Türkiye	78678
19	19	19	Ozan Sokak	NO: 156	İstanbul	Türkiye	24244
20	20	20	Saray Sokak	NO: 256	İstanbul	Türkiye	12345
21	21	21	Sarıkanarya Sokak	NO: 56	İstanbul	Türkiye	52352
22	22	22	Telci Sokak	NO: 36	İstanbul	Türkiye	76867
23	23	23	Sümer Sokak	NO: 76	İstanbul	Türkiye	99676
24	24	24	Şahin Sokak	NO: 86	İstanbul	Türkiye	56574
25	25	25	Şelale Sokak	NO: 6	İstanbul	Türkiye	48456
26	26	26	Şakacı Sokak	NO: 88	İstanbul	Türkiye	24252
27	27	27	Mali Sokak	NO: 77	İstanbul	Türkiye	43425
28	28	28	Geçiçi Sokak	NO: 99	İstanbul	Türkiye	62621
29	29	29	Şenay Sokak	NO: 11	İstanbul	Türkiye	31212
30	30	30	Yüksek Sokak	NO: 22	İstanbul	Türkiye	13231
31	31	31	Feneryolu Sokak	NO:322	Istanbul	Türkiye	34100
32	32	20	Hilmipaşa Sokak	NO: 556	İstanbul	Türkiye	34151

(Figure 3.20.1 Before the delete operation (Selected ID=31))

120 %

Results | Messages

	ID	customerID	streetName	streetNumber	city	country	zipcode
1	1	1	Akpınar Bulvarı	NO: 356	İstanbul	Türkiye	12345
2	2	2	23 Nisan Sokak	NO: 36	İstanbul	Türkiye	16515
3	3	3	Akdeniz Sokak	NO: 6	İstanbul	Türkiye	15615
4	4	4	Atatürk Caddesi	NO: 56	İstanbul	Türkiye	21212
5	5	5	Bayar Caddesi	NO: 5	İstanbul	Türkiye	12122
6	6	6	Begonya Sokak	NO: 36	İstanbul	Türkiye	12445
7	7	7	23 Nisan Sokak	NO: 3	İstanbul	Türkiye	54481
8	8	8	Bayar Sokak	NO: 16	İstanbul	Türkiye	23233
9	9	9	Akpınar Bulvarı	NO: 26	İstanbul	Türkiye	33424
10	10	10	Hilmipaşa Sokak	NO: 556	İstanbul	Türkiye	55352
11	11	11	Güneşli Sokak	NO: 76	İstanbul	Türkiye	23423
12	12	12	Gülay Sokak	NO: 46	İstanbul	Türkiye	12111
13	13	13	İnce Çıkmazı Sokak	NO: 56	İstanbul	Türkiye	53532
14	14	14	Kadıpaşa Sokak	NO: 6	İstanbul	Türkiye	52521
15	15	15	İnönü Caddesi	NO: 56	İstanbul	Türkiye	34223
16	16	16	Kavisli Sokak	NO: 3	İstanbul	Türkiye	35225
17	17	17	Kürkçü Sokak	NO: 351	İstanbul	Türkiye	66622
18	18	18	İnce Çıkmazı Sokak	NO: 316	İstanbul	Türkiye	78678
19	19	19	Ozan Sokak	NO: 156	İstanbul	Türkiye	24244
20	20	20	Saray Sokak	NO: 256	İstanbul	Türkiye	12345
21	21	21	Sarıkanarya Sokak	NO: 56	İstanbul	Türkiye	52352
22	22	22	Telci Sokak	NO: 36	İstanbul	Türkiye	76867
23	23	23	Sümer Sokak	NO: 76	İstanbul	Türkiye	99676
24	24	24	Şahin Sokak	NO: 86	İstanbul	Türkiye	56574
25	25	25	Şelale Sokak	NO: 6	İstanbul	Türkiye	48456
26	26	26	Şakacı Sokak	NO: 88	İstanbul	Türkiye	24252
27	27	27	Mali Sokak	NO: 77	İstanbul	Türkiye	43425
28	28	28	Geçiçi Sokak	NO: 99	İstanbul	Türkiye	62621
29	29	29	Şenay Sokak	NO: 11	İstanbul	Türkiye	31212
30	30	30	Yüksek Sokak	NO: 22	İstanbul	Türkiye	13231
31	32	20	Hilmipaşa Sokak	NO: 556	İstanbul	Türkiye	34151

(Figure 3.20.2 After the delete operation)

21-) sp_deleteCustomer

Definition: It deletes the customer's record based on customerID, firstName and lastName variable.

```
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_deleteCustomer]    Script Date: 27/12/2020 15:24:41 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_deleteCustomer]
    -- Add the parameters for the stored procedure here
    @id int,
    @firstName nvarchar(50),
    @lastName nvarchar(50)

AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    DELETE FROM dbo.[ADDRESS] WHERE customerID=@id
    DELETE FROM dbo.[CUSTOMER] WHERE ID=@id and firstName=@firstName and lastName=@lastName

END
```

(Figure 3.21 Query of the stored procedure)

How It Works?

120 %

Results | Messages

	ID	firstName	lastName	phoneNumber
1	1	Yeliz	Şentürk	+90531226484
2	2	Hilal	Şimşek	+908513568659
3	3	Hüseyin	Kaya	+908512479358
4	4	İsmail	Koç	+908515183821
5	5	Ayşe	Aktaş	+908511156543
6	6	Fatma	Özer	+908513443305
7	7	Turgut	Sarı	+908513559005
8	8	Ali	Doğan	+908213449005
9	9	Berk	Yalçın	+905513549634
10	10	Ahmet	Can	+905513449234
11	11	Mehmet	Çakır	+908512351623
12	12	Aslı	Aksoy	+908517818225
13	13	Ayfer	Tekin	+908512832757
14	14	Fuat	Güler	+908511397104
15	15	Ali	Özcan	+908518345372
16	16	Anıl	Şenay	+908516690684
17	17	Çağrı	Bozkurt	+908515733352
18	18	Veli	Yılmaz	+908514333616
19	19	Burcu	Aslan	+908516470151
20	20	Güneş	Çetin	+908511725530
21	21	Meral	Kılıç	+908511928378
22	22	Kadir	Acun	+908512851992
23	23	Furkan	Aydın	+908516270387
24	24	Kemal	Yıldız	+908513778542
25	25	Berna	Keskin	+908516863830
26	26	Betül	Kaplan	+908511483226
27	27	Ecem	Gül	+908516945789
28	28	Tufan	Özkan	+908511376048
29	29	Arda	Özdemir	+908511373942
30	30	Sefa	Taş	+908517044027
31	31	Ali	Çevik	+905364456622
32	32	Mehmet	Yılmaz	+90821312455

(Figure 3.21.1 Before the delete operation (Selected ID=32))

```
exec sp_deleteCustomer 32, 'Mehmet', 'Yılmaz'
Select * from CUSTOMER
```

120 %

Results | Messages

	ID	firstName	lastName	phoneNumber
1	1	Yeliz	Şentürk	+90531226484
2	2	Hilal	Şimşek	+908513568659
3	3	Hüseyin	Kaya	+908512479358
4	4	İsmail	Koç	+908515183821
5	5	Ayşe	Aktaş	+908511156543
6	6	Fatma	Özer	+908513443305
7	7	Turgut	Sarı	+908513559005
8	8	Ali	Doğan	+908213449005
9	9	Berk	Yalçın	+905513549634
10	10	Ahmet	Can	+905513449234
11	11	Mehmet	Çakır	+908512351623
12	12	Aslı	Aksoy	+908517818225
13	13	Ayfer	Tekin	+908512832757
14	14	Fuat	Güler	+908511397104
15	15	Ali	Özcan	+908518345372
16	16	Anıl	Şenay	+908516690684
17	17	Çağrı	Bozkurt	+908515733352
18	18	Veli	Yılmaz	+908514333616
19	19	Burcu	Aslan	+908516470151
20	20	Güneş	Çetin	+908511725530
21	21	Meral	Kılıç	+908511928378
22	22	Kadir	Acun	+908512851992
23	23	Furkan	Aydın	+908516270387
24	24	Kemal	Yıldız	+908513778542
25	25	Berna	Keskin	+908516863830
26	26	Betül	Kaplan	+908511483226
27	27	Ecem	Gül	+908516945789
28	28	Tufan	Özkan	+908511376048
29	29	Arda	Özdemir	+908511373942
30	30	Sefa	Taş	+908517044027
31	31	Ali	Çevik	+905364456622

(Figure 3.21.2 After the delete operation)

22-) sp_deleteWrongOrder

Definition: It deletes the all-related order information with given orderID.

```sql
USE [TechnicalServiceDatabase]
GO
/****** Object:  StoredProcedure [dbo].[sp_deleteWrongOrder]    Script Date: 27/12/2020 15:52:45 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- =============================================
ALTER PROCEDURE [dbo].[sp_deleteWrongOrder]
    -- Add the parameters for the stored procedure here
    @orderID int
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    DELETE FROM dbo.[ORDERED_PART] WHERE orderID=@orderID
    DELETE FROM dbo.[ORDER] WHERE orderID=@orderID


END
```

(Figure 3.22 Query of the stored procedure)

How It Works?

```
Select * from dbo.[ORDER]
ORDER BY orderID desc

Select * from dbo.[ORDERED_PART]
ORDER BY orderID desc
```

120 %

Results | Messages

	orderID	totalCost	employeeID	orderDate	isConfirmed
1	28	24100.00	5	2020-12-27 15:49:45.163	0
2	27	5000.00	5	2020-12-27 15:49:45.163	0
3	25	0.00	17	2020-12-27 15:49:45.163	0
4	24	0.00	14	2020-12-27 15:49:45.163	0
5	23	0.00	14	2020-12-27 15:49:45.163	0
6	22	0.00	24	2020-12-27 15:49:45.163	0
7	21	0.00	24	2020-12-27 15:49:45.163	0
8	20	0.00	24	2020-12-27 15:49:45.163	0
9	19	0.00	20	2020-12-27 15:49:45.163	0
10	18	0.00	20	2020-12-27 15:49:45.163	0
11	17	0.00	20	2020-12-27 15:49:45.163	0
12	16	0.00	20	2020-12-27 15:49:45.163	0
13	15	0.00	17	2020-12-27 15:49:45.163	0
14	14	0.00	17	2020-12-27 15:49:45.163	0
15	13	0.00	17	2020-12-27 15:49:45.163	0
16	12	500.00	17	2020-12-27 15:49:45.163	0
17	11	0.00	14	2020-12-27 15:49:45.163	0
18	10	0.00	14	2020-12-27 15:49:45.163	0
19	9	0.00	14	2020-12-27 15:49:45.163	0

	orderID	partID	quantity
1	28	1	199
2	28	2	3
3	28	4	39
4	27	1	50
5	25	28	3
6	24	27	2
7	23	20	1
8	22	23	3
9	21	22	2
10	20	19	1
11	19	18	4
12	18	12	1
13	17	7	2
14	16	13	6
15	15	3	5
16	14	9	2
17	13	2	3

(Figure 3.22.1 Before the update operation (Selected ID=28))

```sql
exec sp_deleteWrongOrder 28

Select * from dbo.[ORDER]
ORDER BY orderID desc

Select * from dbo.[ORDERED_PART]
ORDER BY orderID desc
```

120 %

Results | Messages

	orderID	totalCost	employeeID	orderDate	isConfirmed
1	27	5000.00	5	2020-12-27 15:56:03.490	0
2	25	0.00	17	2020-12-27 15:56:03.490	0
3	24	0.00	14	2020-12-27 15:56:03.490	0
4	23	0.00	14	2020-12-27 15:56:03.490	0
5	22	0.00	24	2020-12-27 15:56:03.490	0
6	21	0.00	24	2020-12-27 15:56:03.490	0
7	20	0.00	24	2020-12-27 15:56:03.490	0
8	19	0.00	20	2020-12-27 15:56:03.490	0
9	18	0.00	20	2020-12-27 15:56:03.490	0
10	17	0.00	20	2020-12-27 15:56:03.490	0
11	16	0.00	20	2020-12-27 15:56:03.490	0
12	15	0.00	17	2020-12-27 15:56:03.490	0
13	14	0.00	17	2020-12-27 15:56:03.490	0
14	13	0.00	17	2020-12-27 15:56:03.490	0
15	12	500.00	17	2020-12-27 15:56:03.490	0
16	11	0.00	14	2020-12-27 15:56:03.490	0
17	10	0.00	14	2020-12-27 15:56:03.490	0
18	9	0.00	14	2020-12-27 15:56:03.490	0
19	8	0.00	14	2020-12-27 15:56:03.490	0

	orderID	partID	quantity
1	27	1	50
2	25	28	3
3	24	27	2
4	23	20	1
5	22	23	3
6	21	22	2
7	20	19	1
8	19	18	4
9	18	12	1
10	17	7	2
11	16	13	6
12	15	3	5
13	14	9	2

(Figure 3.22.2 After the delete operation)