**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**Ders : Veri Tabanı Yönetim Sistemleri**

**Dönem : 2021-2022 Güz Dönemi**

**Öğrenci Numarası : B181210076**

**Öğrenci Adı : Muhammet Kemal**

**Öğrenci Soyadı : Güvenç**

**Öğrenci E-Postası :** [**kemal.guvenc@ogr.sakarya.edu.tr**](mailto:kemal.guvenc@ogr.sakarya.edu.tr)

**Tanıtım**

Bu program bir şirketin şubelere verdiği bir yazılımdır. Şubeler bu yazılım aracılığıyla tedarikçilerden mal alımını, müşterilere ürün satımını, işçilerin durumunu görüntüleyebileceği ve bir çok özellik sunar.

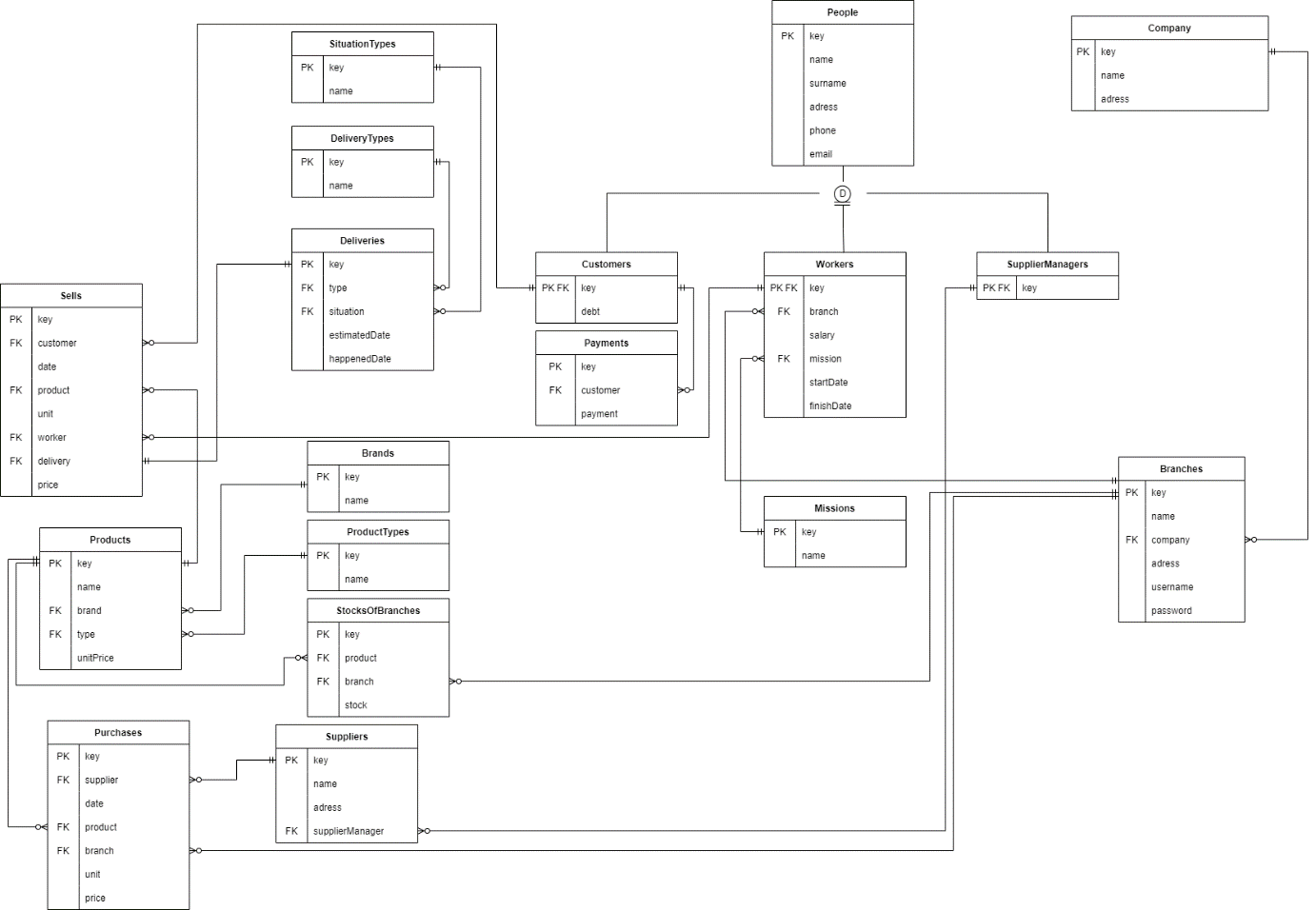
**İş Kuralları**

1. Şirketin adı, adresi ve şubeleri bulunur.
2. Şubelerin de adı ve adresi bulunur. Ayrıca sisteme girmeleri için şirketin onlara verdiği birer adet kullanıcı adı ve şifreleri vardır.
3. Şubelerin çalışanları vardır.
4. Sistemde 3 farklı kişi tipi kayıtlıdır: Çalışan, Müşteri ve Tedarikçi Temsilcisi
5. Kişi sadece bunlardan biri olabilir.
6. Her kişinin adı, soyadı, adresi, telefon numarası ve e-posta adresi vardır.
7. Her müşterinin bir de borç miktarı kayıtlıdır.
8. Çalışanların da ayrıca görevi, maaşı ve işe başlama ve çıkma tarihleri vardır.
9. Çalışanların 5 farklı görevi olabilir: Yönetim, Satış, Kasa, Temizlik, Teslimat
10. Her çalışan sadece bir şubeye bağlıdır.
11. Her ürünün adı, markası, stok adedi, birim fiyatı ve türü bulunur.
12. Her ürünün stok adedi şubeye göre değişebilir.
13. Ürünler tedarikçilerden alınır.
14. Her tedarikçinin adı, adresi ve temsilcisi vardır.
15. Ürünler müşterilere satılılr.
16. Müşteriler ürünleri 2 farklı şekilde alabilirler: Kargo ya da elden
17. Şubeler sadece kendileriyle alakalı olan çalışanları, teslimatları, siparişleri, alımları görürüler.
18. Müşteriler ve tedarikçiler hepsinde ortaktır.
19. Müşteriler isterlerse ürün iadesi yapabilirler.
20. Gerektiğinde tedarikçilerden yapılan alımlar iptal edilebilir.

**İlişkisel Şema**

1. Company(**key: int**, name: string, adress: string)
2. Branches(**key:int**, name: string, company: int, adress: string, username: string, password: string)
3. People(**key:int**, name: string, surname: string, adress: string, phone: string, email: string)
4. Customer(**key: int**, debt: int)
5. Workers(**key: int**, branch: int, salary: int, mission: int, startDate: date, finishDate: date)
6. SupplierManagers(**key: int**)
7. Payments(**key: int**, customer: int, payment: int)
8. Missions(**key: int**, name: string)
9. Brands(**key: int**, name: string)
10. ProductTypes(**key: int**, name: string)
11. Products(**key: int**, name: string, brand: int, type: int, unitPrice: int)
12. StocksOfBranch(**key: int**, product: int, branch: int, stock: int)
13. Suppliers(**key: int**, name: string, adress: string, supplierManager: int)
14. Purchases(**key: int**, supplier: int, date: date, product: int, branch: int, unit: int, price: int)
15. SituationTypes(**key: int**, name: string)
16. DelveryTypes(**key: int**, name: string)
17. Deliveries(**key: int**, type: int, situation: int, estimatedDate: date, happenedDate: date)
18. Sells(**key: int**, customer: int, date: date, product: int, unit: int, worker: int, delivery: int, price: int)

**Varlık Bağıntı Modeli**



**SQL İFADELERİ**

--

-- PostgreSQL database dump

--

-- Dumped from database version 14.1

-- Dumped by pg\_dump version 14.0

SET statement\_timeout = 0;

SET lock\_timeout = 0;

SET idle\_in\_transaction\_session\_timeout = 0;

SET client\_encoding = 'UTF8';

SET standard\_conforming\_strings = on;

SELECT pg\_catalog.set\_config('search\_path', '', false);

SET check\_function\_bodies = false;

SET xmloption = content;

SET client\_min\_messages = warning;

SET row\_security = off;

--

-- Name: CheckBranch(character varying, character varying); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."CheckBranch"("inputUsername" character varying, "inputPassword" character varying) RETURNS integer

LANGUAGE plpgsql

AS $$

DECLARE

output "Branches"%ROWTYPE;

BEGIN

FOR output IN SELECT \* from "Branches" LOOP

IF (md5("inputUsername")=output."username") aND (md5("inputPassword")=output."password") THEN

return output."key";

END IF;

END LOOP;

return -1;

END;

$$;

ALTER FUNCTION public."CheckBranch"("inputUsername" character varying, "inputPassword" character varying) OWNER TO postgres;

--

-- Name: Payment(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."Payment"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE "Customers" SET "debt" = "debt" - NEW."payment"

WHERE "key" = NEW."customer";

RETURN NEW;

END;

$$;

ALTER FUNCTION public."Payment"() OWNER TO postgres;

--

-- Name: Purchase(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."Purchase"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE "StocksOfBranches" SET "stock" = "stock" + NEW."unit"

WHERE "product" = NEW."product" AND "branch" = NEW."branch";

RETURN NEW;

END;

$$;

ALTER FUNCTION public."Purchase"() OWNER TO postgres;

--

-- Name: RemoveCustomer(integer); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemoveCustomer"("customerKey" integer) RETURNS boolean

LANGUAGE plpgsql

AS $$

DECLARE

"row" "Sells"%ROWTYPE;

BEGIN

FOR "row" IN SELECT \* from "Sells" LOOP

IF "row"."customer"="customerKey" THEN

Return FALSE;

END IF;

END LOOP;

DELETE FROM "Customers" WHERE "key"="customerKey";

return TRUE;

END;

$$;

ALTER FUNCTION public."RemoveCustomer"("customerKey" integer) OWNER TO postgres;

--

-- Name: RemoveProduct(integer); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemoveProduct"("productKey" integer) RETURNS boolean

LANGUAGE plpgsql

AS $$

DECLARE

"row" "Purchases"%ROWTYPE;

BEGIN

FOR "row" IN SELECT \* from "Purchases" LOOP

IF "row"."product"="productKey" THEN

Return FALSE;

END IF;

END LOOP;

DELETE FROM "StocksOfBranches"

WHERE "product" = "productKey";

DELETE FROM "Products" WHERE "key"="productKey";

return TRUE;

END;

$$;

ALTER FUNCTION public."RemoveProduct"("productKey" integer) OWNER TO postgres;

--

-- Name: RemovePurchase(integer); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemovePurchase"("purchaseKey" integer) RETURNS boolean

LANGUAGE plpgsql

AS $$

DECLARE

"branchStock" INTEGER;

"purchaseUnit" INTEGER;

"branchKey" INTEGER;

"productKey" INTEGER;

BEGIN

"branchKey" = (Select "branch" FROM "Purchases" WHERE "Purchases"."key" = "purchaseKey");

"productKey" = (Select "product" FROM "Purchases" WHERE "Purchases"."key" = "purchaseKey");

"branchStock" = (SELECT "stock" FROM "StocksOfBranches" WHERE "product" = "productKey" AND "branch" = "branchKey");

"purchaseUnit" = (Select "unit" FROM "Purchases" WHERE "Purchases"."key" = "purchaseKey");

IF ("branchStock" - "purchaseUnit")<0 THEN

Return FALSE;

END IF;

DELETE FROM "Purchases" WHERE "key"="purchaseKey";

return TRUE;

END;

$$;

ALTER FUNCTION public."RemovePurchase"("purchaseKey" integer) OWNER TO postgres;

--

-- Name: RemovePurchaseT(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemovePurchaseT"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE "StocksOfBranches" SET "stock" = "stock" - OLD."unit"

WHERE "product" = OLD."product" AND "branch" = OLD."branch";

RETURN OLD;

END;

$$;

ALTER FUNCTION public."RemovePurchaseT"() OWNER TO postgres;

--

-- Name: RemoveSell(integer); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemoveSell"("sellKey" integer) RETURNS boolean

LANGUAGE plpgsql

AS $$

DECLARE

"row" INT;

BEGIN

"row" := (SELECT "Deliveries"."situation" from "Sells" INNER JOIN "Deliveries" On "Sells"."delivery" = "Deliveries"."key" WHERE "Sells"."key" = "sellKey");

IF "row"='2' THEN

Return FALSE;

END IF;

DELETE FROM "Sells" WHERE "key"="sellKey";

return TRUE;

END;

$$;

ALTER FUNCTION public."RemoveSell"("sellKey" integer) OWNER TO postgres;

--

-- Name: RemoveSellT(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemoveSellT"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE "Customers" SET "debt" = "debt" - OLD."price"

WHERE "key" = OLD."customer";

UPDATE "StocksOfBranches" SET "stock" = "stock" + OLD."unit"

WHERE "product" = OLD."product" AND "branch" = (SELECT "branch" FROM "Workers" WHERE "key" = OLD."worker");

DELETE FROM "Deliveries"

WHERE "key" = OLD."delivery";

return OLD;

END;

$$;

ALTER FUNCTION public."RemoveSellT"() OWNER TO postgres;

--

-- Name: RemoveSupplier(integer); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemoveSupplier"("supplierKey" integer) RETURNS boolean

LANGUAGE plpgsql

AS $$

DECLARE

"row" "Purchases"%ROWTYPE;

BEGIN

FOR "row" IN SELECT \* from "Purchases" LOOP

IF "row"."supplier"="supplierKey" THEN

Return FALSE;

END IF;

END LOOP;

DELETE FROM "Suppliers" WHERE "key"="supplierKey";

return TRUE;

END;

$$;

ALTER FUNCTION public."RemoveSupplier"("supplierKey" integer) OWNER TO postgres;

--

-- Name: RemoveWorker(integer); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."RemoveWorker"("workerKey" integer) RETURNS boolean

LANGUAGE plpgsql

AS $$

DECLARE

"date" DATE;

BEGIN

date:=(select "finishDate" from "Workers" Where "key"="workerKey");

IF date IS NOT NULL THEN

Return FALSE;

END IF;

UPDATE "Workers" SET "finishDate" = CURRENT\_DATE WHERE "key"="workerKey";

return TRUE;

END;

$$;

ALTER FUNCTION public."RemoveWorker"("workerKey" integer) OWNER TO postgres;

--

-- Name: Sell(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."Sell"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE "Customers" SET "debt" = "debt" + NEW."price"

WHERE "key" = NEW."customer";

UPDATE "StocksOfBranches" SET "stock" = "stock" - NEW."unit"

WHERE "product" = NEW."product" AND "branch" = (SELECT "branch" FROM "Workers" WHERE "key" = NEW."worker");

RETURN NEW;

END;

$$;

ALTER FUNCTION public."Sell"() OWNER TO postgres;

--

-- Name: SetHappenedDate(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."SetHappenedDate"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

IF NEW."type" = 2 THEN

NEW."happenedDate" = CURRENT\_DATE;

NEW."estimatedDate" = CURRENT\_DATE;

END IF;

RETURN NEW;

END;

$$;

ALTER FUNCTION public."SetHappenedDate"() OWNER TO postgres;

--

-- Name: SetHappenedDate2(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."SetHappenedDate2"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

IF NEW."situation" != 2 THEN

NEW."happenedDate" = NULL;

END IF;

RETURN NEW;

END;

$$;

ALTER FUNCTION public."SetHappenedDate2"() OWNER TO postgres;

--

-- Name: SetStocksOfBranches(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."SetStocksOfBranches"() RETURNS trigger

LANGUAGE plpgsql

AS $$

DECLARE

"row" "Branches"%ROWTYPE;

BEGIN

FOR "row" IN SELECT \* FROM "Branches" LOOP

INSERT INTO "StocksOfBranches"

("product", "branch", "stock")

VALUES (NEW."key", "row"."key", '0');

END LOOP;

RETURN NEW;

END;

$$;

ALTER FUNCTION public."SetStocksOfBranches"() OWNER TO postgres;

--

-- Name: UpperSurname(); Type: FUNCTION; Schema: public; Owner: postgres

--

CREATE FUNCTION public."UpperSurname"() RETURNS trigger

LANGUAGE plpgsql

AS $$

BEGIN

NEW."surname" = UPPER(NEW."surname");

RETURN NEW;

END;

$$;

ALTER FUNCTION public."UpperSurname"() OWNER TO postgres;

SET default\_tablespace = '';

SET default\_table\_access\_method = heap;

--

-- Name: Branches; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Branches" (

key integer NOT NULL,

name character varying NOT NULL,

company integer NOT NULL,

adress character varying NOT NULL,

password character varying NOT NULL,

username character varying NOT NULL

);

ALTER TABLE public."Branches" OWNER TO postgres;

--

-- Name: Branches\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Branches\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Branches\_key\_seq" OWNER TO postgres;

--

-- Name: Branches\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Branches\_key\_seq" OWNED BY public."Branches".key;

--

-- Name: Brands; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Brands" (

key integer NOT NULL,

name character varying NOT NULL

);

ALTER TABLE public."Brands" OWNER TO postgres;

--

-- Name: Brands\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Brands\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Brands\_key\_seq" OWNER TO postgres;

--

-- Name: Brands\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Brands\_key\_seq" OWNED BY public."Brands".key;

--

-- Name: Company; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Company" (

key integer NOT NULL,

name character varying NOT NULL,

adress character varying NOT NULL

);

ALTER TABLE public."Company" OWNER TO postgres;

--

-- Name: Company\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Company\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Company\_key\_seq" OWNER TO postgres;

--

-- Name: Company\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Company\_key\_seq" OWNED BY public."Company".key;

--

-- Name: Customers; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Customers" (

key integer NOT NULL,

debt real NOT NULL,

CONSTRAINT "debtBigThanZero" CHECK ((debt >= (0)::double precision))

);

ALTER TABLE public."Customers" OWNER TO postgres;

--

-- Name: Deliveries; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Deliveries" (

key integer NOT NULL,

situation integer NOT NULL,

type integer NOT NULL,

"estimatedDate" date NOT NULL,

"happenedDate" date

);

ALTER TABLE public."Deliveries" OWNER TO postgres;

--

-- Name: Deliveries\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Deliveries\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Deliveries\_key\_seq" OWNER TO postgres;

--

-- Name: Deliveries\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Deliveries\_key\_seq" OWNED BY public."Deliveries".key;

--

-- Name: DeliveryTypes; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."DeliveryTypes" (

key integer NOT NULL,

name character varying NOT NULL

);

ALTER TABLE public."DeliveryTypes" OWNER TO postgres;

--

-- Name: DeliveryTypes\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."DeliveryTypes\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."DeliveryTypes\_key\_seq" OWNER TO postgres;

--

-- Name: DeliveryTypes\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."DeliveryTypes\_key\_seq" OWNED BY public."DeliveryTypes".key;

--

-- Name: List-Brands; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Brands" AS

SELECT "Brands".key,

"Brands".name

FROM public."Brands"

ORDER BY "Brands".name;

ALTER TABLE public."List-Brands" OWNER TO postgres;

--

-- Name: People; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."People" (

key integer NOT NULL,

name character varying NOT NULL,

surname character varying NOT NULL,

adress character varying NOT NULL,

phone character varying NOT NULL,

email character varying NOT NULL

);

ALTER TABLE public."People" OWNER TO postgres;

--

-- Name: List-Customers; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Customers" AS

SELECT "People".key AS "Müşteri Numarası",

"People".name AS "Ad",

"People".surname AS "Soyad",

"People".adress AS "Adres",

"People".phone AS "Telefon Numarası",

"People".email AS "E-Posta Adresi",

"Customers".debt AS "Toplam Borç"

FROM (public."Customers"

JOIN public."People" ON (("Customers".key = "People".key)))

ORDER BY "People".name;

ALTER TABLE public."List-Customers" OWNER TO postgres;

--

-- Name: Products; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Products" (

key integer NOT NULL,

name character varying NOT NULL,

brand integer NOT NULL,

type integer NOT NULL,

"unitPrice" integer NOT NULL,

CONSTRAINT "unitPriceBigThanZero" CHECK (("unitPrice" >= 0))

);

ALTER TABLE public."Products" OWNER TO postgres;

--

-- Name: Sells; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Sells" (

key integer NOT NULL,

customer integer NOT NULL,

date date NOT NULL,

product integer NOT NULL,

unit integer NOT NULL,

worker integer NOT NULL,

delivery integer NOT NULL,

price integer NOT NULL,

CONSTRAINT "unitBigThanZero" CHECK ((unit >= 0)),

CONSTRAINT "unitPriceBigThanZero" CHECK ((price >= 0))

);

ALTER TABLE public."Sells" OWNER TO postgres;

--

-- Name: SituationTypes; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."SituationTypes" (

key integer NOT NULL,

name character varying NOT NULL

);

ALTER TABLE public."SituationTypes" OWNER TO postgres;

--

-- Name: List-Deliveries; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Deliveries" AS

SELECT "Deliveries".key AS "Teslimat Kodu",

"Customers".key AS "Müşteri Kodu",

"People".name AS "Müşteri Adı",

"People".surname AS "Müşteri Soyadı",

"People".adress AS "Müşteri Adresi",

"Products".name AS "Ürün Adı",

"Sells".unit AS "Ürün Adedi",

"Sells".date AS "Sipariş Tarihi",

"Deliveries"."estimatedDate" AS "Tahmini Teslimat Tarihi",

"Deliveries"."happenedDate" AS "Gerçekleşen Teslimat Tarihi",

"DeliveryTypes".name AS "Teslimat Şekli",

"SituationTypes".name AS "Teslimat Durumu"

FROM ((((((public."Deliveries"

JOIN public."DeliveryTypes" ON (("Deliveries".type = "DeliveryTypes".key)))

JOIN public."SituationTypes" ON (("Deliveries".situation = "SituationTypes".key)))

JOIN public."Sells" ON (("Sells".delivery = "Deliveries".key)))

JOIN public."Products" ON (("Sells".product = "Products".key)))

JOIN public."Customers" ON (("Sells".customer = "Customers".key)))

JOIN public."People" ON (("Customers".key = "People".key)))

ORDER BY "Sells".date DESC;

ALTER TABLE public."List-Deliveries" OWNER TO postgres;

--

-- Name: List-DeliveryTypes; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-DeliveryTypes" AS

SELECT "DeliveryTypes".key,

"DeliveryTypes".name

FROM public."DeliveryTypes";

ALTER TABLE public."List-DeliveryTypes" OWNER TO postgres;

--

-- Name: Missions; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Missions" (

key integer NOT NULL,

name text NOT NULL

);

ALTER TABLE public."Missions" OWNER TO postgres;

--

-- Name: List-Mission; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Mission" AS

SELECT "Missions".key,

"Missions".name

FROM public."Missions"

ORDER BY "Missions".name;

ALTER TABLE public."List-Mission" OWNER TO postgres;

--

-- Name: ProductTypes; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."ProductTypes" (

key integer NOT NULL,

name character varying NOT NULL

);

ALTER TABLE public."ProductTypes" OWNER TO postgres;

--

-- Name: List-ProductTypes; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-ProductTypes" AS

SELECT "ProductTypes".key,

"ProductTypes".name

FROM public."ProductTypes"

ORDER BY "ProductTypes".name;

ALTER TABLE public."List-ProductTypes" OWNER TO postgres;

--

-- Name: List-Products; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Products" AS

SELECT "Products".key AS "Ürün Kodu",

"Products".name AS "Ad",

"Brands".name AS "Marka",

"ProductTypes".name AS "Tip"

FROM ((public."Products"

JOIN public."Brands" ON (("Products".brand = "Brands".key)))

JOIN public."ProductTypes" ON (("Products".type = "ProductTypes".key)))

ORDER BY "Products".name;

ALTER TABLE public."List-Products" OWNER TO postgres;

--

-- Name: Purchases; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Purchases" (

key integer NOT NULL,

supplier integer NOT NULL,

product integer NOT NULL,

date date NOT NULL,

branch integer NOT NULL,

unit integer NOT NULL,

price integer NOT NULL,

CONSTRAINT "priceBigThanZero" CHECK ((price >= 0)),

CONSTRAINT "unitBigThanZero" CHECK ((unit >= 0))

);

ALTER TABLE public."Purchases" OWNER TO postgres;

--

-- Name: Suppliers; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Suppliers" (

key integer NOT NULL,

name character varying NOT NULL,

adress character varying NOT NULL,

"supplierManager" integer NOT NULL

);

ALTER TABLE public."Suppliers" OWNER TO postgres;

--

-- Name: List-Purchases; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Purchases" AS

SELECT "Purchases".key AS "Alış Kodu",

"Suppliers".name AS "Tedarikçi",

"Products".name AS "Ürün",

"Purchases".unit AS "Ürün Adedi",

"Purchases".date AS "Alış Tarihi"

FROM ((public."Purchases"

JOIN public."Suppliers" ON (("Purchases".supplier = "Suppliers".key)))

JOIN public."Products" ON (("Purchases".product = "Products".key)))

ORDER BY "Purchases".date DESC;

ALTER TABLE public."List-Purchases" OWNER TO postgres;

--

-- Name: Workers; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Workers" (

key integer NOT NULL,

branch integer NOT NULL,

salary integer NOT NULL,

mission integer NOT NULL,

"startDate" date NOT NULL,

"finishDate" date,

CONSTRAINT "bonusRateBigThanZero" CHECK ((salary >= 0))

);

ALTER TABLE public."Workers" OWNER TO postgres;

--

-- Name: List-Sells; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Sells" AS

SELECT "Sells".key AS "Satış Kodu",

"Sells".date AS "Satış Tarihi",

"People".name AS "Müşteri Adı",

"People".surname AS "Müşteri Soyadı",

"Products".name AS "Satın Alınan Ürün",

"Sells".unit AS "Ürün Adedi",

workers.name AS "Satış Yapan Çalışan Adı",

workers.surname AS "Satış Yapan Çalışan Soyadı"

FROM (((((public."Sells"

JOIN public."Products" ON (("Sells".product = "Products".key)))

JOIN public."Customers" ON (("Sells".customer = "Customers".key)))

JOIN public."People" ON (("Customers".key = "People".key)))

JOIN public."Workers" ON (("Sells".key = "Workers".key)))

JOIN public."People" workers ON (("Workers".key = "People".key)))

ORDER BY "Sells".date DESC;

ALTER TABLE public."List-Sells" OWNER TO postgres;

--

-- Name: List-SituationTypes; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-SituationTypes" AS

SELECT "SituationTypes".key,

"SituationTypes".name

FROM public."SituationTypes";

ALTER TABLE public."List-SituationTypes" OWNER TO postgres;

--

-- Name: SupplierManagers; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."SupplierManagers" (

key integer NOT NULL

);

ALTER TABLE public."SupplierManagers" OWNER TO postgres;

--

-- Name: List-Suppliers; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Suppliers" AS

SELECT "Suppliers".key AS "Tedarikçi Kodu",

"Suppliers".name AS "Ad",

"Suppliers".adress AS "Adres",

"People".name AS "Temsilci Adı",

"People".surname AS "Temsilci Soyadı"

FROM ((public."Suppliers"

JOIN public."SupplierManagers" ON (("Suppliers"."supplierManager" = "SupplierManagers".key)))

JOIN public."People" ON (("People".key = "SupplierManagers".key)))

ORDER BY "Suppliers".name;

ALTER TABLE public."List-Suppliers" OWNER TO postgres;

--

-- Name: List-Workers; Type: VIEW; Schema: public; Owner: postgres

--

CREATE VIEW public."List-Workers" AS

SELECT "People".key AS "Çalışan Kodu",

"People".name AS "Ad",

"People".surname AS "Soyad",

"People".adress AS "Adres",

"People".phone AS "Telefon Numarası",

"People".email AS "E-Posta Adresi",

"Branches".name AS "Şube",

"Workers".salary AS "Maaş",

"Missions".name AS "Görev",

"Workers"."startDate" AS "İşe Başlama Tarihi"

FROM (((public."Workers"

JOIN public."People" ON (("Workers".key = "People".key)))

JOIN public."Branches" ON (("Workers".branch = "Branches".key)))

JOIN public."Missions" ON (("Workers".mission = "Missions".key)))

ORDER BY "People".name;

ALTER TABLE public."List-Workers" OWNER TO postgres;

--

-- Name: Missions\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Missions\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Missions\_key\_seq" OWNER TO postgres;

--

-- Name: Missions\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Missions\_key\_seq" OWNED BY public."Missions".key;

--

-- Name: Payments; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."Payments" (

key integer NOT NULL,

customer integer NOT NULL,

payment integer NOT NULL,

CONSTRAINT "paymentBigThanZero" CHECK ((payment >= 0))

);

ALTER TABLE public."Payments" OWNER TO postgres;

--

-- Name: Payments\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Payments\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Payments\_key\_seq" OWNER TO postgres;

--

-- Name: Payments\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Payments\_key\_seq" OWNED BY public."Payments".key;

--

-- Name: People\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."People\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."People\_key\_seq" OWNER TO postgres;

--

-- Name: People\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."People\_key\_seq" OWNED BY public."People".key;

--

-- Name: ProductTypes\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."ProductTypes\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."ProductTypes\_key\_seq" OWNER TO postgres;

--

-- Name: ProductTypes\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."ProductTypes\_key\_seq" OWNED BY public."ProductTypes".key;

--

-- Name: Products\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Products\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Products\_key\_seq" OWNER TO postgres;

--

-- Name: Products\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Products\_key\_seq" OWNED BY public."Products".key;

--

-- Name: Puchases\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Puchases\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Puchases\_key\_seq" OWNER TO postgres;

--

-- Name: Puchases\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Puchases\_key\_seq" OWNED BY public."Purchases".key;

--

-- Name: Sells\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Sells\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Sells\_key\_seq" OWNER TO postgres;

--

-- Name: Sells\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Sells\_key\_seq" OWNED BY public."Sells".key;

--

-- Name: SituationTypes\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."SituationTypes\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."SituationTypes\_key\_seq" OWNER TO postgres;

--

-- Name: SituationTypes\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."SituationTypes\_key\_seq" OWNED BY public."SituationTypes".key;

--

-- Name: StocksOfBranches; Type: TABLE; Schema: public; Owner: postgres

--

CREATE TABLE public."StocksOfBranches" (

key integer NOT NULL,

product integer NOT NULL,

branch integer NOT NULL,

stock integer NOT NULL,

CONSTRAINT "stockBigThanZero" CHECK ((stock >= 0))

);

ALTER TABLE public."StocksOfBranches" OWNER TO postgres;

--

-- Name: StocksOfBranches\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."StocksOfBranches\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."StocksOfBranches\_key\_seq" OWNER TO postgres;

--

-- Name: StocksOfBranches\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."StocksOfBranches\_key\_seq" OWNED BY public."StocksOfBranches".key;

--

-- Name: Suppliers\_key\_seq; Type: SEQUENCE; Schema: public; Owner: postgres

--

CREATE SEQUENCE public."Suppliers\_key\_seq"

AS integer

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE public."Suppliers\_key\_seq" OWNER TO postgres;

--

-- Name: Suppliers\_key\_seq; Type: SEQUENCE OWNED BY; Schema: public; Owner: postgres

--

ALTER SEQUENCE public."Suppliers\_key\_seq" OWNED BY public."Suppliers".key;

--

-- Name: Branches key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Branches" ALTER COLUMN key SET DEFAULT nextval('public."Branches\_key\_seq"'::regclass);

--

-- Name: Brands key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Brands" ALTER COLUMN key SET DEFAULT nextval('public."Brands\_key\_seq"'::regclass);

--

-- Name: Company key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Company" ALTER COLUMN key SET DEFAULT nextval('public."Company\_key\_seq"'::regclass);

--

-- Name: Deliveries key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Deliveries" ALTER COLUMN key SET DEFAULT nextval('public."Deliveries\_key\_seq"'::regclass);

--

-- Name: DeliveryTypes key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."DeliveryTypes" ALTER COLUMN key SET DEFAULT nextval('public."DeliveryTypes\_key\_seq"'::regclass);

--

-- Name: Missions key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Missions" ALTER COLUMN key SET DEFAULT nextval('public."Missions\_key\_seq"'::regclass);

--

-- Name: Payments key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Payments" ALTER COLUMN key SET DEFAULT nextval('public."Payments\_key\_seq"'::regclass);

--

-- Name: People key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."People" ALTER COLUMN key SET DEFAULT nextval('public."People\_key\_seq"'::regclass);

--

-- Name: ProductTypes key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."ProductTypes" ALTER COLUMN key SET DEFAULT nextval('public."ProductTypes\_key\_seq"'::regclass);

--

-- Name: Products key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Products" ALTER COLUMN key SET DEFAULT nextval('public."Products\_key\_seq"'::regclass);

--

-- Name: Purchases key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Purchases" ALTER COLUMN key SET DEFAULT nextval('public."Puchases\_key\_seq"'::regclass);

--

-- Name: Sells key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Sells" ALTER COLUMN key SET DEFAULT nextval('public."Sells\_key\_seq"'::regclass);

--

-- Name: SituationTypes key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."SituationTypes" ALTER COLUMN key SET DEFAULT nextval('public."SituationTypes\_key\_seq"'::regclass);

--

-- Name: StocksOfBranches key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."StocksOfBranches" ALTER COLUMN key SET DEFAULT nextval('public."StocksOfBranches\_key\_seq"'::regclass);

--

-- Name: Suppliers key; Type: DEFAULT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Suppliers" ALTER COLUMN key SET DEFAULT nextval('public."Suppliers\_key\_seq"'::regclass);

--

-- Data for Name: Branches; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Branches" VALUES

(9, 'Sakarya', 1, 'Serdivan', '7b331923dac7e89b6bedd93922b3f2e0', '106b1657e045ce3520ae2c3dbbf2eea0'),

(10, 'İstanbul', 1, 'Fatih', 'e821a8bfc2c786f275e5d5ea94d519a7', 'b4b643cb1547b52057fdd15336581ca8');

--

-- Data for Name: Brands; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Brands" VALUES

(1, 'LG'),

(2, 'Philips'),

(3, 'Sony'),

(4, 'Apple'),

(5, 'Samsung'),

(6, 'Bosch'),

(7, 'Vestel'),

(8, 'Arzum'),

(9, 'Lenovo');

--

-- Data for Name: Company; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Company" VALUES

(1, 'Beyaz Ev', 'İstanbul');

--

-- Data for Name: Customers; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Customers" VALUES

(29, 0),

(28, 9500),

(7, 14500);

--

-- Data for Name: Deliveries; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Deliveries" VALUES

(17, 1, 1, '2021-12-27', NULL),

(15, 2, 2, '2021-12-27', '2021-12-20');

--

-- Data for Name: DeliveryTypes; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."DeliveryTypes" VALUES

(1, 'Kargo'),

(2, 'Elden');

--

-- Data for Name: Missions; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Missions" VALUES

(1, 'Kasa'),

(2, 'Satış'),

(3, 'Temizlik'),

(4, 'Yönetim'),

(5, 'Teslimat');

--

-- Data for Name: Payments; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Payments" VALUES

(4, 28, 500),

(5, 7, 500);

--

-- Data for Name: People; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."People" VALUES

(18, 'Ömer', 'DEMIR', 'Sakarya', '05346948576', 'omer@mail.com'),

(27, 'Sevim', 'ÇELIK', 'Rize', '05389594837', 'sevim@mail.com'),

(33, 'Kemal', 'GÜVENÇ', 'Antalya', '05529485768', 'kemal@mail.com'),

(35, 'Kaya', 'ÇETIN', 'Ankara', '05068585794', 'kaya@mail.com'),

(7, 'Ahmet', 'YAŞAR', 'Istanbul', '05328475694', 'ahmet@mail.com'),

(38, 'Fırat', 'KABUK', 'İstanbul', '05439685740', 'fırat@mail.com'),

(29, 'Rabia', 'KABUK', 'Fatih', '05329604935', 'rabia@mail.com'),

(39, 'Deniz', 'KABAK', 'İstanbul', '05539685768', 'deniz@mail.com'),

(40, 'Ahmet', 'CEYLAN', 'Sakarya', '05437869587', 'ahmet@mail.com'),

(41, 'Mehmet', 'ASD', 'Sakarya', '05436789678', 'mehmet@mail.com'),

(42, 'Macit', 'YıLMAZ', 'Sakarya', '05436758675', 'macit@mail.com'),

(43, 'Mehmet', 'CENGIZ', 'İstanbul', '05436958475', 'mehmet@mail.com'),

(44, 'asd', 'QWE', '213', '213', 'sad'),

(45, 'asd', 'ASD', 'asd', '23', 'asd'),

(46, 'asd', 'ASD', 'asd', '123', 'asd'),

(47, 'asd', 'ASD', 'sad', 'asd', ' '),

(48, 'asd', 'ASD', 'sad', 'asd', ''),

(49, 'asd', 'ASD', 'fds', 'dsf', 'sd'),

(28, 'Fatih', 'CELIKCI', 'Ankara', '05556948574', 'fatih@mail.com'),

(50, 'asd', 'SDF', 'dsf', 'dsf', 'sdf'),

(51, 'sad', 'SDA', 'dsf', 'dsf', 'dsf');

--

-- Data for Name: ProductTypes; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."ProductTypes" VALUES

(1, 'Elektronik'),

(2, 'Oyuncak'),

(3, 'Beyaz Eşye'),

(4, 'Kırtasiye'),

(5, 'Kitap'),

(6, 'Bilgisayar');

--

-- Data for Name: Products; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Products" VALUES

(10, 'Telefon', 4, 1, 10000),

(12, 'Bilgisayar', 4, 1, 15000),

(13, 'Bilgisayar Faresi', 5, 6, 150);

--

-- Data for Name: Purchases; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Purchases" VALUES

(8, 7, 10, '2021-12-20', 10, 3, 15000),

(9, 3, 12, '2021-12-20', 10, 2, 20000),

(10, 5, 13, '2021-12-20', 10, 2, 100);

--

-- Data for Name: Sells; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Sells" VALUES

(14, 28, '2021-12-20', 10, 1, 43, 15, 10000),

(16, 7, '2021-12-20', 12, 1, 39, 17, 15000);

--

-- Data for Name: SituationTypes; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."SituationTypes" VALUES

(2, 'Teslim Edildi'),

(1, 'Hazırlanıyor'),

(3, 'Gönderildi'),

(4, 'Teslim Edilemedi');

--

-- Data for Name: StocksOfBranches; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."StocksOfBranches" VALUES

(14, 13, 10, 2),

(12, 12, 10, 1),

(7, 10, 9, 0),

(8, 10, 10, 2),

(11, 12, 9, 0),

(13, 13, 9, 0);

--

-- Data for Name: SupplierManagers; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."SupplierManagers" VALUES

(18),

(33),

(40),

(45),

(50);

--

-- Data for Name: Suppliers; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Suppliers" VALUES

(3, 'Çetinler', 'İstanbul', 18),

(5, 'Koç Holding', 'İstanbul', 33),

(7, 'Cengizler', 'Sakarya', 40);

--

-- Data for Name: Workers; Type: TABLE DATA; Schema: public; Owner: postgres

--

INSERT INTO public."Workers" VALUES

(27, 9, 4600, 1, '2021-12-06', NULL),

(35, 9, 5000, 1, '2021-12-20', NULL),

(39, 10, 4000, 2, '2021-12-14', NULL),

(42, 10, 3000, 1, '2021-12-20', '2021-12-20'),

(43, 10, 4000, 5, '2021-12-20', NULL),

(46, 10, 123, 3, '2021-12-20', '2021-12-20'),

(51, 10, 2000, 3, '2021-12-20', '2021-12-20');

--

-- Name: Branches\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Branches\_key\_seq"', 10, true);

--

-- Name: Brands\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Brands\_key\_seq"', 10, true);

--

-- Name: Company\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Company\_key\_seq"', 3, true);

--

-- Name: Deliveries\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Deliveries\_key\_seq"', 20, true);

--

-- Name: DeliveryTypes\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."DeliveryTypes\_key\_seq"', 2, true);

--

-- Name: Missions\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Missions\_key\_seq"', 5, true);

--

-- Name: Payments\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Payments\_key\_seq"', 5, true);

--

-- Name: People\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."People\_key\_seq"', 51, true);

--

-- Name: ProductTypes\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."ProductTypes\_key\_seq"', 7, true);

--

-- Name: Products\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Products\_key\_seq"', 15, true);

--

-- Name: Puchases\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Puchases\_key\_seq"', 12, true);

--

-- Name: Sells\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Sells\_key\_seq"', 19, true);

--

-- Name: SituationTypes\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."SituationTypes\_key\_seq"', 4, true);

--

-- Name: StocksOfBranches\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."StocksOfBranches\_key\_seq"', 18, true);

--

-- Name: Suppliers\_key\_seq; Type: SEQUENCE SET; Schema: public; Owner: postgres

--

SELECT pg\_catalog.setval('public."Suppliers\_key\_seq"', 9, true);

--

-- Name: Purchases unique\_Alis\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Purchases"

ADD CONSTRAINT "unique\_Alis\_no" PRIMARY KEY (key);

--

-- Name: Missions unique\_CalisanGorev\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Missions"

ADD CONSTRAINT "unique\_CalisanGorev\_no" PRIMARY KEY (key);

--

-- Name: Workers unique\_Calisan\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Workers"

ADD CONSTRAINT "unique\_Calisan\_no" PRIMARY KEY (key);

--

-- Name: SituationTypes unique\_DurumTur\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."SituationTypes"

ADD CONSTRAINT "unique\_DurumTur\_no" PRIMARY KEY (key);

--

-- Name: SupplierManagers unique\_IletisimGorevlisi\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."SupplierManagers"

ADD CONSTRAINT "unique\_IletisimGorevlisi\_no" PRIMARY KEY (key);

--

-- Name: People unique\_Kisi\_kod; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."People"

ADD CONSTRAINT "unique\_Kisi\_kod" PRIMARY KEY (key);

--

-- Name: Brands unique\_Marka\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Brands"

ADD CONSTRAINT "unique\_Marka\_no" PRIMARY KEY (key);

--

-- Name: Customers unique\_Musteri\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Customers"

ADD CONSTRAINT "unique\_Musteri\_no" PRIMARY KEY (key);

--

-- Name: Payments unique\_Odemeler\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Payments"

ADD CONSTRAINT "unique\_Odemeler\_no" PRIMARY KEY (key);

--

-- Name: Sells unique\_Satis\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Sells"

ADD CONSTRAINT "unique\_Satis\_no" PRIMARY KEY (key);

--

-- Name: Company unique\_Sirket\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Company"

ADD CONSTRAINT "unique\_Sirket\_no" PRIMARY KEY (key);

--

-- Name: StocksOfBranches unique\_StocksOfBranches\_key; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."StocksOfBranches"

ADD CONSTRAINT "unique\_StocksOfBranches\_key" PRIMARY KEY (key);

--

-- Name: Suppliers unique\_Tedarikci\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Suppliers"

ADD CONSTRAINT "unique\_Tedarikci\_no" PRIMARY KEY (key);

--

-- Name: DeliveryTypes unique\_TeslimatTur\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."DeliveryTypes"

ADD CONSTRAINT "unique\_TeslimatTur\_no" PRIMARY KEY (key);

--

-- Name: Deliveries unique\_Teslimat\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Deliveries"

ADD CONSTRAINT "unique\_Teslimat\_no" PRIMARY KEY (key);

--

-- Name: ProductTypes unique\_UrunTur\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."ProductTypes"

ADD CONSTRAINT "unique\_UrunTur\_no" PRIMARY KEY (key);

--

-- Name: Branches unique\_table1\_kod; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Branches"

ADD CONSTRAINT unique\_table1\_kod PRIMARY KEY (key);

--

-- Name: Products unique\_table1\_no; Type: CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Products"

ADD CONSTRAINT unique\_table1\_no PRIMARY KEY (key);

--

-- Name: Deliveries TriggerHappenedDate; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerHappenedDate" BEFORE INSERT ON public."Deliveries" FOR EACH ROW EXECUTE FUNCTION public."SetHappenedDate"();

--

-- Name: Deliveries TriggerHappenedDate2; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerHappenedDate2" BEFORE UPDATE ON public."Deliveries" FOR EACH ROW EXECUTE FUNCTION public."SetHappenedDate2"();

--

-- Name: Payments TriggerPayment; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerPayment" BEFORE INSERT ON public."Payments" FOR EACH ROW EXECUTE FUNCTION public."Payment"();

--

-- Name: Purchases TriggerPurchase; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerPurchase" BEFORE INSERT ON public."Purchases" FOR EACH ROW EXECUTE FUNCTION public."Purchase"();

--

-- Name: Purchases TriggerRemovePurchases; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerRemovePurchases" BEFORE DELETE ON public."Purchases" FOR EACH ROW EXECUTE FUNCTION public."RemovePurchaseT"();

--

-- Name: Sells TriggerRemoveSell; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerRemoveSell" AFTER DELETE ON public."Sells" FOR EACH ROW EXECUTE FUNCTION public."RemoveSellT"();

--

-- Name: Sells TriggerSell; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerSell" BEFORE INSERT ON public."Sells" FOR EACH ROW EXECUTE FUNCTION public."Sell"();

--

-- Name: Products TriggerSetStocksOfBranches; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerSetStocksOfBranches" AFTER INSERT ON public."Products" FOR EACH ROW EXECUTE FUNCTION public."SetStocksOfBranches"();

--

-- Name: People TriggerUpperSurname; Type: TRIGGER; Schema: public; Owner: postgres

--

CREATE TRIGGER "TriggerUpperSurname" BEFORE INSERT OR UPDATE ON public."People" FOR EACH ROW EXECUTE FUNCTION public."UpperSurname"();

--

-- Name: Branches Branches-Company; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Branches"

ADD CONSTRAINT "Branches-Company" FOREIGN KEY (company) REFERENCES public."Company"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Customers Customers-Inheritance; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Customers"

ADD CONSTRAINT "Customers-Inheritance" FOREIGN KEY (key) REFERENCES public."People"(key) MATCH FULL ON UPDATE CASCADE ON DELETE CASCADE;

--

-- Name: Deliveries Deliveries-DeliveryTypes; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Deliveries"

ADD CONSTRAINT "Deliveries-DeliveryTypes" FOREIGN KEY (type) REFERENCES public."DeliveryTypes"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Deliveries Deliveries-SituationTypes; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Deliveries"

ADD CONSTRAINT "Deliveries-SituationTypes" FOREIGN KEY (situation) REFERENCES public."SituationTypes"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Payments Payments-Customers; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Payments"

ADD CONSTRAINT "Payments-Customers" FOREIGN KEY (customer) REFERENCES public."Customers"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Products Products-Brands; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Products"

ADD CONSTRAINT "Products-Brands" FOREIGN KEY (brand) REFERENCES public."Brands"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Products Products-ProductTypes; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Products"

ADD CONSTRAINT "Products-ProductTypes" FOREIGN KEY (type) REFERENCES public."ProductTypes"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Purchases Purchases-Branches; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Purchases"

ADD CONSTRAINT "Purchases-Branches" FOREIGN KEY (branch) REFERENCES public."Branches"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Purchases Purchases-Products; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Purchases"

ADD CONSTRAINT "Purchases-Products" FOREIGN KEY (product) REFERENCES public."Products"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Purchases Purchases-Suppliers; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Purchases"

ADD CONSTRAINT "Purchases-Suppliers" FOREIGN KEY (supplier) REFERENCES public."Suppliers"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Sells Sells-Customers; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Sells"

ADD CONSTRAINT "Sells-Customers" FOREIGN KEY (customer) REFERENCES public."Customers"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Sells Sells-Deliveries; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Sells"

ADD CONSTRAINT "Sells-Deliveries" FOREIGN KEY (delivery) REFERENCES public."Deliveries"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Sells Sells-Products; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Sells"

ADD CONSTRAINT "Sells-Products" FOREIGN KEY (product) REFERENCES public."Products"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Sells Sells-Workers; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Sells"

ADD CONSTRAINT "Sells-Workers" FOREIGN KEY (worker) REFERENCES public."Workers"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: StocksOfBranches StocksOfBranches-Branches; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."StocksOfBranches"

ADD CONSTRAINT "StocksOfBranches-Branches" FOREIGN KEY (branch) REFERENCES public."Branches"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: StocksOfBranches StocksOfBranches-Products; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."StocksOfBranches"

ADD CONSTRAINT "StocksOfBranches-Products" FOREIGN KEY (product) REFERENCES public."Products"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: SupplierManagers SupplierManagers-Inheritance; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."SupplierManagers"

ADD CONSTRAINT "SupplierManagers-Inheritance" FOREIGN KEY (key) REFERENCES public."People"(key) MATCH FULL ON UPDATE CASCADE ON DELETE CASCADE;

--

-- Name: Suppliers Suppliers-SupplierManagers; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Suppliers"

ADD CONSTRAINT "Suppliers-SupplierManagers" FOREIGN KEY ("supplierManager") REFERENCES public."SupplierManagers"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Workers Workers-Branches; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Workers"

ADD CONSTRAINT "Workers-Branches" FOREIGN KEY (branch) REFERENCES public."Branches"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- Name: Workers Workers-Inheritance; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Workers"

ADD CONSTRAINT "Workers-Inheritance" FOREIGN KEY (key) REFERENCES public."People"(key) MATCH FULL ON UPDATE CASCADE ON DELETE CASCADE;

--

-- Name: Workers Workers-Missions; Type: FK CONSTRAINT; Schema: public; Owner: postgres

--

ALTER TABLE ONLY public."Workers"

ADD CONSTRAINT "Workers-Missions" FOREIGN KEY (mission) REFERENCES public."Missions"(key) MATCH FULL ON UPDATE CASCADE ON DELETE RESTRICT;

--

-- PostgreSQL database dump complete

--

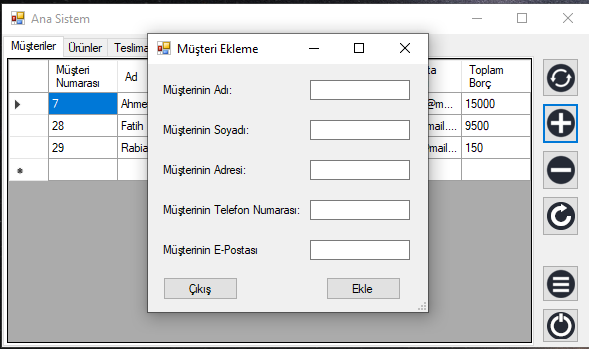
**Saklı Yordamlar**

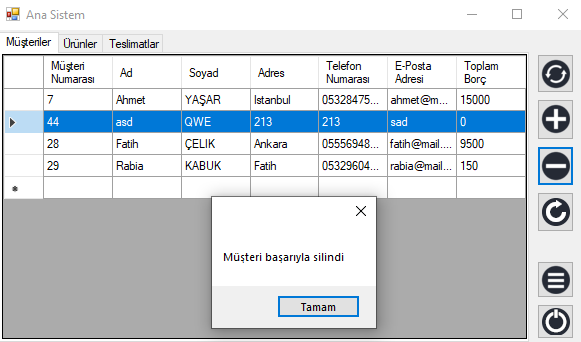
1. CheckBranch => Şubelerin kullanıcı adını ve şifresini kontrol eder.
2. RemoveCustomer => Müşterileri siler fakat önceden siparişi varsa o müşteriyi silmez.
3. RemoveProducts => Ürünleri siler fakat önceden satıldıysa o ürünü silmez.
4. RemoveSupplier => Tedarikçileri siler fakat önceden ondan alım yapıldıysa o tedarikçiyi silmez.
5. RemoveWorker => Çalışanları sistemden komple silmektense sadece çıkış tarihlerini ekler.
6. RemoveSell => Siparişleri siler fakat o sipariş teslim edildiyse o siparişi silmez.
7. RemovePerchase => Alımları siler fakat bu alım silindiğinde ürünün stokunu eksiye düşüyorsa o alımı silmez.

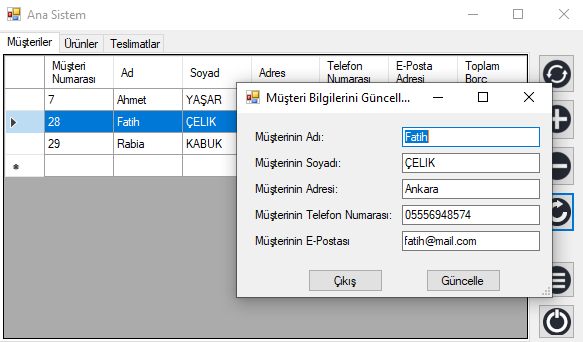
**Tetikleyiciler**

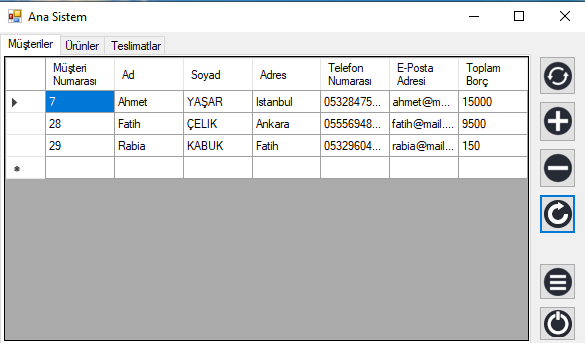
1. SetHappenedDate => Bir teslimatın durumu “Elden” ise siparişin teslim tarihini otomatik olarak o günün tarihi olarak ayarlar.
2. SetHappenedDate2 => Bir teslimatın durumu “Elden” değil ise siparişin teslim tarihini otomatik olarak NULL olacak şekilde ayarlar.
3. Payment => Müşteri ödeme yaptığında borcunu düşürür.
4. UpperSurname => Sisteme bir kişi eklendiğinde soyadının bütün hafleri otomatik olarak büyütülür.
5. SetStocksOfBranches => Bir ürün eklendiğinde o ürünün stoku bütün şubeler için 0 olarak ayarlanır.
6. Purchase => Bir şube bir ürün alımı yaptığında o ürünün stoku o şube için arttırılır.
7. RemovePurchaseT => Bir şube bir ürün alımını sildiğinde o ürünün stoku o şube için tekrardan ayarlanır.
8. Sell => Bir ürün satıldığında o ürünün stoku o şube için azaltılır.
9. RemoveSellT => Bir şube bir ürün satışını sildiğinde o ürünün stoku o şube için tekrardan ayarlanır.

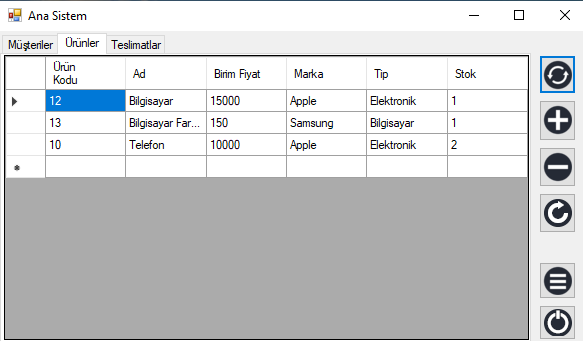
**Ekran Görüntüleri**

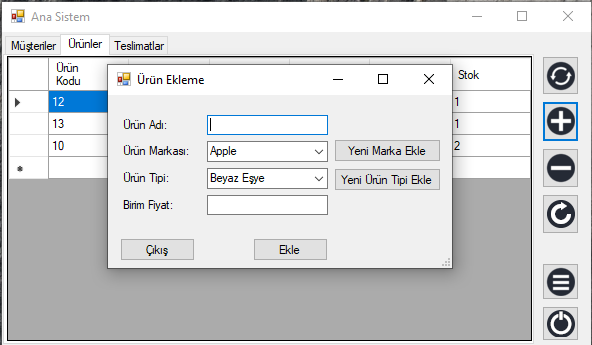


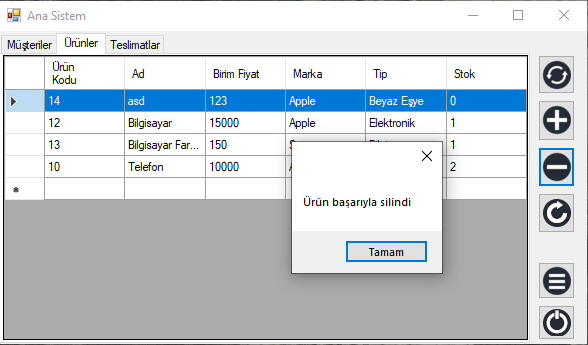


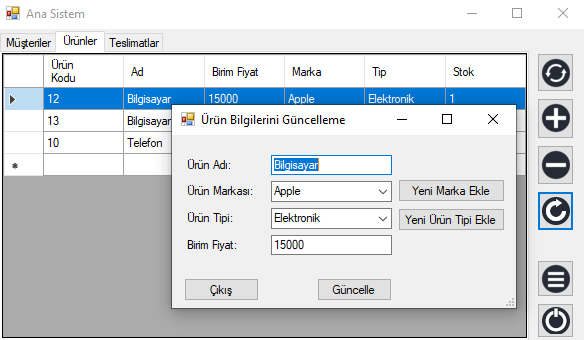


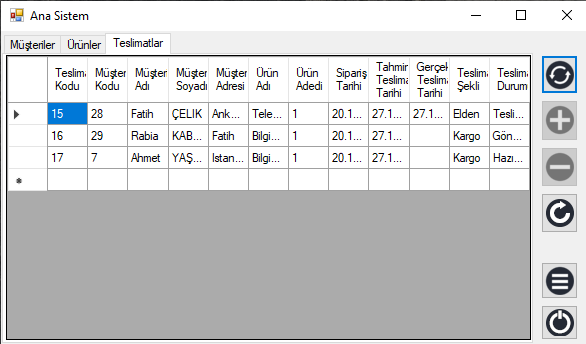


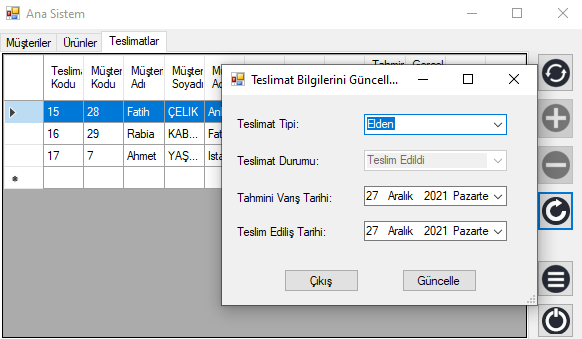


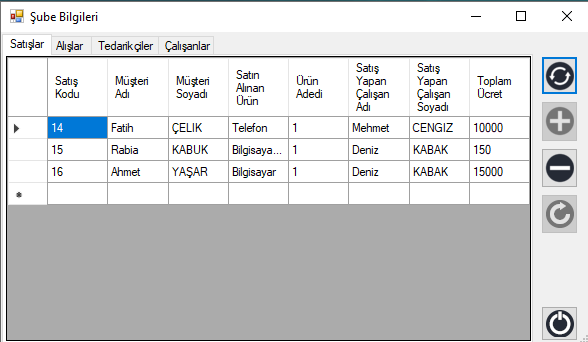


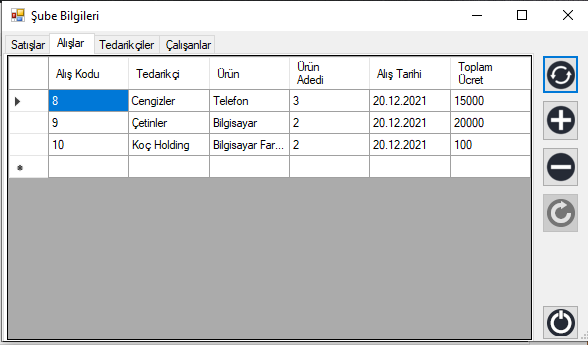


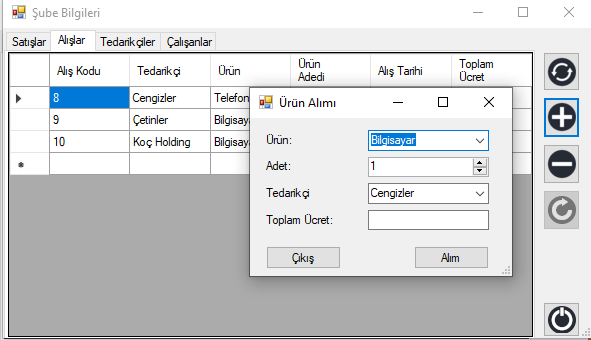


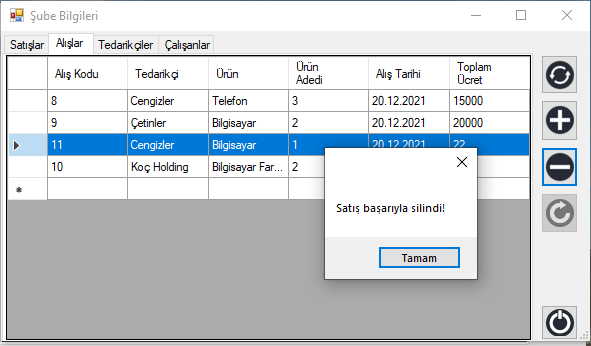


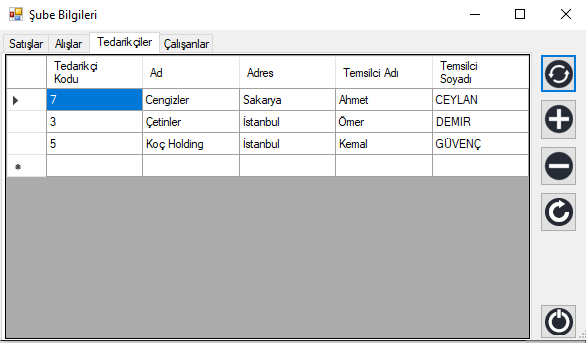


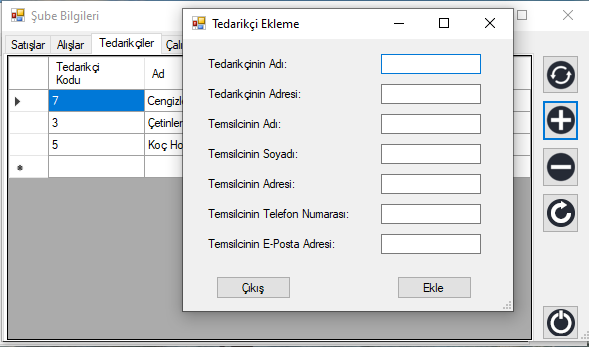


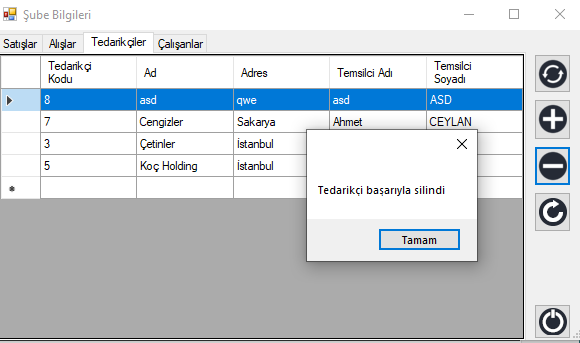


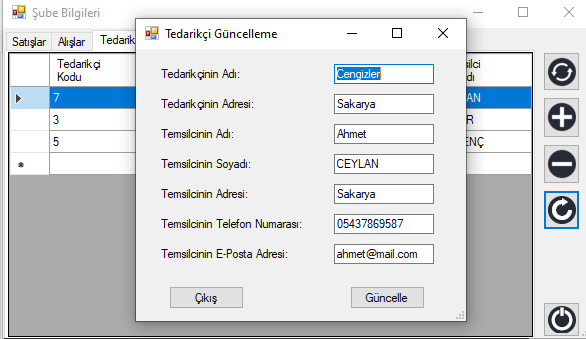


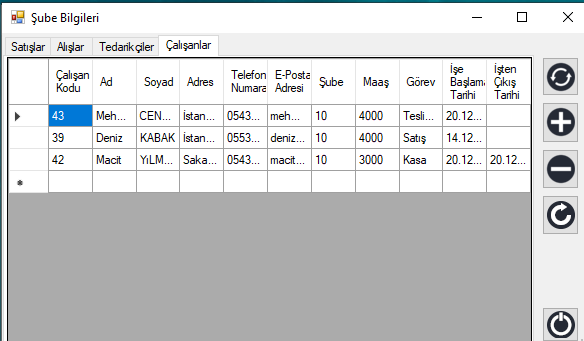


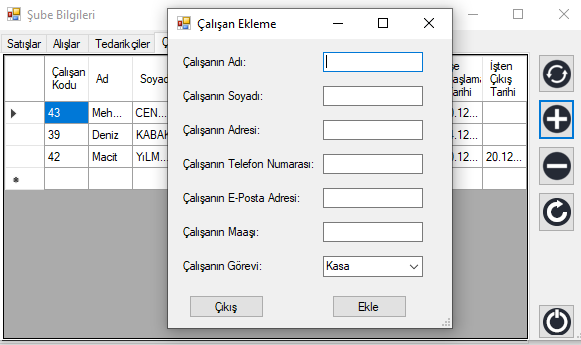


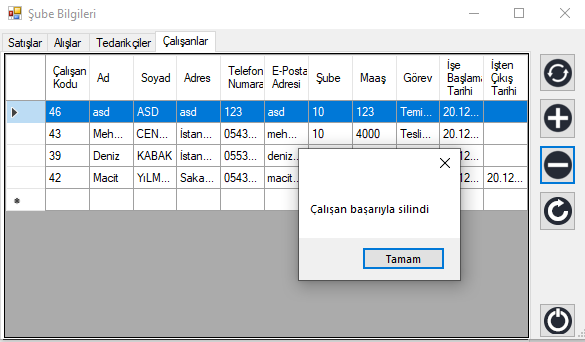


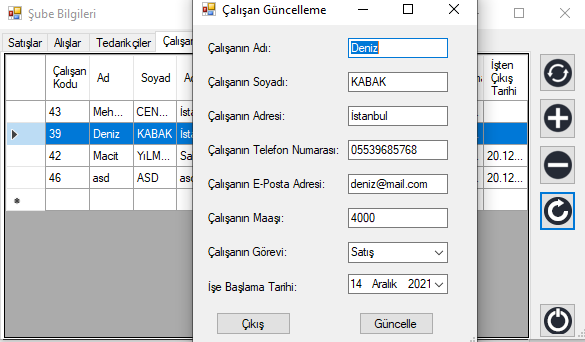


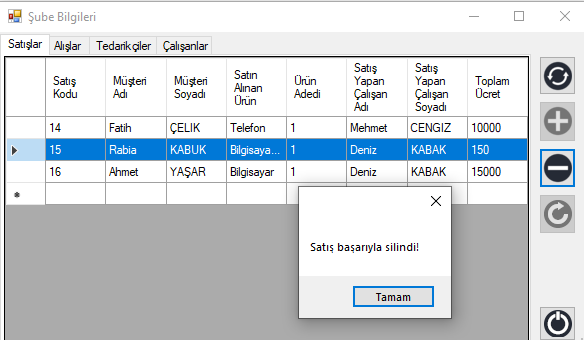












**Kaynak Kodları**

Github: <https://github.com/kemalguvenc/Database-Homework>

**Video**

Youtube:

Video-1 = <https://youtu.be/shPZwxasJvI>

Video-2= <https://youtu.be/fYLoYHwUxxg>

Not: Bilgisayarda yaşadığım bazı sıkıntılardan dolayı videoyu 2 parça halinde yükledim.