

ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА
ФАКУЛТЕТ ПО ИЗЧИСЛИТЕЛНА ТЕХНИКА И АВТОМАТИЗАЦИЯ
Катедра „Софтуерно Инженерство “



Курсова Работа по УИС

Управленски информационни системи

Тема:

Информационна система, подпомагаща управлението на музикален магазин.

Разработил:
Ивайло Пламенов Руменов
Фак. № 23651227

Проверил:
гл. ас. д-р Диян Динев

Съдържание

Задание	3
Увод	4
Проектиране на спрямо бизнес процеси на заданието	4
Архитектура на информационната система	5
Проектиране на базата от данни	6
Програмен код	11
Потребителски интерфейс	11
Заклучение	12
Положение	12

Задание

Управленски информационни системи -Задание 12

Да се създаде информационна система, подпомагаща управлението на музикален магазин. БД да съхранява информация за стоки, клиенти, служители и продажби. В таблиците да се включат атрибути като номер на артикул, вид, година, наименования, изпълнител, жанр, музикална компания, единична цена, наличност, номер на продажба, дата на продажбата, стоки, брой, име на клиент, адрес, телефон, име на служител, позиция, телефон, отдел и др. Базата от данни трябва да е нормализирана. При създаване на таблиците изберете подходящ тип данни и други свойства на полетата. Да се валидират данните. Ограничете броя и наличността да са само положителни числа, а датата на продажбата да е днешна. Наименованията на вида, изпълнителя, жанра и музикалната компания да се избират от списъци. Изберете правилни ключови полета. Свържете таблиците с подходящи релации. В основните таблици да има въведени минимум 10 записа коректни данни. Системата да генерира справки на база информация съхранена в повече от една таблица. Справките да са минимум 5. Например: Да се генерира справка, която да показва последните 5 продажби на стоки издадени последната година. Създадената информационна система да позволява въвеждане, корекция, актуализация и търсене на данни. Направените справки да могат да се експортират в подходящ файлов формат за отпечатване.

Увод

В курсовата работа е разгледано как се реализира информационна система подпомагаща музикален магазин. За реализацията на информационната система е разгледан как се декомпозира идеята за приложението на множество стъпки, като и реализацията на всяка стъпка. Започвайки от анализиране на нужната бизнес логика и тя как повлиява на използваните архитектури и програмни технологии за имплементация. Преминавайки към дълбоко разглеждане как е разработена базата от данни и нейните градивни елементи. Проекта включва и разглеждане на реализацията на бизнес логиката като програмен код. Както и как е тестван и застрахован от провал вътрешната логика на проекта. Проекта включва и разглеждане на как е реализиран потребителския интерфейс, както и как той комуникира с бизнес логика. И финализиране с оценка на трудностите, грешките и подобренията към проекта в негова цялост.

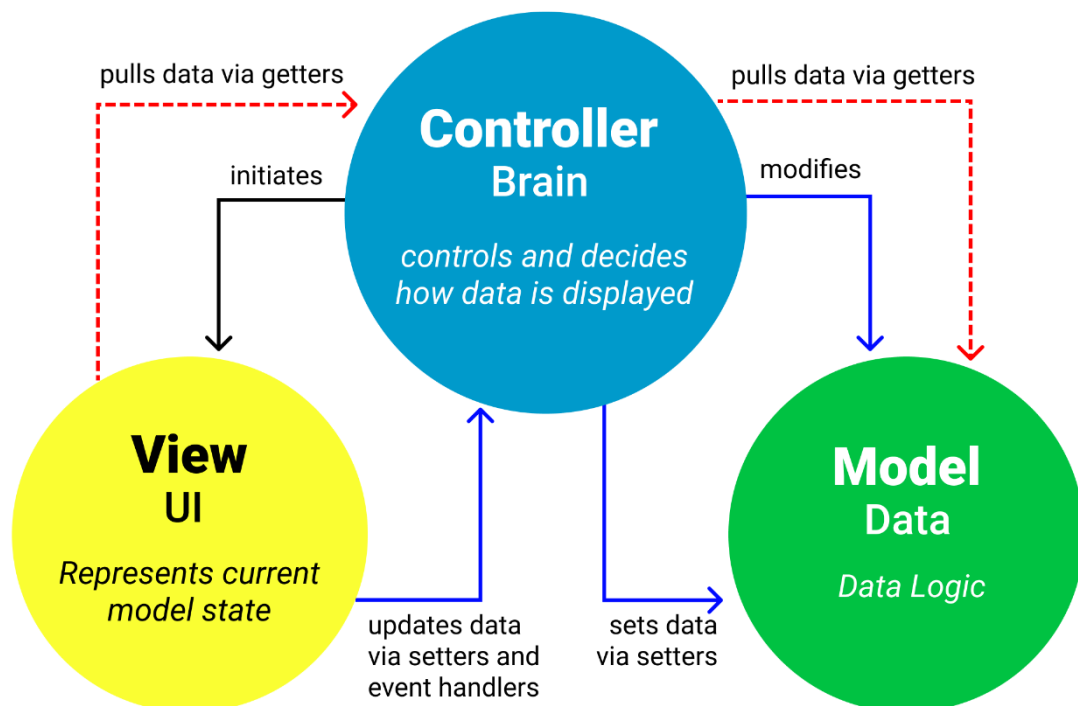
Проектиране на спрямо бизнес процеси на заданието

При започване на задание трябва да се разчертае и цялостната структура спрямо бизнес логиката на информационната система. В този случай е предвидено информационната система да се използва като спомагателно средство на музикален магазин. Тоест цялостната цел на заданието е с комерсиална цел. Знаейки това се примамва към факта че съдържанието и менажирането на стоки и продукти ще е главната цел на информационната система. Както и при комерсиалните приложения за пазаруване се предполага че клиенти ще желаят множеството от артикули да са сортирани и те самите при желание да сортират и търсят спрямо дадени критерий. Както при пазарува ще е нужно да се поддържа активна дадената количка на клиента. Тоест клиента да има свободата да добавя и премахва артикули в количката преди закупуване. И при закупуване на артикули информационната системата трябва да изведе дадените артикули от складова база с данни и да промени количество и да запише поръчката на клиента.

Архитектура на информационната система

След обзора на бизнес логиката можем се стига до факта че информационната система ще бъде от вид Модел Вю Контролер (Фиг. №1). Този модел е често срещан при програмни продукти от разнообразен вид. Той представлява групирани на програмния ко по неговата роля. Така се схематизира всяка единица да е отговорна за едно работно, което също подкрепя един от законите на обектно ориентирано програмиране. При Модел Вю Контролер се изготвят три елемента независими едни от друг. Модела отговорен за бизнес логиката както и връзка с базата от данни. Вю-то което е отговорно за потребителския интерфейс на информационната система. Контролера е последната част от модела и отговаря за комуникационния слой помежду двата елемента. За целта на изпълнението на задачата Контролера е реализиран чрез REST метода на работа и json стандарта за предаване на данни помежду програмните слоеве. REST е метод на реализиране на HTTP заявки помежду два така наречени крайни връзки ("Endpoints"). Често използван е метода REST за комуникация и стандартизиран като доверчив и бърз.

MVC Architecture Pattern



Фиг. №1 Графика на архитектурата Модел Вю Контролер

Проектиране на базата от данни

След приемане на метод на работа се преминава към едни от базовите елементи на модела Модел Вю Контролер и това е базата от данни. За целта на разработване на информационна система за музикален магазин, както и вземайки предвид предишен опит с работата по и с бази от данни, е избрано да се използва PostgreSQL база от данни. Тя използва налог на sql наречен sql. Базата е разгъната локално на машината, на която се разработва проекта. Съществуват облачно базирани услуги за поддържане на база от данни, като neon, digitOcean, cauchebase и много други. Удобството на облачно разгъната база от данни е че множество души могат да работят едновременно по базата но и съществуват проблеми при заключване на достъп до полета и таблици. Но за целта на задачата е предприето базата от данни да е разгъната на локалната машина. Поради липсата от нуждата да се споделя данни, като при екипи от разработчици. Но при имплементация на реална система е силно препоръчано тя да е разгъната на специализиран сървър за менажиране единствено и само за база от данни. Съществуват както и специализирани операционно системи само за менажиране и поддръжка на бази от данни, техните инстанции и мрежовата комуникация нужна за работа с базата.

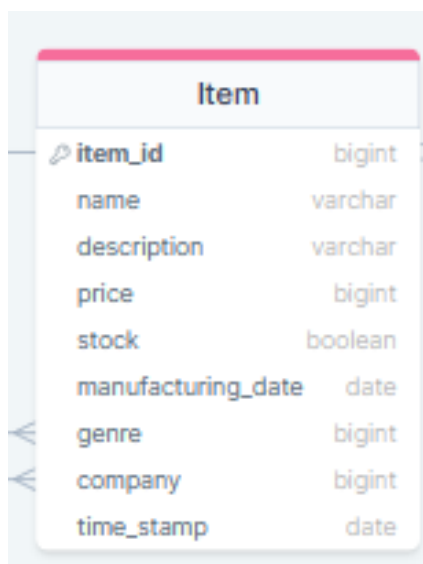
След като е избрана базата за работа и тя къде ще бъде разгъната може да се премине към писане на разработката на базата от данни и това може да се сведе до няколко ключови стъпки:

- Определете целта на база данни. Това ще помогне да се подготвите за останалите стъпки.
- Намиране и организиране на необходимата информация. Събиране на всички типове информация, която може да запишете в базата данни, като например име на продукт и номер на поръчка.
- Разделете информацията в таблици. Разделяне на информацията на елементи на основни обекти или теми, като например продукти или поръчки. След това всеки предмет се превръща в таблица.
- Превръщане на информационните елементи в колони. Решава се каква информация ще се съхраняват във всяка таблица. Всеки елемент става поле и се показва като колона в таблицата. Например таблицата "Служители" може да включва полета като "Фамилно име" и "Дата на наемане".
- Задаване на първични ключове. Избиране първичния ключ на всяка таблица. Първичният ключ е колона, която се използва за еднозначно идентифициране на всеки ред. Пример за това може да бъде ИД на продукт или ИД на поръчка.

- Това е само един от вариантите за разработка на база от данни. Но е преценено че за конкретната задача ще е нужен този метод на разработка на база от данни.

7

След като е представяне черновата и е определено целта на база от данни. Се преминава към отделяне на откритата информация в таблици. След гарирането на нужните елементи и атрибути, в базата от данни, се влиза в конкретиката на дадените атрибути. Осмисля се какви полета са нужни за таблицата и как те ще са обвързани с полета от други таблици. Относно условията на проекта, които гласят че базата от данни ще се употребявал като спомагателно средство на музикален магазин, се извежда факта че трябва да се инстанции устойчива система за следене състоянието на артикулите за продан. Тази функционалност е ядрото на всеки комерсиална информационна система. За това е решено да има таблица “Item” (Фиг. № 3), която е състоене от всичко сходни полета нужни на един артиколи да е годен за продан. От тази таблица се извежда две таблици едната за музикални инструменти „Instrument“ и за грамофонни плочи „Record“. Така системата може да се разпределя класовете на продукти в различни таблици с тяхното наименование, и когато бизнеса си реши да си разшири предметната дейност то може само да дови нова таблица с релация към таблицата с артиколи „Item“. Това от друга страна е недостатък към системата. Тъй като за всяко ново разделение ще е нужна нова таблица и може да се стигне до момент където множество таблици имат релация към таблицата с артиколи. Водейки към голяма опашка при заключване и отключване на достъпа към ресурси.

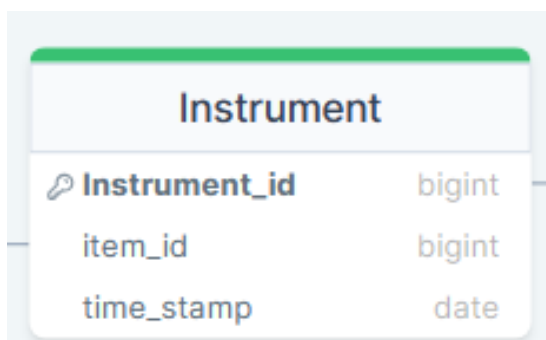


Item	
item_id	bigint
name	varchar
description	varchar
price	bigint
stock	boolean
manufacturing_date	date
genre	bigint
company	bigint
time_stamp	date

Фиг. № 3 Таблицата за Артиколи „Item“

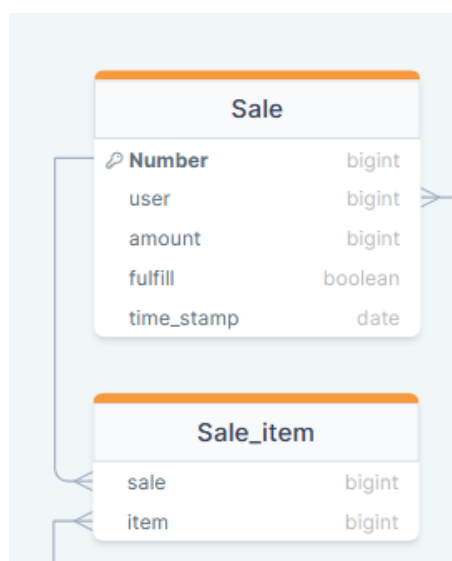
Но за целите на проекта е преценено е че най подходящия вариант за предметната дейност на магазина, която зависи от голяма разновидност при един вид продукти. На пример таблицата за инструменти „Instruments“ (Фиг. № 4), сама дефиниция на какво е инструмент варира от духово до струене. И в тази вариация се специализира допълнително, на пример от какъв материал е направен инструмента, от кой е настроен, дали е електронно съвместен и много други. Точно тази дълбока квалификация на инструментите е причината за разделяне на абстрактния артиколи, като нещо което се продава, и инструмента, като нещо което се използва за свирене. Така лесно се обединяват двете абстракции чрез релация и се получава инструмент за продаване. При това разделение на две абстрактни елемента лесно се добавя и премахнат един от друг. Затова е и възможно да се довят и други абстракции като таблицата за грамофонни плочи „Record“. Както и лесно може да се добавят допълнителни таблици към артикулите. Като е направено за производителя на артиколи с таблицата „Company“. Таблицата „Company“ сама по себе си представлява компания но тази компания в

информационното приложение може да се използва като таблица на производител или друг вид предназначение.



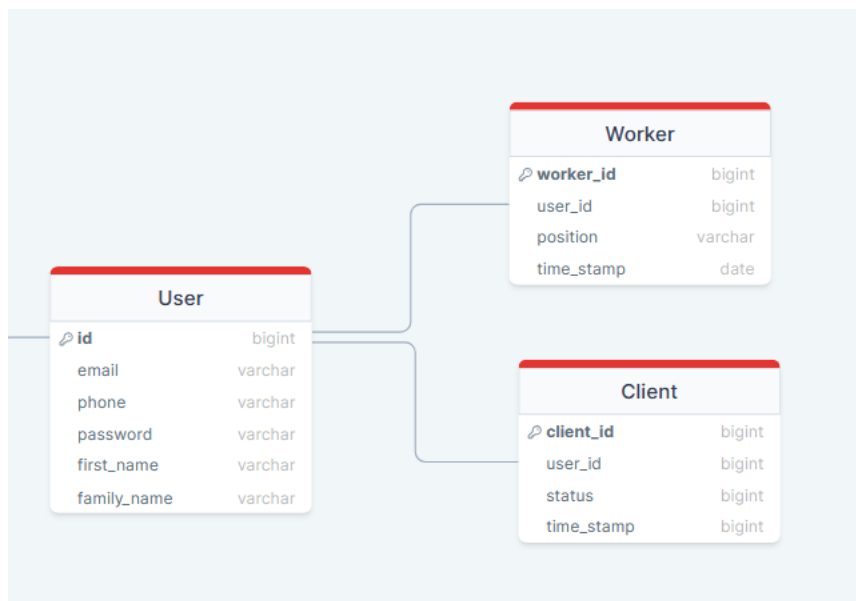
Фиг. № 4 Таблицата за инструменти „Instrument“

Следващата ключова част е самата продажба на артикулите. След като сме запазили правилно артикулите и знаем е са устойчиви може да се помисли как те могат да се продава (Фиг. №5). За тази цел е реализирана така наречената количка за пазаруване, която запазва в себе си релация към артикулите желани от купувача и таблица събирайки информацията за тази неизпълнена покупка. И когато клиента реши да си изпълни поръчка се записва като изпълнена и на клиента се създава нова количка за пазаруване.

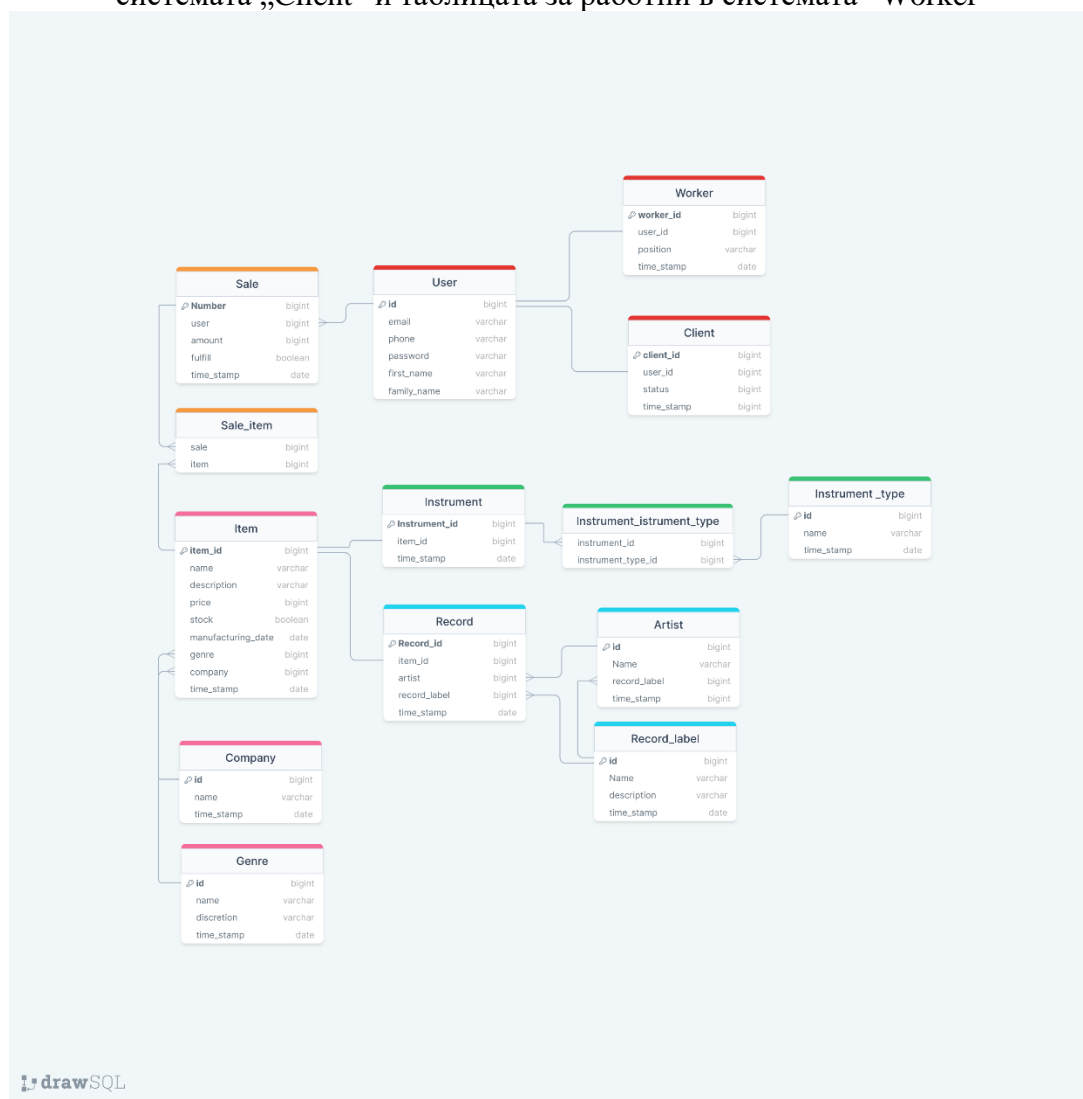


Фиг. №5 Таблиците за кошница за пазара „Sale_item“ и таблицата за продажба „Sale“

Но за да има клиента количка той първо се нуждае от профил, в който ще се отбележи колко поръчки един клиент има и от какво са се състои дадената поръчка. Когато се имплементира модел за профил на потребите то лесно може да се добави профили за работници(Фиг. №6). Това е възможно когато се декомпозира какво е потребител и се раздели на купувач и работник. Те в същности си имат еднакви елементи.



Фиг. №6 Таблицата за потребител на системата „User“, таблицата за клиента на системата „Client“ и таблицата за работни в системата “Worker”



Фиг. №3 Графика на реализираните таблици в базата от данни и техните релации.

Програмен код

След като е инициализиране базата на система може да се премини към вливането на програмния код. Тоест да се реализира бизнес процеси чрез вътрешни алгоритми на системата. За реализирането на това е избрано програмния език java, избран защото е отворен на пазара, представлява устойчивост чрез силно типизиран обектно ориентиран модел на работа и лесен за имплементиране като модел. Но сам по себе си java е слаб затова е имплементиран чрез библиотечната средата Spring. Който позволява улеснен метод за работа с REST и бази от данни едновременно. Той предоставя и вграден начин на тестване на цялостната система от край до край.

Когато имаме бизнес процеси за имплементиране и модела на работа се започва от горе надолу. Тоест се запазва от външните точки (endpoints) на системата. Тези външни точки представляват http uri, който крие зад себе си логиката имплементиране външни. Затова на този крий и само на този слой се правят проверки за коректни данни на входа. След което се преминава към сервизната част на програмата, която представлява алгоритмичната част на програмата. Тук се реализира по някаква форма бизнес логиката на приложение съчета с вътрешно програмна. Вътрешно програмната реализации използва методи на езика за предотвратяване на повторение или други проблеми свързани с обектното ориентирано програмиране. Този слой може множествено пъти да вика набор от методи отговорни за достъп до базата от данни. Тези методи са в така наречен репозитори “repository” и в някой имплементации на модела се използва друг метод за достъп до базата наречен персистентност „persistence“. На за даденото задание е прието да се използват директни заявки към базата нарече queries. Този трислоен подход позволява за висока устойчивост на предложението.

Потребителски интерфейс

Потребителския интерфейс е реализиран чрез уеб базиран метод. Той се гради на javascript езика отговарящ за логика на интерфейса и http съчетан със css. Това е често срещата комбинация за имплементиране на потребителски интерфейс в уеб пространството. Цели този пакет е имплементиран чрез скелета за имплементация наречен angular. Angular е скелет осланиящ процеса на работа в уеб пространството. Той е разработен от Google и е от ней-употребяваните на пазара.

Заклучение

Заданието за представляващ спомагателна информационна система за музикален магазин бе предизвикателна и широко разширяваща множество знания за продукти тяхната логика и как работят едно с друго. Базата от данни използвана е PostgreSQL достъпна чрез String rest application и Angular потребителски интерфейс възпроизведен в уеб браузър. За бъдеще може да се помисли и за добавяне допълнителни абстракции към проекта. Като например опцията за вземане продукт на изплащане. Това лесно може да се постигне тъй като имаме таблица за артикули „Item“, към която може да се добави таблица за период на изплащане който е вързан и към клиента. Друга иновация за процеса на работа, който е лесно за разработване благодарение на модела на имплементация на базата, е отбелязването на служител който да проследява поръчката на клиента. Така всеки клиент ще знае към кой служител може да се обърне за проблеми с поръчката или проблеми с продукта.

Положение

Приложени скриптове за базата от данни

За създаване на базата е използван скрипта

Create.sql

```
CREATE TABLE Ape_user(  
    id BIGINT NOT NULL,  
    email VARCHAR(255) NOT NULL,  
    phone VARCHAR(255) NOT NULL,  
    password VARCHAR(255) NOT NULL,  
    first_name VARCHAR(255) NOT NULL,  
    family_name VARCHAR(255) NOT NULL  
);  
ALTER TABLE  
    Ape_user ADD PRIMARY KEY(id);  
CREATE TABLE Instrument(  
    Instrument_id BIGINT NOT NULL,  
    item_id BIGINT NOT NULL,  
    type BIGINT NOT NULL,  
    time_stamp DATE NOT NULL  
);  
ALTER TABLE  
    Instrument ADD PRIMARY KEY(Instrument_id);  
CREATE TABLE Item(  
    item_id BIGINT NOT NULL,
```

```

        stock BOOLEAN NOT NULL,
        price BIGINT NOT NULL,
        discrimination VARCHAR(255) NULL,
        manufacturing_date DATE NOT NULL,
        name VARCHAR(255) NOT NULL,
        description VARCHAR(255) NOT NULL,
        genre BIGINT NOT NULL,
        company BIGINT NOT NULL,
        time_stamp DATE NOT NULL
    );
    ALTER TABLE
        Item ADD PRIMARY KEY(item_id);
    CREATE TABLE Sale_item(
        sale BIGINT NOT NULL,
        item BIGINT NOT NULL
    );
    CREATE TABLE Instrument_instrument_type(
        instrument_id BIGINT NOT NULL,
        instrument_type_id BIGINT NOT NULL
    );
    CREATE TABLE Sale(
        Number BIGINT NOT NULL,
        ape_user BIGINT NOT NULL,
        amount BIGINT NOT NULL,
        fulfill BOOLEAN NOT NULL,
        time_stamp DATE NOT NULL
    );
    ALTER TABLE
        Sale ADD PRIMARY KEY(Number);
    CREATE TABLE Genre(
        id BIGINT NOT NULL,
        name VARCHAR(255) NOT NULL,
        discretion VARCHAR(255) NOT NULL,
        time_stamp DATE NOT NULL
    );
    ALTER TABLE
        Genre ADD PRIMARY KEY(id);
    CREATE TABLE Company(
        id BIGINT NOT NULL,
        name VARCHAR(255) NOT NULL,
        time_stamp DATE NOT NULL

```

```

);
ALTER TABLE
    Company ADD PRIMARY KEY(id);
CREATE TABLE Artist(
    id BIGINT NOT NULL,
    Name VARCHAR(255) NOT NULL,
    record_label BIGINT NOT NULL,
    time_stamp DATE NOT NULL
);
ALTER TABLE
    Artist ADD PRIMARY KEY(id);
CREATE TABLE Record(
    Record_id BIGINT NOT NULL,
    item_id BIGINT NOT NULL,
    artist BIGINT NOT NULL,
    record_label BIGINT NOT NULL,
    time_stamp DATE NOT NULL
);
ALTER TABLE
    Record ADD PRIMARY KEY(Record_id);
CREATE TABLE Record_label(
    id BIGINT NOT NULL,
    Name VARCHAR(255) NOT NULL,
    description VARCHAR(255) NOT NULL,
    time_stamp DATE NOT NULL
);
ALTER TABLE
    Record_label ADD PRIMARY KEY(id);
CREATE TABLE Worker(
    worker_id BIGINT NOT NULL,
    ape_user BIGINT NOT NULL,
    position VARCHAR(255) NOT NULL,
    time_stamp DATE NOT NULL
);
ALTER TABLE
    Worker ADD PRIMARY KEY(worker_id);
CREATE TABLE Client(
    client_id BIGINT NOT NULL,
    ape_user BIGINT NOT NULL,
    status BIGINT NOT NULL,
    time_stamp BIGINT NOT NULL

```

```

);
ALTER TABLE
    Client ADD PRIMARY KEY(client_id);
CREATE TABLE Instrument_type(
    id BIGINT NOT NULL,
    instrument_type_name VARCHAR(255) NOT NULL,
    time_stamp DATE NOT NULL
);
ALTER TABLE
    Instrument_type ADD PRIMARY KEY(id);
ALTER TABLE
    Instrument_istrument_type ADD CONSTRAINT
instrument_istrument_type_instrument_type_id_foreign FOREIGN
KEY(instrument_type_id) REFERENCES Instrument_type(id);
ALTER TABLE
    Item ADD CONSTRAINT item_company_foreign FOREIGN KEY(company)
REFERENCES Company(id);
ALTER TABLE
    Record ADD CONSTRAINT record_item_id_foreign FOREIGN
KEY(item_id) REFERENCES Item(item_id);
ALTER TABLE
    Sale_item ADD CONSTRAINT sale_item_sale_foreign FOREIGN KEY(sale)
REFERENCES Sale(Number);
ALTER TABLE
    Record ADD CONSTRAINT record_artist_foreign FOREIGN KEY(artist)
REFERENCES Artist(id);
ALTER TABLE
    Instrument ADD CONSTRAINT instrument_item_id_foreign FOREIGN
KEY(item_id) REFERENCES Item(item_id);
ALTER TABLE
    Artist ADD CONSTRAINT artist_record_label_foreign FOREIGN
KEY(record_label) REFERENCES Record_label(id);
ALTER TABLE
    Worker ADD CONSTRAINT worker_user_id_foreign FOREIGN
KEY(ape_user) REFERENCES Ape_user(id);
ALTER TABLE
    Sale_item ADD CONSTRAINT sale_item_item_foreign FOREIGN
KEY(item) REFERENCES Item(item_id);
ALTER TABLE
    Record ADD CONSTRAINT record_record_label_foreign FOREIGN
KEY(record_label) REFERENCES Record_label(id);

```

```

ALTER TABLE
    Client ADD CONSTRAINT client_user_id_foreign FOREIGN KEY(ape_user)
REFERENCES Ape_user(id);
ALTER TABLE
    Item ADD CONSTRAINT item_genre_foreign FOREIGN KEY(genre)
REFERENCES Genre(id);
ALTER TABLE
    Sale ADD CONSTRAINT sale_user_foreign FOREIGN KEY(ape_user)
REFERENCES Ape_user(id);
ALTER TABLE
    Instrument_istrument_type ADD CONSTRAINT
instrument_istrument_type_instrument_id_foreign FOREIGN KEY(instrument_id)
REFERENCES Instrument(Instrument_id);

```

За вмъкване на тестова информация е използван скрита:

Isert.sql

/*

Creating 5 Users

*/

```

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (1, 'ivo@mail.com', '089671253', 'ivo12345678', 'Ivaylo', 'Rumenov');

```

```

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (2, 'asen@mail.com', '089412425', 'asen123asen', 'Asen', 'Hristove');

```

```

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (3, 'damqn@mail.com', '09984932', 'damqnShefa', 'Damqn', 'Mihailov');

```

```

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (4, 'hasan@mail.com', '08931265', 'hasendej', 'Hasan', 'Hassanov');

```

```

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (5, 'user@mail.com', '08999999', '1234', 'User1', 'User1');

```

```

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (6, 'muamed@mail.com', '0894234', 'muhadqnina123', 'Muhamed', 'Imal');

```

```

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)

```



```

VALUES (7, 'IvanZvezdev@gmail.com', '0347573648', '112323231', 'Ivan', 'Zvezdev');

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (8, 'KokoNikolov@mail.com', '08123123543', 'kdopasaoi', 'Koko', 'Nikolov');

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (9, 'CecaIvanova@hotmail.com', '0892342176', 'Mamboamericano', 'Ceca',
'Ivanova');

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (10, 'DamqnHristov@abv.bg', '089123789', 'Carqnakoona', 'Damqn',
'Hristov');

INSERT INTO Ape_user(id, email, phone, password, first_name, family_name)
VALUES (11, 'Marakew213@outlook.com', '08398345', 'Dubail123', 'Samoil',
'Kovachev');
/*
    Creating 3 Workers
*/

INSERT INTO Worker(worker_id,ape_user,position,time_stamp)
VALUES (1,1,'Meneger','2023-10-28');

INSERT INTO Worker(worker_id,ape_user,position,time_stamp)
VALUES (2,2,'HR','2023-10-28');

INSERT INTO Worker(worker_id,ape_user,position,time_stamp)
VALUES (3,3,'Support','2023-10-28');

/*
    Creating 2 Clients
*/

INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (1,4,1,1);

INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (2,5,2,1);

INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (3,6,1,1);

```

```
INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (4,7,2,1);
```

```
INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (5,8,1,1);
```

```
INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (6,9,2,1);
```

```
INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (7,10,2,1);
```

```
INSERT INTO Client(client_id,ape_user,status,time_stamp)
VALUES (8,11,2,1);
```

```
/*
    Creating 3 Instrument types
*/
```

```
INSERT INTO Instrument_type(id,instrument_type_name,time_stamp)
values (1,'Gittar','2023-10-28');
```

```
INSERT INTO Instrument_type(id,instrument_type_name,time_stamp)
values (2,'Drums','2023-10-28');
```

```
INSERT INTO Instrument_type(id,instrument_type_name,time_stamp)
values (3,'Brass','2023-10-28');
```

```
/*
    Creating 3 Genres
*/
```

```
INSERT INTO Genre(id,name,discretion,time_stamp)
values (1,'Rock','you know','2023-10-28');
```

```
INSERT INTO Genre(id,name,discretion,time_stamp)
values (2,'Jazz','you know','2023-10-28');
```

```
INSERT INTO Genre(id,name,discretion,time_stamp)
values (3,'Pop','you know','2023-10-28');
```

```
INSERT INTO Genre(id,name,discretion,time_stamp)
values (4,'Reggae','you know','2023-10-28');
```

```
INSERT INTO Genre(id,name,discretion,time_stamp)
values (5,'Mettal','you know','2023-10-28');
```

```
/*
    Creating 3 companies
*/
```

```
INSERT INTO Company(id,name,time_stamp)
values (1,'Febder','2023-10-28');
```

```
INSERT INTO Company(id,name,time_stamp)
values (2,'LPD','2023-10-28');
```

```
INSERT INTO Company(id,name,time_stamp)
values (3,'Yamaha','2023-10-28');
```

```
INSERT INTO Company(id,name,time_stamp)
values (4,'Perl','2023-10-28');
```

```
INSERT INTO Company(id,name,time_stamp)
values (5,'Marpex','2023-10-28');
```

```
INSERT INTO Company(id,name,time_stamp)
values (6,'Thumb','2023-10-28');
```

```
/*
    Creating 2 guitars and one bass Instruments.
*/
```

```
INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (1,TRUE,32,'Fender_Squier_affinity_Series_Jazz_Bass.jpg','2013-10-
28','Fender_Squier_Classic','Some diescripton',1,1,'2023-10-28');
```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (2,TRUE,54,'Fender_Squier_Affinity.jpg','2013-10-
28','Fender_Squier_Affinity','Some diescripton',1,1,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (3,TRUE,65,'Fender_Squier_Bass.jpg','2013-10-28','Yamaha_YAS','Some
diescripton',2,3,'2023-10-28');

```

```

/*
    Creating 3 Records
*/

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (4,TRUE,32,'Guns_N_Rosses_GreatestHits.jpg','2013-10-
28','Guns_N_Rosses_GreatestHits','Some diescripton',2,3,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (5,TRUE,43,'Nirvana_Nevermind.JPG','2013-10-
28','Nirvana_Nevermind','Some diescripton',2,3,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (6,TRUE,15,'Queen_Greatest_Hits_1.jpg','2013-10-
28','Queen_Greatest_Hits_1','Some diescripton',2,3,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (7,TRUE,65,'Yamaha_YAS_280_S.jpg','2013-10-28','Yamaha_YAS','Some
diescripton',2,3,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (8,TRUE,45,'Yamaha_YAS_280.jpg','2013-10-28','Yamaha_YAS','Some
diescripton',2,3,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (9,TRUE,34,'Yamaha_YEP_642_TS.jpg','2013-10-28','Yamaha_YAS','Some
diescripton',2,3,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (10,TRUE,47,'Yamaha_YEP_642.jpg','2013-10-28','Yamaha_YAS','Some
diescripton',2,3,'2023-10-28');

```

/* Drum items */

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (11,TRUE,500,'thumb_base_fdb9291a.jpg','2013-10-28','Tama MBSS55-SKA
Starclassic','Some diescripton',1,6,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (12,TRUE,1340,'Ymapex-armory-ocean-sunset-ar628sfujg.jpg','2013-10-
28','Ymapex-armory','Some diescripton',1,5,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (13,TRUE,1120,'EXL725S-C246-large.jpg','2013-10-28','Perl','Some
diescripton',1,4,'2023-10-28');

```

/* Second set of records*/

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (14,TRUE,123,'Tyler_The_Creator_Igor.jpg','2013-10-28','Tyler The Creator
Igor','Some diescripton',3,6,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (15,TRUE,32,'Tama_Impala_Currents.jpg','2013-10-28','Tama Impala
Currents','Some diescripton',1,5,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (16,TRUE,54,'The_Bearles_Abbey_Read.JPG','2013-10-28','The Bearles Abbey
Read','Some diescripton',1,4,'2023-10-28');

```

/* Metalica*/

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (17,TRUE,44,'Metakkica_Garage_Inc.jpg','2013-10-28','Metakkica Garage
Inc','Some diescripton',5,6,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (18,TRUE,35,'Metallica.JPG','2013-10-28','Kill dem all','Some
diescripton',5,5,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (19,TRUE,56,'Metallica_Hardwired_To_Self_Destruct.JPG','2013-10-
28','Hardwired To Self Destruct','Some diescripton',5,4,'2023-10-28');

```

/* Bob marly */

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (20,TRUE,46,'Bob_Marley_Natty_Dread_Tour_75.jpg','2013-10-28','Natty
Dread Tour','Some diescripton',4,6,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (21,TRUE,75,'Bob_Marley_Uprising_Live.jpg','2013-10-28','Uprising
Live','Some diescripton',4,5,'2023-10-28');

```

```

INSERT INTO
Item(item_id,stock,price,discrimination,manufacturing_date,name,description,genre,company
,time_stamp)
values (22,TRUE,45,'Bob+Marley_Songs_Of_Freedom.jpg','2013-10-28','Songs Of
Freedom','Some diescripton',4,4,'2023-10-28');

```

```

/*
Creating 3 Instruments
*/

```

```

INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (1,1,1,'2023-10-28');

```

```

INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (2,2,2,'2023-10-28');

```

```

INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (3,3,1,'2023-10-28');

```

```

INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (4,7,1,'2023-10-28');

```

```

INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (5,8,1,'2023-10-28');

```

```

INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (6,9,1,'2023-10-28');

```

```

INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)

```

```
VALUES (7,10,1,'2023-10-28');
```

```
INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (8,11,2,'2023-10-28');
```

```
INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (9,12,2,'2023-10-28');
```

```
INSERT INTO Instrument(Instrument_id,item_id,type,time_stamp)
VALUES (10,13,2,'2023-10-28');
```

```
/*
```

```
    Creating 3 Instruments
```

```
*/
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (1,1);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (2,1);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (3,3);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (4,3);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (5,3);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (6,3);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (7,3);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (8,2);
```

```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (9,2);
```



```
INSERT INTO Instrument_instrument_type(instrument_id,instrument_type_id)
VALUES (10,2);
```

```
/*
```

```
    Creating 3 Record label
```

```
*/
```

```
INSERT INTO Record_label(id,Name,description,time_stamp)
VALUES (1,'Wiener Records','Some diescripton','2023-10-28');
```

```
INSERT INTO Record_label(id,Name,description,time_stamp)
VALUES (2,'Kanine Records','Some diescripton','2023-10-28');
```

```
INSERT INTO Record_label(id,Name,description,time_stamp)
VALUES (3,'Aftermath Entertainment','Some diescripton','2023-10-28');
```

```
/*
```

```
    Creating 3 Artists
```

```
*/
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (1,'GunsNRoses',1,'2023-10-28');
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (2,'Nirvana',1,'2023-10-28');
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (3,'Queen',1,'2023-10-28');
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (4,'Tyler The Creator',1,'2023-10-28');
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (5,'Tema',1,'2023-10-28');
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (6,'The Bearles',1,'2023-10-28');
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (7,'Metalica',1,'2023-10-28');
```

```
INSERT INTO Artist(id,Name,record_label,time_stamp)
VALUES (8,'Bob Marley',1,'2023-10-28');
```

```
/*
    Creating 3 Records
*/
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (1,4,1,1,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (2,5,2,1,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (3,6,3,3,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (4,14,4,1,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (5,15,5,1,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (6,16,6,3,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (7,17,7,2,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (8,18,7,2,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (9,19,7,2,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (10,20,8,1,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (11,21,8,1,'2023-10-28');
```

```
INSERT INTO Record(Record_id,item_id,artist,record_label,time_stamp)
VALUES (12,22,8,1,'2023-10-28');
```

```
/*
    Creating 10 Sale
*/
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (1,4,44,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (2,4,44,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (3,4,123,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (4,4,432,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (5,4,425,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (6,4,52,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (7,4,36,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (8,4,16,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (9,4,134,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
values (10,4,543,TRUE,'2023-10-28');
```

```
INSERT INTO Sale(Number,ape_user,amount,fulfill,time_stamp)
```

```
values (12,5,543,TRUE,'2023-10-28');  
/*  
    Creating 3 Sale_item  
*/
```

```
INSERT INTO Sale_item(sale,item)  
values (1,1);
```

```
INSERT INTO Sale_item(sale,item)  
values (1,2);
```

```
INSERT INTO Sale_item(sale,item)  
values (1,3);
```

```
INSERT INTO Sale_item(sale,item)  
values (2,1);
```

```
INSERT INTO Sale_item(sale,item)  
values (2,3);
```

```
INSERT INTO Sale_item(sale,item)  
values (4,2);
```

```
INSERT INTO Sale_item(sale,item)  
values (5,6);
```

```
INSERT INTO Sale_item(sale,item)  
values (6,7);
```

```
INSERT INTO Sale_item(sale,item)  
values (7,7);
```

```
INSERT INTO Sale_item(sale,item)  
values (8,8);
```

```
INSERT INTO Sale_item(sale,item)  
values (9,9);
```

```
INSERT INTO Sale_item(sale,item)  
values (10,10);
```

```
INSERT INTO Sale_item(sale,item)
values (10,1);
```

```
INSERT INTO Sale_item(sale,item)
values (10,1);
```

За бизнес логита са използвани следните inset, create, update, delete:

Select.sql

```
/*
    Get all Instruments
*/
```

```
SELECT
    it.instrument_type_name,
        i3.name    as    item_name,c.name,i3.stock,i3.price,i3.manufacturing_date,
i3.discrimination
    ,g.name as gener_name
FROM instrument
join public.item i3 on i3.item_id = instrument.item_id
join public.genre g on g.id = i3.genre
join  public.instrument_istrument_type  iit  on  instrument.instrument_id  =
iit.instrument_id
join public.instrument_type it on it.id = iit.instrument_type_id
join public.company c on c.id = i3.company;
```

```
insert into instrument_istrument_type(instrument_id, instrument_type_id)
VALUES(10,1);
```

```
select * from genre;
/*
    Get all records
*/
```

```
SELECT i.name as item_name ,i.price,i.stock,a.name as artist_name,rl.name as
record_label_name from record
join public.artist a on a.id = record.artist
join public.record_label rl on rl.id = record.record_label
join public.item i on i.item_id = record.item_id;
```

```

/*
    Get all artist records
*/
SELECT artist.name as artist_name, rl.name as record_label_name from artist
    join public.record r on artist.id = r.artist
join public.record_label rl on artist.record_label = rl.id;

/*
    Get artist by name
*/
SELECT artist.name as artist_name, rl.name as record_label_name from artist
join public.record r on artist.id = r.artist
join public.record_label rl on artist.record_label = rl.id
where artist.name='Queen'
Limit 1;
/*
    todo
    sell item
        1-add to sale_item
        2-add to sale
        3-finish
    see all sales and user to dem
    WEB
    add drop down to admin meny
    add id to the tables & order them by id
    add user profile

*/
/*
    All sales for a user
*/
SELECT sale.number, au.email from sale
join public.ape_user au on au.id = sale.ape_user
where au.id=4
group by au.email, sale.number;

SELECT sale.number , sale.amount, sale.time_stamp, sale.fulfill from sale
join public.ape_user au on au.id = sale.ape_user
where au.id=4
group by au.email, sale.number;
/*

```

```

    Update user status
*/
    update client set status=2 where ape_user=1;
/*
Most sold item
*/
SELECT count(sale_item.item) as count_,i.name, i.discrimination as img_url from
sale_item
    join public.item i on sale_item.item=i.item_id
group by i.name, i.discrimination
order by count_ desc
LIMIT 1;
/*
    Sale a item
*/
/* Step 0: get users curent shopping cart if he hsae one*/

SELECT s.number from ape_user
join public.sale s on ape_user.id = s.ape_user
where s.fulfill=true and s.ape_user=4;

SELECT client.status from client
    where client_id =1;

/* Step 1: make a Sale and add a item*/

insert into sale(number, ape_user, amount, fulfill, time_stamp)
values (11,2,0,false, CURRENT_DATE);

/* Stem 2: Persist clients shopping cart id as sale.number
and add new items as the clients adds them to the cart
where 1 is the new item
where 11 is the users shopping cart*/
insert into sale_item(sale, item)
values (11,1);
/*delete item from shopping cart*/
delete from Sale_item where sale_item.sale=? && sale_item.item=?;
/* Step 3: Finish the order and set to user*/
update sale set fulfill=true where number=?;

/* Get users shpoing card whit items by user id*/

```

```

select si.item from sale
join public.sale_item si on sale.number = si.sale
join public.ape_user au on sale.ape_user = au.id
where au.id=4 and sale.fulfill=false;

/*
    Get by Instrument Type
*/
/* Select all instrument types for drop down*/
SELECT instrument_type_name from instrument_type;
/* Get all Brass instruments*/
SELECT
    it.instrument_type_name,
        i3.name    as    item_name,c.name,i3.stock,i3.price,i3.manufacturing_date,
i3.discrimination
        ,g.name as gener_name
FROM instrument
    join public.item i3 on i3.item_id = instrument.item_id
    join public.genre g on g.id = i3.genre
        join public.instrument_istrument_type iit on instrument.instrument_id =
iit.instrument_id
        join public.instrument_type it on it.id = iit.instrument_type_id
        join public.company c on c.id = i3.company
WHERE it.id=3;

/*
    Get by Records Genre
*/
/* Select all genres for drop down*/
SELECT name from genre;

SELECT i.name as item_name ,i.price,i.stock,
    a.name as artist_name,rl.name as record_label_name from record
        join public.artist a on a.id = record.artist
        join public.record_label rl on rl.id = record.record_label
        join public.item i on i.item_id = record.item_id
where i.genre=2;
/*
    See all used sales
*/

```



```

select sale.amount,sale.time_stamp as date,au.email as user from sale
join public.ape_user au on au.id = sale.ape_user
join public.client c on au.id = c.ape_user
where sale.fulfill=true;

/*
    Drop down selects
*/

select artist.name from artist;
select company.name from company;
SELECT name from genre;
SELECT instrument_type_name from instrument_type;
select rl."name" from record_label rl;
/* All record names*/
select i."name" from record r
join public.item i on i.item_id = r.item_id ;
/* All instrument names*/
select i."name" from instrument i2
join public.item i on i.item_id = i2.item_id ;

/*
    user profile
*/

SELECT * from ape_user
join public.client cl on cl.ape_user=ape_user.id
where ape_user.id=4;

/*
    All DELETES
*/
/* Delete Intrument */
delete          from          instrument_istrument_type          where
instrument_istrument_type.instrument_id=8;
delete from instrument where instrument.instrument_id=8;
delete from item where item.item_id=11;

/*delete item from shopping cart*/
delete from Sale_item where sale_item.sale=? && sale_item.item=?;

```

```

/*
    Delete record
*/
delete from record where record.Record_id=12;
delete from item where item_id=22;

select i.item_id from record
join public.item i on i.item_id = record.item_id
where record_id=11;

select record_id from record
join public.item i on i.item_id = record.item_id
where i.item_id=18;

select instrument_id from instrument
join public.item i on i.item_id = instrument.item_id
where i.item_id=2;

delete from instrument_istrument_type where instrument_id=1;
delete from instrument where instrument_id=1;
delete from sale_item where item=1;
delete from item where item_id=1;

```

За затриване на базата от данни и техните полета е използван следния скрипт

Drop.sql

```

DROP TABLE Client;
DROP TABLE Worker;
DROP TABLE Sale_item;
DROP TABLE Sale;
DROP TABLE User;
DROP TABLE Instrument_istrument_type;
DROP TABLE Record;
DROP TABLE Artist;
DROP TABLE Record_label;
DROP TABLE Instrument_type;
Drop TABLE Instrument;
DROP TABLE Item;
DROP TABLE Genre;

```

DROP TABLE Company;