Ресторанти
IFDb
Internet Food Database

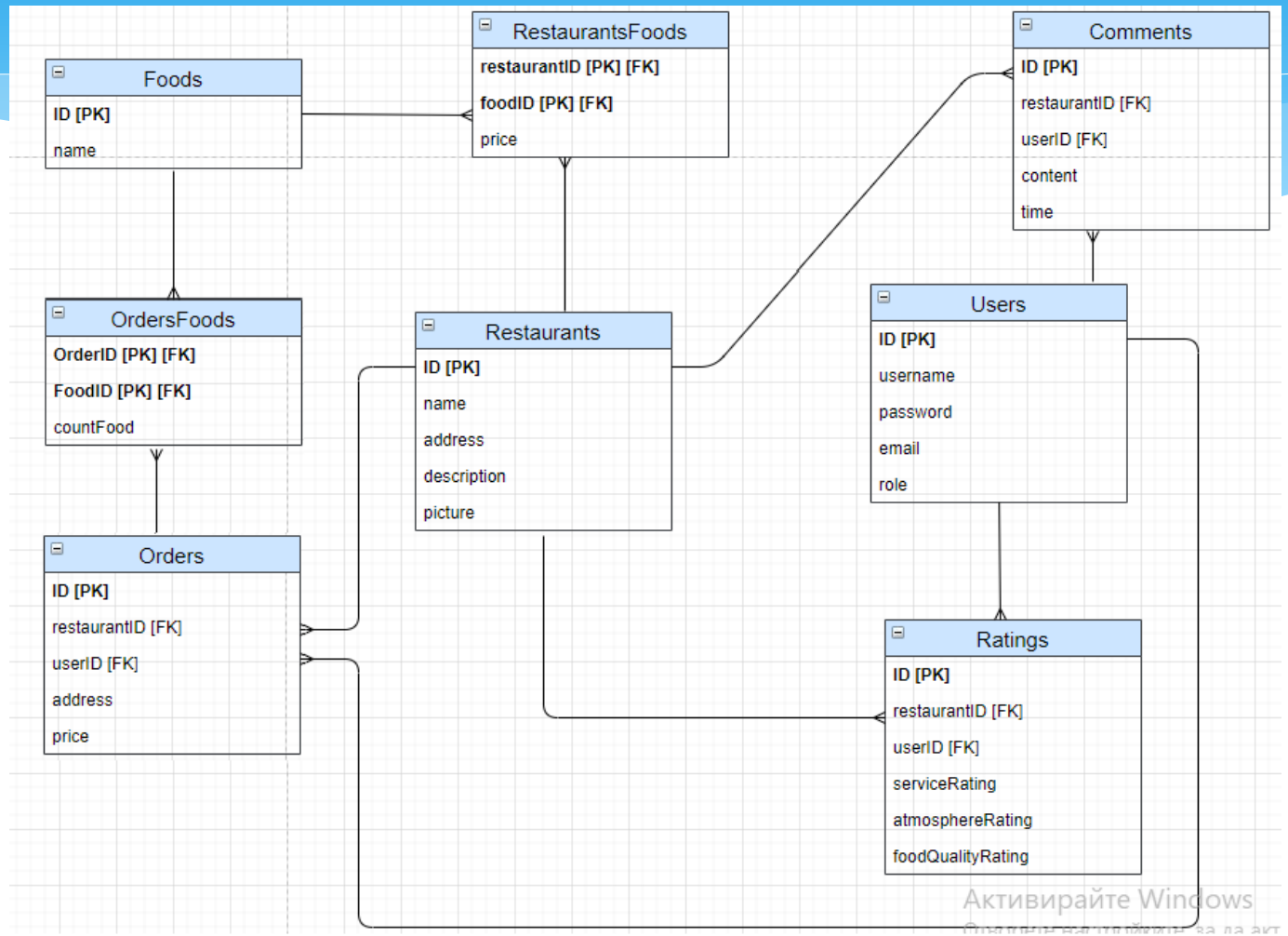
Изготвили:
Живко Георгиев, 62304
Васил Василев, 62348
Владислав Стефанов, 62349

Основна цел на проекта

* Основната цел на IFDb е да предоставя достъп до информация за ресторанти и дава начин за търсене по предлагани ястия.
* IFDb дава възможност на регистрирани потребители да поръчват храна онлайн, да дават оценки и коментари на даден ресторант.
* Допълнително всеки потребител може да има различни роли:
  * Admin
  * Moderator
  * Restaurant Page Manager,

Релационна схема

Решението за IFDb включва създаване на база данни с релационна схема:



Таблици

* Restaurants – съдържа информация за ресторантите
* Users – съдържа информация за потребителите
* Foods – показва различните храни, които предлагат ресторантите (може да се ползват при търсене като тагове)

Таблици

* Orders – съдържа информация за поръчките (от кой потребител до кой ресторант)
* Comments – показва информация за коментарите (от кой потребител за кой ресторант)
* Ratings - показва информация за оценките (от кой потребител за кой ресторант)
* OrdersFoods – дава информация за поръчаните храни в дадена поръчка
* RestaurantsFoods – предоставя информация за това кой ресторант какви ястия предлага (меню)

Заявки за дефиниране на релациите

```
CREATE TABLE Restaurants (
        ID INT PRIMARY KEY NOT NULL,
        name VARCHAR(256) NOT NULL,
        address VARCHAR(256) NOT NULL,
        description VARCHAR(256),
        picture VARCHAR(256)
)
```

Заявки за дефиниране на релациите

```
CREATE TABLE Users (
        ID INT PRIMARY KEY NOT NULL,
        username VARCHAR(256) NOT NULL,
        password VARCHAR(256) NOT NULL,
        email VARCHAR(256) NOT NULL,
        role VARCHAR(256) NOT NULL
)
CREATE TABLE RestaurantsFoods (
        restaurantID INT NOT NULL,
        foodID INT NOT NULL,
        price DECIMAL(6,2) NOT NULL DEFAULT 0
)
```

Заявки за добавяне на примерно съдържание

* `INSERT INTO USERS(ID, username, password, email, role) VALUES (1, 'vasilnv', 'alabalaa', 'vasil@gmail.com', 'User')`
* `INSERT INTO Restaurants(ID, name, address, description, picture) VALUES (1, 'Nacional', 'Studentski komplex 42', 'top food', NULL)`
* `INSERT INTO FOODS(ID, name) VALUES (1, 'Musaka')`
* `INSERT INTO ORDERS(ID, restaurantID, userID, address, price) VALUES (1, 1, 1, 'Studenski komplex №8', 10)`
* `INSERT INTO Comments(ID, restaurantID, userID, content, time) VALUES (1, 1, 1, 'fantastic food and staff', '2019-09-27')`
* `INSERT INTO Ratings(ID, restaurantID, userID, serviceRating, atmosphereRating, foodQualityRating) VALUES (1, 1, 1, 5, 5, 5)`

Добавяне на ограничения на ниво таблица

```
ALTER TABLE OrdersFoods
ADD CONSTRAINT pk_orders_foods PRIMARY KEY(orderID, foodId);
```

Един User може да влезе в системата като Admin, Moderator, Restaurant Page Manager или User

```
ALTER TABLE Users
ADD CONSTRAINT role
CHECK (role IN ('Admin', 'Moderator', 'Restaurant Page Manager', 'User'))
```

Добавяне на ограничения на ниво таблица

```
ALTER TABLE Ratings
ADD CONSTRAINT fk_to_users FOREIGN KEY(userID)
REFERENCES Users(ID)
ON DELETE CASCADE
```

Ресторантите се характеризират с уникална комбинация от име и адрес

```
ALTER TABLE Restaurants
ADD CONSTRAINT uq_name_address UNIQUE(name, address)
```

Прости заявки

Напишете заявка, която изкарва информацията за тези поръчки, чиято цена е по-голяма от 10 лева.

```
SELECT * FROM Orders WHERE price > 10
```

	ID	restaurantID	userID	address	price
1	2	2	3	Studenski komplex №8	12.00

Напишете заявка, която изкарва username и email за всеки един от потребителите

```
SELECT username as Name, email from Users
```

	Name	email
1	vasilnv	vasil@gmail.com
2	jivkomg	jivko@gmail.com
3	vladislav	vladi@gmail.com
4	gogo	gogo@abv.bg
5	jojo	jojo@gmail.com
6	pesho	pesho@gmail....

Прости заявки

Напишете заявка, която изкарва всичката информация за рейтингите, където рейтинга на обслужването и атмосферата е повече от 3, подредено възходящо по рейтинг на обслужването

```
SELECT * FROM Ratings WHERE servicerating > 3 AND
atmosphererating > 3 ORDER BY servicerating ASC
```

	ID	restaurantID	userID	serviceRating	atmosphereRating	foodQualityRating
1	5	4	4	4	5	5
2	1	1	1	5	5	5
3	2	1	1	5	5	5

Заявки върху две или повече релации

Напишете заявка, която изкарва различните имена и рейтинги на атмосферата, където рейтинга на атмосферата е 5.

```
SELECT DISTINCT name, atmosphererating FROM Ratings,
Restaurants WHERE atmosphererating = 5 AND
Ratings.restaurantID = Restaurants.id
```

Подзаявки

Напишете заявка, която изкарва ID на храната, ресторанта и цената, на която храната се предлага в ресторантите, чиито имена започват с N

```
SELECT foodID, restaurantID, price FROM RestaurantsFoods
WHERE restaurantid IN (SELECT id FROM Restaurants WHERE
name LIKE 'N%')
```

	foodID	restaurantID	price
1	1	1	12.00
2	1	2	13.00
3	2	1	5.00
4	3	1	6.00
5	4	2	7.00

Подзаявки

Напишете заявка, която изкарва най-евтината храна във всички ресторанти и нейното ID

```
SELECT foodID, price FROM RestaurantsFoods WHERE price <=
ALL (SELECT price FROM RestaurantsFoods)
```

	foodID	price
1	6	2.00

Примери със съединения

Напишете заявка, която изкарва информацията за потребителите и коментарите, които са написали

```
SELECT * FROM Comments c JOIN Users u ON c.userid = u.id
```

	ID	restaurantID	userID	content	time	ID	username	password	email	role
1	1	1	1	fantastic food and staff	2019-09-27	1	vasilnv	alabalaa	vasil@gmail.com	User
2	2	1	2	fantastic food and staff	2019-09-12	2	jivkomg	alabalal	jivko@gmail.com	User
3	3	2	3	fantastic food and staff	2019-09-01	3	vladislav	alablaal	vladi@gmail.com	User
4	4	3	1	fantastic food and staff	2019-12-05	1	vasilnv	alabalaa	vasil@gmail.com	User
5	5	4	2	fantastic food and staff	2019-06-15	2	jivkomg	alabalal	jivko@gmail.com	User

Напишете заявка, която изкарва информация за името и адреса на ресторантите и username на потребителите, оценили ги със средна оценка по-висока от 4

```
SELECT username, Restaurants.name, Restaurants.address FROM Ratings JOIN USERS ON Users.ID =
Ratings.userID JOIN Restaurants ON Ratings.restaurantID = Restaurants.ID WHERE
(serviceRating + atmosphereRating + foodQualityRating) / 3 >= 4
```

	username	name	address
1	vasilnv	Nacional	Studentski komplex 42
2	vasilnv	Nacional	Studentski komplex 42
3	vladislav	Nacional	Tsarigradsko
4	gogo	Shtastli...	bul. "Vitosha" 27

Групиране и агрегация

Напишете заявка, която изкарва информация за името на дадена храна и броят на ресторантите, които я предлагат, като записите са подредени низходящо по броя на ресторантите.

```
SELECT COUNT(rf.restaurantID) AS restaurantsCount, f.name AS foodName
FROM RestaurantsFoods AS rf
JOIN Foods AS f ON f.ID = rf.foodID
GROUP BY f.name
ORDER BY restaurantsCount DESC
```

	restaurantsCount	foodName
1	4	Musaka
2	2	Spaghetti Bolognese
3	2	Tarator
4	1	Soup
5	1	Bob
6	1	Meat rolls

Групиране и агрегация

Напишете заявка, която изкарва име и адрес на ресторантите, както и осреднения рейтинг от всички рейтинги, които е получил, подредени по осреднения рейтинг низходящо

```
SELECT AVG((serviceRating + atmosphereRating + foodQualityRating) / 3) AS
averageOverallRating, r.name AS restaurantName, r.address AS
restaurantAddress
FROM Ratings AS rat
JOIN Restaurants AS r ON restaurantID = r.ID
GROUP BY r.name, r.address
ORDER BY averageOverallRating DESC
```

	averageOverallRating	restaurantName	restaurantAddress
1	4	Shtastlivetsa	bul. "Vitosha" 27
2	4	Nacional	Studentski komplex 42
3	4	Nacional	Tsarigradsko
4	3	Talent's Rest...	Boulevard "Evlogi i ...

Групиране и агрегация

Напишете заявка, която изкарва броя на потребителите в системата

```
SELECT COUNT(ID) AS usersCount
FROM Users
```

	usersCount
1	6

Напишете заявка, която изкарва средния рейтинг за качеството на храната на всички ресторанти, които са получили рейтинг, подредени низходящо по среден рейтинг за качеството на храната

```
SELECT AVG(foodQualityRating) AS averageFoodQualityRating, r.name AS restaurantName,
r.address AS restaurantAddress
FROM Ratings AS rat
JOIN Restaurants AS r ON restaurantID = r.ID
GROUP BY r.name, r.address
ORDER BY averageFoodQualityRating DESC
```

	averageFoodQualityRating	restaurantName	restaurantAddress
1	5	Talent's Restaurant	Boulevard "Evlogi i Hristo Georgiev"
2	5	Shtastlivetsa	bul. "Vitosha" 27
3	5	Nacional	Studentski komplex 42
4	5	Nacional	Tsarigradsko

Изгледи, индекси, тригери

* За улесняване на някои заявки има изгледи за:
  * Поръчките
  * Коментарите
  * Оценките
* Създадени са индекси за увеличаване на бързината на част от заявките
* Добавени са тригери за автоматичното пресмятане на някои полета, което намалява броя на заявките, които се правят към базата данни и тяхната сложност

Индекси

```
CREATE INDEX idx_username
ON Users(username)

CREATE INDEX idx_restaurant__by_name
ON Restaurants(name, address)

CREATE INDEX idx_ratings_by_restaurants
ON Ratings(restaurantID, userID)
```

Изгледи

Изглед за това кой потребител от къде си е поръчал храна, на какъв адрес иска да бъде доставена и каква е цената на поръчката

```
CREATE VIEW dbo.orders_view AS
SELECT r.name AS restaurantName, r.address AS restaurantAddress, u.username
AS clientName, o.address AS clientAddress, o.price AS price
FROM Orders o
INNER JOIN Restaurants AS r ON r.ID = o.restaurantID
INNER JOIN Users AS u ON u.ID = o.userID
```

Изгледи

изглед на храната, която всеки ресторант предлага

```
CREATE VIEW dbo.restaurants_foods_view AS
SELECT r.name AS restaurantName, r.address AS
restaurantAddress, f.name AS foodName
FROM RestaurantsFoods AS rf
INNER JOIN Restaurants AS r ON r.ID = rf.restaurantID
INNER JOIN Foods AS f ON f.ID = rf.foodID
```

Тригери

тригер за автоматично задаване на датата при добавяне или промяна на коментар

```
CREATE TRIGGER comments_add_time
ON Comments
AFTER INSERT, UPDATE
AS
        UPDATE Comments
        SET time = GETDATE()
        WHERE ID IN (SELECT ID FROM inserted)
```

Тригери

```
CREATE TRIGGER orders_calculate_price
ON OrdersFoods
AFTER INSERT, DELETE, UPDATE
AS
        UPDATE o
        SET o.price = o.price + (
                SELECT COALESCE(SUM(rf.price * i.countFood), 0)
                FROM RestaurantsFoods AS rf
                JOIN inserted AS i ON rf.foodID = i.foodID
                WHERE o.restaurantID = rf.restaurantID AND o.ID = i.orderID
        ) - (
                SELECT COALESCE(SUM(rf.price * d.countFood), 0)
                FROM RestaurantsFoods AS rf
                JOIN deleted AS d ON rf.foodID = d.foodID
                WHERE o.restaurantID = rf.restaurantID AND o.ID = d.orderID
        )
        FROM Orders AS o
        WHERE o.ID IN (SELECT orderID FROM inserted UNION SELECT orderID FROM deleted)
```

тригер за изчисляване
на общата цена на дадена поръчка

Възможности за развитие

* Системата има възможности за развитие. Например може да не разчита на външна услуга да се занимава с поръчките, а самата система да го прави. Тогава биха се появили още таблици в базата от данни, като например доставчици и други.
* Схемата вероятно има възможност да бъде опростена, а също така и да бъде добавена нова функционалност, като например при поръчване на вече добавена храна към поръчката, count-а да се увеличава, а не да се отбелязва, че вече е направена тази поръчка.
* При бъдещи модификации на системата е възможно тя да поддържа повече от един потребител, използващи един и същ мейл или повече от един потребител с един и същ username.

Поуки и трудности

* Трудностите, които срещнахме бяха най-вече на ниво организация
* Като цяло можеше по-добре да обмислим първоначалното разпределение на задачите, за да не се налага в хода на правенето на проекта да правим промени
* Също така е една идея по-трудно да работим синхронизирано по проекта, използвайки zoom и facebook
* Можеше по-добре да обмислим още при дефинирането на релациите имената на колоните и техните ограничения, тъй като открихме, че цената на такива грешки впоследствие се оказва голяма по отношение на времето

Благодарим за вниманието!