Podstawy baz danych

Projekt:
System wspomagania firmy świadczącej usługi gastronomiczne

Jakub Szymczak, Szymon Budziak, Łukasz Wala

1. Propozycja funkcji realizowanych przez system

1.1 Menadżer restauracji:

1) wgląd i możliwość modyfikacji oferty/menu,
2) wgląd i możliwość modyfikacji listy pracowników,
3) ustalanie rabatów,
4) dostęp do raportów, faktur i statystyk generowanych przez system,
5) możliwość dodawania, usuwania oraz realizacji rezerwacji.

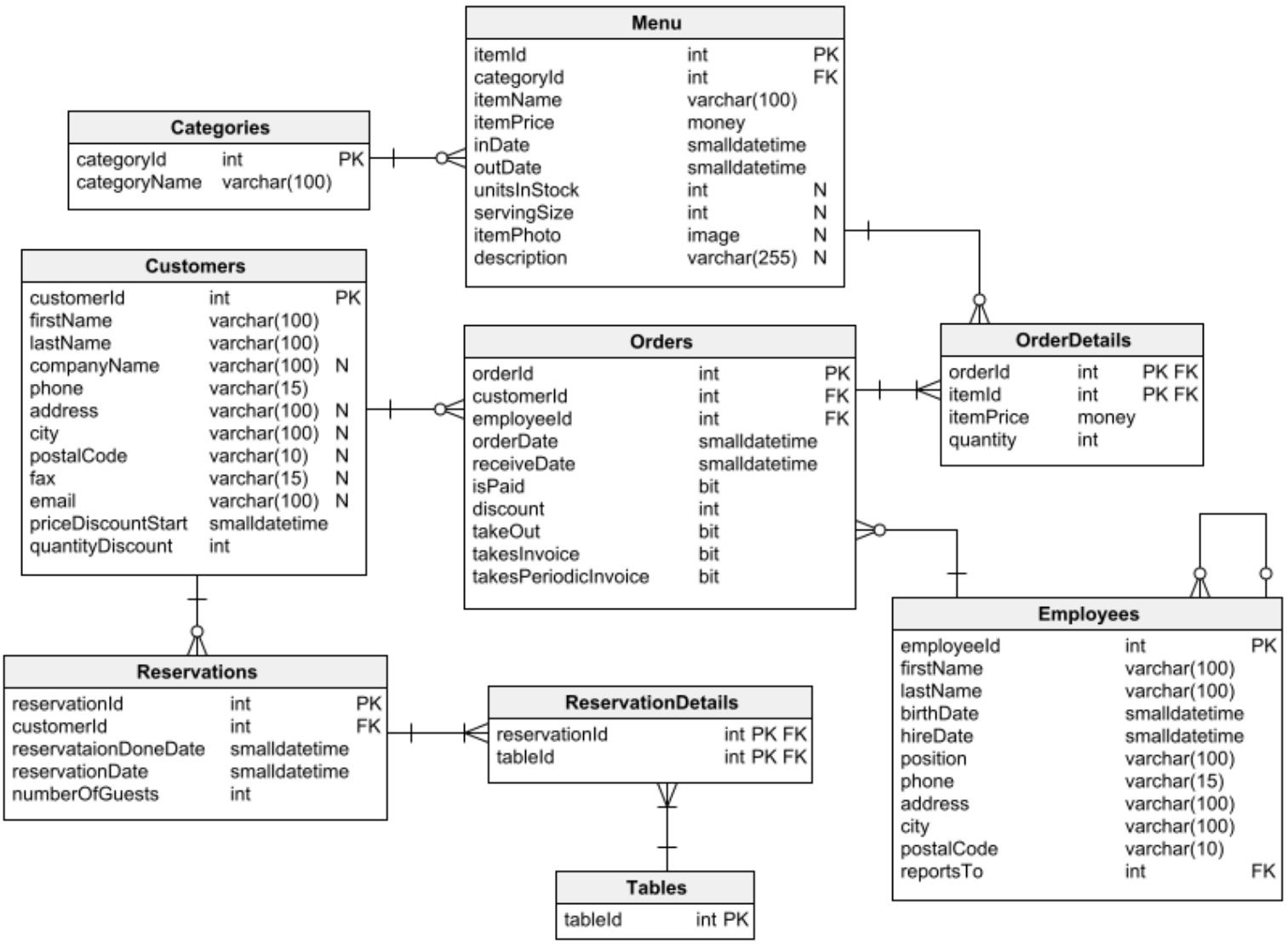
1.2 Kelner/ kasjer:

1) możliwość dodawania, usuwania oraz realizacji rezerwacji,
2) możliwość dodawania zamówień,
3) wgląd do menu.

1.3 Klient:

1) wgląd do menu,
2) możliwość rezerwacji stolików i zamówienia jedzenia na wynos.

2. Schemat bazy danych



3. Tabele

3.1 Tabela Categories

Tabela zawiera informacje na temat kategorii, do których należą elementy menu.

categoryId - identyfikator danej kategorii, autoinkrementowany,
categoryName - nazwa danej kategorii.

```
-- Table: Categories
CREATE TABLE Categories (
    categoryId int NOT NULL IDENTITY(1, 1),
    categoryName varchar(100)  NOT NULL,
    CONSTRAINT Categories_pk PRIMARY KEY  (categoryId)
);
```

3.2 Tabela Tables

Tabela przechowująca informacje o stolikach.

tableId - identyfikator stolika.

```
-- Table: Tables
CREATE TABLE Tables (
    tableId int  NOT NULL IDENTITY(1, 1),
    CONSTRAINT Tables_pk PRIMARY KEY  (tableId)
);
```

3.3 Tabela Customers

Tabela zawiera informacje na temat klientów firmy, pozwala na stwierdzenie, czy klient jest indywidualny, czy reprezentuje firmę, jego informacje kontaktowe.

customerId - identyfikator klienta, autoinkremetowany,
firstName, lastName - imię i nazwisko klienta,
companyName - nazwa firmy, jeżeli klient ją reprezentuje, NULL w innym przypadku,
phone, fax, email - numer telefonu, email i fax klienta,
address, city, postalCode - dane adresowe klienta,
priceDiscountStart - data rozpoczęcia działania rabatu drugiego typu,
quantityDiscount - wartość oznaczająca, czy klienta obejmuje rabat pierwszego typu.

```
-- Table: Customers
CREATE TABLE Customers (
    customerId int  NOT NULL IDENTITY(1, 1),
    firstName varchar(100)  NOT NULL,
    lastName varchar(100)  NOT NULL,
    companyName varchar(100)  NULL,
    phone varchar(15)  NOT NULL,
    address varchar(100)  NULL,
    city varchar(100)  NULL,
    postalCode varchar(10)  NULL,
    fax varchar(15)  NULL CHECK (fax IS NULL OR LEN(fax)
= 10),
    email varchar(100)  NULL CHECK (email IS NULL OR
email LIKE '%_@%_.%_'),
    priceDiscountStart smalldatetime  NOT NULL,
    quantityDiscount decimal(4, 4)  NOT NULL  CHECK
(quantityDiscount BETWEEN 0 AND 1.0),
    CONSTRAINT Customers_pk PRIMARY KEY  (customerId)
);
```

3.4 Tabela Employees

Tabela zawiera informacje na temat pracowników firmy, ich dane kontaktowe i adresowe oraz to, komu podlegają.

employeeId - identyfikator pracownika,
firstName, lastName - imię i nazwisko pracownika,
birthDate - data urodzenia pracownika,
hireDate - data przyjęcia pracownika do pracy,
position - stanowisko pracownika w firmie,
address, city, postalCode - dane adresowe pracownika,
reportsTo - pracownik, który nadzoruje danego pracownika.

```
-- Table: Employees
CREATE TABLE Employees (
    employeeId int  NOT NULL IDENTITY(1, 1),
    firstName varchar(100)  NOT NULL,
    lastName varchar(100)  NOT NULL,
    birthDate smalldatetime  NOT NULL CHECK (birthDate
BETWEEN '19000101' AND GETDATE()),
    hireDate smalldatetime  NOT NULL CHECK (hireDate <
GETDATE()),
    position varchar(100)  NOT NULL,
    phone varchar(15)  NOT NULL,
    address varchar(100)  NOT NULL,
    city varchar(100)  NOT NULL,
    postalCode varchar(10)  NOT NULL,
    reportsTo int  NOT NULL,
    CONSTRAINT Employees_pk PRIMARY KEY  (employeeId)
);

-- Reference: Employees_Employees (table: Employees)
ALTER TABLE Employees ADD CONSTRAINT Employees_Employees
    FOREIGN KEY (reportsTo)
    REFERENCES Employees (employeeId);
```

3.5 Tabela Menu

Tabela przechowuje dane o produktach zawartych w menu.

itemId - identyfikator danego produktu.
categoryId - identyfikator kategorii, do której należy produkt,
itemName - nazwa danego produktu,
itemPrice - cena danego produktu,
inDate - data, od której dany produkt jest w obecnym menu restauracji,
outDate - data, do której dany produkt jest w obecnym menu restauracji,
unitsInStock - ilość produktu obecna w magazynie,
servingSize - wielkość produktu (waga/objętość),
itemPhoto - zdjęcie danego produktu,
description - opis danego produktu.

```
-- Table: Menu
CREATE TABLE Menu (
    itemId int  NOT NULL IDENTITY(1, 1),
    categoryId int  NOT NULL,
    itemName varchar(100)  NOT NULL,
    itemPrice money  NOT NULL CHECK (itemPrice > 0),
    inDate smalldatetime  NOT NULL,
    outDate smalldatetime  NOT NULL CHECK (outDate >
GETDATE()),
    unitsInStock int  NULL CHECK (unitsInStock IS NULL
OR unitsInStock >= 0),
    servingSize int  NULL CHECK (unitsInStock IS NULL OR
servingSize > 0),
    itemPhoto image  NULL,
    description varchar(255)  NULL,
    CONSTRAINT Menu_pk PRIMARY KEY  (itemId)
);

-- Reference: Menu_Categories (table: Menu)
ALTER TABLE Menu ADD CONSTRAINT Menu_Categories
    FOREIGN KEY (categoryId)
    REFERENCES Categories (categoryId);
```

```
-- Reference: Menu_OrderDetails (table: OrderDetails)
ALTER TABLE OrderDetails ADD CONSTRAINT Menu_OrderDetails
    FOREIGN KEY (itemId)
    REFERENCES Menu (itemId);
```

3.6 Tabela Orders

Tabela przechowuje informacje o zamówieniach wykonanych przez klientów. Dzięki niej jesteśmy w stanie dowiedzieć się o szczegółach danego zamówienia, które zostało złożone przez danego klienta, np. jaki pracownik je obsłużył, data zamówienia, czy odbiera zamówienie na wynos oraz czy odbiera fakturę lub paragon.

orderId - identyfikator zamówienia, wartość autoinkrementowana,
customerId - identyfikator klienta,
employeeId - identyfikator pracownika,
orderDate - data złożenia zamówienia,
receiveDate - data odebrania zamówienia,
isPaid - wartość identyfikująca czy zamówienie zostało złożone,
discount - procent jaki został nadany klientowi na całe zamówienie,
takeOut - wartość identyfikująca czy klient odbiera zamówienie na wynos,
takesInvoice - wartość identyfikująca czy klient odbiera fakturę czy paragon,
takesPeriodicInvoice - wartość identyfikująca czy klient odbiera okresową fakturę czy paragon,

```
-- Table: Orders
CREATE TABLE Orders (
      orderId int  NOT NULL IDENTITY(1, 1),
      customerId int  NOT NULL,
      employeeId int  NOT NULL,
      orderDate smalldatetime  NOT NULL CHECK (orderDate <= GETDATE()),
      receiveDate smalldatetime  NOT NULL,
      isPaid bit  NOT NULL,
      discount int  NOT NULL CHECK (discount BETWEEN 0 AND 1.0),
      takeOut bit  NOT NULL,
      takesInvoice bit  NOT NULL,
      takesPeriodicInvoice bit  NOT NULL,
      CONSTRAINT Orders_pk PRIMARY KEY  (orderId)
);
```

```
-- Reference: Orders_Customers (table: Orders)
ALTER TABLE Orders ADD CONSTRAINT Orders_Customers
    FOREIGN KEY (customerId)
    REFERENCES Customers (customerId);
-- Reference: Orders_Employees (table: Orders)
ALTER TABLE Orders ADD CONSTRAINT Orders_Employees
    FOREIGN KEY (employeeId)
    REFERENCES Employees (employeeId);
```

3.7 Tabela OrderDetails

Tabela przechowująca informacje o szczegółach danego zamówienia.

orderId - identyfikator zamówienia,
itemId - identyfikator produktu,
itemPrice - cena produktu,
quantity - ilość danego produktu.

```
-- Table: OrderDetails
CREATE TABLE OrderDetails (
     orderId int  NOT NULL IDENTITY(1, 1),
     itemId int  NOT NULL,
     itemPrice money  NOT NULL CHECK (itemPrice > 0),
     quantity int  NOT NULL CHECK (quantity > 0),
     CONSTRAINT OrderDetails_pk PRIMARY KEY
(orderId,itemId)
);

-- Reference: OrderDetails_Orders (table: OrderDetails)
ALTER TABLE OrderDetails ADD CONSTRAINT
OrderDetails_Orders
    FOREIGN KEY (orderId)
    REFERENCES Orders (orderId);
```

3.8 Tabela Reservations

Tabela przechowuje informacje dotyczące rezerwacji dokonanych przez klientów.

reservationId - identyfikator danej rezerwacji,
customerId - identyfikator klienta, który dokonał danej rezerwacji,
reservationDoneDate - data, w której klient dokonał rezerwacji,
reservationDate - data, na którą klient dokonał rezerwacji,
numberOfGuests - ilość gości wchodzących na daną rezerwację.

```
-- Table: Reservations
CREATE TABLE Reservations (
    reservationId int  NOT NULL IDENTITY(1, 1),
    customerId int  NOT NULL,
    reservationDoneDate smalldatetime  NOT NULL CHECK
(reservationDoneDate <= GETDATE()),
    reservationDate smalldatetime  NOT NULL,
    numberOfGuests int  NOT NULL,
    CONSTRAINT Reservations_pk PRIMARY KEY
(reservationId)
);

-- Reference: Reservations_Customers (table:
Reservations)
ALTER TABLE Reservations ADD CONSTRAINT
Reservations_Customers
    FOREIGN KEY (customerId)
    REFERENCES Customers (customerId);
```

3.9 Tabela ReservationDetails

Tabela przechowuje detale rezerwacji zawartych w tabeli Reservations.

reservationId - identyfikator danej rezerwacji,
tableId - numer identyfikacyjny stołu, na który dokonana jest rezerwacja.

```
-- Table: ReservationDetails
CREATE TABLE ReservationDetails (
    reservationId int  NOT NULL,
    tableId int  NOT NULL,
    CONSTRAINT ReservationDetails_pk PRIMARY KEY
(reservationId,tableId)
);

-- Reference: ReservationDetails_Reservations (table:
ReservationDetails)
ALTER TABLE ReservationDetails ADD CONSTRAINT
ReservationDetails_Reservations
    FOREIGN KEY (reservationId)
    REFERENCES Reservations (reservationId);

-- Reference: ReservationDetails_Tables (table:
ReservationDetails)
ALTER TABLE ReservationDetails ADD CONSTRAINT
ReservationDetails_Tables
    FOREIGN KEY (tableId)
    REFERENCES Tables (tableId);
```