System zarządzania restauracją

Database model documentation

Spis treści:

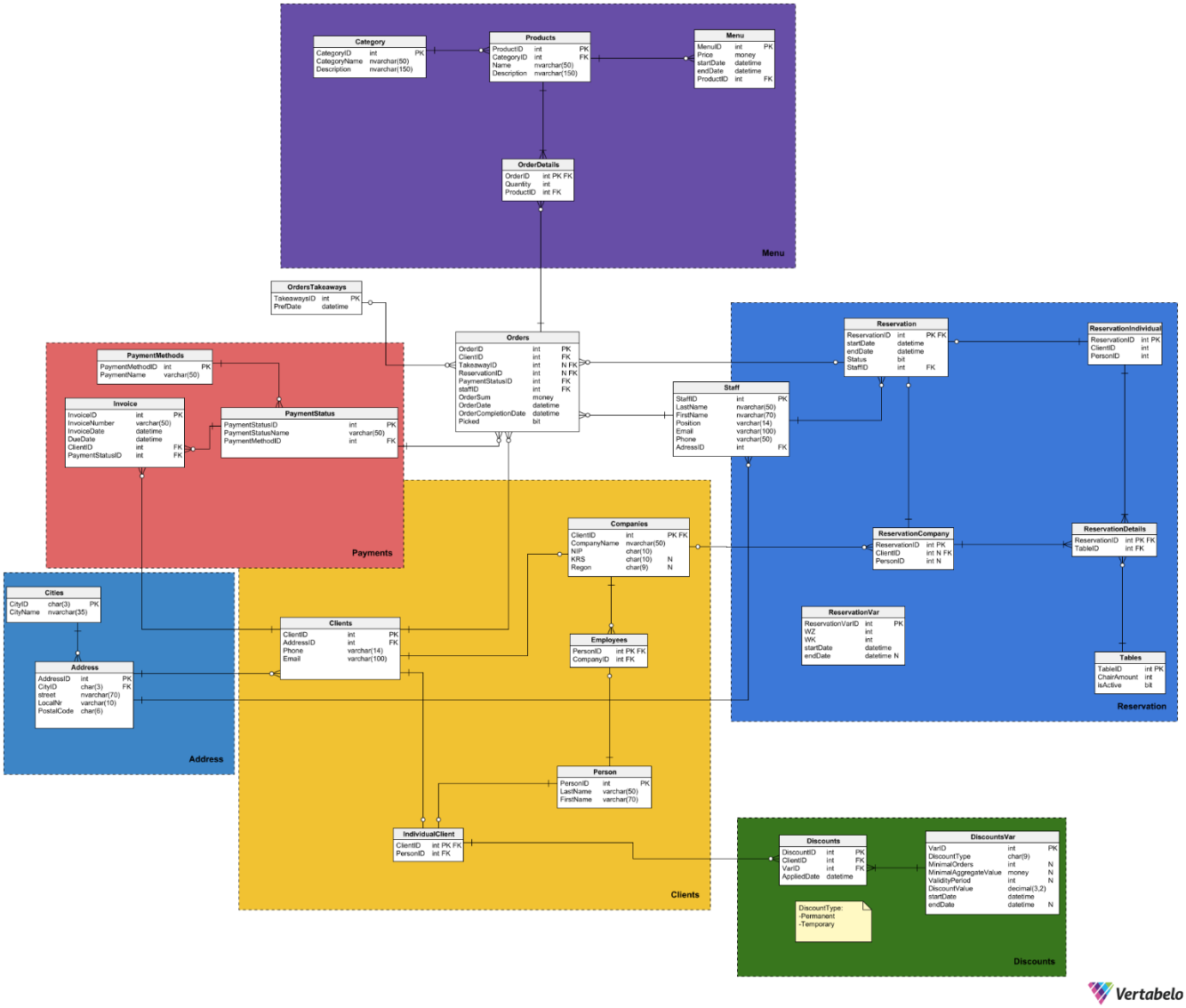
1. Opis modelu

Model name: `System zarządzania restauracją`

Version: `2.4`

Database engine: `Microsoft SQL Server`

1.1. Schemat bazy



2. Opisy Tabel

2.1. Tabela Orders

Przechowuje informacje o zamówieniach.

- Klucz główny: OrderID
- Klucze obce: ClientID (do tabeli Clients), TakeawayID (do tabeli OrdersTakeaways), ReservationID (do tabeli Reservation), PaymenttStatusID (do tabeli PaymentStatus), staffID (do tabeli Staff) , PaymentMethodID(do tabeli PaymentMethods)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
OrderID	int	Not null	Numer ID zamówienia
ClientID	int	Not null	Numer ID klienta
TakeawayID	int	null	Numer ID na wynos
InvoiceID	Int	Null	Numer Faktury do zamówienia
ReservationID	int	null	Numer ID rezerwacji
PaymentStatusID	int	Not null	Numer ID statusu zapłaty
PaymentMethodID	int	Not null	Numer ID metody płatności
staffID	int	Not null	Numer ID personelu restauracji
OrderSum	money	Not null	Wartość zamówienia
OrderDate	datetime	Not null	Data złożenia zamówienia
OrderCompletionDate	datetime	null	Data skompletowania zamówienia
OrderStatus	varchar(15)	Not null	Status zamówienia - Pending - Accepted - Completed - Denied - Picked - Cancelled

```
CREATE TABLE Orders (
    OrderID int  NOT NULL IDENTITY (1,1),
    ClientID int  NOT NULL,
    TakeawayID int  NULL,
    InvoiceID int  NULL,
    ReservationID int  NULL,
    PaymentStatusID int  NOT NULL,
    PaymentMethodID int  NOT NULL,
    staffID int  NOT NULL,
    OrderSum money  NOT NULL check ( OrderSum > 0 ),
    OrderDate datetime  NOT NULL default getdate(),
    OrderCompletionDate datetime  NULL ,
    OrderStatus varchar(15) NOT NULL
check (OrderStatus in ('pending', 'accepted', 'completed', 'denied', 'picked',
'cancelled')),
    CONSTRAINT validDateOrders check ( (OrderCompletionDate >= OrderDate)  or (Or-
derCompletionDate is null)),
    CONSTRAINT Orders_pk PRIMARY KEY  (OrderID)
);
```

2.2. Tabela OrdersTakeaways

Przechowuje informacje o zamówieniu na wynos

- Klucz główny: TakeawaysID

Nazwa kolumny	Typy Danych	Czy null	Co przechowuje
TakeawaysID	int	Not null	Numer ID zamówienia na wynos
PrefDate	datetime	Not null	Preferowaną date odbioru zamówienia

```
CREATE TABLE OrdersTakeaways (
    TakeawaysID int  NOT NULL IDENTITY (1,1),
    PrefDate datetime  NOT NULL check (PrefDate >= getdate()),
    CONSTRAINT OrdersTakeaways_pk PRIMARY KEY  (TakeawaysID)
);
```

2.3. Tabela Reservation

Przechowuje informacje o aktualnych rezerwacjach

- Klucz główny: ReservationID
- Klucze obce: StaffID (do tabeli Staff)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ReservationID	int	Not null	Numer ID

			rezerwacji
startDate	datetime	Not null	Data rozpoczęcia rezerwacji
endDate	datetime	Not null	Data zakończenia rezerwacji
Status	Varchar(15)	Not null	Status danej rezerwacji - Pending - Accepted - Denied - Cancelled - Waiting
StaffID	int	Not null	Numer ID personelu

```
CREATE TABLE Reservation (
    ReservationID int  NOT NULL  IDENTITY (1,1),
    startDate datetime  NOT NULL,
    endDate datetime  NOT NULL ,
    Status varchar(15)  NOT NULL default 'waiting',
    StaffID int  NOT NULL,
    constraint validStatus
check (Status in ('pending', 'accepted', 'denied', 'cancelled', 'waiting')),
    CONSTRAINT validDateReservation  check(startDate < endDate),
    CONSTRAINT Reservation_pk PRIMARY KEY  (ReservationID)
);
```

2.4. Tabela ReservationIndividual

Przechowuje informacje o rezerwacjach osób indywidualnych

- Klucz główny: ReservationID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ReservationID	int	Not null	Numer ID rezerwacji
ClientID	int	Not null	Numer ID klienta
PersonID	int	Not null	Numer ID osoby

```
CREATE TABLE ReservationIndividual (
    ReservationID int  NOT NULL,
    ClientID int  NOT NULL,
    PersonID int  NOT NULL,
    CONSTRAINT ReservationIndividual_pk PRIMARY KEY  (ReservationID)
);
```

2.5. Tabela ReservationCompany

Przechowuje informacje o rezerwacjach firmowych

- Klucz główny: ReservationID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ReservationID	int	Not null	Numer ID rezerwacji
ClientID	int	Not null	Numer ID klienta
PersonID	int	Not null	Numer ID osoby

```
CREATE TABLE ReservationCompany (
    ReservationID int  NOT NULL,
    ClientID int  NULL,
    PersonID int  NULL,
    CONSTRAINT ReservationCompany_pk PRIMARY KEY  (ReservationID)
);
```

2.6. Tabela ReservationDetails

Łączy rezerwacje z stolikami dla nich

- Klucz główny: ReservationID
- Klucze obce: TableID (do tabeli Tables)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ReservationID	int	Not null	Numer ID rezerwacjio
TableID	int	Not null	Numer ID stolika

```
CREATE TABLE ReservationDetails (
    ReservationID int  NOT NULL,
    TableID int  NOT NULL,
    CONSTRAINT ReservationDetails_pk PRIMARY KEY  (ReservationID)
);
```

2.7. Tabela Tables

Przechowuje informacje o stolikach

- Klucz główny: TableID
- Klucze obce: TableID (do tabeli Tables)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
TableID	int	Not null	Numer ID stolika
ChairAmount	int	Not null	Liczbe krzeseł
isActive	bit	Not null	Czy stolik jest aktywny. Czy nie stoi na zapleczu

```
CREATE TABLE Tables (
    TableID int  NOT NULL,
    ChairAmount int  NOT NULL check (ChairAmount >= 2),
    isActive bit  NOT NULL default 1,
    CONSTRAINT Tables_pk PRIMARY KEY  (TableID)
);
```

2.8. Tabela ReservationVar

Przechowuje informacje o zmiennych do rezerwacji

- Klucz główny: ReservationVarID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ReservationVarID	int	Not null	Numer ID zmiennej dotyczącej rezerwacji
WZ	int	Not null	Minimalna liczba zamówień potrzebna do rezerwację
WK	int	Not null	Minimalna kwota potrzebna do

			rezerwację
startDate	datetime	Not null	Data obowiązywania zmiennej
endDate	datetime	null	Data zakończenia obowiązywania zmiennej

```
CREATE TABLE ReservationVar (
    ReservationVarID int  NOT NULL IDENTITY (1,1),
    WZ int  NOT NULL check ( WZ > 0 ),
    WK int  NOT NULL check (WK > 0),
    startDate datetime  NOT NULL,
    endDate datetime  NULL,
    CONSTRAINT check(startDate < endDate or endDate is NULL),
    CONSTRAINT ReservationVar_pk PRIMARY KEY  (ReservationVarID)
);
```

2.9. Tabela Clients

Przechowuje informacje o klientach

- Klucz główny: ClientID
- Klucze obce: AddressID (do tabeli Address)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ClientID	int	Not null	Numer ID klienta
AddressID	int	Not null	Numer ID adressów
Phone	varchar(14)	Not null	Numer telefonu
Email	varchar(100)	Not null	Adres email

```
CREATE TABLE Clients (
    ClientID int  NOT NULL IDENTITY (1,1),
    AddressID int  NOT NULL,
    Phone varchar(14)  NOT NULL UNIQUE
                                check (isnumeric(Phone) = 1 and len(Phone) >= 9),
    Email varchar(100)  NOT NULL UNIQUE check( Email like '%[@]%[.]%'),
    CONSTRAINT Clients_pk PRIMARY KEY  (ClientID)
);
```

2.10. Tabela Companies

Przechowuje informacje o klientach firmowych

- Klucz główny: ClientID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ClientID	int	Not null	Numer ID klienta
CompanyName	nvarchar(50)	Not null	Nazwe firmy
NIP	char(10)	Not null	Numer NIP
KRS	char(10)	null	Numer KRS
Regon	char(9)	null	Numer REGON

```
CREATE TABLE Companies (
    ClientID int  NOT NULL,
    CompanyName nvarchar(50)  NOT NULL UNIQUE,
    NIP char(10)  NOT NULL UNIQUE check(NIP like '[0-9][0-9][0-9][0-9][0-9][0-9][0-
9][0-9][0-9][0-9]'),
    KRS char(10)  NULL UNIQUE check(KRS like '[0-9][0-9][0-9][0-9][0-9][0-9][0-
9][0-9][0-9][0-9]'),
    Regon char(9)  NULL UNIQUE check(Regon like '[0-9][0-9][0-9][0-9][0-9][0-9][0-
9][0-9][0-9]'),
    CONSTRAINT Companies_pk PRIMARY KEY  (ClientID)
);
```

2.11. Tabela Employees

Przechowuje informacje o pracownikach danej firmy

- Klucz główny: PersonID
- Klucze obce: CompanyID (do tabeli Companies), PersonID (do tabeli Employees)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
PersonID	int	Not null	Numer ID osoby
CompanyID	int	Not null	Numer ID Firmy

```
CREATE TABLE Employees (
    PersonID int  NOT NULL,
    CompanyID int  NOT NULL,
    CONSTRAINT Employees_pk PRIMARY KEY  (PersonID)
);
```

2.12. Tabela Person

Przechowuje informacje o osobach

- Klucz główny: PersonID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
PersonID	int	Not null	Numer ID osoby
LastName	varchar(50)	Not null	Imie osoby
FirstName	varchar(70)	Not null	Nazwisko osoby

```
CREATE TABLE Person (
    PersonID int  NOT NULL,
    LastName varchar(50)  NOT NULL,
    FirstName varchar(70)  NOT NULL,
    CONSTRAINT Person_pk PRIMARY KEY  (PersonID)
);
```

2.13. Tabela IndividualClient

Przechowuje informacje o klientach indywidualnych

- Klucz główny: ClientID
- Klucze obce: PersonID (do tabeli Person)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ClientID	int	Not null	Numer ID klienta
PersonID	int	Not null	Numer ID osoby

```
CREATE TABLE IndividualClient (
    ClientID int  NOT NULL,
    PersonID int  NOT NULL,
    CONSTRAINT IndividualClient_pk PRIMARY KEY  (ClientID)
);
```

2.14. Tabela Category

Przechowuje informacje o kategoriach produktów

- Klucz główny: CategoryID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
CategoryID	int	Not null	Numer ID kategorii
CategoryName	nvarchar(50)	Not null	Nazwe kategorii
Description	nvarchar(150)	Not null	Opis Kategorii

```
CREATE TABLE Category (
    CategoryID int  NOT NULL IDENTITY (1,1),
    CategoryName nvarchar(50)  NOT NULL,
    Description nvarchar(150)  NOT NULL,
    CONSTRAINT Category_pk PRIMARY KEY  (CategoryID)
);
```

2.15. Tabela Products

Przechowuje informacje o produktach oferowanych przez restauracje

- Klucz główny: ProductID
- Klucze obce: CategoryID (do tabeli Category)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ProductID	int	Not null	Numer ID produktu
CategoryID	int	Not null	Numer ID kategorii
Name	nvarchar(50)	Not null	Nazwe produktu
Description	nvarchar(150)	Not null	Opis produktu
IsAvailable	Bit	Not null	Mówimy o tym czy dany produkt jest dostępny. Chodzi o produkty np sezonowe

```
CREATE TABLE Products (
    ProductID int  NOT NULL IDENTITY (1,1),
    CategoryID int  NOT NULL,
    Name nvarchar(50)  NOT NULL,
    Description nvarchar(150)  NOT NULL default 'brak opisu' ,
    IsAvailable bit NOT NULL default 1,
    CONSTRAINT Products_pk PRIMARY KEY  (ProductID)
);
```

2.16. Tabela Menu

Przechowuje informacje o menu oferowanym przez restauracje w danym okresie

- Klucz główny: ID
- Klucze obce: ProductID (do tabeli Products)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
ID	int	Not null	Klucz główny
MenuID	int	Not null	Numer ID menu
Price	money	Not null	Cene danego produktu w danym nemu
startDate	datetime	Not null	Data od kiedy obowiązuje menu
endDate	datetime	null	Data do kiedy obowiązuje menu. Może być tak, że menu nie ma końca obowiązywania
ProductID	int	Not null	Numer ID produktu

```
CREATE TABLE Menu (
    ID int IDENTITY (1,1),
    MenuID int  NOT NULL ,
    Price money  NOT NULL check ( Price > 0 ),
    startDate datetime  NOT NULL default getdate(),
    endDate datetime NULL ,
    ProductID int  NOT NULL,
    CONSTRAINT validDateMenu
        check((dateadd(day, 14 ,startDate) < endDate and endDate is not null)
        or endDate is null),
    CONSTRAINT Menu_pk PRIMARY KEY  (ID)
);
```

2.17. Tabela OrderDetails

Przechowuje informacje o szczegółach danego zamówienia

- Klucz główny: OrderID
- Klucze obce: ProductID (do tabeli Products), OrderID (do tabeli Orders)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
OrderID	int	Not null	Numer ID zamówienia

Quantity	int	Not null	Ilość danego produktu w danym zamówieniu
ProductID	int	Not null	Numer ID produktu

```
CREATE TABLE OrderDetails (
    OrderID int  NOT NULL,
    Quantity int  NOT NULL check ( Quantity > 0 ),
    ProductID int  NOT NULL,
    CONSTRAINT OrderDetails_pk PRIMARY KEY  (OrderID)
);
```

2.18. Tabela Discounts

Przechowuje informacje o zniżka dla klientów indywidualnych

- Klucz główny: DiscountID
- Klucze obce: ClientID (do tabeli IndividualClient), VarID (do tabeli DiscountsVar)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
DiscountID	int	Not null	Numer ID zniżki
ClientID	int	Not null	Numer ID klienta
VarID	int	Not null	Numer ID zmiennych dotyczących tego zamówienia
AppliedDate	datetime	Not null	Data od kiedy obowiązuje zniżka
isUsed	Bit	Null	Jeśli jest to jednorazowa zniżka to posiada wartości czy jest już nałożona na klienta

```
CREATE TABLE Discounts (
    DiscountID int  NOT NULL IDENTITY (1,1),
    ClientID int  NOT NULL,
    VarID int  NOT NULL,
    AppliedDate datetime  NOT NULL ,
    isUsed bit NULL default 0,
    CONSTRAINT Discounts_pk PRIMARY KEY  (DiscountID)
);
```

2.19. Tabela DiscountsVar

Przechowuje informacje o zmiennych dotyczących zniżek dla klientów indywidualnych

- Klucz główny: VarID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
VarID	int	Not null	Numer ID zmiennej
DiscountType	char(9)	Not null	Typ zniżki: - Tymczasowa - Trwała
MinimalOrders	int	null	Najmniższa liczba zamówień aby zniżka zaczęła obowiązywać. Dotyczy zniżki tymczasowej
MinimalAggregateValue	money	null	Najmniższa sumaryczna kwota wydana na zamówienia
ValidityPeriod	int	null	Ilość dni w jakich dotyczy zniżka. Dotyczy zniżki tymczasowej
DiscountValue	decimal(3,2)	Not null	Wartość zniżki
startDate	datetime	Not null	Data od kiedy dane zmienne obowiązywały
endDate	datetime	null	Data kiedy zmienne skończyły obowiązywać

```
CREATE TABLE DiscountsVar (
    VarID int  NOT NULL IDENTITY (1,1),
    DiscountType char(9)  NOT NULL
                        check (DiscountType in ('Permanent', 'Temporary')),
    MinimalOrders int  NULL,
    MinimalAggregateValue money  NULL,
    ValidityPeriod int  NULL,
    DiscountValue decimal(3,2)  NOT NULL
                        CHECK ( DiscountValue >= 0 and DiscountValue <= 1 ),
    startDate datetime  NOT NULL DEFAULT getdate(),
    endDate datetime  NULL check(endDate IS NULL or startDate < endDate),
    CONSTRAINT DiscountsVar_pk PRIMARY KEY  (VarID)
);
```

2.20. Tabela PaymentStatus

Przechowuje informacje o statusie opłat zamówienia

- Klucz główny: PaymentStatusID

- Klucze obce: PaymentMethodID (dotyczy tabeli PaymentMethods)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
PaymentStatusID	int	Not null	Numer ID statusu płatności zamówienia
PaymentStatusName	varchar(50)	Not null	Nazwa statusu płatności

```
-- Table: PaymentStatus
CREATE TABLE PaymentStatus (
    PaymentStatusID int  NOT NULL IDENTITY (1,1),
    PaymentStatusName varchar(50)  NOT NULL default 'Unpaid',
    CONSTRAINT PaymentStatus_pk PRIMARY KEY  (PaymentStatusID)
);
```

2.21. Tabela Invoice

Przechowuje informacje o fakturach

- Klucz główny: InvoiceID
- Klucze obce: PaymentStatusID (dotyczy tabeli PaymentStatus), ClientID (dotyczy tabeli Clients), PaymentMethodID(do tabeli PaymentMethod)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
InvoiceID	int	Not null	Numer ID fakttury
InvoiceNumber	varchar(50)	Not null	Numer faktury
InvoiceDate	datetime	Not null	Data wystawienia faktury
DueDate	datetime	Not null	Termin zapłaty
ClientID	int	Not null	Numer ID klienta
PaymentStatusID	int	Not null	Numer ID statusu płatności faktury
PaymentMethodID	int	Not null	Numer ID metody płatności faktury

```
CREATE TABLE Invoice (
    InvoiceID int  NOT NULL IDENTITY (1,1),
    InvoiceNumber varchar(50)  NOT NULL UNIQUE,
    InvoiceDate datetime  NOT NULL,
    DueDate datetime  NOT NULL,
    ClientID int  NOT NULL,
    PaymentStatusID int  NOT NULL,
    CONSTRAINT Invoice_pk PRIMARY KEY  (InvoiceID)
);
```

2.22. Tabela Cities

Przechowuje informacje o miastach

- Klucz główny: CityID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
CityID	char(3)	Not null	Numer ID miasta. Trzy pierwsze litery miasta. Jak już istnieje to bierze kolejne.
CityName	nvarchar(35)	Not null	Nazwa miasta

```
CREATE TABLE Cities (
    CityID INT  NOT NULL IDENTITY (1,1),
    CityName nvarchar(35)  NOT NULL,
    CONSTRAINT Cities_pk PRIMARY KEY  (CityID)
);
```

2.23. Tabela Address

Przechowuje informacje o Adresach

- Klucz główny: AddressID
- Klucze obce: CityID (dotyczy tabeli Cities)

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
AddressID	int	Not null	Numer ID adresu
CityID	char(3)	Not null	Numer ID miasta. Opis w tabeli Cities.
street	nvarchar(70)	Not null	Nazwa ulicy

LocalNr	varchar(10)	Not null	Numer Domu wraz z np. 10A itp.
PostalCode	char(6)	Not null	Kod pocztowy w postaci XX-XXX.

```
CREATE TABLE Address (
    AddressID int  NOT NULL IDENTITY (1,1),
    CityID INT NOT NULL,
    street nvarchar(70)  NOT NULL,
    LocalNr varchar(10)  NOT NULL check(localNr like '[0-9]%'),
    PostalCode char(6)  NOT NULL
                check(PostalCode like '[0-9][0-9]-[0-9][0-9][0-9]'),
    CONSTRAINT Address_pk PRIMARY KEY  (AddressID)
);
```

2.24. Tabela PaymentMethods

Przechowuje informacje o metodach płatności

- Klucz główny: PaymentMethodID

Nazwa kolumny	Typy danych	Czy null	Co przechowuje
PaymentMethodID	int	Not null	Numer ID metody płatności
PaymentName	varchar(50)	Not null	Nazwa metody

```
CREATE TABLE PaymentMethods (
    PaymentMethodID int  NOT NULL IDENTITY (1,1),
    PaymentName varchar(50)  NOT NULL,
    CONSTRAINT PaymentMethods_pk PRIMARY KEY  (PaymentMethodID)
);
```

2.25. Table Staff

Przechowuje informacje o personelu

- Klucz główny: `StaffID`

Column name	Type	Properties	Description
`StaffID`	`int`	Not null	Numer ID pracownika
`LastName`	`nvarchar(50)`	Not null	Nazwisko pracownika
`FirstName`	`nvarchar(70)`	Not null	Imię pracownika

Position	varchar(14)	Not null	Stanowisko jakie pracownik zajmuje w firmie
Email	varchar(100)	Not null	Adres email
Phone	varchar(50)	Not null	Numer telefonu
AdressID	int	Not null	Numer ID adresu

```
CREATE TABLE Staff (
    StaffID int  NOT NULL IDENTITY (1,1),
    LastName nvarchar(50)  NOT NULL,
    FirstName nvarchar(70)  NOT NULL,
    Position varchar(50)  NOT NULL,
    Email varchar(100)  NOT NULL ,
    Phone varchar(14)  NOT NULL UNIQUE
                                check( isnumeric(Phone) = 1 and len(Phone) >= 9),
    AdressID int  NOT NULL UNIQUE check( Email like '%[@]%[.]%'),
    CONSTRAINT Staff_pk PRIMARY KEY  (StaffID)
);
```

3. Referencje

3.1. Reference Products_Category

Category	0..*	Products
CategoryID	<->	CategoryID

```
alter table Products
    add constraint Products_Category
        foreign key (CategoryID) references Category
            on update cascade
```

3.2. Reference Menu_Products

Products	0..*	Menu
ProductID	<->	ProductID

```
alter table Menu
    add constraint Menu_Products
        foreign key (ProductID) references Products
            on update cascade
```

3.3. Reference OrderDetails_Products

Products	1..*	OrderDetails
ProductID	<->	ProductID

```
alter table OrderDetails
    add constraint OrderDetails_Products
        foreign key (ProductID) references Products
            on update cascade
```

3.4. Reference Orders_OrdersTakeaways

OrdersTakeaways	0..*	Orders
TakeawaysID	<->	TakeawayID

```
alter table Orders
    add constraint Orders_OrdersTakeaways
        foreign key (TakeawayID) references OrdersTakeaways
            on update cascade
```

3.5. Reference OrderDetails_Orders

Orders	0..*	OrderDetails
OrderID	<->	OrderID

```
alter table OrderDetails
    add constraint OrderDetails_Orders
        foreign key (OrderID) references Orders
            on update cascade
```

3.6. Reference Discounts_DiscountsVar

DiscountsVar	1..*	Discounts
VarID	<->	VarID

```
alter table Discounts
    add constraint Discounts_DiscountsVar
        foreign key (VarID) references DiscountsVar
            on update cascade
```

3.7. Reference Discounts_IndividualClient

IndividualClient	0..*	Discounts
ClientID	<->	ClientID

```
alter table Discounts
    add constraint Discounts_IndividualClient
        foreign key (ClientID) references IndividualClient
            on update cascade
```

3.8. Reference Clients_IndividualClient

Clients	0..1	IndividualClient
ClientID	<->	ClientID

```
alter table IndividualClient
    add constraint Clients_IndividualClient
        foreign key (ClientID) references Clients
            on update cascade
```

3.9. Reference IndividualClient_Person

Person	0..1	IndividualClient
PersonID	<->	PersonID

```
alter table IndividualClient
    add constraint IndividualClient_Person
        foreign key (PersonID) references Person
            on update cascade
```

3.10. Reference Employees_Person

Person	0..1	Employees
PersonID	<->	PersonID

```
alter table Employees
    add constraint Employees_Person
        foreign key (PersonID) references Person
            on update cascade
```

3.11. Reference Companies_Clients

Clients	0..1	Companies
ClientID	<->	ClientID

```
alter table Companies
    add constraint Companies_Clients
        foreign key (ClientID) references Clients
```

3.12. Reference Employees_Companies

Companies	0..*	Employees
ClientID	<->	CompanyID

```
alter table Employees
    add constraint Employees_Companies
        foreign key (CompanyID) references Companies
            on update cascade
```

3.13. Reference Orders_Clients

Clients	0..*	Orders
ClientID	<->	ClientID

```
alter table Orders
    add constraint Orders_Clients
        foreign key (ClientID) references Clients
            on update cascade
```

3.14. Reference ReservationCompany_Companies

Companies	0..*	ReservationCompany
ClientID	<->	ClientID

```
alter table ReservationCompany
    add constraint ReservationCompany_Companies
        foreign key (ClientID) references Companies
            on update cascade
```

3.15. Reference Reservation_ReservationCompany

ReservationCompany	0..1	Reservation
ReservationID	<->	ReservationID

```
alter table Reservation
    add constraint Reservation_ReservationCompany
        foreign key (ReservationID) references ReservationCompany
```

3.16. Reference Orders_Reservation

Reservation	0..*	Orders
ReservationID	<->	ReservationID

```
alter table Orders
    add constraint Orders_Reservation
        foreign key (ReservationID) references Reservation
            on update cascade
```

3.17. Reference ReservationDetails_Tables

Tables	0..*	ReservationDetails
TableID	<->	TableID

```
alter table ReservationDetails
    add constraint ReservationDetails_Tables
        foreign key (TableID) references Tables
            on update cascade
```

3.18. Reference ReservationDetails_ReservationCompany

ReservationCompany	1..*	ReservationDetails
ReservationID	<->	ReservationID

```
alter table ReservationDetails
    add constraint ReservationDetails_ReservationCompany
        foreign key (ReservationID) references ReservationCompany
            on update cascade
```

3.19. Reference ReservationDetails_ReservationIndividual

ReservationIndividual	1..*	ReservationDetails
ReservationID	<->	ReservationID

```
alter table ReservationDetails
    add constraint ReservationDetails_ReservationIndividual
        foreign key (ReservationID) references ReservationIndividual
            on update cascade
```

3.20. Reference Reservation_ReservationIndividual

ReservationIndividual	0..1	Reservation
ReservationID	<->	ReservationID

```
alter table Reservation
    add constraint Reservation_ReservationIndividual
        foreign key (ReservationID) references ReservationIndividual
```

3.21. Reference Orders_PaymentStatus

PaymentStatus	0..*	Orders
PaymentStatusID	<->	PaymentStatusID

```
alter table Orders
    add constraint Orders_PaymentStatus
        foreign key (PaymentStatusID) references PaymentStatus
            on update cascade
```

3.22. Reference Invoice_PaymentStatus

PaymentStatus	0..*	Invoice
PaymentStatusID	<->	PaymentStatusID

```
alter table Invoice
    add constraint Invoice_PaymentStatus
        foreign key (PaymentStatusID) references PaymentStatus
            on update cascade
```

3.23. Reference Invoice_Clients

Clients	0..*	Invoice
ClientID	<->	ClientID

```
alter table Invoice
    add constraint Invoice_Clients
        foreign key (ClientID) references Clients
            on update cascade
```

3.24. Reference Clients_Address

Address	0..*	Clients
AddressID	<->	AddressID

```
alter table Clients
    add constraint Clients_Address
        foreign key (AddressID) references Address
            on update cascade
```

3.25. Reference Address_Cities

Cities	0..*	Address
CityID	<->	CityID

```
alter table Address
    add constraint Address_Cities
        foreign key (CityID) references Cities
            on update cascade
```

3.26. Reference Orders_staff

Staff	0..*	Orders
StaffID	<->	staffID

```
alter table Orders
    add constraint Orders_staff
        foreign key (staffID) references Staff
            on update cascade
```

3.27. Reference Staff_Address

Address	0..*	Staff
AddressID	<->	AdressID

```
ALTER TABLE Staff ADD CONSTRAINT Staff_Address
    FOREIGN KEY (AddressID)
    REFERENCES Address (AddressID);
```

3.28. Reference Reservation_Staff

Staff	0..*	Reservation
StaffID	<->	StaffID

```
alter table Reservation
    add constraint Reservation_Staff
        foreign key (StaffID) references Staff
```

3.29. Reference Orders_PaymentMethod

```
ALTER TABLE Orders ADD CONSTRAINT Orders_PaymentMethods
    FOREIGN KEY (PaymentMethodID)
    REFERENCES PaymentMethods (PaymentMethodID);
```

3.30. Reference Invoice_PaymentMethod

```
ALTER TABLE Invoice ADD CONSTRAINT Invoice_PaymentMethods
    FOREIGN KEY (PaymentMethodID)
    REFERENCES PaymentMethods (PaymentMethodID);
```

3.31. Reference Orders_Invoice

```
-- Reference: Orders_Invoice (table: Orders)
ALTER TABLE Orders ADD CONSTRAINT Orders_Invoice
    FOREIGN KEY (InvoiceID)
    REFERENCES Invoice (InvoiceID);
```