

Zadanie projektowe PBD 2025/26, WI

Wojciech Cieřlik, Jakub Głód, Michał Graboř

Spis treści

1	Przechowywane informacje	5
2	Warunki integralności	7
3	Schemat bazy danych	8
4	Opis struktury bazy	9
4.1	Części	9
4.2	Dostawcy	10
4.3	Dostawy_Części	11
4.4	Firmy	12
4.5	Kategorie	12
4.6	Klienci	13
4.7	Klienci_indywidualni	14
4.8	Magazyn_części	15
4.9	Magazyn_produktów	16
4.10	Materiały	17
4.11	Metody_płatności	17
4.12	Płatności	18
4.13	Pracownicy	19
4.14	Produkcja	21
4.15	Produkty	22
4.16	Struktura_produktu	23
4.17	Stanowiska	23
4.18	Szczegóły_produktu	24
4.19	Szczegóły_zamówienia	25
4.20	Zamówienia	26
4.21	Państwa	27
4.22	Miejscowości	27
4.23	Realizacja_Dostawy	28
4.24	Stan_zamówienia	29
4.25	Stany	29
5	Generowane Dane	30
6	Widoki	31
6.1	Klienci_pełne_informacje (JG)	31
6.2	Informacje_zamówienia (JG)	33
6.3	Miesięczna_sprzedaż (JG)	34
6.4	Statystyki_dostawców (JG)	35
6.5	Aktualne_Koszty_Produkcji (MG)	36
6.6	Efektywność_Pracowników_Sprzedaż (WC)	37
6.7	Dostawy_pełne_informacje (WC)	38
6.8	Maksymalna_Możliwa_Produkcja (WC)	39
6.9	Historia_Produkcji (WC)	39
6.10	Struktura_Produktów_Z_Części (JG)	40

7	Raporty	41
7.1	Raport_Koszty_Jednostkowe_Miesięcznie (JG)	41
7.2	Raport_Koszty_Jednostkowe_Kwartalnie (JG)	42
7.3	Raport_Koszty_Jednostkowe_Rocznie (JG)	43
7.4	Raport_Koszty_Produkcji_Miesięcznie (WC)	44
7.5	Raport_Koszty_Produkcji_Kwartalnie (WC)	45
7.6	Raport_Koszty_Produkcji_Roczne (WC)	46
7.7	Raport_Stany_Magazynowe_Produktów (JG)	47
7.8	Raport_Stany_Magazynowe_Części (JG)	48
7.9	Raport_Historia_Zamówień_Klientów (MG)	49
7.10	Raport_Zamówienia_Klientów_W_Czasie (WC)	50
7.11	Raport_Sprzedaż_Tygodniowo (JG)	51
7.12	Raport_Plan_Wytworzenia_Produktów_Dzienny (MG)	52
8	Procedury	53
8.1	Procedura_Zamówienia_Klienta_W_Przedziale (JG)	53
8.2	Procedura_Koszty_Produkcji_W_Przedziale (JG)	54
8.3	Procedura_Sprzedaż_W_Przedziale (JG)	55
8.4	Procedura_Zlecenie_Produkcji (WC)	56
8.5	Procedura_Zamówienie_Części (WC)	57
8.6	Procedura_Sprawdz_Dostępnosc_Dla_Zamowienia (MG)	58
8.7	Procedura_Zloz_Zamowienie_Z_Produkcja (MG)	59
9	Triggery	60
9.1	tr_dostawyczesci_update_magazyn (WC)	60
9.2	tr_stanzamowienia_jedenaktywny (MG)	61
9.3	tr_produkcja_zakonczona (WC)	61
9.4	tr_szczegoly_produktu_domykanie (WC)	62
9.5	tr_szczegoly_produktu_kontrola_chronologii (WC)	62
10	Funkcje	63
10.1	Funkcja_Wartość_Zamówienia (MG)	63
10.2	Funkcja_Czas_Realizacji_Produkcji (MG)	63
10.3	Funkcja_Adres_Klienta (MG)	64
10.4	Funkcja_Czy_Wymaga_Uzupełnienia (WC)	65
10.5	Funkcja_Czy_Można_Wyprodukować (MG)	66
10.6	Funkcja_Brakujące_Części (WC)	67
10.7	Funkcja_Szacowany_Termin_Realizacji (JG)	68
10.8	Funkcja_Analiza_Dostępnosci_Zamowienia (JG)	69
11	Użytkownicy i uprawnienia	70
11.1	Administrator (JG)	70
11.2	Kierownik sprzedaży (JG)	71
11.3	Kierownik produkcji (WC)	72
11.4	Analitik (MG)	73
11.5	Kierownik kadr (JG)	73
12	Indeksy	74

12.1 Zamówienia	74
12.2 Produkcja	74
12.3 Struktura_produkту	74
12.4 Magazyn_produkтуw	74
12.5 Magazyn_części	74
12.6 Klienci	74
12.7 Szczegóły zamówienia	75
12.8 Produkty	75
12.9 Dostawy_Części	75

1 Przechowywane informacje

O produktach

- nazwa produktu
- kategoria produktu
- koszt produkcji
- czas produkcji
- cena
- opis

O częściach z jakich tworzone są produkty

- typ części (metal, tworzywo, elementy łączące)
- ilość części potrzebnych na dany produkt
- koszt wykonania
- czas wykonania

O stanie magazynu

- ile jednostek produktu znajduje się w magazynie
- ile części do produkcji znajduje się w magazynie
- minimalne/maksymalne ilości produktu w magazynie

O produkcji (zał. tařmowa)

- data zlecenia produkcji
- data produkcji
- ilość produktu
- czas produkcji

O klientach

- imię, nazwisko / nazwa firmy
- adres
- dane kontaktowe
- dane do faktury

O zamówieniach

- data zamówienia
- data wykonania zamówienia
- nazwa zamawianego produktu
- ilość zamawianego produktu
- cena zamawianego produktu
- rabat
- informacje o płatnościach

O pracownikach

- imię, nazwisko
- adres
- dane kontaktowe
- reports_to

O dostawcach części

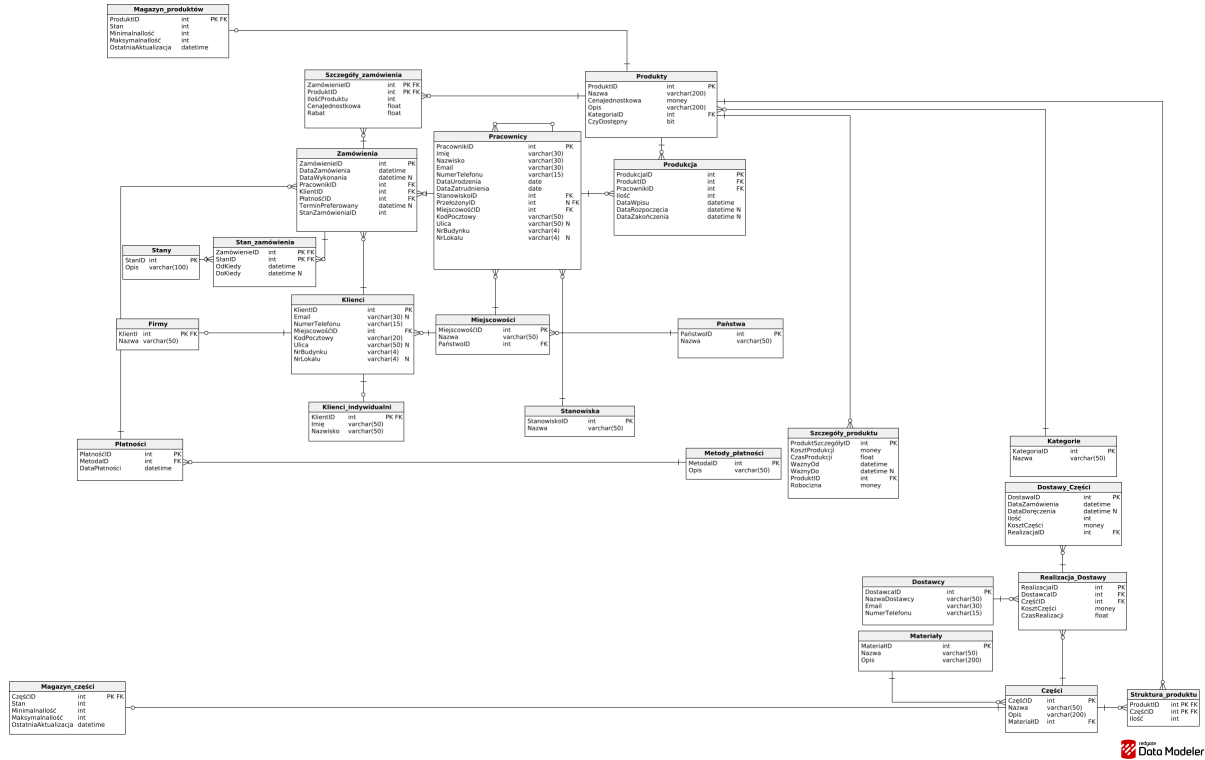
- nazwa dostawcy
- dane kontaktowe

2 Warunki integralności

- **Integralność encji:** każda tabela ma klucz główny NOT NULL. Dodana autoinkrementacja dla odpowiednich kluczy głównych.
- **Integralność referencyjna:** wszystkie klucze obce są zdefiniowane z relacjami (schemat).
- **Integralność domenowa:** sprawdzenia wartości w kolumnach (CHECK).
- **Integralność NOT NULL:** ważne kolumny mają wymagane wartości.
- **Integralność unikalności:** klucze główne i niektóre pola (np. email, numer telefonu) są unikalne (UNIQUE).

3 Schemat bazy danych

Wygenerowany za pomocą Redgate Data Modeler



4 Opis struktury bazy

4.1 Części

Opis: Informacja o częściach, z jakich zbudowane są produkty.

Kolumna	Typ	Opis
CzęśćID	int	Identyfikator części (PK)
Nazwa	varchar(50)	Nazwa części
Opis	varchar(200)	Krótki opis części
MateriałID	int	Materiał wykonania (FK → Materiały.MateriałID)
DostawaID	int	Aktualne informacje o dostawie (FK → Szczegóły_dostawy.DostawaID)

UNIQUE: Nazwa

```
CREATE TABLE [dbo].[Czesc](
[CzescID] [int] IDENTITY(1,1) NOT NULL,
[Nazwa] [varchar](50) NOT NULL,
[Opis] [varchar](200) NOT NULL,
[MaterialID] [int] NOT NULL,
CONSTRAINT [PK_Czesc] PRIMARY KEY CLUSTERED
(
[CzescID] ASC
) ON [PRIMARY],
CONSTRAINT [IX_Czesc] UNIQUE NONCLUSTERED
(
[Nazwa] ASC
) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Czesc] WITH CHECK ADD CONSTRAINT
[FK_Czesc_Materialy] FOREIGN KEY([MaterialID])
REFERENCES [dbo].[Materialy] ([MaterialID])

ALTER TABLE [dbo].[Czesc] CHECK CONSTRAINT [FK_Czesc_Materialy]
```

4.2 Dostawcy

Opis: Informacja o dostawcach dowożących części.

Kolumna	Typ	Opis
DostawcaID	int	Identyfikator dostawcy (PK)
NazwaDostawcy	varchar(50)	Nazwa firmy
Email	varchar(30)	E-mail
NumerTelefonu	varchar(15)	Numer telefonu

UNIQUE: Email, NumerTelefonu

CHECK: Email min. 5 znaków ze znakami @ i .

NumerTelefonu w formacie [+00 000 000 000]

```
CREATE TABLE [dbo].[Dostawcy](
[DostawcaID] [int] IDENTITY(1,1) NOT NULL,
[NazwaDostawcy] [varchar](50) NOT NULL,
[Email] [varchar](30) NOT NULL,
[NumerTelefonu] [varchar](15) NOT NULL,
CONSTRAINT [PK_Dostawcy] PRIMARY KEY CLUSTERED
(
[DostawcaID] ASC
) ON [PRIMARY],
CONSTRAINT [IX_Dostawcy] UNIQUE NONCLUSTERED ([Email] ASC) ON
[PRIMARY],
CONSTRAINT [IX_Dostawcy_1] UNIQUE NONCLUSTERED ([NumerTelefonu]
ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Dostawcy] WITH CHECK ADD CONSTRAINT
[CK_Dostawcy] CHECK
(((Email] like '%@.%' AND NOT [Email] like '%@%' AND NOT
[Email] like '@%')
AND NOT [Email] like '%.@" AND len([Email])>=(5) AND
[NumerTelefonu] like
'+[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'))

ALTER TABLE [dbo].[Dostawcy] CHECK CONSTRAINT [CK_Dostawcy]
```

4.3 Dostawy_Części

Opis: Rzeczywiste dostawy części.

Kolumna	Typ	Opis
DostawaID	int	Identyfikator dostawy (PK)
DataZamówienia	datetime	Data zamówienia
DataDoręczenia	datetime	Data doręczenia (możliwy NULL)
Ilość	int	Ilość dostarczonych części
KosztCzęści	money	Koszt dostawy
RealizacjaID	int	Odniesienie do szablonu realizacji (FK → Realizacja_Dostawy.RealizacjaID)

CHECK: DataDoręczenia NULL lub > DataZamówienia; Ilość > 0, KosztCzęści > 0

DEFAULT: DataZamówienia – aktualna data

```
CREATE TABLE [dbo].[Dostawy_Czesci](
[DostawaID] [int] NOT NULL,
[DataZamowienia] [datetime] NOT NULL,
[DataDoreczenia] [datetime] NULL,
[Ilosc] [int] NOT NULL,
[KosztCzesci] [money] NOT NULL,
[RealizacjaID] [int] NOT NULL,
CONSTRAINT [PK_Dostawy_Czesci] PRIMARY KEY CLUSTERED
([DostawaID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Dostawy_Czesci] ADD CONSTRAINT
[DF_Dostawy_Czesci_DataZamowienia] DEFAULT (getdate()) FOR
[DataZamowienia]

ALTER TABLE [dbo].[Dostawy_Czesci] WITH CHECK ADD CONSTRAINT
[FK_Dostawy_Czesci_Realizacja_Dostawy] FOREIGN
KEY([RealizacjaID])
REFERENCES [dbo].[Realizacja_Dostawy] ([RealizacjaID])

ALTER TABLE [dbo].[Dostawy_Czesci] WITH CHECK ADD CONSTRAINT
[CK_Dostawy_Czesci] CHECK ((([DataDoreczenia] IS NULL OR
[DataZamowienia]<[DataDoreczenia]) AND [Ilosc]>(0) AND
[KosztCzesci]>(0))
```

4.4 Firmy

Opis: Informacje o firmach.

Kolumna	Typ	Opis
KlientID	int	Identyfikator firmy (PK, FK → Klienci.KlientID)
Nazwa	varchar(50)	Nazwa firmy

```
CREATE TABLE [dbo].[Firma](
[KlientID] [int] NOT NULL,
[Nazwa] [varchar](50) NOT NULL,
CONSTRAINT [PK_Firma] PRIMARY KEY CLUSTERED ([KlientID] ASC) ON
[PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Firma] WITH CHECK ADD CONSTRAINT
[FK_Firma_Klienci]
FOREIGN KEY([KlientID]) REFERENCES [dbo].[Klienci] ([KlientID])

ALTER TABLE [dbo].[Firma] CHECK CONSTRAINT [FK_Firma_Klienci]
```

4.5 Kategorie

Opis: Informacje o kategoriach produktów.

Kolumna	Typ	Opis
KategoriaID	int	Identyfikator kategorii (PK)
Nazwa	varchar(50)	Nazwa kategorii

UNIQUE: Nazwa

```
CREATE TABLE [dbo].[Kategorie](
[KategoriaID] [int] IDENTITY(1,1) NOT NULL,
[Nazwa] [varchar](50) NOT NULL,
CONSTRAINT [PK_Kategorie] PRIMARY KEY CLUSTERED ([KategoriaID]
ASC) ON [PRIMARY],
CONSTRAINT [IX_Kategorie] UNIQUE NONCLUSTERED ([Nazwa] ASC) ON
[PRIMARY]
) ON [PRIMARY]
```

4.6 Klienci

Opis: Informacje o klientach.

Kolumna	Typ	Opis
KlientID	int	Identyfikator klienta (PK)
Email	varchar(30)	E-mail
NumerTelefonu	varchar(15)	Numer telefonu
MiejscowoćID	int	Miejscowoć (FK → MiejscowoćID)
KodPocztowy	varchar(20)	Kod pocztowy
Ulica	varchar(50)	Ulica (możliwy NULL)
NrBudynku	varchar(4)	Numer budynku
NrLokalu	varchar(4)	Numer lokalu (możliwy NULL)

UNIQUE: Email, NumerTelefonu

CHECK: Email min. 5 znaków ze znakami @ i .

NumerTelefonu w formacie [+00 000 000 000]

```
CREATE TABLE [dbo].[Klienci](
[KlientID] [int] IDENTITY(1,1) NOT NULL,
[Email] [varchar](30) NOT NULL,
[NumerTelefonu] [varchar](15) NOT NULL,
[MiejscowoscID] [int] NOT NULL,
[KodPocztowy] [varchar](20) NOT NULL,
[Ulica] [varchar](50) NULL,
[NrBudynku] [varchar](4) NOT NULL,
[NrLokalu] [varchar](4) NULL,
CONSTRAINT [PK_Klienci] PRIMARY KEY CLUSTERED ([KlientID] ASC)
ON [PRIMARY],
CONSTRAINT [IX_Klienci_2] UNIQUE NONCLUSTERED ([Email] ASC) ON
[PRIMARY],
CONSTRAINT [IX_Klienci_3] UNIQUE NONCLUSTERED ([NumerTelefonu]
ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Klienci] WITH CHECK ADD CONSTRAINT
[FK_Klienci_Miejscowosci]
FOREIGN KEY([MiejscowoscID]) REFERENCES [dbo].[Miejscowosci]
([MiejscowoscID])

ALTER TABLE [dbo].[Klienci] WITH CHECK ADD CONSTRAINT
[CK_Klienci] CHECK
(((Email like '%@%.' AND NOT [Email] like '%@%.' AND NOT
[Email] like '%@'
AND NOT [Email] like '%.@" AND len([Email])>=(5) AND
[NumerTelefonu] like
'+[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'))
```

4.7 Klienci_indywidualni

Opis: Informacje o klientach indywidualnych.

Kolumna	Typ	Opis
KlientID	int	Identyfikator klienta (PK, FK → Klienci.KlientID)
Imię	varchar(50)	Imię
Nazwisko	varchar(50)	Nazwisko

```
CREATE TABLE [dbo].[Klienci_indywidualni](
[KlientID] [int] NOT NULL,
[Imię] [varchar](50) NOT NULL,
[Nazwisko] [varchar](50) NOT NULL,
CONSTRAINT [PK_Klienci_indywidualni] PRIMARY KEY CLUSTERED
([KlientID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Klienci_indywidualni] WITH CHECK ADD
CONSTRAINT
[FK_Klienci_indywidualni_Klienci] FOREIGN KEY([KlientID])
REFERENCES [dbo].[Klienci] ([KlientID])
```

4.8 Magazyn_części

Opis: Stan magazynowy części.

Kolumna	Typ	Opis
CzęśćID	int	Identyfikator części (PK, FK → CzęśćID)
Stan	int	Aktualna ilość
MinimalnaIlość	int	Minimalny stan
MaksymalnaIlość	int	Maksymalny stan
OstatniaAktualizacja	datetime	Data ostatniej dostawy

CHECK: $\text{Stan} \geq 0$, $\text{MinimalnaIlość} < \text{MaksymalnaIlość}$, $\text{MinimalnaIlość} \geq 0$, $\text{MaksymalnaIlość} > 0$

DEFAULT: OstatniaAktualizacja – aktualna data

```
CREATE TABLE [dbo].[Magazyn_czesci](
[CzescID] [int] NOT NULL,
[Stan] [int] NOT NULL,
[MinimalnaIlosc] [int] NOT NULL,
[MaksymalnaIlosc] [int] NOT NULL,
[OstatniaAktualizacja] [datetime] NOT NULL,
CONSTRAINT [PK_Magazyn_czesci] PRIMARY KEY CLUSTERED ([CzescID]
ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Magazyn_czesci] ADD CONSTRAINT
[DF_Magazyn_czesci_OstatniaAktualizacja] DEFAULT (getdate()) FOR
[OstatniaAktualizacja]

ALTER TABLE [dbo].[Magazyn_czesci] WITH CHECK ADD CONSTRAINT
[FK_Magazyn_czesci_Czesc] FOREIGN KEY([CzescID]) REFERENCES
[dbo].[Czesc] ([CzescID])

ALTER TABLE [dbo].[Magazyn_czesci] WITH CHECK ADD CONSTRAINT
[CK_Magazyn_czesci] CHECK (([Stan]>=(0) AND
[MinimalnaIlosc]<[MaksymalnaIlosc]
AND [MinimalnaIlosc]>=(0) AND [MaksymalnaIlosc]>(0)))
```

4.9 Magazyn_produktów

Opis: Stan magazynowy produktów.

Kolumna	Typ	Opis
ProduktID	int	Identyfikator produktu (PK, FK → Produkty.ProduktID)
Stan	int	Aktualna ilość
MinimalnaIlość	int	Minimalny stan
MaksymalnaIlość	int	Maksymalny stan
OstatniaAktualizacja	datetime	Data ostatniej dostawy

CHECK: $\text{Stan} \geq 0$, $\text{MinimalnaIlość} < \text{MaksymalnaIlość}$, $\text{MinimalnaIlość} \geq 0$, $\text{MaksymalnaIlość} > 0$

DEFAULT: OstatniaAktualizacja – aktualna data

```
CREATE TABLE [dbo].[Magazyn_produktow](
[ProduktID] [int] NOT NULL,
[Stan] [int] NOT NULL,
[MinimalnaIlosc] [int] NOT NULL,
[MaksymalnaIlosc] [int] NOT NULL,
[OstatniaAktualizacja] [datetime] NOT NULL,
CONSTRAINT [PK_Magazyn_produktow] PRIMARY KEY CLUSTERED
    ([ProduktID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Magazyn_produktow] ADD CONSTRAINT
[DF_Magazyn_produktow_OstatniaAktualizacja] DEFAULT (getdate())
    FOR [OstatniaAktualizacja]

ALTER TABLE [dbo].[Magazyn_produktow] WITH CHECK ADD CONSTRAINT
[FK_Magazyn_produktow_Produkty] FOREIGN KEY([ProduktID])
REFERENCES [dbo].[Produkty] ([ProduktID])

ALTER TABLE [dbo].[Magazyn_produktow] WITH CHECK ADD CONSTRAINT
[CK_Magazyn_produktow] CHECK (([Stan]>=(0) AND
    [MinimalnaIlosc]<[MaksymalnaIlosc]
AND [MinimalnaIlosc]>=(0) AND [MaksymalnaIlosc]>(0)))
```


4.10 Materiały

Opis: Informacje o materiałach.

Kolumna	Typ	Opis
MateriałID	int	Identyfikator materiału (PK)
Nazwa	varchar(50)	Nazwa materiału
Opis	varchar(200)	Opis materiału

UNIQUE: Nazwa

```
CREATE TABLE [dbo].[Materialy](  
[MaterialID] [int] IDENTITY(1,1) NOT NULL,  
[Nazwa] [varchar](50) NOT NULL,  
[Opis] [varchar](200) NOT NULL,  
CONSTRAINT [PK_Materialy] PRIMARY KEY CLUSTERED ([MaterialID]  
ASC) ON [PRIMARY],  
CONSTRAINT [IX_Materialy] UNIQUE NONCLUSTERED ([Nazwa] ASC) ON  
[PRIMARY]  
) ON [PRIMARY]
```

4.11 Metody_płatności

Opis: Słownik metod płatności.

Kolumna	Typ	Opis
MetodaID	int	Identyfikator metody (PK)
Opis	varchar(50)	Opis metody

```
CREATE TABLE [dbo].[Metody_platnosci](  
[MetodaID] [int] NOT NULL,  
[Opis] [varchar](50) NOT NULL,  
CONSTRAINT [PK_Metody_platnosci] PRIMARY KEY CLUSTERED  
([MetodaID] ASC) ON [PRIMARY]  
) ON [PRIMARY]
```

4.12 Płatności

Opis: Informacje o płatnościach.

Kolumna	Typ	Opis
PłatnośćID	int	Identyfikator płatności (PK)
MetodaID	int	Metoda płatności (FK → Metody_płatności.MetodaID)
DataPłatności	datetime	Termin płatności

```
CREATE TABLE [dbo].[Platnosci](
  [PlatnoscID] [int] IDENTITY(1,1) NOT NULL,
  [MetodaID] [int] NOT NULL,
  [DataPlatnosci] [datetime] NOT NULL,
  CONSTRAINT [PK_Platnosci] PRIMARY KEY CLUSTERED ([PlatnoscID]
    ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Platnosci] WITH CHECK ADD CONSTRAINT
  [FK_Platnosci_Metody_platnosci] FOREIGN KEY([MetodaID])
  REFERENCES [dbo].[Metody_platnosci] ([MetodaID])
```

4.13 Pracownicy

Opis: Informacje o pracownikach.

Kolumna	Typ	Opis
PracownikID	int	Identyfikator pracownika (PK)
Imię	varchar(30)	Imię
Nazwisko	varchar(30)	Nazwisko
Email	varchar(30)	E-mail
NumerTelefonu	varchar(15)	Numer telefonu
DataUrodzenia	date	Data urodzenia
DataZatrudnienia	date	Data zatrudnienia
StanowiskoID	int	Stanowisko (FK → Stanowiska.StanowiskoID)
PrzełożonyID	int	Przełożony (FK → Pracownicy.PracownikID, możliwy NULL)
MiejscowośćID	int	Miejscowość (FK → Miejscowości.MiejscowośćID)
KodPocztowy	varchar(50)	Kod pocztowy
Ulica	varchar(50)	Ulica (możliwy NULL)
NrBudynku	varchar(4)	Numer budynku
NrLokalu	varchar(4)	Numer lokalu (możliwy NULL)

UNIQUE: Email, NumerTelefonu

CHECK: DataUrodzenia < DataZatrudnienia; Email min. 5 znaków ze znakami @ i .
NumerTelefonu w formacie [+00 000 000 000]

```
CREATE TABLE [dbo].[Pracownicy](
[PracownikID] [int] IDENTITY(1,1) NOT NULL,
[Imie] [varchar](30) NOT NULL,
[Nazwisko] [varchar](30) NOT NULL,
[Email] [varchar](30) NOT NULL,
[NumerTelefonu] [varchar](15) NOT NULL,
[DataUrodzenia] [date] NOT NULL,
[DataZatrudnienia] [date] NOT NULL,
[StanowiskoID] [int] NOT NULL,
[PrzelozonyID] [int] NULL,
[MiejscowoscID] [int] NOT NULL,
[KodPocztowy] [varchar](50) NOT NULL,
[Ulica] [varchar](50) NULL,
[NrBudynku] [varchar](4) NOT NULL,
[NrLokalu] [varchar](4) NULL,
CONSTRAINT [PK_Pracownicy] PRIMARY KEY CLUSTERED ([PracownikID]
ASC) ON [PRIMARY],
CONSTRAINT [IX_Pracownicy] UNIQUE NONCLUSTERED ([Email] ASC) ON
[PRIMARY],
CONSTRAINT [IX_Pracownicy_1] UNIQUE NONCLUSTERED
([NumerTelefonu] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Pracownicy] WITH CHECK ADD CONSTRAINT
[FK_Pracownicy_Miejscowosci] FOREIGN KEY([MiejscowoscID])
REFERENCES [dbo].[Miejscowosci] ([MiejscowoscID])

ALTER TABLE [dbo].[Pracownicy] WITH CHECK ADD CONSTRAINT
[FK_Pracownicy_Pracownicy] FOREIGN KEY([PrzelozonyID])
REFERENCES [dbo].[Pracownicy] ([PracownikID])

ALTER TABLE [dbo].[Pracownicy] WITH CHECK ADD CONSTRAINT
[CK_Pracownicy] CHECK
((([DataUrodzenia]<[DataZatrudnienia] AND ([Email] like '%@%.%'
AND NOT [Email] like '%@%@%' AND NOT [Email] like '@%'
AND NOT [Email] like '%.@' AND len([Email])>=(5))
AND [NumerTelefonu] like
'+[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'))
```

4.14 Produkcja

Opis: Informacje o zleceniach produkcyjnych.

Kolumna	Typ	Opis
ProdukcjaID	int	Identyfikator zlecenia (PK)
ProduktID	int	Produkt (FK → Produkty.ProduktID)
PracownikID	int	Odpowiedzialny pracownik (FK → Pracownicy.PracownikID)
Iloř	int	Iloř wyprodukowana
DataWpisu	datetime	Data wpisu
DataRozpoczęcia	datetime	Data rozpoczęcia (możliwy NULL)
DataZakończenia	datetime	Data zakończenia (możliwy NULL)

CHECK: Iloř > 0, DataRozpoczęcia NULL lub ≥ DataWpisu, DataZakończenia NULL lub ≥ DataRozpoczęcia

DEFAULT: DataWpisu – aktualna data

```
CREATE TABLE [dbo].[Produkcja](
[ProdukcjaID] [int] IDENTITY(1,1) NOT NULL,
[ProduktID] [int] NOT NULL,
[PracownikID] [int] NOT NULL,
[Ilosc] [int] NOT NULL,
[DataWpisu] [datetime] NOT NULL,
[DataRozpoczecia] [datetime] NULL,
[DataZakonczenia] [datetime] NULL,
CONSTRAINT [PK_Produkcja] PRIMARY KEY CLUSTERED ([ProdukcjaID]
ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Produkcja] ADD CONSTRAINT
[DF_Produkcja_DataWpisu]
DEFAULT (getdate()) FOR [DataWpisu]

ALTER TABLE [dbo].[Produkcja] WITH CHECK ADD CONSTRAINT
[CK_Produkcja] CHECK ((([Ilosc]>(0)
AND ([DataRozpoczecia] IS NULL OR [DataWpisu]<=[DataRozpoczecia])
AND ([DataZakonczenia] IS NULL OR [DataRozpoczecia] IS NULL
OR [DataRozpoczecia]<=[DataZakonczenia])))
```

4.15 Produkty

Opis: Informacje o produktach.

Kolumna	Typ	Opis
ProduktID	int	Identyfikator produktu (PK)
Nazwa	varchar(200)	Nazwa produktu
CenaJednostkowa	money	Cena jednostkowa
Opis	varchar(200)	Opis produktu
KategoriaID	int	Kategoria (FK → Kategorie.KategoriaID)
CzyDostępný	bit	Dostępność

CHECK: CenaJednostkowa > 0; UNIQUE: Nazwa; DEFAULT: CzyDostępný – 1 (dostępny)

```
CREATE TABLE [dbo].[Produkty](
[ProduktID] [int] IDENTITY(1,1) NOT NULL,
[Nazwa] [varchar](200) NOT NULL,
[CenaJednostkowa] [money] NOT NULL,
[Opis] [varchar](200) NOT NULL,
[KategoriaID] [int] NOT NULL,
[CzyDostepny] [bit] NOT NULL,
CONSTRAINT [PK_Produkty] PRIMARY KEY CLUSTERED ([ProduktID] ASC)
ON [PRIMARY],
CONSTRAINT [IX_Produkty] UNIQUE NONCLUSTERED ([Nazwa] ASC) ON
[PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Produkty] ADD CONSTRAINT
[DF_Produkty_CzyDostepny]
DEFAULT ((1)) FOR [CzyDostepny]

ALTER TABLE [dbo].[Produkty] WITH CHECK ADD CONSTRAINT
[FK_Produkty_Kategorie]
FOREIGN KEY([KategoriaID]) REFERENCES [dbo].[Kategorie]
([KategoriaID])

ALTER TABLE [dbo].[Produkty] WITH CHECK ADD CONSTRAINT
[CK_Produkty]
CHECK (([CenaJednostkowa]>(0)))
```

4.16 Struktura_produktu

Opis: Powiązanie produktów z częściami.

Kolumna	Typ	Opis
ProduktID	int	Produkt (PK, FK → Produkty.ProduktID)
CzęśćID	int	Część (PK, FK → Części.CzęśćID)
Ilość	int	Ilość części w produkcie

CHECK: Ilość > 0

```
CREATE TABLE [dbo].[Struktura_produktu](
[ProduktID] [int] NOT NULL,
[CzescID] [int] NOT NULL,
[Ilosc] [int] NOT NULL,
CONSTRAINT [PK_ProduktySzczegoly] PRIMARY KEY CLUSTERED
([ProduktID] ASC, [CzescID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Struktura_produktu] WITH CHECK ADD CONSTRAINT
[CK_Struktura_produktu] CHECK (([Ilosc]>(0)))
```

4.17 Stanowiska

Opis: Informacje o stanowiskach pracy.

Kolumna	Typ	Opis
StanowiskoID	int	Identyfikator stanowiska (PK)
Nazwa	varchar(50)	Nazwa stanowiska

UNIQUE: Nazwa

```
CREATE TABLE [dbo].[Stanowiska](
[StanowiskoID] [int] IDENTITY(1,1) NOT NULL,
[Nazwa] [varchar](50) NOT NULL,
CONSTRAINT [PK_Stanowiska] PRIMARY KEY CLUSTERED ([StanowiskoID]
ASC) ON [PRIMARY],
CONSTRAINT [IX_Stanowiska] UNIQUE NONCLUSTERED ([Nazwa] ASC) ON
[PRIMARY]
) ON [PRIMARY]
```

4.18 Szczegóły_produktu

Opis: Parametry produkcji produktu.

Kolumna	Typ	Opis
ProduktSzczegółyID	int	Identyfikator (PK)
KosztProdukcji	money	Koszt
CzasProdukcji	float	Czas (w godzinach)
WażnyOd	datetime	Data od
WażnyDo	datetime	Data do (możliwy NULL)
ProduktID	int	Produkt (FK → Produkty.ProduktID)
Robocizna	money	Stawka za wykonanie

CHECK: KosztProdukcji > 0, CzasProdukcji > 0, Robocizna > 0, WażnyDo NULL lub > WażnyOd

DEFAULT: WażnyOd

```
CREATE TABLE [dbo].[Szczegoly_produktu](
[ProduktSzczegyID] [int] IDENTITY(1,1) NOT NULL,
[KosztProdukcji] [money] NOT NULL,
[CzasProdukcji] [float] NOT NULL,
[WaznyOd] [datetime] NOT NULL,
[WaznDo] [datetime] NULL,
[ProduktID] [int] NOT NULL,
[Robocizna] [money] NOT NULL,
CONSTRAINT [PK_Produkty_szczegoly] PRIMARY KEY CLUSTERED
([ProduktSzczegyID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Szczegoly_produktu] ADD CONSTRAINT
[DF_Szczegoly_produktu_WaznyOd] DEFAULT (getdate()) FOR [WaznyOd]

ALTER TABLE [dbo].[Szczegoly_produktu] WITH CHECK ADD CONSTRAINT
[CK_Szczegoly_produktu] CHECK (([KosztProdukcji]>(0) AND
[CzasProdukcji]>(0)
AND [WaznyOd]<[WaznDo] AND [Robocizna]>(0)))
```


4.19 Szczegóły_zamówienia

Opis: Pozycje zamówień.

Kolumna	Typ	Opis
ZamówienieID	int	Zamówienie (PK, FK → Zamówienia.ZamówienieID)
ProduktID	int	Produkt (PK, FK → Produkty.ProduktID)
IlośćProduktu	int	Ilość
CenaJednostkowa	float	Cena
Rabat	float	Rabat

CHECK: IlośćProduktu > 0, CenaJednostkowa > 0, Rabat ≥ 0, Rabat ≤ 1

DEFAULT: Rabat = 0 (brak rabatu)

```
CREATE TABLE [dbo].[Szczegoly_zamowienia](
[ZamowienieID] [int] NOT NULL,
[ProduktID] [int] NOT NULL,
[IloscProduktu] [int] NOT NULL,
[CenaJednostkowa] [float] NOT NULL,
[Rabat] [float] NOT NULL,
CONSTRAINT [PK_Szczegoly_zamowienia] PRIMARY KEY CLUSTERED
([ZamowienieID] ASC, [ProduktID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Szczegoly_zamowienia] ADD CONSTRAINT
[DF_Szczegoly_zamowienia_Rabat] DEFAULT ((0)) FOR [Rabat]

ALTER TABLE [dbo].[Szczegoly_zamowienia] WITH CHECK ADD
CONSTRAINT
[CK_Szczegoly_zamowienia] CHECK ((([IloscProduktu]>(0) AND
[CenaJednostkowa]>(0)
AND [Rabat]>=(0) AND [Rabat]<=(1))))
```

4.20 Zamówienia

Opis: Informacje o zamówieniach.

Kolumna	Typ	Opis
ZamówienieID	int	Identyfikator zamówienia (PK)
DataZamówienia	datetime	Data zamówienia
DataWykonania	datetime	Data wykonania
PracownikID	int	Obsługujący pracownik (FK → Pracownicy.PracownikID)
KlientID	int	Klient (FK → Klienci.KlientID)
PłatnośćID	int	Płatność (FK → Płatności.PłatnośćID)
TerminPreferowany	datetime	Preferowany termin (możliwy NULL)
StanZamówieniaID	int	Aktualny stan zamówienia (FK → Stany.StanID)

CHECK: DataWykonania NULL lub > DataZamówienia, TerminPreferowany > GETDATE()

```
CREATE TABLE [dbo].[Zamowienia](
[ZamowienieID] [int] IDENTITY(1,1) NOT NULL,
[DataZamowienia] [datetime] NOT NULL,
[DataWykonania] [datetime] NULL,
[PracownikID] [int] NOT NULL,
[KlientID] [int] NOT NULL,
[PlatnoscID] [int] NOT NULL,
[TerminPreferowany] [datetime] NULL,
[StanZamowieniaID] [int] NOT NULL,
CONSTRAINT [PK_Zamowienia] PRIMARY KEY CLUSTERED ([ZamowienieID]
ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Zamowienia] WITH CHECK ADD CONSTRAINT
[CK_Zamowienia] CHECK
(((DataWykonania IS NULL OR [DataZamowienia]<[DataWykonania])
AND [TerminPreferowany]>getdate()))
```

4.21 Państwa

Opis: Lista państw.

Kolumna	Typ	Opis
PaństwoID	int	Identyfikator państwa (PK)
Nazwa	varchar(50)	Nazwa państwa

UNIQUE: Nazwa

```
CREATE TABLE [dbo].[Panstwa](
[PaństwoID] [int] IDENTITY(1,1) NOT NULL,
[Nazwa] [varchar](50) NOT NULL,
CONSTRAINT [PK_Panstwa] PRIMARY KEY CLUSTERED ([PaństwoID] ASC)
ON [PRIMARY],
CONSTRAINT [IX_Panstwa] UNIQUE NONCLUSTERED ([Nazwa] ASC) ON
[PRIMARY]
) ON [PRIMARY]
```

4.22 Miejscowości

Opis: Lista miejscowości.

Kolumna	Typ	Opis
MiejscowośćID	int	Identyfikator miejscowości (PK)
Nazwa	varchar(50)	Nazwa miejscowości
PaństwoID	int	Państwo (FK → Państwa.PaństwoID)

UNIQUE: Nazwa

```
CREATE TABLE [dbo].[Miejscowosci](
[MiejscowoscID] [int] IDENTITY(1,1) NOT NULL,
[Nazwa] [varchar](50) NOT NULL,
[PaństwoID] [int] NOT NULL,
CONSTRAINT [PK_Miejscowosci] PRIMARY KEY CLUSTERED
([MiejscowoscID] ASC) ON [PRIMARY],
CONSTRAINT [IX_Miejscowosci] UNIQUE NONCLUSTERED ([Nazwa] ASC)
ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Miejscowosci] WITH CHECK ADD CONSTRAINT
[FK_Miejscowosci_Panstwa] FOREIGN KEY([PaństwoID])
REFERENCES [dbo].[Panstwa] ([PaństwoID])
```

4.23 Realizacja_Dostawy

Opis: Oczekiwane parametry dostaw części od dostawców.

Kolumna	Typ	Opis
RealizacjaID	int	Identyfikator realizacji (PK)
DostawcaID	int	Dostawca (FK → Dostawcy.DostawcaID)
CzęśćID	int	Częć (FK → Części.CzęćID)
KosztCzęści	money	Koszt części
CzasRealizacji	float	Czas realizacji (w godzinach)

CHECK: KosztCzęści > 0, CzasRealizacji > 0

```
CREATE TABLE [dbo].[Realizacja_Dostawy](
[RealizacjaID] [int] NOT NULL,
[DostawcaID] [int] NOT NULL,
[CzescID] [int] NOT NULL,
[KosztCzesci] [money] NOT NULL,
[CzasRealizacji] [float] NOT NULL,
CONSTRAINT [PK_Realizacja_Dostawy] PRIMARY KEY CLUSTERED
    ([RealizacjaID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Realizacja_Dostawy] WITH CHECK ADD CONSTRAINT
[CK_Realizacja_Dostawy] CHECK (([KosztCzesci]>(0) AND
[CzasRealizacji]>(0)))
```

4.24 Stan_zamówienia

Opis: Historia stanów zamówień.

Kolumna	Typ	Opis
ZamówienieID	int	Zamówienie (PK, FK → Zamówienia.ZamówienieID)
StanID	int	Stan (PK, FK → Stany.StanID)
OdKiedy	datetime	Data rozpoczęcia stanu
DoKiedy	datetime	Data zakończenia stanu (możliwy NULL)

CHECK: DoKiedy NULL lub DoKiedy > OdKiedy

```
CREATE TABLE [dbo].[Stan_zamowienia](
[ZamowienieID] [int] NOT NULL,
[StanID] [int] NOT NULL,
[OdKiedy] [datetime] NOT NULL,
[DoKiedy] [datetime] NULL,
CONSTRAINT [PK_Stan_zamowienia] PRIMARY KEY CLUSTERED
([ZamowienieID] ASC, [StanID] ASC) ON [PRIMARY]
) ON [PRIMARY]

ALTER TABLE [dbo].[Stan_zamowienia] WITH CHECK ADD CONSTRAINT
[CK_Stan_zamowienia] CHECK (([DoKiedy] IS NULL OR
[OdKiedy]<[DoKiedy]))
```

4.25 Stany

Opis: Słownik stanów zamówień.

Kolumna	Typ	Opis
StanID	int	Identyfikator stanu (PK, UNIQUE)
Opis	varchar(100)	Opis stanu

```
CREATE TABLE [dbo].[Stany](
[StanID] [int] NOT NULL,
[Opis] [varchar](100) NOT NULL,
CONSTRAINT [PK_Stany] PRIMARY KEY CLUSTERED ([StanID] ASC) ON
[PRIMARY],
CONSTRAINT [IX_Stany] UNIQUE NONCLUSTERED ([StanID] ASC) ON
[PRIMARY]
) ON [PRIMARY]
```

5 Generowane Dane

Dane zostały wygenerowane skryptem napisanym w Pythonie, gdzie użyto bibliotek `faker` i `random`.

Dla kaędej tabeli wygenerowano następującą liczbę rekordów:

- Części: 40
- Dostawcy: 5
- Dostawy części: 15
- Kategorie: 10
- Klienci: 20 (indywidualni: 12, firmy: 8)
- Magazyn części: 40
- Magazyn produktów: 20
- Materiały: 15
- Metody płatności: 5
- Miejscowości: 15
- Państwa: 5
- Płatności: 50
- Pracownicy: 15
- Produkcja: 25
- Produkty: 20
- Struktura produktu: 111
- Stanowiska: 10
- Szczegóły produktu: 41
- Szczegóły_zamówień: 88
- Zamówienia: 50
- Realizacja dostaw: 15
- Stan zamówienia: 296
- Stany: 6

6 Widoki

6.1 Klienci_pełne_informacje (JG)

Opis: Szczegółowe informacje o kliencie.

Kolumna	Opis
ID klienta	Identyfikator klienta
Klient indywidualny / firma	Klient jest klientem indywidualnym lub firmą
Imię i nazwisko / nazwa firmy	Imię i nazwisko klienta indywidualnego lub nazwa firmy
Email	E-mail
Numer telefonu	Numer telefonu
Państwo	Państwo
Miejscowość	Miejscowość
Kod pocztowy	Kod pocztowy
Ulica	Ulica
NrBudynku	Numer budynku
NrLokalu	Numer lokalu

```
create view Klienci_pelne_informacje
as (
select
k.KlientID as 'ID klienta',
'Klient indywidualny' as 'Klient indywidualny / firma',
ki.Imie + ' ' + ki.Nazwisko as 'Imie i nazwisko / nazwa firmy',
k.Email,
k.NumerTelefonu as 'Numer telefonu',
p.Nazwa as 'Panstwo',
m.Nazwa as 'Miejscowosc',
k.KodPocztowy as 'Kod pocztowy',
k.Ulica,
k.NrBudyunku as 'Nr Budyunku',
k.NrLokalu as 'Nr lokalu'
from Klienci k
join Klienci_indywidualni ki on k.KlientID = ki.KlientID
join Miejscowosci m on k.MiejscowoscID = m.MiejscowoscID
join Panstwa p on m.PanstwoID = p.PanstwoID

union

select
k.KlientID as 'ID klienta',
'Firma' as 'Klient indywidualny / firma',
f.Nazwa as 'Imie i nazwisko / nazwa firmy',
k.Email,
k.NumerTelefonu as 'Numer telefonu',
p.Nazwa as 'Panstwo',
m.Nazwa as 'Miejscowosc',
k.KodPocztowy as 'Kod pocztowy',
k.Ulica,
k.NrBudyunku as 'Nr Budyunku',
k.NrLokalu as 'Nr lokalu'
from Klienci k
join Firmy f on k.KlientID = f.KlientID
join Miejscowosci m on k.MiejscowoscID = m.MiejscowoscID
join Panstwa p on m.PanstwoID = p.PanstwoID
)
```


6.2 Informacje_zamówienia (JG)

Opis: Szczegółowe informacje o zamówieniach.

Kolumna	Opis
ID zamówienia	Identyfikator zamówienia
Data zamówienia	Data zamówienia
Data wykonania	Data wykonania zamówienia
Termin preferowany	Preferowany termin
Pracownik	Imię i nazwisko pracownika
Klient	Imię i nazwisko klienta indywidualnego lub nazwa firmy
Wartość zamówienia	Łączna wartość zamówienia z uwzględnieniem rabatu
Łączna liczba produktów	Łączna liczba produktów w zamówieniu
Metoda płatności	Metoda płatności

```
create view Informacje_zamowienia
as (
select
z.ZamowienieID as 'ID zamowienia',
z.DataZamowienia as 'Data zamowienia',
z.DataWykonania as 'Data wykonania',
z.TerminPreferowany as 'Termin preferowany',
p.Imie + ' ' + p.Nazwisko as 'Pracownik',
isnull(ki.Imie + ' ' + ki.Nazwisko, f.Nazwa) as 'Klient',
round(sum(szz.IloscProduktu * szz.CenaJednostkowa *
(1-szz.Rabat)), 2)
as 'Wartosc zamowienia',
sum(szz.IloscProduktu) as 'Laczna liczba produktow',
mp.Opis as 'Metoda platnosci'
from Zamowienia z
join Pracownicy p on z.PracownikID = p.PracownikID
join Platnosci pl on z.PlatnoscID = pl.PlatnoscID
join Metody_platnosci mp on pl.MetodaID = mp.MetodaID
join Szczegoly_zamowienia szz on z.ZamowienieID =
szz.ZamowienieID
join Klienci k on z.KlientID = k.KlientID
left join Klienci_indywidualni ki on k.KlientID = ki.KlientID
left join Firmy f on k.KlientID = f.KlientID
group by z.ZamowienieID, z.DataZamowienia, z.DataWykonania,
z.TerminPreferowany, p.Imie, p.Nazwisko,
ki.Imie, ki.Nazwisko, f.Nazwa, mp.Opis
)
```

6.3 Miesięczna_sprzedaż (JG)

Opis: Informacje o sprzedaży miesięcznej dla każdej kategorii.

Kolumna	Opis
Rok	Rok sprzedaży
Miesiąc	Miesiąc sprzedaży
Kategoria produktu	Kategoria produktów sprzedawanych
Liczba zamówień	Liczba zamówień dla danej kategorii w danym roku i miesiącu
Łączna liczba produktów	Łączna liczba produktów dla danej kategorii w danym roku i miesiącu
Wartość zamówień	Sumaryczna wartość zamówień
Wartość rabatów	Sumaryczna wartość rabatów
Średnia cena jednostkowa	Średnia cena jednostkowa

```
create view Miesieczna_sprzedaz
as (
select
year(z.DataZamowienia) as 'Rok',
month(z.DataZamowienia) as 'Miesiac',
k.Nazwa as 'Kategoria produktu',
count(distinct z.ZamowienieID) as 'Liczba zamowien',
sum(sz.IloscProduktu) as 'Laczna liczba produktow',
round(sum(sz.IloscProduktu * sz.CenaJednostkowa * (1 -
    sz.Rabat)), 2)
    as 'Wartosc zamowien',
round(sum(sz.IloscProduktu * sz.CenaJednostkowa * sz.Rabat), 2)
    as 'Wartosc rabatow',
round(avg(sz.CenaJednostkowa), 2) as 'Srednia cena jednostkowa'
from Zamowienia z
join Szczegoly_zamowienia sz on z.ZamowienieID = sz.ZamowienieID
join Produkty p on sz.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
group by year(z.DataZamowienia), month(z.DataZamowienia), k.Nazwa
)
```

6.4 Statystyki_dostawców (JG)

Opis: Statystyki dostawców części.

Kolumna	Opis
ID dostawcy	Identyfikator dostawcy
Nazwa dostawcy	Nazwa dostawcy
Numer telefonu	Numer telefonu
Liczba dostaw	Liczba dostaw
Łączna liczba dostarczonych części	Łączna liczba dostarczonych części
Łączna wartość dostaw	Łączna wartość dostaw
Średni czas realizacji	Średni czas realizacji dostaw
Pierwsza dostawa	Pierwsza dostawa dostawcy
Ostatnia dostawa	Ostatnia dostawa dostawcy

```
create view Statystyki_dostawcow
as (
select
d.DostawcaID as 'ID dostawcy',
d.NazwaDostawcy as 'Nazwa dostawcy',
d.Email,
d.NumerTelefonu as 'Numer telefonu',
count(distinct dc.DostawaID) as 'Liczba dostaw',
sum(dc.Ilosc) as 'Łączna liczba dostarczonych części',
round(sum(dc.KosztCzesci * dc.Ilosc), 2) as 'Łączna wartość
    dostaw',
round(avg(rd.CzasRealizacji), 2) as 'Średni czas realizacji',
min(dc.DataDoreczenia) as 'Pierwsza dostawa',
max(dc.DataDoreczenia) as 'Ostatnia dostawa'
from Dostawcy d
left join Realizacja_Dostawy rd on d.DostawcaID = rd.DostawcaID
left join Dostawy_Czesci dc on rd.RealizacjaID = dc.RealizacjaID
GROUP BY d.DostawcaID, d.NazwaDostawcy, d.Email, d.NumerTelefonu
)
```

6.5 Aktualne_Koszty_Produkcji (MG)

Opis: Zestawienie aktualnie obowiązujących kosztów produkcji i marży dla produktów.

Kolumna	Opis
Nazwa produktu	Nazwa produktu
Cena sprzedaży	Cena jednostkowa produktu (katalogowa)
Koszt materiałowy	Koszt materiałów potrzebnych do produkcji
Robocizna	Koszt pracy ludzkiej przy produkcji
Łączny koszt produkcji	Suma kosztów materiałowych i robocizny
Szacowany czas produkcji	Czas potrzebny na wytworzenie (w godzinach)
Marża kwotowa	Różnica między ceną sprzedaży a kosztem produkcji

```
create view Aktualne_Koszty_Produkcji
as (
select
p.Nazwa as 'Nazwa produktu',
p.CenaJednostkowa as 'Cena sprzedaży',
sp.KosztProdukcji as 'Koszt materiałowy',
sp.Robocizna,
(sp.KosztProdukcji + sp.Robocizna) as 'Łączny koszt produkcji',
sp.CzasProdukcji as 'Szacowany czas produkcji',
(p.CenaJednostkowa - (sp.KosztProdukcji + sp.Robocizna)) as
'Marża kwotowa'
from Produkty p
join Szczegoly_produktu sp on p.ProduktID = sp.ProduktID
where getdate() >= sp.WaznyOd
and (sp.WaznDo IS NULL OR getdate() <= sp.WaznDo)
)
```

6.6 Efektywnoř Pracowników Sprzedaź (WC)

Opis: Raport dla kadry zarzadzajacej pokazujacy skutecznoř pracowników obslugujacych zamówienia.

Kolumna	Opis
ID pracownika	Identyfikator pracownika
Pracownik	Imię i nazwisko pracownika
Stanowisko	Stanowisko zajmowane przez pracownika
Liczba obsłużonych zamówień	Całkowita liczba zamówień przypisanych do pracownika
Łączna wartość sprzedaży	Sumaryczna wartość zrealizowanych zamówień (po rabatach)
Średnia wartość zamówienia	Średnia kwota pojedynczego zamówienia

```
create view Efektywnosc_Pracownikow_Sprzedaz
as (
select
p.PracownikID as 'ID pracownika',
p.Imie + ' ' + p.Nazwisko as 'Pracownik',
s.Nazwa as 'Stanowisko',
count(distinct z.ZamowienieID) as 'Liczba obsluzonych zamowien',
round(sum(sz.IloscProduktu * sz.CenaJednostkowa * (1 -
sz.Rabat)), 2)
as 'Laczna wartosc sprzedazy',
round(avg(sz.IloscProduktu * sz.CenaJednostkowa * (1 -
sz.Rabat)), 2)
as 'Srednia wartosc zamowienia'
from Pracownicy p
join Stanowiska s on p.StanowiskoID = s.StanowiskoID
join Zamowienia z on p.PracownikID = z.PracownikID
join Szczegoly_zamowienia sz on z.ZamowienieID = sz.ZamowienieID
group by p.PracownikID, p.Imie, p.Nazwisko, s.Nazwa
)
```

6.7 Dostawy_pelne_informacje (WC)

Opis: Widok pokazuje szczegółowe informacje o dostawach części wraz z nazwą części i dostawcą.

```
create view Dostawy_pelne_informacje
as (
select
dc.DostawaID,
dc.DataZamowienia as 'Data zamowienia',
dc.DataDoreczenia as 'Data doreczenia',
dc.Ilosc,
dc.KosztCzesci as 'Koszt czesci',
c.Nazwa as 'Czesc',
d.NazwaDostawcy as 'Dostawca'
from Dostawy_Czesci dc
join Realizacja_Dostawy rd on dc.RealizacjaID = rd.RealizacjaID
join Czesc c on rd.CzescID = c.CzescID
join Dostawcy d on rd.DostawcaID = d.DostawcaID
)
```

6.8 Maksymalna_Mozliwa_Produkcja (WC)

Opis: Dla kaędego produktu okreřla, ile sztuk moęna wyprodukować bazując na aktualnym stanie magazynu części.

```
create view Maksymalna_Mozliwa_Produkcja
as (
select
p.Nazwa as 'Produkt',
case
when min(floor(mc.Stan / sp.Ilosc)) > 0 then min(floor(mc.Stan
/ sp.Ilosc))
else 0
end as 'Liczba',
case
when min(floor(mc.Stan / sp.Ilosc)) > 0 then 'Mozliwe do
produkcji'
else 'Brak czesci'
end as 'Status produkcji'
from Produkty p
join Struktura_produkту sp on p.ProduktID = sp.ProduktID
join Magazyn_czesci mc on sp.CzescID = mc.CzescID
group by p.Nazwa
)
```

6.9 Historia_Produkcji (WC)

Opis: Historia zakończonych zleceń produkcyjnych wraz z kosztami produkcji obowiązującymi w momencie wpisu zlecenia.

```
create view Historia_Produkcji
as (
select
pr.ProdukcjaID,
prod.Nazwa as 'Produkt',
pr.Ilosc,
pr.DataWpisu,
pr.DataRozpoczecia,
pr.DataZakonczenia,
sp.KosztProdukcji as 'Koszt produkcji',
sp.Robocizna as 'Koszt robocizny',
pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna) as 'Laczny koszt
produkcji'
from Produkcja pr
join Produkty prod on pr.ProduktID = prod.ProduktID
join Szczegoly_produkту sp on prod.ProduktID = sp.ProduktID
and pr.DataWpisu >= sp.WaznyOd
and (sp.WaznDo is null or pr.DataWpisu <= sp.WaznDo)
where pr.DataZakonczenia is not null
)
```

6.10 Struktura_Produktów_Z_Części (JG)

Opis: Struktura każdego produktu – ile i jakie części zawiera produkt.

```
create view Struktura_Produktow_Z_Czesci
as
select
p.Nazwa as 'Nazwa produktu',
p.Opis as 'Opis produktu',
sp.Ilosc as 'Ilosc czesci w produkcji',
c.Nazwa as 'Nazwa czesci',
c.Opis as 'Opis czesci'
from Struktura_produkту sp
join Produkty p on sp.ProduktID = p.ProduktID
join Czesc c on c.CzescID = sp.CzescID
```


7 Raporty

7.1 Raport_Koszty_Jednostkowe_Miesięcznie (JG)

Opis: Koszt produkcji jednostkowy ujęty miesięcznie.

```
create view Raport_Koszty_Jednostkowe_Miesiecznie
as
select
year(pr.DataZakonczenia) as 'Rok',
month(pr.DataZakonczenia) as 'Miesiac',
p.ProduktID,
p.Nazwa as 'Nazwa produktu',
k.Nazwa as 'Kategoria',
count(pr.ProdukcjaID) as 'Liczba produkcji',
sum(pr.Ilosc) as 'Wyprodukowanych sztuk',
sum(pr.Ilosc * sp.KosztProdukcji) as 'Laczne koszty materialow',
sum(pr.Ilosc * sp.Robocizna) as 'Laczna robocizna',
sum(pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna)) as 'Laczny
    koszt produkcji',
avg(sp.KosztProdukcji + sp.Robocizna) as 'Sredni koszt
    jednostkowy',
avg(p.CenaJednostkowa) - avg(sp.KosztProdukcji + sp.Robocizna)
    as 'Srednia marza'
from Produkcja pr
join Produkty p on pr.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
join Szczegoly_produkту sp on p.ProduktID = sp.ProduktID
    and pr.DataZakonczenia between sp.WaznyOd and
        isnull(sp.WaznDo, getdate())
where pr.DataZakonczenia is not null
group by year(pr.DataZakonczenia), month(pr.DataZakonczenia),
p.ProduktID, p.Nazwa, k.Nazwa
```

7.2 Raport_Koszty_Jednostkowe_Kwartalnie (JG)

Opis: Koszt produkcji jednostkowy ujęty kwartalnie.

```
create view Raport_Koszty_Jednostkowe_Kwartalnie
as
select
year(pr.DataZakonczenia) as 'Rok',
datepart(quarter, pr.DataZakonczenia) as 'Kwartal',
p.ProduktID,
p.Nazwa as 'Nazwa produktu',
k.Nazwa as 'Kategoria',
count(pr.ProdukcjaID) as 'Liczba produkcji',
sum(pr.Ilosc) as 'Wyprodukowanych sztuk',
sum(pr.Ilosc * sp.KosztProdukcji) as 'Laczne koszty materialow',
sum(pr.Ilosc * sp.Robocizna) as 'Laczna robocizna',
sum(pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna)) as 'Laczny
    koszt produkcji',
avg(sp.KosztProdukcji + sp.Robocizna) as 'Sredni koszt
    jednostkowy',
avg(p.CenaJednostkowa) - avg(sp.KosztProdukcji + sp.Robocizna)
    as 'Srednia marza'
from Produkcja pr
join Produkty p on pr.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
join Szczegoly_produkту sp on p.ProduktID = sp.ProduktID
    and pr.DataZakonczenia between sp.WaznyOd and
        isnull(sp.WaznDo, getdate())
where pr.DataZakonczenia is not null
group by year(pr.DataZakonczenia), datepart(quarter,
    pr.DataZakonczenia),
    p.ProduktID, p.Nazwa, k.Nazwa
```

7.3 Raport_Koszty_Jednostkowe_Rocznie (JG)

Opis: Koszt produkcji jednostkowy ujęty rocznie.

```
create view Raport_Koszty_Jednostkowe_Rocznie
as
select
year(pr.DataZakonczenia) as 'Rok',
p.ProduktID,
p.Nazwa as 'Nazwa produktu',
k.Nazwa as 'Kategoria',
count(pr.ProdukcjaID) as 'Liczba produkcji',
sum(pr.Ilosc) as 'Wyprodukowanych sztuk',
sum(pr.Ilosc * sp.KosztProdukcji) as 'Laczne koszty materialow',
sum(pr.Ilosc * sp.Robocizna) as 'Laczna robocizna',
sum(pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna)) as 'Laczny
    koszt produkcji',
avg(sp.KosztProdukcji + sp.Robocizna) as 'Sredni koszt
    jednostkowy',
avg(p.CenaJednostkowa) - avg(sp.KosztProdukcji + sp.Robocizna)
    as 'Srednia marza'
from Produkcja pr
join Produkty p on pr.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
join Szczegoly_produkту sp on p.ProduktID = sp.ProduktID
    and pr.DataZakonczenia between sp.WaznyOd and
        isnull(sp.WaznDo, getdate())
where pr.DataZakonczenia is not null
group by year(pr.DataZakonczenia), p.ProduktID, p.Nazwa, k.Nazwa
```

7.4 Raport_Koszty_Produkcji_Miesięcznie (WC)

Opis: Koszt produkcji dla danych kategorii ujęty miesięcznie.

```
create view Raport_Koszty_Produkcji_Miesiecznie
as (
select
year(pr.DataZakonczenia) as 'Rok',
month(pr.DataZakonczenia) as 'Miesiac',
k.KategoriaID,
k.Nazwa as 'Kategoria',
count(pr.ProdukcjaID) as 'Liczba produkcji',
sum(pr.Ilosc) as 'Laczna ilosc sztuk',
sum(pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna)) as 'Laczny
koszt produkcji',
avg(sp.KosztProdukcji + sp.Robocizna) as 'Sredni koszt
jednostkowy',
sum(pr.Ilosc * sp.CzasProdukcji) as 'Laczny czas produkcji
(godziny)'
from Produkcja pr
join Produkty p on pr.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
join Szczegoly_produktu sp on p.ProduktID = sp.ProduktID
and pr.DataZakonczenia between sp.WaznyOd and
isnull(sp.WaznDo, getdate())
where pr.DataZakonczenia is not null
group by year(pr.DataZakonczenia), month(pr.DataZakonczenia),
k.KategoriaID, k.Nazwa
)
```

7.5 Raport_Koszty_Produkcji_Kwartalnie (WC)

Opis: Koszt produkcji dla danych kategorii ujęty kwartalnie.

```
create view Raport_Koszty_Produkcji_Kwartalnie
as (
select
year(pr.DataZakonczenia) as 'Rok',
datepart(quarter, pr.DataZakonczenia) as 'Kwartal',
k.KategoriaID,
k.Nazwa as 'Kategoria',
count(pr.ProdukcjaID) as 'Liczba produkcji',
sum(pr.Ilosc) as 'Laczna ilosc sztuk',
sum(pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna)) as 'Laczny
    koszt produkcji',
avg(sp.KosztProdukcji + sp.Robocizna) as 'Sredni koszt
    jednostkowy',
sum(pr.Ilosc * sp.CzasProdukcji) as 'Laczny czas produkcji
    (godziny)'
from Produkcja pr
inner join Produkty p on pr.ProduktID = p.ProduktID
inner join Kategorie k on p.KategoriaID = k.KategoriaID
inner join Szczegoly_produktu sp on p.ProduktID = sp.ProduktID
    and pr.DataZakonczenia between sp.WaznyOd and
        isnull(sp.WaznDo, getdate())
where pr.DataZakonczenia is not null
group by year(pr.DataZakonczenia), datepart(quarter,
    pr.DataZakonczenia),
    k.KategoriaID, k.Nazwa
)
```

7.6 Raport_Koszty_Produkcji_Roczne (WC)

Opis: Koszt produkcji dla danych kategorii ujęty rocznie.

```
create view Raport_Koszty_Produkcji_Roczne
as (
select
year(pr.DataZakonczenia) as 'Rok',
k.KategoriaID,
k.Nazwa as 'Kategoria',
count(pr.ProdukcjaID) as 'Liczba produkcji',
sum(pr.Ilosc) as 'Laczna ilosc sztuk',
sum(pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna)) as 'Laczny
koszt produkcji',
avg(sp.KosztProdukcji + sp.Robocizna) as 'Sredni koszt
jednostkowy',
sum(pr.Ilosc * sp.CzasProdukcji) as 'Laczny czas produkcji
(godziny)'
from Produkcja pr
inner join Produkty p on pr.ProduktID = p.ProduktID
inner join Kategorie k on p.KategoriaID = k.KategoriaID
inner join Szczegoly_produktu sp on p.ProduktID = sp.ProduktID
and pr.DataZakonczenia between sp.WaznyOd and
isnull(sp.WaznDo, getdate())
where pr.DataZakonczenia is not null
group by year(pr.DataZakonczenia), k.KategoriaID, k.Nazwa
)
```

7.7 Raport_Stany_Magazynowe_Produktów (JG)

Opis: Raport obrazujący bieżące stany magazynowe produktów wraz z informacją o konieczności produkcji.

```
create view Raport_Stany_Magazynowe_Produktow
as
select
mp.ProduktID,
p.Nazwa as 'Nazwa produktu',
k.Nazwa as 'Kategoria',
mp.Stan as 'Aktualny stan',
mp.MinimalnaIlosc as 'Minimalna ilosc',
mp.MaksymalnaIlosc as 'Maksymalna ilosc',
mp.OstatniaAktualizacja as 'Ostatnia aktualizacja',
case
    when mp.Stan < mp.MinimalnaIlosc then 'Ponizej normy!'
    when mp.Stan < mp.MinimalnaIlosc * 1.5 then 'Niski stan'
    when mp.Stan > mp.MaksymalnaIlosc then 'Nadmiar'
    else 'W normie'
end as 'Status magazynu',
mp.MaksymalnaIlosc - mp.Stan as 'Maksymalna liczba do produkcji',
p.CenaJednostkowa * mp.Stan as 'Wartosc magazynu'
from Magazyn_produkow mp
join Produkty p on mp.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
```

7.8 Raport_Stany_Magazynowe_Czeřci (JG)

Opis: Raport obrazujący bieżące stany magazynowe części wraz z informacją o konieczności domówienia.

```
create view Raport_Stany_Magazynowe_Czesci
as
select
mc.CzescID,
c.Nazwa as 'Nazwa czesci',
m.Nazwa as 'Material',
mc.Stan as 'Aktualny stan',
mc.MinimalnaIlosc as 'Minimalna ilosc',
mc.MaksymalnaIlosc as 'Maksymalna ilosc',
mc.OstatniaAktualizacja as 'Ostatnia aktualizacja',
case
    when mc.Stan < mc.MinimalnaIlosc then 'Ponizej normy!'
    when mc.Stan < mc.MinimalnaIlosc * 1.5 then 'Niski stan'
    when mc.Stan > mc.MaksymalnaIlosc then 'Nadmiar'
    else 'W normie'
end as 'Status magazynu'
from Magazyn_czesci mc
join Czesc c on mc.CzescID = c.CzescID
join Materialy m on c.MaterialID = m.MaterialID
```


7.9 Raport_Historia_Zamówień_Klientów (MG)

Opis: Historia zamówień klientów wraz z wyceną produktów i statusem realizacji.

```
create view Raport_Historia_Zamowien_Klientow
as
select
z.ZamowienieID,
z.DataZamowienia,
z.DataWykonania,
kl.KlientID,
coalesce(ki.Imie + ' ' + ki.Nazwisko, f.Nazwa) as 'Nazwa
    klienta',
case when ki.KlientID is not null then 'Indywidualny' else
    'Firma' end as 'Typ klienta',
kl.Email,
p.Nazwa as 'Nazwa produktu',
kat.Nazwa as 'Kategoria',
sz.IloscProduktu,
sz.CenaJednostkowa,
sz.Rabat,
sz.IloscProduktu * sz.CenaJednostkowa * (1 - sz.Rabat) as
    'Wartosc pozycji',
st.Opis as 'Status zamowienia'
from Zamowienia z
join Klienci kl on z.KlientID = kl.KlientID
left join Klienci_indywidualni ki on kl.KlientID = ki.KlientID
left join Firmy f on kl.KlientID = f.KlientID
join Szczegoly_zamowienia sz on z.ZamowienieID = sz.ZamowienieID
join Produkty p on sz.ProduktID = p.ProduktID
join Kategorie kat on p.KategoriaID = kat.KategoriaID
join Stany st on z.StanZamowieniaID = st.StanID
```

7.10 Raport_Zamówienia_Klientów_W_Czasie (WC)

Opis: Liczba zakończonych zamówień, wartość netto i średni rabat klientów w ujęciu miesięcznym, kwartalnym i rocznym.

```
create view Raport_Zamowienia_Klientow_W_Czasie
as
select
year(z.DataZamowienia) as 'Rok',
datepart(quarter, z.DataZamowienia) as 'Kwartal',
month(z.DataZamowienia) as 'Miesiac',
kl.KlientID,
coalesce(ki.Imie + ' ' + ki.Nazwisko, f.Nazwa) as 'Nazwa
    klienta',
count(distinct z.ZamowienieID) as 'Liczba zamowien',
sum(sz.IloscProduktu * sz.CenaJednostkowa * (1 - sz.Rabat)) as
    'Wartosc netto',
round(avg(sz.Rabat) * 100, 2) as 'Sredni rabat procent'
from Zamowienia z
join Klienci kl on z.KlientID = kl.KlientID
left join Klienci_indywidualni ki on kl.KlientID = ki.KlientID
left join Firmy f on kl.KlientID = f.KlientID
join Szczegoly_zamowienia sz on z.ZamowienieID = sz.ZamowienieID
join Stan_zamowienia stz on z.ZamowienieID = stz.ZamowienieID
join Stany s on stz.StanID = s.StanID
where s.StanID = 6
group by grouping sets (
    (year(z.DataZamowienia), datepart(quarter, z.DataZamowienia),
    month(z.DataZamowienia), kl.KlientID,
    coalesce(ki.Imie + ' ' + ki.Nazwisko, f.Nazwa)),
    (year(z.DataZamowienia), datepart(quarter, z.DataZamowienia),
    kl.KlientID, coalesce(ki.Imie + ' ' + ki.Nazwisko, f.Nazwa)),
    (year(z.DataZamowienia), kl.KlientID,
    coalesce(ki.Imie + ' ' + ki.Nazwisko, f.Nazwa))
)
```

7.11 Raport_Sprzedaz_Tygodniowo (JG)

```
create view Raport_Sprzedaz_Tygodniowo as
select
year(z.DataZamowienia) as 'Rok',
datepart(week, z.DataZamowienia) as 'Numer tygodnia',
format(dateadd(day, 1 - datepart(weekday, z.DataZamowienia),
  z.DataZamowienia), 'yyyy-MM-dd') as 'Poczatek tygodnia',
k.KategoriaID,
k.Nazwa as 'Kategoria',
count(distinct z.ZamowienieID) as 'Liczba zamowien',
sum(sz.IloscProduktu) as 'Sprzedanych sztuk',
sum(sz.IloscProduktu * sz.CenaJednostkowa) as 'Wartosc brutto',
sum(sz.IloscProduktu * sz.CenaJednostkowa * (1 - sz.Rabat)) as
  'Wartosc netto',
sum(sz.IloscProduktu * sz.CenaJednostkowa * sz.Rabat) as
  'Udzielone rabaty'
from Zamowienia z
join Szczegoly_zamowienia sz on z.ZamowienieID = sz.ZamowienieID
join Produkty p on sz.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
group by year(z.DataZamowienia), datepart(week,
  z.DataZamowienia),
dateadd(day, 1 - datepart(weekday, z.DataZamowienia),
  z.DataZamowienia),
k.KategoriaID, k.Nazwa
```

7.12 Raport_Plan_Wytworzenia_Produktów_Dzienny (MG)

Opis: Monitorowanie postępów produkcji poszczególnych produktów w cyklach dziennych.

```
CREATE OR ALTER VIEW Raport_Plan_Wytworzenia_Produktow_Dzienny AS
SELECT
  CAST(pr.DataWpisu AS DATE) AS Dzień,
  CASE DATEPART(WEEKDAY, pr.DataWpisu)
    WHEN 1 THEN 'Niedziela'
    WHEN 2 THEN 'Poniedzialek'
    WHEN 3 THEN 'Wtorek'
    WHEN 4 THEN 'Sroda'
    WHEN 5 THEN 'Czwartek'
    WHEN 6 THEN 'Piatek'
    WHEN 7 THEN 'Sobota'
  END AS DzieńTygodnia,
  p.ProduktID,
  p.Nazwa AS NazwaProduktu,
  k.Nazwa AS Kategoria,
  COUNT(pr.ProdukcjaID) AS LiczbaZleceń,
  SUM(pr.Ilość) AS ZaplanowanaIlość,
  SUM(CASE WHEN pr.DataRozpoczęcia IS NOT NULL AND
    pr.DataZakończenia IS NULL
    THEN pr.Ilość ELSE 0 END) AS WTrakcieProdukcji,
  SUM(CASE WHEN pr.DataZakończenia IS NOT NULL THEN pr.Ilość ELSE
    0 END) AS Zrealizowano,
  SUM(CASE WHEN pr.DataRozpoczęcia IS NULL THEN pr.Ilość ELSE 0
    END) AS Oczekujące,
  ROUND(CAST(SUM(CASE WHEN pr.DataZakończenia IS NOT NULL THEN
    pr.Ilość ELSE 0 END)
    AS FLOAT) / NULLIF(SUM(pr.Ilość), 0) * 100, 2) AS
    ProcentRealizacji
FROM Produkcja pr
INNER JOIN Produkty p ON pr.ProduktID = p.ProduktID
INNER JOIN Kategorie k ON p.KategoriaID = k.KategoriaID
GROUP BY CAST(pr.DataWpisu AS DATE), DATEPART(WEEKDAY,
  pr.DataWpisu),
  p.ProduktID, p.Nazwa, k.Nazwa
GO
```

8 Procedury

8.1 Procedura_Zamówienia_Klienta_W_Przedziale (JG)

Opis: Procedura wyświetlająca zamówienia danego klienta w danym przedziale czasowym.

Parametry: @KlientID, @DataOd, @DataDo

```
create or alter procedure
    Procedura_Zamowienia_Klienta_W_Przedziale
@KlientID int,
@DataOd datetime,
@DataDo datetime
as
begin
select
z.ZamowienieID,
z.DataZamowienia,
p.Nazwa as 'Produkt',
sz.IloscProduktu,
sz.CenaJednostkowa,
sz.Rabat * 100 as 'Rabat procent',
sz.IloscProduktu * sz.CenaJednostkowa * (1 - sz.Rabat) as
    'Wartosc',
st.Opis as 'Status'
from Zamowienia z
join Szczegoly_zamowienia sz on z.ZamowienieID = sz.ZamowienieID
join Produkty p on sz.ProduktID = p.ProduktID
join Stany st on z.StanZamowieniaID = st.StanID
where z.KlientID = @KlientID
and z.DataZamowienia between @DataOd and @DataDo
order by z.DataZamowienia desc
end

-- Przyklad uzycia:
EXEC Procedura_Zamowienia_Klienta_W_Przedziale
@KlientID = 2,
@DataOd = '2024-07-12',
@DataDo = '2026-07-12'
```

8.2 Procedura_Koszty_Produkcji_W_Przedziale (JG)

Opis: Procedura wyświetlająca koszty produkcji w danym przedziale czasowym.

Parametry: @DataOd, @DataDo

```
create or alter procedure Procedura_Koszty_Produkcji_W_Przedziale
@DataOd datetime,
@DataDo datetime
as
begin
select
k.Nazwa as 'Kategoria',
count(pr.ProdukcjaID) as 'Liczba produkcji',
sum(pr.Ilosc) as 'Wyprodukowanych sztuk',
sum(pr.Ilosc * sp.KosztProdukcji) as 'Koszty materialow',
sum(pr.Ilosc * sp.Robocizna) as 'Koszty robocizny',
sum(pr.Ilosc * (sp.KosztProdukcji + sp.Robocizna)) as 'Koszty
    calkowite'
from Produkcja pr
join Produkty p on pr.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
join Szczegoly_produktu sp on p.ProduktID = sp.ProduktID
    and pr.DataZakonczenia between sp.WaznyOd and
        isnull(sp.WaznDo, getdate())
where pr.DataZakonczenia between @DataOd and @DataDo
group by k.Nazwa
end
```

8.3 Procedura_Sprzedaz_W_Przedziale (JG)

Opis: Procedura wyświetlająca wartości sprzedaży w danym przedziale czasowym dla każdej z kategorii.

Parametry: @DataOd, @DataDo

```
create or alter procedure Procedura_Sprzedaz_W_Przedziale
@DataOd datetime,
@DataDo datetime
as
begin
select
k.Nazwa as 'Kategoria',
count(distinct z.ZamowienieID) as 'Liczba zamowien',
count(distinct z.KlientID) as 'Liczba klientow',
sum(sz.IloscProduktu) as 'Sprzedanych sztuk',
sum(sz.IloscProduktu * sz.CenaJednostkowa) as 'Przychod brutto',
sum(sz.IloscProduktu * sz.CenaJednostkowa * (1 - sz.Rabat)) as
    'Przychod netto',
sum(sz.IloscProduktu * sz.CenaJednostkowa * sz.Rabat) as
    'Wartosc rabatow'
from Zamowienia z
join Szczegoly_zamowienia sz on z.ZamowienieID = sz.ZamowienieID
join Produkty p on sz.ProduktID = p.ProduktID
join Kategorie k on p.KategoriaID = k.KategoriaID
where z.DataZamowienia between @DataOd and @DataDo
group by k.Nazwa
end
```

8.4 Procedura_Zlecenie_Produkcji (WC)

Opis: Procedura obsługująca proces zlecenia produkcji produktu.

Parametry: @ProduktID, @IloścID, @PracownikID

```
create procedure Procedura_Zlecenie_Produkcji
@produktid int,
@ilosc int,
@pracownikid int
as
begin
set nocount on;
begin tran;

if exists (
    select 1
    from struktura_produkту sp
    join magazyn_czesci mc on mc.czescid = sp.czescid
    where sp.produktid = @produktid
    and mc.stan < sp.ilosc * @ilosc
)
begin
    raiserror('brak wystarczajacej ilosci czesci w magazynie', 16,
        1);
    rollback;
    return;
end

update mc
set mc.stan = mc.stan - sp.ilosc * @ilosc
from magazyn_czesci mc
join struktura_produkту sp on mc.czescid = sp.czescid
where sp.produktid = @produktid;

insert into produkcja (produktid, pracownikid, ilosc, datawpisu)
values (@produktid, @pracownikid, @ilosc, getdate());

commit;
end;
```


8.5 Procedura_Zamówienie_Części (WC)

Opis: Rejestruje nowe zamówienie dostawy części w bazie danych.

Parametry: @realizacjaid, @ilosc, @koszt, @datadoreczenia (opcjonalny)

```
create procedure Procedura_Zamowienie_Czesci
@realizacjaid int,
@ilosc int,
@koszt money,
@datadoreczenia datetime = null
as
begin
set nocount on;

insert into dostawy_czesci
(datazamowienia, datadoreczenia, ilosc, kosztczesci,
    realizacjaid)
values (getdate(), @datadoreczenia, @ilosc, @koszt,
    @realizacjaid);
end;
```

8.6 Procedura_Sprawdz_Dostepnosc_Dla_Zamowienia (MG)

Opis: Weryfikuje wykonalnořć zamówienia. Zwraca dwa zestawy danych: szczegółową listę produktów oraz ogółny raport dostepnořci i terminowořci.

Parametry: @ZamowienieID

```
CREATE OR ALTER PROCEDURE
    [dbo].[Procedura_Sprawdz_Dostepnosc_Dla_Zamowienia]
@ZamowienieID INT
AS
BEGIN
    SET NOCOUNT ON;

    IF NOT EXISTS (SELECT 1 FROM Zamowienia WHERE ZamowienieID =
        @ZamowienieID)
    BEGIN
        RAISERROR('Zamowienie o podanym ID nie istnieje.', 16, 1);
        RETURN;
    END

    DECLARE @TerminPreferowany DATETIME;
    SELECT @TerminPreferowany = TerminPreferowany FROM Zamowienia
    WHERE ZamowienieID = @ZamowienieID;

    SELECT * FROM
        dbo.Funkcja_Analiza_Dostepnosci_Zamowienia(@ZamowienieID);

    SELECT
        @ZamowienieID AS ZamowienieID,
        @TerminPreferowany AS TerminPreferowany,
        SUM(CASE WHEN Status = 'Dostepny' THEN 1 ELSE 0 END) AS
            ProduktowDostepnych,
        SUM(CASE WHEN Status = 'Do produkcji' THEN 1 ELSE 0 END) AS
            ProduktowDoProdukcji,
        SUM(CASE WHEN Status = 'Brak czesci' THEN 1 ELSE 0 END) AS
            ProduktowBezCzesci,
        CASE
            WHEN SUM(CASE WHEN CzyMozliweWTerminie = 0 THEN 1 ELSE 0 END)
                > 0 THEN 'NIE'
            ELSE 'TAK'
        END AS RealizacjaWTerminieMozliwa,
        MAX(SzacowanyTermin) AS NajpozniejszyTermin
    FROM dbo.Funkcja_Analiza_Dostepnosci_Zamowienia(@ZamowienieID);
END
GO
```

8.7 Procedura _Zloz _Zamowienie _Z _Produkcja (MG)

Opis: Inicjuje proces zamówienia poprzez utworzenie nagłowa zamówienia i powiązanej płatności.

Parametry: @KlientID, @PracownikID, @MetodaPlatnosciID, @TerminPreferowany (opcjonalny)

```
CREATE OR ALTER PROCEDURE
    [dbo].[Procedura_Zloz_Zamowienie_Z_Produkcja]
    @KlientID INT,
    @PracownikID INT,
    @MetodaPlatnosciID INT,
    @TerminPreferowany DATETIME = NULL
AS
BEGIN
    SET NOCOUNT ON;
    BEGIN TRY
        BEGIN TRANSACTION;

        IF NOT EXISTS (SELECT 1 FROM Klienci WHERE KlientID = @KlientID)
        BEGIN
            RAISERROR('Klient o podanym ID nie istnieje.', 16, 1);
            RETURN;
        END

        DECLARE @PlatnoscID INT;
        INSERT INTO Platnosci (MetodaID, DataPlatnosci)
        VALUES (@MetodaPlatnosciID, GETDATE());
        SET @PlatnoscID = SCOPE_IDENTITY();

        DECLARE @ZamowienieID INT;
        INSERT INTO Zamowienia (DataZamowienia, PracownikID, KlientID,
            PlatnoscID, TerminPreferowany, StanZamowieniaID)
        VALUES (GETDATE(), @PracownikID, @KlientID, @PlatnoscID,
            @TerminPreferowany, 1);
        SET @ZamowienieID = SCOPE_IDENTITY();

        INSERT INTO Stan_zamowienia (ZamowienieID, StanID, OdKiedy,
            DoKiedy)
        VALUES (@ZamowienieID, 1, GETDATE(), NULL);

        COMMIT TRANSACTION;

        SELECT @ZamowienieID AS NoweZamowienieID,
            'Zamowienie utworzone. Dodaj produkty za pomoca
            Procedura_Dodaj_Produkt_Z_Produkcja.'
            AS Komunikat,
            @TerminPreferowany AS TerminPreferowany;

    END TRY
    BEGIN CATCH
```

```
ROLLBACK TRANSACTION;  
THROW;  
END CATCH  
END  
GO
```

9 Triggery

9.1 tr_dostawyczesci_update_magazyn (WC)

Opis: Trigger aktualizujący stan magazynu części po dodaniu lub zrealizowaniu dostawy.

```
create trigger tr_dostawyczesci_update_magazyn  
on dostawy_czesci  
after insert, update  
as  
begin  
set nocount on;  
  
update mc  
set  
mc.stan = mc.stan + i.ilosc,  
mc.ostatniaaktualizacja = getdate()  
from magazyn_czesci mc  
join realizacja_dostawy rd on rd.czescid = mc.czescid  
join inserted i on i.realizacjaaid = rd.realizacjaaid  
left join deleted d on d.dostawaaid = i.dostawaaid  
where  
i.datadoreczenia is not null  
and (d.dostawaaid is null or d.datadoreczenia is null);  
end;
```

9.2 tr_stanzamowienia_jedenaktywny (MG)

Opis: Trigger kontrolujący, aby każde zamówienie miało tylko jeden aktywny stan.

```
create trigger tr_stanzamowienia_jedenaktywny
on stan_zamowienia
after insert, update
as
begin
set nocount on;

if exists (
    select zamowienieid
    from stan_zamowienia
    where dokiedy is null
    group by zamowienieid
    having count(*) > 1
)
begin
    rollback transaction;
    raiserror('zamowienie moze miec tylko jeden aktywny stan', 16,
        1);
end
end;
```

9.3 tr_produkcja_zakonczona (WC)

Opis: Trigger aktualizujący stan magazynu produktów po zakończeniu produkcji.

```
create trigger tr_produkcja_zakonczona
on produkcja
after update
as
begin
set nocount on;

update mp
set
mp.stan = mp.stan + i.ilosc,
mp.ostatniaaktualizacja = getdate()
from magazyn_produkow mp
join inserted i on mp.produktid = i.produktid
join deleted d on d.produkcjaid = i.produkcjaid
where
d.datazakonczenia is null
and i.datazakonczenia is not null;
end;
```

9.4 tr_szczegoly_produkту_domykanie (WC)

Opis: Trigger aktualizujący poprzednie rekordy szczegółów produktu po dodaniu nowego wpisu.

```
create trigger tr_szczegoly_produkту_domykanie
on szczegoly_produkту
after insert
as
begin
set nocount on;

update sp
set sp.WaznDo = dateadd(day, -1, i.WaznyOd)
from szczegoly_produkту sp
join inserted i on sp.ProduktID = i.ProduktID
where sp.ProduktSzczegyID <> i.ProduktSzczegyID
and (sp.WaznDo is null or sp.WaznDo > i.WaznyOd);
end;
```

9.5 tr_szczegoly_produkту_kontrola_chronologii (WC)

Opis: Trigger wymuszający chronologiczne wprowadzanie szczegółów produktu.

```
create trigger tr_szczegoly_produkту_kontrola_chronologii
on szczegoly_produkту
after insert, update
as
begin
set nocount on;

if exists (
    select 1
    from szczegoly_produkту sp
    join inserted i on sp.ProduktID = i.ProduktID
    where sp.ProduktSzczegyID <> i.ProduktSzczegyID
    and sp.WaznyOd > i.WaznyOd
)
begin
    rollback transaction;
    raiserror('nie mozna wstawic lub zmienic szczegolow produktu
na date
wcześniejsza niz istniejące nowsze wpisy', 16, 1);
end
end;
```

10 Funkcje

10.1 Funkcja_Wartoř_Zamówienia (MG)

Opis: Oblicza całkowitą wartość zamówienia (z rabatami), zwraca typ MONEY.

```
CREATE OR ALTER FUNCTION
    Funkcja_Wartosc_Zamowienia(@ZamowienieID INT)
RETURNS MONEY
AS
BEGIN
    DECLARE @Wartosc MONEY;

    SELECT @Wartosc = SUM(IloscProduktu * CenaJednostkowa * (1 -
        Rabat))
    FROM Szczegoly_zamowienia
    WHERE ZamowienieID = @ZamowienieID;

    RETURN ISNULL(@Wartosc, 0);
END
GO

-- Przyklad uzycia:
SELECT dbo.Funkcja_Wartosc_Zamowienia(10);
```

10.2 Funkcja_Czas_Realizacji_Produkcji (MG)

Opis: Szacowany czas produkcji w godzinach, zwraca typ FLOAT.

```
CREATE OR ALTER FUNCTION
    [dbo].[Funkcja_Czas_Realizacji_Produkcji]
(@ProduktID INT, @Ilosc INT)
RETURNS FLOAT
AS
BEGIN
    DECLARE @Czas FLOAT;

    SELECT @Czas = CzasProdukcji * @Ilosc
    FROM Szczegoly_produktu
    WHERE ProduktID = @ProduktID AND WaznDo IS NULL;

    RETURN ISNULL(@Czas, 0);
END
GO

-- Przyklad uzycia:
SELECT dbo.Funkcja_Czas_Realizacji_Produkcji(1, 5);
```

10.3 Funkcja_Adres_Klienta (MG)

Opis: Pełny adres klienta, zwraca typ VARCHAR.

```
CREATE OR ALTER FUNCTION [dbo].[Funkcja_Adres_Klienta](@KlientID
    INT)
RETURNS VARCHAR(200)
AS
BEGIN
    DECLARE @Adres VARCHAR(200);

    SELECT @Adres =
        ISNULL(k.Ulica + ', ', '') +
        k.NrBudynku +
        ISNULL('/', ' ') + k.NrLokalu, '' + ', ' +
        k.KodPocztowy + ', ' +
        m.Nazwa + ', ' +
        p.Nazwa
    FROM Klienci k
    INNER JOIN Miejscowosci m ON k.MiejscowoscID = m.MiejscowoscID
    INNER JOIN Panstwa p ON m.PanstwoID = p.PanstwoID
    WHERE k.KlientID = @KlientID;

    RETURN @Adres;
END
GO

-- Przykład użycia:
SELECT dbo.Funkcja_Adres_Klienta(4);
```


10.4 Funkcja_Czy_Wymaga_Uzupełnienia (WC)

Opis: Sprawdza, czy stan magazynowy produktu spadł poniżej minimalnego poziomu, zwraca typ BIT.

```
CREATE OR ALTER FUNCTION
    [dbo].[Funkcja_Czy_Wymaga_Uzupełnienia](@ProduktID INT)
RETURNS BIT
AS
BEGIN
    DECLARE @Wymaga BIT = 0;

    IF EXISTS (
        SELECT 1 FROM Magazyn_produkow
        WHERE ProduktID = @ProduktID AND Stan < MinimalnaIlosc
    )
    BEGIN
        SET @Wymaga = 1;
    END

    RETURN @Wymaga;
END
GO
```

10.5 Funkcja_Czy_Mozna_Wyprodukowac (MG)

Opis: Sprawdza, czy można wyprodukować produkt z części w magazynie, zwraca typ BIT.

```
CREATE OR ALTER FUNCTION [dbo].[Funkcja_Czy_Mozna_Wyprodukowac]
(@ProduktID INT, @Ilosc INT)
RETURNS BIT
AS
BEGIN
    DECLARE @Mozliwe BIT = 1;

    IF EXISTS (
        SELECT 1
        FROM Struktura_produkту sp
        INNER JOIN Magazyn_czesci mc ON sp.CzescID = mc.CzescID
        WHERE sp.ProduktID = @ProduktID
        AND mc.Stan < (sp.Ilosc * @Ilosc)
    )
    BEGIN SET @Mozliwe = 0; END

    IF NOT EXISTS (SELECT 1 FROM Struktura_produkту WHERE ProduktID
        = @ProduktID)
    BEGIN SET @Mozliwe = 0; END

    RETURN @Mozliwe;
END
GO
```

10.6 Funkcja _Brakujace _Części (WC)

Opis: Zwraca tabelę z informacjami o tym, ile i jakich części brakuje dla danej ilości produktu.

```
CREATE OR ALTER FUNCTION [dbo].[Funkcja_Brakujace_Czesci]
(@ProduktID INT, @Ilosc INT)
RETURNS TABLE
AS
RETURN
(
SELECT
c.CzescID,
c.Nazwa AS NazwaCzesci,
sp.Ilosc * @Ilosc AS WymaganaIlosc,
ISNULL(mc.Stan, 0) AS StanMagazynowy,
(sp.Ilosc * @Ilosc) - ISNULL(mc.Stan, 0) AS Brakuje
FROM Struktura_produktu sp
INNER JOIN Czesc c ON sp.CzescID = c.CzescID
LEFT JOIN Magazyn_czesci mc ON c.CzescID = mc.CzescID
WHERE sp.ProduktID = @ProduktID
AND ISNULL(mc.Stan, 0) < (sp.Ilosc * @Ilosc)
);
GO

-- Przyklad uzycia:
SELECT * FROM dbo.Funkcja_Brakujace_Czesci(1, 100);
```

10.7 Funkcja_Funkcja_Szacowany_Termin_Realizacji (JG)

Opis: Zwraca datę szacowanego terminu realizacji dla danej ilości produktu.

```
CREATE OR ALTER FUNCTION
    [dbo].[Funkcja_Szacowany_Termin_Realizacji]
(@ProduktID INT, @Ilosc INT)
RETURNS DATETIME
AS
BEGIN
    DECLARE @CzasProdukcjiGodziny FLOAT;
    DECLARE @GodzinyPracyDziennie FLOAT = 8.0;
    DECLARE @SzacowanaDataZakonczenia DATETIME;

    SELECT @CzasProdukcjiGodziny = ISNULL(CzasProdukcji, 1.0) *
        @Ilosc
    FROM Szczegoly_produktu
    WHERE ProduktID = @ProduktID AND WaznDo IS NULL;

    DECLARE @LiczbaDni INT =
        CEILING(ISNULL(@CzasProdukcjiGodziny, 8) /
            @GodzinyPracyDziennie) + 1;

    SET @SzacowanaDataZakonczenia = DATEADD(DAY, @LiczbaDni,
        GETDATE());
    RETURN @SzacowanaDataZakonczenia;
END
GO
```

10.8 Funkcja_Analiza_Dostepnosci_Zamowienia (JG)

Opis: Zwraca tabelę z informacjami o dostępności lub możliwości wyprodukowania sztuk produktu w kontekście danego zamówienia.

```
CREATE OR ALTER FUNCTION
    [dbo].[Funkcja_Analiza_Dostepnosci_Zamowienia]
(@ZamowienieID INT)
RETURNS TABLE
AS
RETURN
(
    SELECT
        sz.ProduktID,
        p.Nazwa AS NazwaProduktu,
        sz.IloscProduktu AS ZamowionaIlosc,
        ISNULL(mp.Stan, 0) AS StanMagazynowy,
        CASE
            WHEN ISNULL(mp.Stan, 0) >= sz.IloscProduktu THEN 0
            ELSE sz.IloscProduktu - ISNULL(mp.Stan, 0)
        END AS DoProdukcji,
        CASE
            WHEN ISNULL(mp.Stan, 0) >= sz.IloscProduktu THEN 'Dostepny'
            WHEN dbo.Funkcja_Czy_Mozna_Wyprodukowac(sz.ProduktID,
                sz.IloscProduktu - ISNULL(mp.Stan, 0)) = 1 THEN 'Do
                produkcji'
            ELSE 'Brak czesci'
        END AS Status,
        dbo.Funkcja_Szacowany_Termin_Realizacji(
            sz.ProduktID,
            CASE WHEN sz.IloscProduktu > ISNULL(mp.Stan, 0)
                THEN sz.IloscProduktu - ISNULL(mp.Stan, 0) ELSE 0 END
        ) AS SzacowanyTermin,
        z.TerminPreferowany,
        CASE
            WHEN ISNULL(mp.Stan, 0) >= sz.IloscProduktu THEN 1
            ELSE dbo.Funkcja_Czy_Mozliwa_Realizacja_W_Terminie(
                sz.ProduktID,
                sz.IloscProduktu,
                ISNULL(z.TerminPreferowany, DATEADD(DAY, 30, GETDATE()))
            )
        END AS CzyMozliweWTerminie
    FROM Szczegoly_zamowienia sz
    INNER JOIN Produkty p ON sz.ProduktID = p.ProduktID
    INNER JOIN Zamowienia z ON sz.ZamowienieID = z.ZamowienieID
    LEFT JOIN Magazyn_produktow mp ON sz.ProduktID = mp.ProduktID
    WHERE sz.ZamowienieID = @ZamowienieID
);
GO
```

11 Użytkownicy i uprawnienia

11.1 Administrator (JG)

Uprawnienia:

- CONTROL na całej bazie danych
- Może tworzyć, modyfikować i usuwać tabele, widoki, procedury, funkcje
- Zarządza użytkownikami i rolami
- Ma dostęp do wszystkich danych

```
CREATE ROLE administrator;  
GRANT CONTROL ON DATABASE::[u_grabos] TO administrator;
```

11.2 Kierownik sprzedaży (JG)

Uprawnienia: SELECT, INSERT, UPDATE na tabelach zamówień, klientów i płatności; SELECT na widokach sprzedażowych; wykonywanie procedur sprzedażowych.

```
CREATE ROLE KierownikSprzedazy;

GRANT SELECT, INSERT, UPDATE ON Zamowienia TO KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Szczegoly_zamowienia TO
    KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Klienci TO KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Klienci_indywidualni TO
    KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Firmy TO KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Platnosci TO KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Metody_platnosci TO
    KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Stany TO KierownikSprzedazy;
GRANT SELECT, INSERT, UPDATE ON Stan_zamowienia TO
    KierownikSprzedazy;

GRANT SELECT ON Raport_Sprzedaz_Tygodniowo TO KierownikSprzedazy;
GRANT SELECT ON Raport_Sprzedaz_Miesiecznie TO
    KierownikSprzedazy;
GRANT SELECT ON Raport_Zamowienia_Klientow_W_Czasie TO
    KierownikSprzedazy;
GRANT SELECT ON Informacje_zamowienia TO KierownikSprzedazy;
GRANT SELECT ON Miesieczna_sprzedaz TO KierownikSprzedazy;
GRANT SELECT ON Raport_Historia_Zamowien_Klientow TO
    KierownikSprzedazy;
GRANT SELECT ON Klienci_pelne_informacje TO KierownikSprzedazy;

GRANT EXECUTE ON Procedura_Sprzedaz_W_Przedziale TO
    KierownikSprzedazy;
GRANT EXECUTE ON Procedura_Zamowienia_Klienta_W_Przedziale TO
    KierownikSprzedazy;
```

11.3 Kierownik produkcji (WC)

Upewnienia: SELECT, INSERT, UPDATE na tabelach produkcyjnych; SELECT na widokach produkcyjnych; wykonywanie procedur i funkcji produkcyjnych.

```
CREATE ROLE KierownikProdukcji;

GRANT SELECT, INSERT, UPDATE ON Produkcja TO KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Szczegoly_produktu TO
    KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Struktura_produktu TO
    KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Magazyn_produktow TO
    KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Magazyn_czesci TO
    KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Czesc TO KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Materialy TO KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Produkty TO KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Kategorie TO KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Dostawy_Czesci TO
    KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Realizacja_Dostawy TO
    KierownikProdukcji;
GRANT SELECT, INSERT, UPDATE ON Dostawcy TO KierownikProdukcji;

GRANT EXECUTE ON Procedura_Koszty_Produkcji_W_Przedziale TO
    KierownikProdukcji;
GRANT EXECUTE ON Funkcja_Czas_Realizacji_Produkcji TO
    KierownikProdukcji;
GRANT EXECUTE ON Funkcja_Czy_Wymaga_Uzupelnienia TO
    KierownikProdukcji;
```


11.4 Analityk (MG)

Uprawnienia: SELECT na tabelach analitycznych; SELECT na wszystkich widokach raportowych; wykonywanie procedur i funkcji niemodyfikujących danych.

```
CREATE ROLE Analityk;

GRANT SELECT ON Zamowienia TO Analityk;
GRANT SELECT ON Szczegoly_zamowienia TO Analityk;
GRANT SELECT ON Produkty TO Analityk;
GRANT SELECT ON Kategorie TO Analityk;
GRANT SELECT ON Produkcja TO Analityk;
GRANT SELECT ON Szczegoly_produkту TO Analityk;
GRANT SELECT ON Klienci TO Analityk;
GRANT SELECT ON Klienci_indywidualni TO Analityk;
GRANT SELECT ON Firmy TO Analityk;

-- Widoki
GRANT SELECT ON Raport_Sprzedaz_Tygodniowo TO Analityk;
GRANT SELECT ON Raport_Sprzedaz_Miesiecznie TO Analityk;
GRANT SELECT ON Raport_Zamowienia_Klientow_W_Czasie TO Analityk;
GRANT SELECT ON Raport_Koszty_Produkcji_Miesiecznie TO Analityk;
GRANT SELECT ON Raport_Stany_Magazynowe_Czesci TO Analityk;
GRANT SELECT ON Raport_Stany_Magazynowe_Produktow TO Analityk;
GRANT SELECT ON Historia_Produkcji TO Analityk;
GRANT SELECT ON Efektywnosc_Pracownikow_Sprzedaz TO Analityk;

-- Procedury i funkcje
GRANT EXECUTE ON Procedura_Sprzedaz_W_Przedziale TO Analityk;
GRANT EXECUTE ON Procedura_Koszty_Produkcji_W_Przedziale TO
    Analityk;
GRANT EXECUTE ON Procedura_Zamowienia_Klienta_W_Przedziale TO
    Analityk;
GRANT EXECUTE ON Funkcja_Adres_Klienta TO Analityk;
GRANT EXECUTE ON Funkcja_Czas_Realizacji_Produkcji TO Analityk;
GRANT EXECUTE ON Funkcja_Wartosc_Zamowienia TO Analityk;
```

11.5 Kierownik kadr (JG)

Uprawnienia: SELECT, INSERT, UPDATE na tabelach pracowników; SELECT na widoku efektywności pracowników.

```
CREATE ROLE KierownikKadr;

GRANT SELECT, INSERT, UPDATE ON Pracownicy TO KierownikKadr;
GRANT SELECT, INSERT, UPDATE ON Stanowiska TO KierownikKadr;
GRANT SELECT ON Efektywnosc_Pracownikow_Sprzedaz TO
    KierownikKadr;
```

12 Indeksy

12.1 Zamówienia

```
CREATE INDEX IX_Zamowienia_KlientData  
ON Zamowienia(KlientID, DataZamowienia DESC);  
  
CREATE INDEX IX_Zamowienia_DataStatus  
ON Zamowienia(DataZamowienia, StanZamowieniaID)  
INCLUDE (KlientID, PracownikID, PlatnoscID);
```

12.2 Produkcja

```
CREATE INDEX IX_Produkcja_Status  
ON Produkcja(DataRozpoczecia, DataZakonczenia)  
INCLUDE (ProduktID, Ilosc, PracownikID);
```

12.3 Struktura_produktu

```
CREATE INDEX IX_Struktura_Produkt  
ON Struktura_produktu(ProduktID)  
INCLUDE (CzescID, Ilosc);  
  
CREATE INDEX IX_Struktura_Czesc  
ON Struktura_produktu(CzescID)  
INCLUDE (ProduktID, Ilosc);
```

12.4 Magazyn_produktów

```
CREATE INDEX IX_MagazynProduktow_Stan  
ON Magazyn_produktow(Stan);
```

12.5 Magazyn_części

```
CREATE INDEX IX_MagazynCzesci_Stan  
ON Magazyn_czesci(Stan);
```

12.6 Klienci

```
CREATE INDEX IX_Klienci_Miejscowosc  
ON Klienci(MiejscowoscID)  
INCLUDE (Email, NumerTelefonu);
```

12.7 Szczegóły zamówienia

```
CREATE INDEX IX_Szczegoly_ZamowieniaProdukt  
ON Szczegoly_zamowienia(ZamowienieID, ProduktID)  
INCLUDE (IloscProduktu, CenaJednostkowa, Rabat);
```

12.8 Produkty

```
CREATE INDEX IX_Produkty_Kategoria  
ON Produkty(KategoriaID, CzyDostepny)  
INCLUDE (Nazwa, CenaJednostkowa);
```

12.9 Dostawy_Części

```
CREATE INDEX IX_DostawyCzesci_Data  
ON Dostawy_Czesci(DataZamowienia, DataDoreczenia)  
INCLUDE (Ilosc, RealizacjaID);
```