```
CREATE DATABASE komunikacja_miejska
USE komunikacja_miejska

CREATE TABLE Dzialy (
IdDzialu INT,
NazwaDzialu VARCHAR(50) NOT NULL,
Adres VARCHAR(50) NOT NULL,
NumerTelefonu VARCHAR(9) NOT NULL,
PRIMARY KEY (IdDzialu),
)

CREATE TABLE Stanowiska (
Stanowisko VARCHAR (50),
IdDzialu INT NOT NULL,
Pensja MONEY NOT NULL,
Hierarchia INT NOT NULL,
PRIMARY KEY (Stanowisko),
FOREIGN KEY (IdDzialu) REFERENCES Dzialy (IdDzialu),
)



CREATE TABLE Pracownicy (
IdPracownika INT,
Imie VARCHAR (50) NOT NULL,
Nazwisko VARCHAR (50) NOT NULL,
PESEL CHAR(11) UNIQUE NOT NULL,
DataZatrudnienia DATE NOT NULL,
NumerPrawaJazdy CHAR(10) NULL,
Stanowisko VARCHAR(50) NOT NULL,
IdPrzelozonego INT REFERENCES Pracownicy(IdPracownika),
PRIMARY KEY (IdPracownika),
FOREIGN KEY (Stanowisko) REFERENCES Stanowiska (Stanowisko) - DODAŁAM KLUCZ OBCY
)


CREATE TABLE Urlopy_Pracowników (
IdPracownika INT,
OdKiedy DATE,
DoKiedy DATE NOT NULL,
PRIMARY KEY (IdPracownika, OdKiedy),
FOREIGN KEY (IdPracownika) REFERENCES Pracownicy (IdPracownika),
CONSTRAINT CK1 CHECK (DoKiedy > OdKiedy)
)


CREATE TABLE RodzajeBiletow (
RodzajBiletu VARCHAR(50), -- np. ulgowy, normalny
Opis VARCHAR(256), --kogo dotyczą jakie ulgi, dokładny opis
PRIMARY KEY (RodzajBiletu)
)

CREATE TABLE Bilety (
NazwaBiletu VARCHAR(50),  -- np. 20-minutowy, jednoprzejazdowy, 90-minutowy
RodzajBiletu VARCHAR(50), -- np. ulgowy, normalny
Cena MONEY NOT NULL,
PRIMARY KEY (NazwaBiletu, RodzajBiletu),
FOREIGN KEY (RodzajBiletu) REFERENCES RodzajeBiletow (RodzajBiletu)
```

```
)



CREATE TABLE Przystanki (
IdPrzystanku INT,
NazwaPrzystanku VARCHAR(50) NOT NULL,
Ulica VARCHAR(50) NOT NULL,
Rodzaj VARCHAR(50), -- np. na żądanie
PRIMARY KEY (IdPrzystanku)
)


CREATE TABLE Linie (
NumerLinii INT,
RodzajLinii VARCHAR(50) NOT NULL, -- np.nocna, przyśpieszona
IdPrzystankuPoczatkowego INT NOT NUll,
IdPrzystankuKoncowego INT NOT NUll,
PRIMARY KEY (NumerLinii),
FOREIGN KEY (IdPrzystankuPoczatkowego) REFERENCES Przystanki (IdPrzystanku),
FOREIGN KEY (IdPrzystankuKoncowego) REFERENCES Przystanki (IdPrzystanku)
)


CREATE TABLE Trasy (
NumerLinii INT,
NumerPorządkowyPrzystanku INT,
IdPrzystanku INT NOT NULL,
PRIMARY KEY (NumerLinii, NumerPorządkowyPrzystanku),
FOREIGN KEY (NumerLinii) REFERENCES Linie (NumerLinii),
FOREIGN KEY (IdPrzystanku) REFERENCES Przystanki (IdPrzystanku)
)


CREATE TABLE Zajezdnie (
IdZajezdni INT,
NazwaZajezdni VARCHAR(50) NOT NULL,
AdresZajezdni VARCHAR(100) NOT NULL,
PRIMARY KEY (IdZajezdni),
)


CREATE TABLE Autobusy (
IdAutobusu INT,
NrRejestracyjny VARCHAR(10) UNIQUE NOT NULL,
OdKiedy DATETIME NOT NULL,
IloscMiejscSiedzacych INT NOT NULL,
IloscMiejscStojacych INT NOT NULL,
Rodzaj VARCHAR(20) NOT NULL, --nisko-, wysokopodłowgowy
Biletomat BIT NOT NULL, --tak lub nie
MiejsceDlaRowerow BIT NOT NULL, --TAK LUB NIE
MiejsceDlaWozkow BIT NOT NULL, --tak lub nie
Przegląd VARCHAR(20),--atualny lub nie
DataAktualnegoPrzegladu DATETIME NOT NULL,
IdZajezdni INT NOT NULL,
PRIMARY KEY (IdAutobusu),
```

```
FOREIGN KEY (IdZajezdni) REFERENCES Zajezdnie (IdZajezdni),
)


CREATE TABLE Obsluga_techniczna (
IdPojazdu INT,
NrRejestracyjny VARCHAR(10) UNIQUE NOT NULL,
OdKiedy DATETIME NOT NULL,
DataAktualnegoPrzeglądu DATETIME NOT NULL,
Rodzaj VARCHAR(50),
PRIMARY KEY (IdPojazdu),
)


CREATE TABLE DziennikAwarii (
IdAutobusu INT,
IdPojazduObslugi INT,
Data DATE,
[Opis zdarzenia] VARCHAR(256),
PRIMARY KEY (IdAutobusu, IdPojazduObslugi, Data),
FOREIGN KEY (IdAutobusu) REFERENCES Autobusy (IdAutobusu),
FOREIGN KEY (IdPojazduObslugi) REFERENCES Obsluga_Techniczna (IdPojazdu)
)


CREATE TABLE Kursy (
IdKursu INT,
NumerLinii INT NOT NULL,
Kierunek INT NOT NULL, --czyli przystanek końcowy
PRIMARY KEY (IdKursu),
FOREIGN KEY (NumerLinii) REFERENCES Linie (NumerLinii),
)


CREATE TABLE Rozklady (
IdPrzystanku INT,
IdKursu INT,
Godzina TIME(0),
Dzien VARCHAR(10), --powszedni, sobota, (niedziela), święta
PRIMARY KEY (IdPrzystanku, IdKursu, Godzina, Dzien),
FOREIGN KEY (IdPrzystanku) REFERENCES Przystanki (IdPrzystanku),
FOREIGN KEY (IdKursu) REFERENCES Kursy (IdKursu),
)


CREATE TABLE Pojazdy_i_Linie (
IdAutobusu INT,
NumerLinii INT NOT NULL,
PRIMARY KEY (IdAutobusu, NumerLinii),
FOREIGN KEY (IdAutobusu) REFERENCES Autobusy (IdAutobusu),
FOREIGN KEY (NumerLinii) REFERENCES Linie (NumerLinii),
)


CREATE TABLE Grafiki_Pracownikow (
IdKursu INT,
IdPracownika INT,
Godzina TIME(0),
```

```
PRIMARY KEY (IdKursu, IdPracownika, Godzina),
FOREIGN KEY (IdKursu) REFERENCES Kursy (IdKursu),
FOREIGN KEY (IdPracownika) REFERENCES Pracownicy (IdPracownika),
)


WITH MyCTE (Poczatek, Koniec, Odleglosc)
AS (
SELECT IdPrzelozonego, IdPracownika, 1
FROM Pracownicy
WHERE IdPrzelozonego IS NOT NULL
UNION ALL
SELECT MyCTE.Poczatek, P.IdPracownika,
MyCTE.Odleglosc + 1
FROM MyCTE JOIN Pracownicy AS P
ON MyCTE.Koniec = P.IdPrzelozonego
)
SELECT Poczatek AS IdPrzelozonego, Koniec AS IdPodwladnego INTO Przelozeni  FROM MyCTE
ORDER BY Poczatek, Koniec

-- WYZWALACZE

GO
CREATE TRIGGER tr1_instead_of_insert on Pracownicy
INSTEAD OF INSERT
AS
       DECLARE @COUNT1 INT
       SELECT @COUNT1 = COUNT(*) FROM dbo.Pracownicy
       DECLARE @COUNT_IN INT
       SELECT @COUNT_IN = COUNT(*) FROM inserted
       INSERT INTO Pracownicy
    SELECT * FROM   inserted
    WHERE  EXISTS (SELECT 1
WHERE  ((Stanowisko = N'Kierowca' AND NumerPrawaJazdy IS NOT NULL) OR (Stanowisko <>
N'Kierowca')))
       DECLARE @COUNT2 INT
       SELECT @COUNT2 = COUNT(*) FROM dbo.Pracownicy

IF ( @COUNT_IN + @COUNT1 > @COUNT2 )
       PRINT 'Error: Nie wszystkie rekordy zostały dodane! Kierowca musi posiadac numer
prawa jazdy!'
ELSE
       PRINT 'Rekord(y) został dodany.'
GO

CREATE TRIGGER tr2_instead_of_insert ON Autobusy
INSTEAD OF INSERT
AS
       DECLARE @NrZajezdni INT
       SELECT @NrZajezdni = IdZajezdni FROM Inserted
       DECLARE @LiczbaAutobusow INT
       SELECT @LiczbaAutobusow = COUNT(IdAutobusu) FROM Autobusy WHERE
IdZajezdni=@NrZajezdni
IF @LiczbaAutobusow <10
BEGIN
       INSERT INTO Autobusy
       SELECT * FROM inserted
       PRINT 'Oddano autobus do uzytku'
```

```
	END
ELSE
	BEGIN
	PRINT 'Limit osiągniety! Umiesc autobus w innej zajezdni.'
	END
GO


CREATE TRIGGER tr3_instead_of_insert
ON Grafiki_Pracownikow
INSTEAD OF INSERT
AS
	DECLARE @czaspracy TIME(0)
	SELECT @czaspracy = Godzina FROM inserted
	DECLARE @IdKursuG INT
	SELECT @IdKursuG = IdKursu FROM inserted
IF EXISTS (SELECT IdPracownika =@Pracownik FROM Pracownicy WHERE Stanowisko NOT LIKE
'kierowca' )
	PRINT 'DODANO!'
ELSE IF EXISTS (SELECT IdPracownika =@Pracownik FROM Pracownicy WHERE Stanowisko LIKE
'kierowca' )
BEGIN

	IF EXISTS (SELECT * FROM Rozklady WHERE IdKursu = @IdKursuG AND Godzina =
@czaspracy)
		BEGIN
		INSERT INTO Grafiki_Pracownikow
		SELECT * FROM inserted
		PRINT 'DODANO!'
		END
	ELSE
	BEGIN
		PRINT 'Nie można dodać rekordu! Podany kurs lub godzina odjazdu nie istnieje
w rozkładzie!'
	END
	END
ELSE BEGIN
	PRINT 'DODANO!'
	END
GO

CREATE TRIGGER tr4_instead_of_update ON Stanowiska
INSTEAD OF UPDATE
AS
	DECLARE @hierarchia INT
	SELECT @hierarchia = Hierarchia FROM inserted
	DECLARE @pensja MONEY
	SELECT @pensja = Pensja FROM inserted
IF EXISTS
(SELECT * FROM Stanowiska WHERE Hierarchia + 1 = @hierarchia AND Pensja<=@pensja)
	BEGIN
	PRINT 'Wprowadzana wartość pensji jest za wysoka dla tego stanowiska!'
	END
ELSE IF EXISTS
(SELECT * FROM Stanowiska WHERE Hierarchia - 1 = @hierarchia AND Pensja>=@pensja)
	BEGIN
	PRINT 'Wprowadzana wartość pensji jest za niska dla tego stanowiska!'
	END
```

```
ELSE
	BEGIN
	UPDATE Stanowiska
	SET Pensja = @pensja
	WHERE Hierarchia = @hierarchia
	PRINT 'Zaktualizowano!'
	END
GO

CREATE TRIGGER tr5_after_update ON Autobusy
AFTER UPDATE
AS
	DECLARE @aktualnadata DATETIME
	SELECT @aktualnadata = GETDATE()
	DECLARE @dataprzegladu DATETIME
	SELECT @dataprzegladu = DataAktualnegoPrzegladu FROM inserted
IF @dataprzegladu>=@aktualnadata
	BEGIN
	UPDATE Autobusy
	SET Przegląd = 'aktualny'
	WHERE IdAutobusu = (SELECT IdAutobusu FROM inserted)
	PRINT 'Zaktualizowano stan przeglądu!'
	END
IF EXISTS
(SELECT * FROM Autobusy WHERE DataAktualnegoPrzegladu<@aktualnadata)
	BEGIN
	UPDATE Autobusy
	SET Przegląd = 'nieaktualny'
	WHERE DataAktualnegoPrzegladu<@aktualnadata
	PRINT 'Upłynął termin ważności przeglądu technicznego!'
	END
GO

--FUNKCJE

CREATE FUNCTION RozkladDlaPrzystanku (@nazwa VARCHAR(50), @numer INT)
RETURNS @RozkladPrzystankow TABLE (GodzinaOdjazdu TIME(0), Kierunek VARCHAR(50)
) AS BEGIN
	DECLARE @przystanek INT
	SET @przystanek = (SELECT IdPrzystanku FROM Przystanki WHERE NazwaPrzystanku =
@nazwa)
	INSERT INTO @RozkladPrzystankow
	SELECT RK.Godzina AS [Godzina Odjazdu], P.NazwaPrzystanku AS Kierunek FROM (
	SELECT R.Godzina, K.Kierunek FROM Rozklady AS R LEFT JOIN Kursy AS K
	ON R.IdKursu = K.IdKursu
	WHERE K.NumerLinii = @numer AND R.IdPrzystanku = @przystanek)
	AS RK LEFT JOIN Przystanki AS P
	ON RK.Kierunek = P.IdPrzystanku
	RETURN
END
GO

CREATE FUNCTION JakieLinie (@nazwa VARCHAR(50))
RETURNS @LinieKierunki TABLE (NumerLinii INT, Kierunek VARCHAR(50)
) AS BEGIN
	INSERT INTO @LinieKierunki
	SELECT KPR.NumerLinii, PP.NazwaPrzystanku AS Kierunek FROM (
	SELECT K.NumerLinii, K.Kierunek FROM Kursy AS K RIGHT JOIN (
```

```
	SELECT DISTINCT R.IdKursu FROM Rozklady AS R LEFT JOIN Przystanki AS P
	ON R.IdPrzystanku = P.IdPrzystanku
	WHERE P.NazwaPrzystanku = @nazwa ) AS PR
	ON K.IdKursu = PR.IdKursu) AS KPR LEFT JOIN
	Przystanki AS PP
	ON KPR.Kierunek = PP.IdPrzystanku
	RETURN
END
GO

CREATE FUNCTION JakiePrzystanki (@numer INT)
RETURNS @RozkladPrzystankow TABLE (NazwaPrzystanku VARCHAR(50)
) AS BEGIN
	INSERT INTO @RozkladPrzystankow
	SELECT P.NazwaPrzystanku FROM (
	SELECT IdPrzystanku FROM Trasy
	WHERE NumerLinii = @numer) AS T LEFT JOIN Przystanki AS P
	ON T.IdPrzystanku = P.IdPrzystanku
	RETURN
END
GO


CREATE FUNCTION PrzystankiZUlicy (@nazwa VARCHAR(50))
RETURNS @RozkladPrzystankow TABLE (NazwaPrzystanku VARCHAR(50))
AS BEGIN
	INSERT INTO @RozkladPrzystankow
	SELECT NazwaPrzystanku FROM Przystanki
	WHERE Ulica = @nazwa
	RETURN
END
GO

CREATE FUNCTION AutobusyDoPrzegladu ()
RETURNS @Autobusy TABLE (IdAutobusu INT, [Data aktualnego przegladu] DATE
) as BEGIN
	INSERT INTO @Autobusy
	SELECT IdAutobusu, DataAktualnegoPrzegladu FROM Autobusy
	WHERE DATEDIFF(MONTH, GETDATE(), DataAktualnegoPrzegladu) <= 1
	RETURN
END
GO

CREATE FUNCTION JakaLiniaDojechac (@nazwa1 VARCHAR(50), @nazwa2 VARCHAR(50))
RETURNS @Linie TABLE ([Numer Linii] INT
) AS BEGIN
	INSERT INTO @Linie
	SELECT R.NumerLinii FROM (
	SELECT K.NumerLinii, COUNT(*) AS Licznik FROM (
	SELECT T.NumerLinii, P.NazwaPrzystanku FROM Trasy AS T LEFT JOIN Przystanki AS P
	ON T.IdPrzystanku = P.IdPrzystanku
	WHERE P.NazwaPrzystanku = @nazwa1 OR P.NazwaPrzystanku = @nazwa2) AS K
	GROUP BY K.NumerLinii) AS R
	WHERE R.Licznik = 2
	RETURN
END
GO
```

```
--WIDOKI

CREATE VIEW AutobusyDlaUzytkownika AS
SELECT IdAutobusu, IloscMiejscSiedzacych, IloscMiejscStojacych, Rodzaj, Biletomat,
MiejsceDlaRowerow, MiejsceDlaWozkow FROM Autobusy
GO

CREATE VIEW KierowcyWPracy AS
SELECT DISTINCT P.IdPracownika, Imie, Nazwisko FROM Pracownicy P LEFT JOIN
Urlopy_Pracowników U ON P.IdPracownika=U.IdPracownika
WHERE Stanowisko = 'kierowca' AND (P.IdPracownika NOT IN (SELECT IdPracownika FROM
Urlopy_Pracowników) OR DoKiedy<GETDATE()OR (DoKiedy>GETDATE() AND OdKiedy>GETDATE()))
GO

CREATE VIEW LinieNocne AS
SELECT * FROM Linie WHERE RodzajLinii LIKE 'nocna'
GO

CREATE VIEW RozkladySwieta AS
SELECT * FROM Rozklady WHERE Dzien LIKE 'święta'
GO

--PROCEDURY

CREATE PROCEDURE ZmianaCen
(@zmiana FLOAT)
AS
BEGIN
UPDATE Bilety SET Cena = ROUND((Cena + (Cena * @zmiana)),1)
END
GO

CREATE PROCEDURE IleAwarii
(@OdKiedy DATE, @DoKiedy DATE, @ile INT )
AS
BEGIN
SELECT P.IdAutobusu, P.Data, P.[Opis zdarzenia] FROM DziennikAwarii AS P LEFT JOIN
(SELECT IdAutobusu, COUNT(*) AS Licznik FROM DziennikAwarii
WHERE Data BETWEEN @OdKiedy AND @DoKiedy
GROUP BY IdAutobusu) AS K
ON P.IdAutobusu = K.IdAutobusu
WHERE K.Licznik >= @ile AND P.Data BETWEEN @OdKiedy AND @DoKiedy
END
GO

CREATE PROCEDURE AktualizacjaDaty (@numer INT, @data DATETIME)
AS
BEGIN
UPDATE Obsluga_techniczna
SET DataAktualnegoPrzeglądu = @data
WHERE IdPojazdu = @numer
END
GO

CREATE PROCEDURE Ile_kursow (@dzien VARCHAR(10))
AS
BEGIN
```

```
SELECT DISTINCT K.NumerLinii, COUNT(*) AS LiczbaKursow FROM (Rozklady R JOIN Kursy K ON
R.IdKursu=K.IdKursu) JOIN Linie L ON K.NumerLinii=L.NumerLinii
WHERE R.IdPrzystanku=L.IdPrzystankuPoczatkowego AND R.Dzien=@dzien
GROUP BY K.NumerLinii, Dzien
END
GO

CREATE PROCEDURE LiniaKryteria
(@rodzaj VARCHAR(50), @biletomat BIT, @rower BIT, @wozek BIT)
AS
BEGIN

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 1 AND @rower = 1 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy'
AND Biletomat = 1 AND MiejsceDlaRowerow = 1 AND MiejsceDlaWozkow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 1 AND @rower = 1 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy'
AND Biletomat = 1 AND MiejsceDlaRowerow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 1 AND @rower = 0 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy' AND Biletomat = 1 AND
MiejsceDlaWozkow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 0 AND @rower = 1 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy' AND MiejsceDlaRowerow = 1
AND MiejsceDlaWozkow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 1 AND @rower = 1 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE Biletomat = 1 AND MiejsceDlaRowerow = 1 AND
MiejsceDlaWozkow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 1 AND @rower = 0 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy' AND Biletomat = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 0 AND @rower = 0 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy' AND MiejsceDlaWozkow = 1)
AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 0 AND @rower = 1 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy' AND MiejsceDlaRowerow = 1)
AS S
ON PL.IdAutobusu = S.IdAutobusu
```

```sql
IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 1 AND @rower = 1 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE Biletomat = 1 AND MiejsceDlaRowerow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 1 AND @rower = 0 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE Biletomat = 1 AND MiejsceDlaWozkow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 0 AND @rower = 1 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE MiejsceDlaRowerow = 1 AND MiejsceDlaWozkow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'niskopodlogowy' AND @biletomat = 0 AND @rower = 0 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE rodzaj = 'niskopodlogowy') AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 1 AND @rower = 0 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE Biletomat = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 0 AND @rower = 1 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE MiejsceDlaRowerow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 0 AND @rower = 0 AND @wozek = 1)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy WHERE MiejsceDlaWozkow = 1) AS S
ON PL.IdAutobusu = S.IdAutobusu

IF (@rodzaj = 'wysokopodlogowy' AND @biletomat = 0 AND @rower = 0 AND @wozek = 0)
SELECT DISTINCT PL.NumerLinii FROM Pojazdy_i_Linie AS PL RIGHT JOIN (
SELECT IdAutobusu FROM Autobusy) AS S
ON PL.IdAutobusu = S.IdAutobusu
END
GO
```