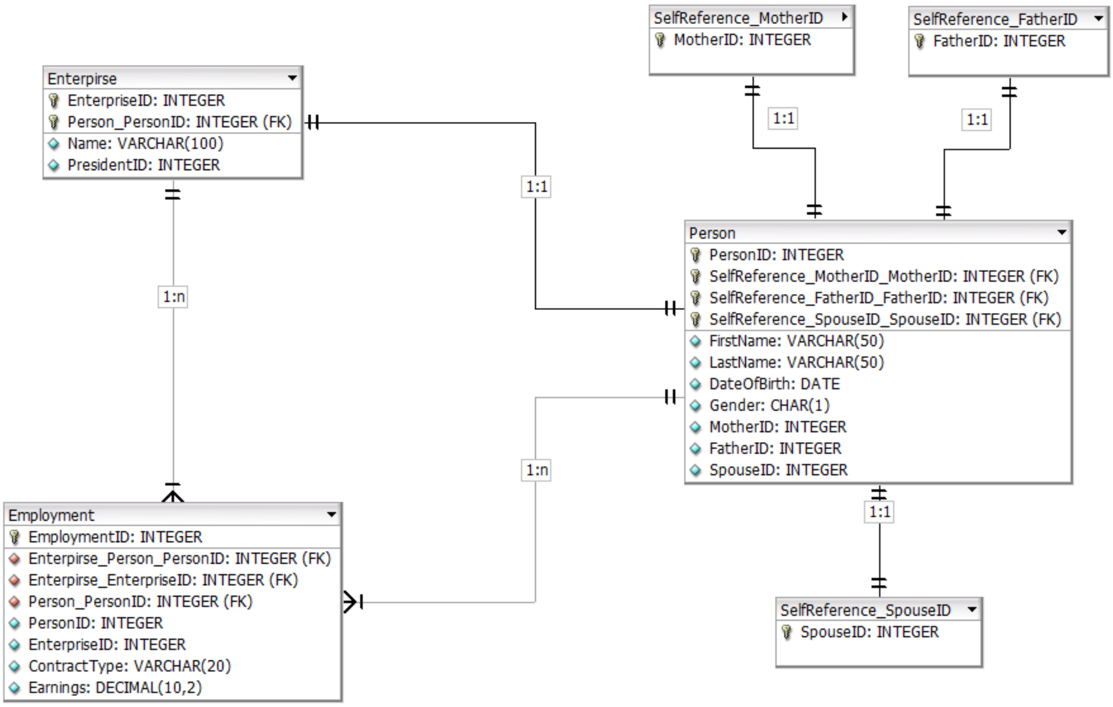
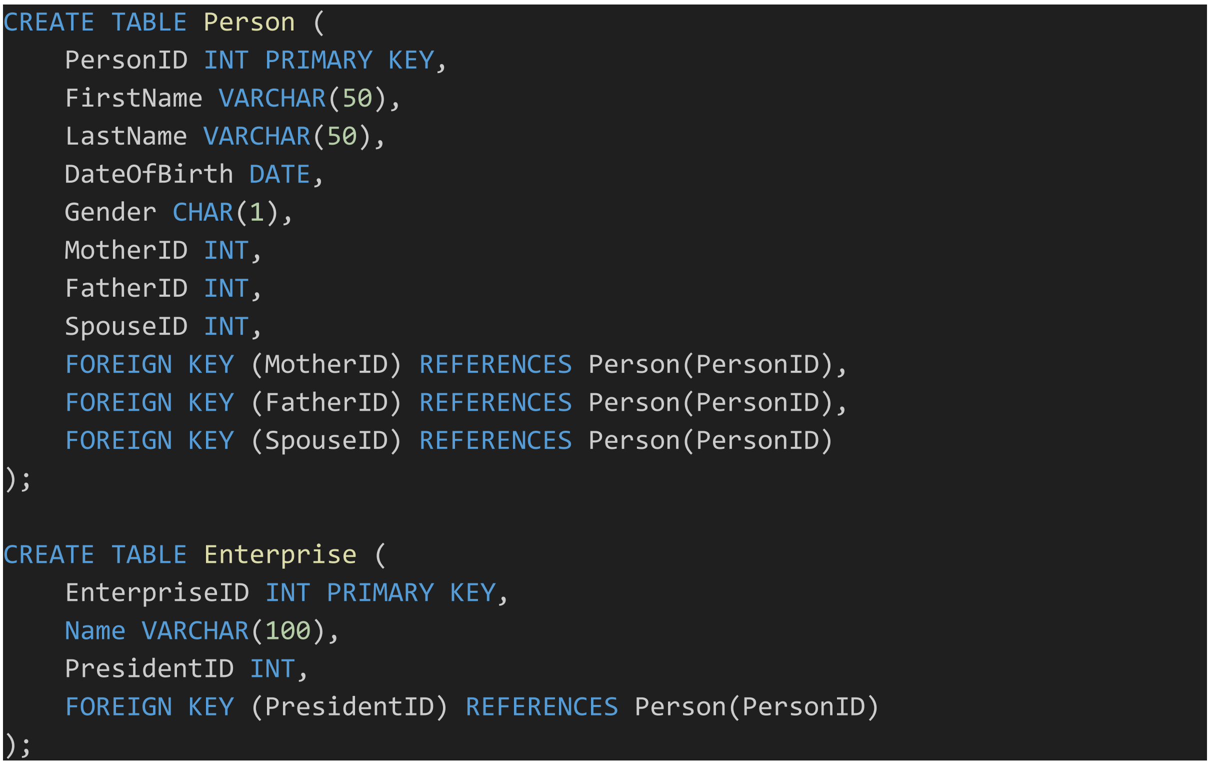
Zadanie 2

Diagram ERD:



Kod SQL Do stworzenia tabel z diagramu



```
CREATE TABLE Person (
    PersonID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DateOfBirth DATE,
    Gender CHAR(1),
    MotherID INT,
    FatherID INT,
    SpouseID INT,
    FOREIGN KEY (MotherID) REFERENCES Person(PersonID),
    FOREIGN KEY (FatherID) REFERENCES Person(PersonID),
    FOREIGN KEY (SpouseID) REFERENCES Person(PersonID)
);

CREATE TABLE Enterprise (
    EnterpriseID INT PRIMARY KEY,
    Name VARCHAR(100),
    PresidentID INT,
    FOREIGN KEY (PresidentID) REFERENCES Person(PersonID)
);
```

```sql
CREATE TABLE Employment (
    EmploymentID INT PRIMARY KEY,
    PersonID INT,
    EnterpriseID INT,
    ContractType VARCHAR(20),
    Earnings DECIMAL(10, 2),
    FOREIGN KEY (PersonID) REFERENCES Person(PersonID),
    FOREIGN KEY (EnterpriseID) REFERENCES Enterprise(EnterpriseID)
);

CREATE INDEX idx_person_gender ON Person(Gender);
CREATE INDEX idx_person_motherid ON Person(MotherID);
CREATE INDEX idx_person_fatherid ON Person(FatherID);
CREATE INDEX idx_employment_contracttype ON Employment(ContractType);
CREATE INDEX idx_employment_earnings ON Employment(Earnings);
CREATE INDEX idx_person_spouseid ON Person(SpouseID);
```

W celu **optymalizacji** działania tej bazy danych najlepiej zindeksować kolumny które są samo-odwołaniami tabeli Person, gdzie każda osoba może mieć jednego Ojca (inny rekord typu Person), Matkę lub Partnera. Co ważne mamy tutaj możliwość ciągów rodzinnych, prababcia, babcia itd. Dlatego też zindeksowanie tych kolumn jest najważniejsze. Ważne są też zarobki które mogą pojawiać się w kilku różnych rekordach dla każdej osoby, oraz typy kontraktu. Indeksowanie po płci również może przyśpieszyć czas działania kwerend.

Zadanie 1)

```sql
SELECT P1.FirstName, P1.LastName
FROM Person P1
JOIN Person P2 ON P1.PersonID = P2.MotherID OR P1.PersonID = P2.FatherID
JOIN Person P3 ON P2.PersonID = P3.MotherID OR P2.PersonID = P3.FatherID
WHERE P3.Gender = 'F'
GROUP BY P1.PersonID, P1.FirstName, P1.LastName
ORDER BY COUNT(P3.PersonID) DESC
LIMIT 1;
```

Zadanie 2)

Średnia z obu umów

```sql
SELECT
    AVG(CASE WHEN E.ContractType = 'Contract' THEN 1 ELSE 0 END) AS
AvgContractEmployees,
    AVG(CASE WHEN E.ContractType = 'Employment' THEN 1 ELSE 0 END) AS
AvgEmploymentContractEmployees,
    AVG(E.Earnings) AS AvgEarnings
FROM Employment E;
```

Osobne średnie w zależności od typu umowy

```
SELECT
    AVG(CASE WHEN E.ContractType = 'Contract' THEN E.Earnings ELSE NULL END)
AS AvgContractEarnings,
    AVG(CASE WHEN E.ContractType = 'Employment' THEN E.Earnings ELSE NULL END)
AS AvgEmploymentEarnings,
    AVG(CAST(CASE WHEN E.ContractType = 'Contract' THEN E.PersonID ELSE NULL
END AS FLOAT)) AS AvgContractEmployees,
    AVG(CAST(CASE WHEN E.ContractType = 'Employment' THEN E.PersonID ELSE NULL
END AS FLOAT)) AS AvgEmploymentContractEmployees
FROM (
    SELECT E1.EnterpriseID, E1.ContractType, E1.Earnings, E1.PersonID
    FROM Employment E1
    GROUP BY E1.EnterpriseID, E1.PersonID, E1.ContractType, E1.Earnings
) E;
```

Zadanie 3)

```
WITH FamilyEarnings AS (
    SELECT
        P1.PersonID, P1.FirstName, P1.LastName,
        COALESCE(SUM(E1.Earnings), 0) AS PersonEarnings,
        COALESCE(SUM(E2.Earnings), 0) AS SpouseEarnings,
        COALESCE(SUM(E3.Earnings), 0) AS ChildEarnings,
        COALESCE(SUM(E4.Earnings), 0) AS ChildSpouseEarnings
    FROM
        Person P1
        LEFT JOIN Person P2 ON P1.SpouseID = P2.PersonID
        LEFT JOIN Person P3 ON P1.PersonID = P3.MotherID OR P1.PersonID =
P3.FatherID
        LEFT JOIN Person P4 ON P3.SpouseID = P4.PersonID
        LEFT JOIN Employment E1 ON P1.PersonID = E1.PersonID
        LEFT JOIN Employment E2 ON P2.PersonID = E2.PersonID
        LEFT JOIN Employment E3 ON P3.PersonID = E3.PersonID
        LEFT JOIN Employment E4 ON P4.PersonID = E4.PersonID
    GROUP BY P1.PersonID, P1.FirstName, P1.LastName
)
SELECT FirstName, LastName,
       (PersonEarnings + SpouseEarnings + ChildEarnings + ChildSpouseEarnings)
AS TotalFamilyEarnings
FROM FamilyEarnings
ORDER BY TotalFamilyEarnings
LIMIT 1;
```