Katedra
Informatyki i Automatyki
Politechniki Rzeszowskiej

Inżynieria Oprogramowania

1.ERD i ORM’y

Opracował: Maciej Penar

Spis treści

1. Zanim zaczniemy

Zrelaksować się i przypomnieć sobie teorię dot. Diagramów ERD. Można wybrać dowolną notację, ale notacja Chen'a oraz Martin'a są preferowane.

Materiały do ERD:

- Info o różnych notacjach: https://pl.wikipedia.org/wiki/Notacja_Martina
- Materiały z Lucidchart: https://www.lucidchart.com/pages/ER-diagram-symbols-and-meaning
- Wprowadzenie do Systemów Baz Danych, rozdział 3, Elmasri Navathe

Oprogramowanie:

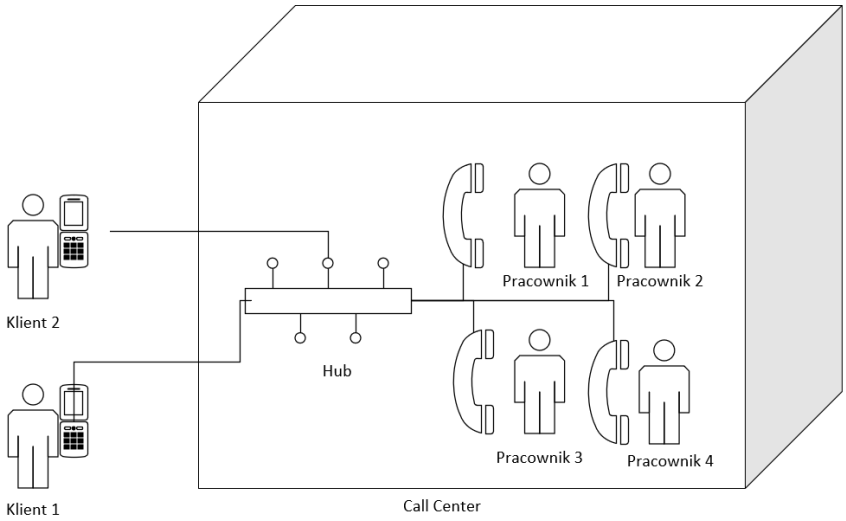
- Lucidchart (oprogramowanie w modelu SasS do modelowania)
- Enterprise Architect (http://www.sparxsystems.com/products/ea/)
- Visual Paradigm (https://www.visual-paradigm.com/download/community.jsp)
- Visio (https://products.office.com/pl-pl/visio/flowchart-software)

2. (8 pkt) ERD

Pro-tip: najlepiej zainstalować któryś z programów (lub użyć Lucidchart) bo będzie dużo copy-paste'a .

Pro-tip 2: istnieje duża dowolność, ale nie komplikować nadmiernie. Proszę.

1. (1.5 pkt) Narysuj diagram ERD wraz z atrybutami reprezentujący dane dot. Filmów oraz gwiazd filmowych. Zweryfikuj model pod kątem następujących pytań:
    a. Czy można ustalić reżysera filmu?
    b. Czy można ustalić obsadę filmu?
    c. Czy można ustalić gwiazdy które współpracują z reżyserem?
    d. Czy można ustalić gatunek filmu?
    e. Itp.
2. (1 pkt) Narysuj diagram ERD wraz z atrybutami reprezentujący dane zbierane dla hotelu (np. klient, rezerwacje). Jakie pytania mogą zweryfikować mikro-świat?
3. (1 pkt) Narysuj diagram ERD wraz z atrybutami reprezentujący dane zbierane dla wypożyczalni filmów. Jakie pytania mogą zweryfikować mikro-świat?
4. (0.5 pkt) Jaka jest [zasadnicza] różnica pomiędzy zadaniem 2 a 3?
5. (4 pkt) Narysować diagram ERD dla uproszczonego systemu zgłoszeń w Call Center z rysunku 1.
    a. Call Center mniej więcej tak że mamy zarezerwowany 1 numer pod który dzwonią klienci. System przekierowuje rozmowę na wolne stanowisko (Pracownika).
    b. W systemach zgłoszeń zazwyczaj jest tak że wydzielone jest kilka tzw. Lin wsparcia (Support Line). Najpierw klient łączy się z tzw. Pierwszą linią wsparcia i jeśli konsultant (pracownik) nie jest w stanie sobie poradzić to klient jest przekierowany na drugą (kolejną) linię wspracia (tzw. Eskalacja).
    c. Serwisy działają tak, że konsultanci wypełniają formularz zgłoszeniowy / zamawiają kuriera odbierającego sprzęt.
    d. Zazwyczaj zgłoszeniom w takich systemach nadawany jest pewien priorytet. Konsultanci mogą powiedzieć „Oddzwonimy".
    e. Zazwyczaj śledzony jest czas połączeń który kontrastowany jest z czasem który konsultant przesiedział bezczynnie.
    f. Załóżmy że interesuje nas efektywność Call Center – czyli:
        i. Ile wykonywanych jest połączeń?
        ii. Jak długie są połączenia?
        iii. Ile mamy dostępnych konsultantów? Którzy są nierobami?
        iv. Śledzenie zgłoszeń – np. wg. Priorytetu. Czy nasze Call Center udało się rozwiązać problem (zamknąć zgłoszenie) czy byliśmy niekompetentni (odrzucono zgłoszenie). Która linia wsparcia była aktywna?
        v. Załóżmy dwa rodzaje zgłoszeń: takie które konsultanci są w stanie rozwiązać i zamknąć ad-hoc, oraz takie które opisują proces (* patrz niżej). Zauważmy że oba zgłoszenia można opisać wspólnymi atrybutami np. **czasem zgłoszenia,** a także (prawdopodobnie) wspólnymi związkami



Rys 1. Call Center

Komentarz odnośnie procesu (*): serwisanci tak działają – dzwonimy z zepsutym laptopem – zgłoszenie staje się **otwarte**, kurier zostaje zamówiony. Mija kilka dni, kurier przyjeżdża, odbiera sprzęt, zawozi do serwisantów – zmieniają oni zgłoszenie na status np. **odebrano sprzęt**. Jak naprawią, zamieniają zgłoszenie na **gotowe do wysyłki** – za co odpowiada inny dział. Znowu zamówiony jest kurier i sprzęt trafia do klienta z powrotem, po czym zgłoszenie jest ustawiane na **zamknięte.**

3. (4 pkt) ORM'y

SŁOWO WSTĘPU:

W tej części spróbujemy wykonać coś przydatnego: stworzyć aplikację która posiada wbudowaną bazę danych (Embedded Database) i napisać kod który wykonuje proste operacje na niej. Wykorzystamy do tego C#, bazę danych **SQLLite** oraz ORM zwany **ORMLite**.

Żeby zacząć działać trzeba pobrać VisualStudio – jak macie możliwość to w wersji Enterprise, jak nie to wersję Community którą można pobrać: https://www.visualstudio.com/pl/downloads/

Ja korzystam z:

Dodatkowo potrzebujecie klienta SQLite (.exe) który dostępny jest na na repo albo na oficjalnej stronie:

- Oficjalna strona: https://www.sqlite.org/index.html
- Repo; https://github.com/mpenarprz/InzynieriaOprogramowaniaI4/tree/master/Laboratorium/tools

Podsumowując potrzebujecie:

- Visual Studio z C#
- SQLite.exe

PROJEKT POMOCNICZY

Do pomocy zostawiłem na repo projekt, który **powinien** odpalać się bez narzekania.

Projekt jest tu:
https://github.com/mpenarprz/InzynieriaOprogramowaniaI4/tree/master/Laboratorium/src/OrmTest

Jak odpalić;

1. Po otwarciu **OrmTest.sln** w Visual Studio powinniśmy zobaczyć komunikat: „Some NuGet packages are missing from this solution. Click to restore from you online package sources."
2. Klikamy **Restore**
3. Jeśli mamy odpowiednie SDK to wszystko powinno działać
4. Naciskamy **Start**
5. W ścieżce podanej w zmiennej **ConnectionString** powinniśmy znaleźć plik bazy danych – na repo to **sqlite.db**

Najlepiej debugować kod linijka po linijce i podglądać jak w zmiennych **addressOne, addressTwo, personOne, personTwo, personThree** zmieniają się dane (w tym identyfikatory).

Równolegle warto użyć programu **sqlite.exe** żeby podejrzeć zawartość Bazy Danych, za pomocą wyrażeń SELECT.

Wykonuje się komendą w konsoli:

```
.\sqlite3.exe [ścieżka do bazy danych]

.\sqlite3.exe sqlite.db
```

Otworzy nam się shell SQLite'a.

Najważniejsza komenda: **.help**

Po wpisaniu **.tables** wyświetli nam się lista tabel:

```
sqlite> .tables
Addresses  Persons
```

Po wpisaniu **.fullschema** wyświetlą nam się definicje tabel, można sprawdzić czy ORM nie zrobił czegoś podejrzanego.

```
sqlite> .fullschema
CREATE TABLE IF NOT EXISTS "Addresses"
(
  "Id" INTEGER PRIMARY KEY AUTOINCREMENT,
  "Text" VARCHAR(8000) NOT NULL
);
CREATE TABLE IF NOT EXISTS "Persons"
(
  "Id" INTEGER PRIMARY KEY AUTOINCREMENT,
  "Name" VARCHAR(8000) NOT NULL,
  "Surname" VARCHAR(8000) NOT NULL,
  "Age" INTEGER NOT NULL DEFAULT (-1),
  "AddressId" INTEGER NOT NULL,
CONSTRAINT CHK_Age CHECK (Age >= -1)
);
```

Definicje te są zbudowane z klas tzw. Modelu – zostaną skomentowane one w dalszej części: https://github.com/mpenarprz/InzynieriaOprogramowaniaI4/tree/master/Laboratorium/src/OrmTest/OrmTest/Model

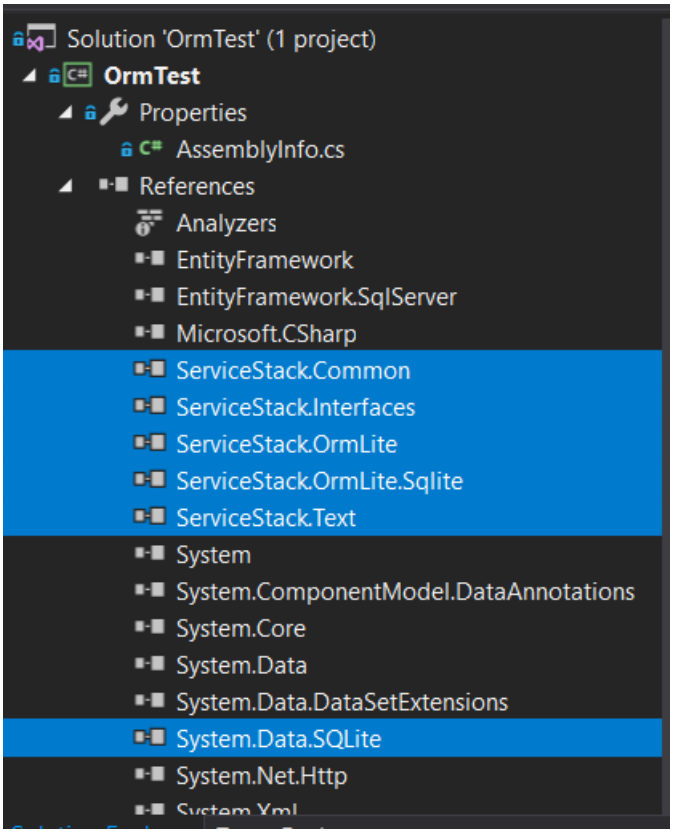
SQLite wspiera wyrażenia SQL'a (dziwne), które piszemy **tylko żeby zweryfikować poprawność działania** ORM'a.

```
sqlite> SELECT * FROM Persons;
1|Maciej|Penar|17|1
3|Róża|Wesoła|21|2
```

JAK ZROBIĆ TO SAMO OD ZERA

Mniej więcej tak:

1. Utworzyć nowy projekt (File -> New -> Project -> Console App (.NET Core) C#
2. Dodać referencję do bibilotek ORM'a
    a. I metoda:
        i. Tools -> Nuget Packet Manager -> Package Manager Console
        ii. Wpisać: Install-Package ServiceStack.OrmLite.Sqlite
        iii. Zostaną pobrane i dołączone biblioteki ORMLite'a i SQLite'a
    b. II metoda:
        i. Tools -> Nuget Packet Manager -> Manage NuGet Packages for Solution
        ii. Wybrać zakładkę Browse
        iii. Znaleźć: ServiceStack.OrmLite.Sqlite
        iv. Zainstalować
3. W drzewku powinniśmy zobaczyć następujące biblioteki



4. **Tworzymy klasy modelu –** najtrudniejszy moment
5. Piszemy program który:
    a. Tworzy fabrykę: `var dbFactory = new OrmLiteConnectionFactory(…)`
    b. Otwiera połączenie do BD: `using (var db = dbFactory.Open())`
    c. Tworzy tabele: `db.CreateTableIfNotExists<Typ>();`
    d. Tworzymy logikę operującą na klasach modelu

KLASY MODELU

Tu można znaleźć co możemy wykorzystać do modelowania klas:

https://github.com/ServiceStack/ServiceStack.OrmLite

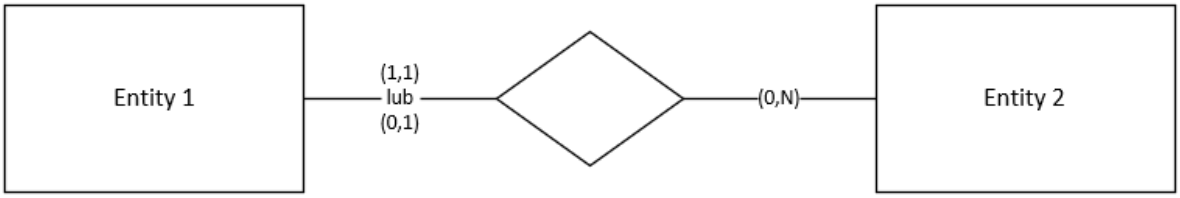
Nie. Nie czytajcie całości. Znajdźcie sekcję: **Create Tables Schemas**

ZADANIE (TU JEST 4 PKT)

Wybrać 2 niezłożone byty. Zamodelować dowolne dwa byty np. samochód – kolor.

Różnych sytuacji powinno być ich w grupie tyle ile osób.

Chodzi o implementację tej sytuacji ERD:



Napisać program który:

1. (1 pkt) Modeluje w/w sytuację
2. Generuje losowe dane
3. (1 pkt) Wykonuje insert >= 1000 rekordów (np. samochody)
4. (1 pkt) Każdemu rekordowi przyporządkowuje inny byt (np. samochodowi przyporządkuje jeden kolor)
5. Sformułować warunek, nazwijmy go W1
6. Usunąć wszystkie rekordy po warunku W1
7. (1 pkt) Wykazać że zapytanie **SELECT** COUNT(*) **FROM** Table **WHERE** W1 zwraca 0.