Моделиране на данни и E/R диаграми (упражнение)

1. Увод

1.1 Моделирането на данни ни помага да опишем обектите от реалния свят (ентитети), техните свойства (атрибути) и връзките помежду им. E/R диаграмите са прост начин да визуализираме тези връзки преди да създадем таблиците.

2. Основни понятия

- **Ентитет**: обект от реалния свят, който искаме да съхраняваме (напр. *Student*).
- **Атрибут**: характеристика на ентитета (напр. *FullName*).
- **Идентификатор**: атрибут(и), които уникално различават записите (PK).
- **Връзка**: как ентитетите се свързват (1:1, 1:N, N:M).
- **Кардиналност**: какъв е броят на участието (единичен/много).

2.1 Кардиналност накратко

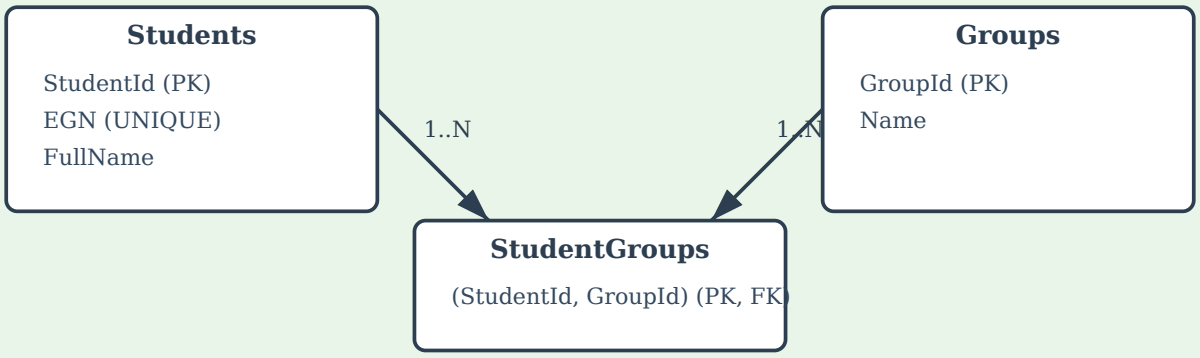
Какво е: колко елемента от единия тип могат да са свързани с елемент от другия.

- **1:1**: един към един (напр. човек ↔ паспорт).
- **1:N**: един към много (напр. клиент ↔ поръчки).
- **N:M**: много към много (напр. студент ↔ група).
- **0..N, 1..N**: 0 означава по избор, 1 означава задължително участие.

3. Прости примери с E/R диаграми

3.1 Студенти и групи (N:M чрез междинна таблица)

Идея: Един студент може да е в много групи, а една група има много студенти.



3.2 Поръчки: Клиент (1:N) Поръчки

Идея: Един клиент има много поръчки; всяка поръчка принадлежи на един клиент.

4. От ER към таблици (релационен модел)

- **1:N**: FK в таблицата от страната „много“.
- **N:M**: междинна таблица с композитен PK от двата FK.
- **1:1**: рядко; често се реализира с уникален FK.

Пример (SQL): Студенти, Групи, Членства

```
CREATE TABLE Students (
  StudentId SERIAL PRIMARY KEY,
  EGN CHAR(10) UNIQUE NOT NULL,
  FullName VARCHAR(100) NOT NULL
);

CREATE TABLE `Groups` (
  GroupId SERIAL PRIMARY KEY,
  Name VARCHAR(100) NOT NULL
);

CREATE TABLE StudentGroups (
  StudentId INT NOT NULL,
  GroupId INT NOT NULL,
  JoinedAt TIMESTAMP NOT NULL DEFAULT NOW(),
  PRIMARY KEY (StudentId, GroupId),
  FOREIGN KEY (StudentId) REFERENCES Students(StudentId) ON DELETE CASCADE,
  FOREIGN KEY (GroupId) REFERENCES `Groups`(GroupId) ON DELETE CASCADE
);
```

5. Кратки задачи

- Нарисувай ER диаграма за библиотека: Readers, Books, Loans (N:M).
- Добави атрибути и избери идентификатори (PK/FK).
- Реши как ще реализираш връзките в таблици.

6. Заключение

6.1 E/R диаграмите улесняват преминаването от идея към правилен релационен модел с ясни ключове и връзки.

7. JOIN: свързване на таблици

Идея: JOIN комбинира редове от свързани таблици (по PK/FK), за да получим смислени резултати.

7.1 INNER JOIN (N:M чрез междинна таблица)

```
-- Студентите с техните групи
SELECT s.FullName, g.Name AS GroupName
FROM Students s
JOIN StudentGroups sg ON sg.StudentId = s.StudentId
JOIN `Groups` g ON g.GroupId = sg.GroupId
ORDER BY s.FullName, g.Name;
```

7.2 LEFT JOIN (всички студенти, дори без група)

```
SELECT s.FullName, g.Name AS GroupName
FROM Students s
LEFT JOIN StudentGroups sg ON sg.StudentId = s.StudentId
LEFT JOIN `Groups` g ON g.GroupId = sg.GroupId
ORDER BY s.FullName;
```

Съвет: INNER JOIN връща само съвпадения; LEFT JOIN връща всички от лявата таблица + NULL за липсващи съвпадения.

7.3 RIGHT JOIN (всички групи, дори без студенти)

```
SELECT g.Name AS GroupName, s.FullName
FROM `Groups` g
RIGHT JOIN StudentGroups sg ON sg.GroupId = g.GroupId
RIGHT JOIN Students s ON s.StudentId = sg.StudentId
ORDER BY g.Name, s.FullName;
```

7.4 GROUP BY + COUNT (брой студенти по група)

```
SELECT g.Name AS GroupName, COUNT(sg.StudentId) AS StudentCount
FROM `Groups` g
LEFT JOIN StudentGroups sg ON sg.GroupId = g.GroupId
GROUP BY g.GroupId, g.Name
ORDER BY g.Name;
```

XAMPP импорт на данни: отворете phpMyAdmin → изберете база `school_db` → Import → изберете файла `01_seed_students_groups.sql` от папката на урока → Go.