Let’s learn about this using a familiar example. You are asked to build a software for Scaler which can handle some base requirements.

The requirements are as follows:

1. Scaler will have multiple **batches**.
2. For each **batch**, we need to store the name, start month and current instructor.
3. Each **batch** of Scaler will have multiple **students**.
4. Each batch has multiple **classes**.
5. For each **class**, store the name, date and time, instructor of the class.
6. For every **student**, we store their name, graduation year, University name, email, phone number.
7. Every student has a **buddy**, who is also a student.
8. A student may move from one **batch** to another.
9. For each batch a **student** moves to, the date of starting is stored.
10. Every student has a **mentor**.
11. For every **mentor**, we store their name and current company name.
12. Store information about all **mentor sessions** (time, duration, student, mentor, student rating, mentor rating).
13. For every **batch**, store if it is an Academy-batch or a DSML-batch.

Steps:

1. Create Tables:
    - Batches
    - Students
    - Classes
    - Mentors
    - Mentor_Sessions

2. Add **Primary Key** and other **attributes about entity**:
    - **Batches(batch_id,** name, start_month, batch_instructor**)**
    - **Students(student_id,** name, grad_year, univ_name, email, phone_number**)**
    - **Classes(class_id,** name, date, time, class_instructor**)**
    - **Mentors(mentor_id,** name, current_company**)**
    - **Mentor_Sessions(mentor_session_id,** time, duration, student_id, mentor_id, student_rating, mentor_rating**)**

3. Represent Relations (Using Cardinality):

    - 1:1 → Add column on any side
    - 1:M → Add column on M's side.
    - M:1 → Add column on M's side.
    - M:M → Create a mapping table.

    - Specify cardinalitites:
    - Batches : Students → 1:M (Add a foreign key in students table)
    - Batches : Classes → 1:M (Add foreign key in classes table)

    - **Batches(batch_id,** name, start_month, batch_instructor, **batch_type_id)**

- **Students(student_id,** name, grad_year, univ_name, email, phone_number, **batch_id, buddy_id)**
- **Classes(class_id,** name, date, time, class_instructor, **batch_id)**
- **Mentors(mentor_id,** name, current_company**)**
- **Mentor_Sessions(mentor_session_id,** time, duration, **student_id**, **mentor_id**, student_rating, mentor_rating**)**

Creating lookup tables here:

- **Student_batch_History**(**batch_history_id**, student_id, batch_id, start_Date)
- **Batch_types**(**batch_type_id**, batch_type)

4. Mention FK and Indexes:

- Batches(**batch_type_id** references Batch_Types(**batch_type_id**))
- Students(**batch_id** references batches(**batch_id**))
- Classes(**batch_id** references batches(**batch_id**))

5. Indexes:
    - Batches → Index on Primay Key i.e **batch_id**

Case Study: Netflix Schema Design

Problem Statement
Design Database Schema for a system like Netflix with following Use Cases.
Use Cases

1. Netflix has **users**.
2. Every **user** has an email and a password.
3. **Users** can create **profiles** to have separate independent environments.
4. Each **profile** has a name and a type. **Type** can be KID or ADULT.
5. There are multiple **videos** on netflix.
6. For each **video**, there will be a title, description and a cast.
7. A **cast** is a list of **actors** who were a part of the **video**. For each **actor** we need to know their **name** and list of videos they were a part of.
8. For every **video**, for any **profile** who watched that video, we need to know the **status** (COMPLETED/ IN PROGRESS).
9. For every **profile** for whom a video is in progress, we want to know their last watch timestamp.

1. Create Tables:
    - Users
    - Profiles
    - Profile_Types
    - Videos
    - Actors
    - Status

2. Add Primary Key and other **attributes about entity**:
    - Users(**user_id**, email_ password)
    - Profiles(**profile_id**, name, **profile_type_id**)
    - Profile_Types(**profile_type_id**, type) → Adult, Kids
    - Videos(**video_id**, title, description, cast)
    - Actors(**actor_id**, name, list_of_video)
    - Status(**status_id**, status_type) → Completed/In-Progress

3. Represent Relations (Using Cardinality):
    - Users(**user_id**, email_ password)
    - Profiles(**profile_id**, name, **user_id, profile_type_id**)
    - Profile_Types(**profile_type_id**, type) → Adult, Kids
    - Videos(**video_id**, title, description)
    - Actors(**actor_id**, name)
    - Status(**status_id**, status_type) → Completed/In-Progress

→ Lookup tables:
    - Video_Actors(**video_id, actor_id**) → Cast
    - Profile_Video_info(**profile_id, video_id**, status_id, timestamp)

4. Mention FK and Indexes:
    -