Lab 1.

Part 1.

Task 1.1.

Relation A: Employee

1.

Superkeys:

1) EmpID
2) SSN
3) Email
4) Phone, Name
5) SSN, Name
6) Email, Name

2. Candidate keys

- EmpId ✓
- SSN ✓
- Email ✓
- Phone ✓
- Name
- Salary

3.

I would choose Emp Id as a Primary key. Because it is often shorter than SSN and unique. Also, email can change over time. So Emp Id is a safe and practical variant.

4.

According to the shown data, phone numbers can not be repeated. However, in theory in some companies office workers can divide phone among themselves.

Relation B : Course Registration

1, 2

(Student Id, Course Code, Section, Semester, Year)

- Student Id - registration of a student
- Course Code - in order to differentiate courses
- Section - in one semester of one course can be several sections and student

can registering in one exact section.

- Semester and Year - year complements semester. Because, student can't take the same section in that semester. But there can be like that, student A ~~take~~ took section A in fall 2024, and then take that section in spring 2025 or fall 2025.

3.

If there exists unique identificator of section in a semestr (for example, OfferingId), then (studentID, OfferingId) - candidate. If section is unique in the course and the semestr, then (studentID, courseCode, Semestr, Year, Section) - main candidate.

Task 1.2

1. Student.AdvisorID → Professor.ProfID
(AdviserID ссылается на профессора - эдвайзера)

2. Professor.Department → Department.DeptCode
Professor works in Department

3. Enrollment.StudentID → Student.StudentID
запись о зачислении студента

4. Enrollment.CourseID → Course.CourseID

5. Course.DepartmentCode → Departmend.DeptCode

6. Department.ChairID → Professor.ProfId

Part 2 ER Diagram Construction

Task 2.1. Hospital Management System

1. Entities (strong, weak)

Strong Entities (they have their unique ID (identificator)

- Patient
- Doctor
- Department
- Appointment
- Prescription
- Hospital Room (weak, Department)

Weak Entities (can exist only by relations with others)

it could be phone Number of Specialization of doctors (multi-valued)

2. Attributes (simple, composite, multi-valued or derived)

Patient

ID - simple (PK)
Name - simple
Birthdate - simple
address - composite
phone number - (can be several / multi-valued)
insurance info. - (simple / composite)

Doctor

ID - simple (PK)

Name - simple

specialization - multi-valued

phone numbers - simple / multi-valued

office locations - simple / multi-valued

Departments

Code - simple (Primary key)

Name - simple

location - simple

Appointments

Patient ID - Fk, simple (Foreign key)

Doctor ID - Fk, simple

Date / - simple

Time - simple

Purpose - simple

Notes - simple (optional

Appointment ID - Pk, simple

Prescriptions

Prescription ID - PK, simple

Patien ID - Fk, simple

Doctor ID - Fk, simple

Medication Name - simple

Dose - simple

Instructions - simple

Hospital Rooms

Room ID - PK, simple

Department Code - Fk, simple

3. Relationships with their cardinalities

1:1, 1:N, M:N

1. Patient → Appointment 1:N
2. Doctor → Appointment 1:N
3. Doctor → Specialization M:N
4. Patient → Prescriptions 1:N
5. Doctor → Prescriptions 1:N
6. Doctor → Department N:1

7. Department - Hospital Room 1:N

4. ER Diagram

WWW.draw.io

Task 2.2. E-commerce Platform

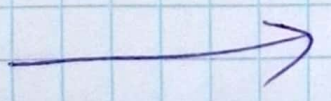
1. ER Diagram

Entities (сущности)

- Customer (PK: CustomerID)
- Order (PK: OrderId)
- OrderItem (PK: {OrderId, ProductId})
  (Позиция заказа (количество, цена))
- Product (PK: ProductId)
- Category (PK: CategoryId) - категории товаров
- Vendor (PK: VendorId) (поставщики)
- Reviews (PK: ReviewId) - (отзывы)
- Inventory (PK: ProductId)
- Address (PK: AddressId) - (Customer may have billing, shipping)

Attributes

- Customer → Name, Email, Billing Address Id (Fk)
  Shipping Address Id (fk)
- Order → Date, Status, Customer Id (Fk)
- Order Item → Quantity, Price At Order Time
- Product → Name, Description, Price, Category Id (Fk),
  Vendor (Fk).
- Category → Name
- Vendor → Name, Contact Info
- Review → Rating, Comment, Customer Id (Fk),
  Product Id (Fk).
- Inventory → Stock Level
- Address → street, City, State, Zip



2. Weak Entity

Order Item is weak entity. Because, it's PK is composite (составной). I mean we can't have Order Item, if we don't have Order Id and Product Id.

3. M:N

Customer - Product by Review.

One customer can write a lot of reviews

One product can have a lot of reviews from a lot of customers

Part 4. Normalization Workshop

Task 4.1 Denormalized Table Analysis

StudentProject (StudentID, StudentName, StudentMajor, ProjectID, ProjectTitle, ProjectType, SupervisorID, SupervisorName, SupervisorDept, Role, HoursWorked, StartDate, EndDate)

1. Functional Dependencies

- StudentID → StudentName, StudentMajor
- ProjectID → ProjectTitle, ProjectType, SupervisorID
- SupervisorID → SupervisorName, SupervisorDept,
- (StudentID, ProjectID) → Role, HoursWorked, StartDate, EndDate
- ProjectID → SupervisorID

2. Identify problems

Redundancy: if one Supervisor leads more than 1 project, then Supervisor Name will be repeat in every of those projects.

Update: if we change Supervisor Name, we are going to change SupervisorID in every project.

Insert: if we want to add new project without any students, then we will have to enter null StudenID or we can't if it is necessarily.

Delete: if the last student ego out. and we delete Student ID, then there is a chance to delete whole project Data unless this data is in other Data Bases.

3. Apply 1NF

There are two things that may violate 1NF.

They are Role and HoursWorked.

Because, if one student can have more than one role, then ith the table might be like this

Role	Hours Worked
A, B	7h, 3h

But other than that everything is good.

4. Apply 2NF

1) Primary key - (StudentID, ProjectID)

2) StudentName, StudentMajor — StudentID

Partial dependency

ProjectTitle, ProjectType, supervisorID

|

ProjectID

So, this dependencis violates 2NF

3) 1NF → 2NF

- Student (PK Student ID, Student Name, Student Major)
- Project (PK Project ID, Project Title, Project Type, FK Supervisor ID)
- Supervisor (PK Supervisor ID, Supervisor Name, Supervisor Dept)
- Student Project (FK Student ID, FK Project ID, Role, Hours Worked, Start Date, End Date, PK (Student ID, Project ID)

5. 2NF → 3NF

Project → Supervisor ID

- Student (Student ID, student Name, student Major)
- Supervisor (Supervisor ID, Supervisor Name, Supervisor Dept)
- Project (Project ID, Project Title, Project Type, [Supervisor ID INT FK References Supervisor (Supervisor ID)])

Student Project (PK (Student ID, Project ID)), Student ID FK
Project ID FK, Role, Hours Worked, Start Date,
End Date)

Task 4.2. Advanced Normalization

Course Schedule (Student ID, Student Major,
Course ID, Course Name,
Instructor ID, Instructor Name,
TimeSlot, Room, Building)

1. PK -

[redacted]

minimal unique record of actual visit -
- (Student ID, Course ID, Timeslot)

room ↔ TimeSlot

CourseSection - (Course ID, Timeslot, Room)
it is better to create a new attribute
which is called CourseSektion.

Course Section (Section ID (Pk), Course ID (Fk),
Instructor ID (Fk), Time Slot, Room)

Enrollment (Student ID (Fk), Section ID (Fk),
Primary key (Student ID, Section ID))

Primary key – (Student ID, Section ID)

2. FD

StudenID → Student Major,

Coutse ID → Cocurse Name

Instruetor ID → Instructor Name

TimeSlot, Room, Building → Instructor ID

3. BCNF

The table is not in BCNF.

Because, there exists attributs like Student ID, which is not super key.

4. Decomposition

1. Student (PK Student ID, Student Major)
2. Instructor (PK Instructor ID, Instructor Name)
3. Course (PK Course ID, Course Name)
4. Room (PK Room, Building)
5. Couse Section (PK Section ID, FK Course ID, FK Instructor ID, Timeslot, FK Room)
6. Enrollment (FK Student ID, FK Section ID, PK (Student ID, Section ID)

5.

- Decomposition saves the information (lossless) Because, during regenerating tables via PK|FK we can regenerate original attributes (Student Major from Student)
- No dependencies are lost, transitive dependencies are resolved through separate tables. (разрешены через отдельные таблицы).

If BCNF is decomposed correctly, then there will no be any losses of info. The only thing is that if in original table were inconsistent data, like iCourse ID has several CourseName, then decomposing detects that and request a resolution of conflict.