Task 1.1

Relation A: Employee:

1. Superkeys – {EmpID}, {SSN}, {Email}, {Phone}, {Name}, {Salary}, {EmpID, SSN}, {EmpID, Email} …
2. {EmpID}, {SSN}, {Email}, {Phone}, {Name}, {Salary}
3. EmpID – ID itself says that it is unique.
4. If we are considering every possible situation, two employees indeed can have the same phone number (wife and husband are working in the same company and share phone, it might seem very strange, but it still could happen), but in real life this situation is very unlikely. And based on the sample data, there is no mutual phone number between employees.

Relation B: Course Registration:

1. {StudentID, CourseCode, Section, Semester, Year}
2. StudendID – to identify student,

   CourseCode – to identify course

   Section – to identify section of a course(for student not to take the same section of a course in one semester)

   Semester – to identify in which semester student takes the course for him not to take the same section.

   Year – to identify in which year does student take the course

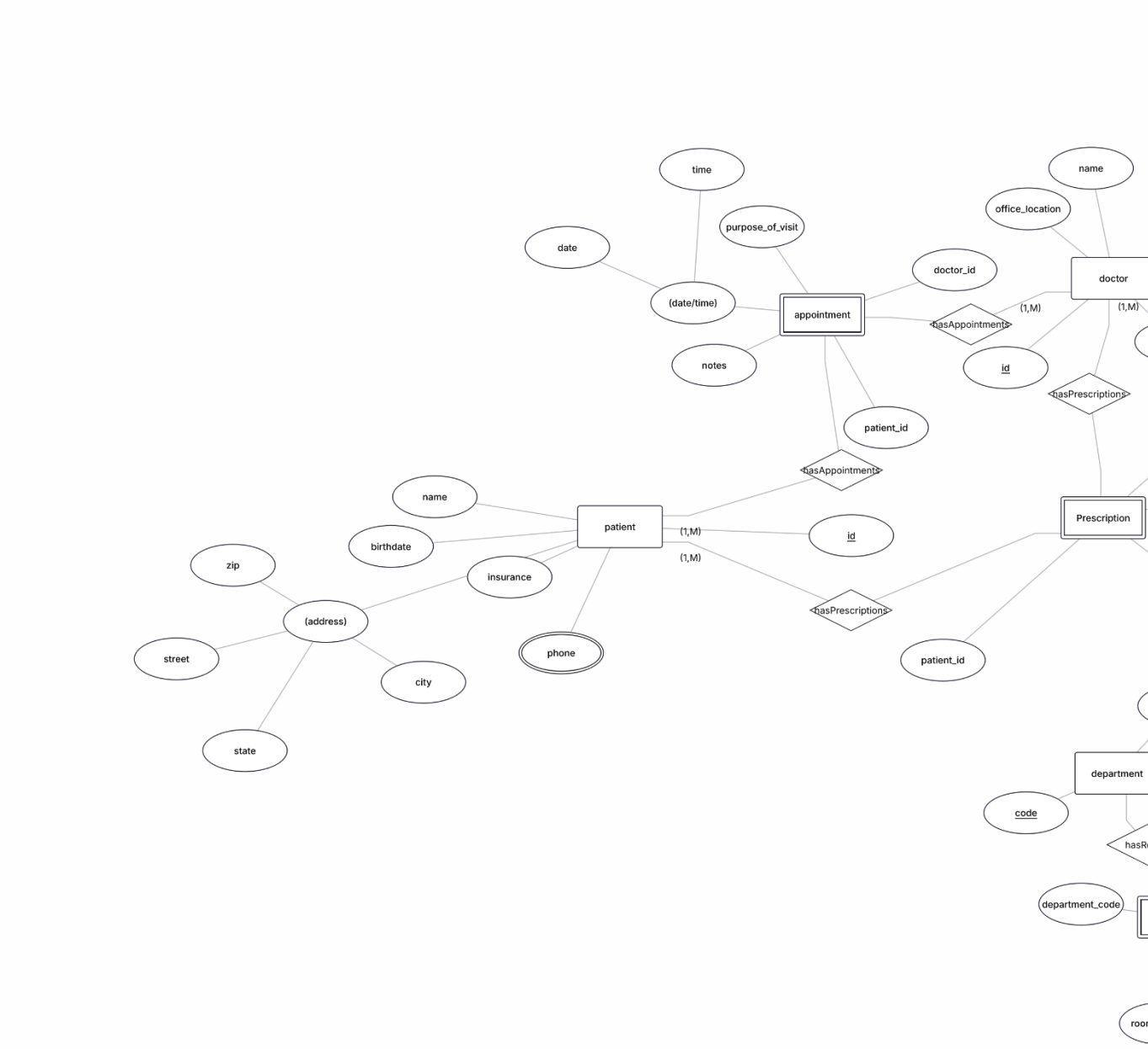
3. No additional keys

Task 1.2

1. Student table has Major field that is a foreign key to Department
   Professor table has Department field that is a foreign key to Department
   Course table has DepartmentCode field that is a foreign key to Department
   Enrollment table has StudentID and CourseID foreign keys to Student and Course tables.

Task 2.1

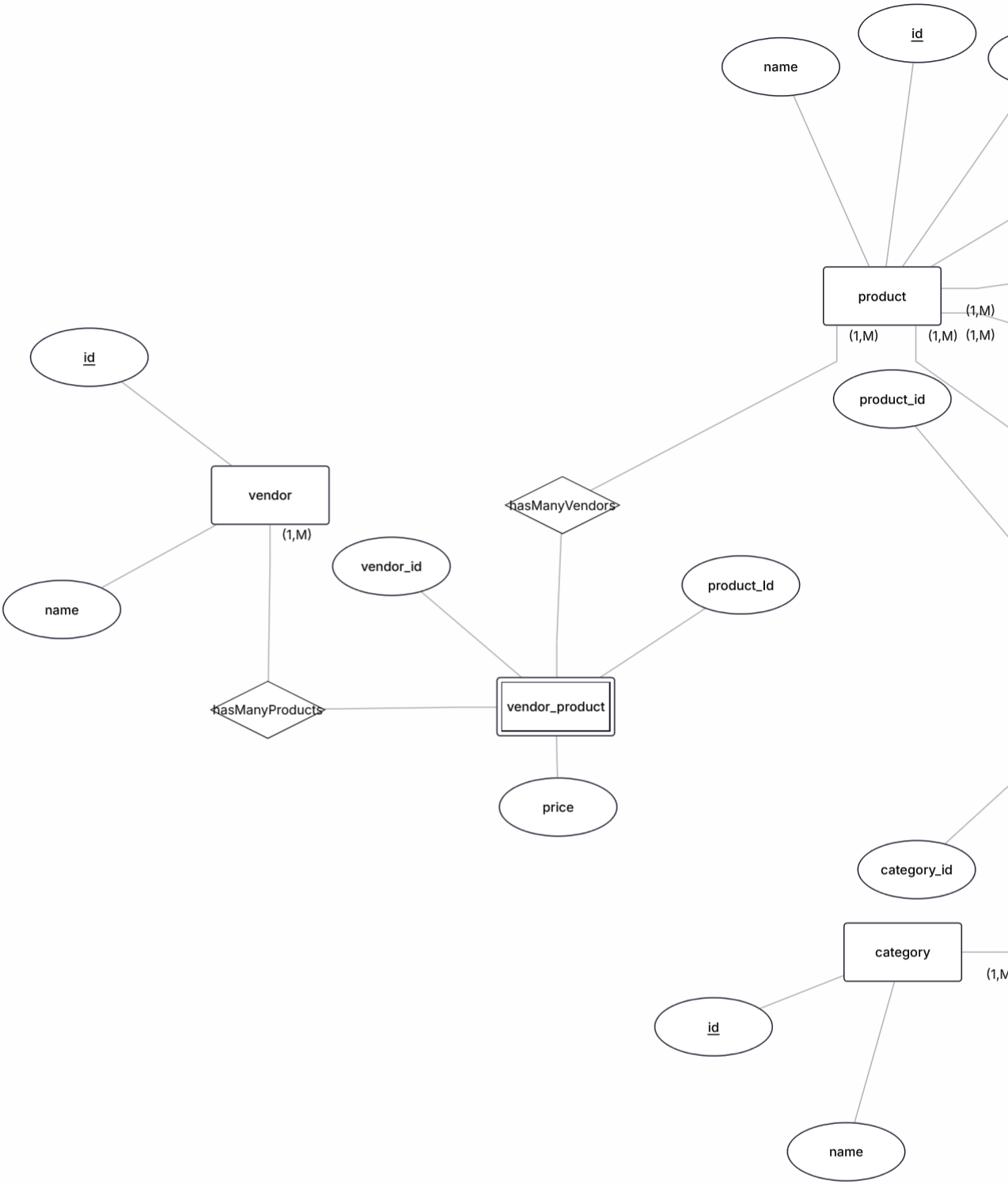
Url: ***https://ibb.co.com/KpGCrGJ9***



Task 2.2

Url: ***https://ibb.co.com/s9YZbx34***

2. order_item is weak entity because of being dependent on product and order
3. reviews entity implements many-to-many relationships between customers and products and has additional attributes.


id
name
product
(1,M)
(1,M)
(1,M)
(1,M)
id
product_id
vendor
(1,M)
hasManyVendors
name
vendor_id
product_Id
hasManyProducts
vendor_product
price
category_id
category
(1,M
id
name

Task 4.1

1. StudentID -> StudentName, StudentMajor

   ProjectID -> ProjectTitle, ProjectType

   SupervisorID -> SupervisorName, SupervisorDept

2. For every students project - student name and student major redundancy, for every supervisor – supervisor name and supervisor department redundancy.
   When updating students name we have to update all rows with him, for example, student with id=1 changed his name to some other, then if that student has n projects, then we have to update n rows.
   When inserting we have to place studentID, but if we have no suitable students for that project, then we cannot insert that row. For example, we have someone that wanted to create studentproject and after a while attach some student to that project, but our system does not allow.
   When deleting project from table we are losing data about student.
3. There are no 1NF violations, all values are single-valued and atomic.
   Decomposing StudentProject table to 4 tables – Student (id (PK), name, major), Project (id (PK), title, type, supervisor_id(FK)), Supervisor (id (PK), name, dept), StudentProject (student_id (FK), project_id (FK), role, hours_worked, startdate, enddate).
   2NF: All non-key attributes dependent on PK(Student (id), Project (id), Supervisor (id), StudentProject is a pivot table that relates all others with foreign keys)
   3NF: no transitive dependencies – Student: id -> name, major; Project: id -> title, type; Supervisor: id -> name, dept; StudentProject: {student_id, project_id} -> role, hours_worked, startdate.

Task 4.2

1. {StudentID, CourseID}
2. StudentID -> StudentMajor
   CourseID -> CourseName, InstructorID, TimeSlot, Room
   InstructorID -> InstructorName
   Room -> Building
   PK -> evr else
3. Table is not in BCNF due to transitive dependeny violation – CourseId -> InstructorID -> InstructorName, thus CourseID -> InstructorName
4. Decomposing – Student (id (PK), major), Course (id (PK), name, instructor_id (FK), timeslot, room (FK)) Instructor (id (PK), name), Room(room (PK), building), CourseSchedule (student_id (FK), course_id (FK))

Task 5.1

url:

https://ibb.co.com/bMr7gqJ2