Question 1:

1.1 – Required Data Constraints

- ENO must be an Index and the primary key.
- DNO must be an Index and the primary key.
- PNO must be an Index and the primary key.

1.2 – Entity Integrity Constraints

- Table Employee: FName, LName, Job
- Table Department: DName, Location
- Table Project: PName, Budget

1.3 – Referential Integrity Constraints

- Table Employee: DNO (as FK)

1.4 – Domain Constraint

- Table Project: Budget

1.5 – Enterprise-Level Constraints

- I would say none.

Question 2:

2.1 – Tables with Attributes:

- Employee
  - ENO – Employee
  - FName – Employee
  - LName – Employee
  - DNO – Department
- Department
  - DNO – Department
  - DName – Department
  - Location – Department
  - EmployeeIds – Employee
- Project
  - PNO – Project
  - Pname – Project
  - Budget – Project
- ProjectAssignments (Self-Added Associative Table)
  - AssignmentId – ProjectAssignments
  - ENO – Employee
  - PNO – Project

2.2 – Primary Keys

- Table: Employee – Key: ENO
- Table: Department – Key: DNO
- Table: Project – Key: PNO

- Table: ProjectAssignments – Key: AssignmentId

2.3 – Foreign Key Relationships

- DNO (Base table = Department, Foreign Key Table = Employee)
- PNO (Base table = Project, Foreign Key Table = Employee)

2.4 – Associative Table

- I created a table called ProjectAssignments (PK AssignmentId, FK ENO, FK DNO, AssignmentDate, CompletionDate).

2.5 – Handle Associative Table Attributes

- To handle the many projects (M) an Employee can have and that Projects don't have any Employees (N), I grabbed the ENO from the Employee and the PNO from the Project and added them as Foreign Keys on the Associative Table ProjectAssignments to bridge which Employee is working on X Projects, where X is some arbitrary number.

2.6 – Derived Attribute

- I noticed Start_date and job were derived and I decided to add Start_date to ProjectAssignment and renamed it to AssignmentDate. For Job I moved it over to Employee. That's what made most sense to me to do.

2.7 – Denormalize or Split

- No, I didn't denormalize or split any tables. The most I did was I moved the job column to the Employee table, but that is something that makes sense for Employee to have anyway and doesn't compromise any normalization.

Question 3:

Student (WNumber, FName, LName, Address, DOB, Balance, AmountDue, AmountPaid)

Now I'd split it into two and split the sensitive financial columns from the Student table.

Student (WNumber, Fname, LName, Address, DOB)

StudentTuitionInfo(WNumber, Balance, AmountDue, AmountPaid)