# TASK – 10

Normalizing Databases using Functional Dependencies up to Third Normal Form

* Aim:

To normalize the below relation and create the simplified table with suitable constraint.

University(UniversityID, UName, UAddress, Contact\_No, DepartmentID, DName, HOD, CourseID, CName, Duration, Credits, StudentID, SFName, SLName, Age, DOB, Gender, Email, Phone, ProfessorID, PFName, PLName, Qualification, Experience, PEmail, PContact, SubjectID, SubName, SubCode, ClassID, ClassTime, RoomNo, ExamID, ExamDate, Result).

a) Apply the functional dependency, normalize to 1NF

b) Normalize the relations using FD+ and α+.

c) Find the minimal cover, canonical cover.

d) Normalize to 2NF, add/alter constraints if necessary.

e) Normalize to 3NF, add/alter constraints if necessary.

## Procedure:

Normalize the given relation and create simplified tables with suitable constraints. We need to identify the functional dependencies and separate them into different tables. Normalization involves breaking down the data into smaller, related tables to minimize data redundancy and maintain data integrity. Let's identify the functional dependencies:

* Functional Dependency:
* UniversityID → UName, UAddress, Contact\_No
* DepartmentID → DName, HOD, UniversityID
* CourseID → CName, Duration, Credits, DepartmentID
* StudentID → SFName, SLName, Age, DOB, Gender, Email, Phone, CourseID
* ProfessorID → PFName, PLName, Qualification, Experience, PEmail, PContact, DepartmentID
* SubjectID → SubName, SubCode, CourseID
* ClassID → ClassTime, RoomNo, CourseID, ProfessorID
* ExamID → ExamDate, Result, SubjectID, StudentID
* Now, we can create simplified tables:
* University (UniversityID [PK], UName, UAddress, Contact\_No)
* Department (DepartmentID [PK], DName, HOD, UniversityID [FK])
* Course (CourseID [PK], CName, Duration, Credits, DepartmentID [FK])
* Student (StudentID [PK], SFName, SLName, Age, DOB, Gender, Email, Phone, CourseID [FK])
* Professor (ProfessorID [PK], PFName, PLName, Qualification, Experience, PEmail, PContact, DepartmentID [FK])
* Subject (SubjectID [PK], SubName, SubCode, CourseID [FK])
* Class (ClassID [PK], ClassTime, RoomNo, CourseID [FK], ProfessorID [FK])
* Exam (ExamID [PK], ExamDate, Result, SubjectID [FK], StudentID [FK])

In these tables, [PK] denotes the primary key, [FK] denotes the foreign key, and suitable constraints should be added to maintain data integrity.

Create tables for all non-prime attributes using α+ (Alpha Plus allows grouping attributes based on their functional dependencies and candidate keys). The candidate keys in this case are UniversityID, DepartmentID, CourseID, StudentID, ProfessorID, SubjectID, ClassID, and ExamID.

University Table: UniversityID (PK), UName, UAddress, Contact\_No

Department Table: DepartmentID (PK), DName, HOD, UniversityID (FK)

Course Table: CourseID (PK), CName, Duration, Credits, DepartmentID (FK)

Student Table: StudentID (PK), SFName, SLName, Age, DOB, Gender, Email, Phone, CourseID (FK)

Professor Table: ProfessorID (PK), PFName, PLName, Qualification, Experience, PEmail, PContact, DepartmentID (FK)

Subject Table: SubjectID (PK), SubName, SubCode, CourseID (FK)

Class Table: ClassID (PK), ClassTime, RoomNo, CourseID (FK), ProfessorID (FK)

Exam Table: ExamID (PK), ExamDate, Result, SubjectID (FK), StudentID (FK)

## First Normal Form:

The given relation into the First Normal Form (1NF), to need to ensure that each attribute (column) contains atomic (indivisible) values, and there are no repeating groups or arrays. Based on the provided relation, it appears that each attribute already contains atomic values, so there are no repeating groups to eliminate.

## Second Normal Form:

To determine whether the given relation is in the Second Normal Form (2NF), we need to check two conditions:  
1. The relation must already be in 1NF (First Normal Form).  
2. All non-prime attributes (attributes not part of any candidate key) must be fully functionally dependent on the entire primary key.

It appears that the potential candidate keys could be:  
1. UniversityID  
2. DepartmentID  
3. CourseID  
4. StudentID  
5. ProfessorID  
6. SubjectID  
7. ClassID  
8. ExamID

Next, we need to check if all non-prime attributes are fully functionally dependent on their respective candidate key(s).

## Third Normal Form:

To determine whether the given relation is in the Third Normal Form (3NF), need to check two conditions:  
1. The relation must already be in the Second Normal Form (2NF).  
2. There should be no transitive dependencies between non-prime attributes and candidate keys.

The given relation satisfies the conditions of the Second Normal Form (2NF). Now, let's check for transitive dependencies:

* UniversityID → UName, UAddress, Contact\_No
* DepartmentID → DName, HOD, UniversityID
* CourseID → CName, Duration, Credits, DepartmentID
* StudentID → SFName, SLName, Age, DOB, Gender, Email, Phone, CourseID
* ProfessorID → PFName, PLName, Qualification, Experience, PEmail, PContact, DepartmentID
* SubjectID → SubName, SubCode, CourseID
* ClassID → ClassTime, RoomNo, CourseID, ProfessorID
* ExamID → ExamDate, Result, SubjectID, StudentID

There are no transitive dependencies in the above functional dependencies, as all non-prime attributes are directly dependent on their primary keys.

Thus, with the elimination of transitive dependencies, the relation now satisfies the conditions of the Third Normal Form (3NF).

## Result:

Thus the normalization of the given relation is created the simplified tables with suitable constraint successfully.