FAST ALUMNI MANAGEMENT SYSTEM

DB PROJECT MID EVALUATION

MEMBERS:
22K-4504 Muhammad Ashir
22K-4471 Muhammad Ashar Ali
22K-4625 Sayal Baig

DATA REQUIRED:

- User_ID
- User_CNIC
- User_FullName
- User_NuID
- User_DoB
- User_EmailID
- User_MobileNo
- User_PasswordHashed
- User_Campus
- User_CurrentCountry
- User_CurrentCity
- User_Profile_Picture
- User_Role
- User_IsActive
- User_DateCreated
- User_Lastupdated
- User_Department

- Alumni_ProfileID
- Alumni_UserID
- Alumni_Organization
- Alumni_CurrentPosition
- Alumni_LinkedInProfile
- Alumni_Industry
- Alumni_ProfessionalSummary
- Alumni_PrivacySettings

- Student_UserID
- Student_CurrentYear
- Student_CurrentSemester
- Student_GPA

- Event_ID
- Event_Name
- Event_Description
- Event_Date

DATA REQUIRED

- Event_Time
- Event_Location
- Event_OrganizerID
- Event_MaxParticipants
- Event_RegistrationDeadline
- Event_banner
- Event_ParticipantCount
- Event_CreatedDate
- Event_Type

- Registration_ID
- Registration_EventID
- Registration_UserID
- Registration_Date
- Registration_TicketNumber
- Registration_Status

- Donation_ID
- Donation_UserID
- Donation_Amount
- Donation_PaymentMethod
- Donation_TransactionID
- Donation_Date
- Donation_Note

- Job_ID
- Job_Type
- Job_SalaryRange
- Job_ExperienceLevel
- Job_Title
- Job_Description
- Job_CompanyName
- Job_Location
- Job_Applicationlink
- Job_PostedBy
- Job_DatePosted
- Job_deadline

- Mentorship_ID
- Mentor_ID
- Mentee_ID
- Mentorship_StartDate
- MentorShip_EndDate
- Mentorship_Objectives
- Mentorship_status

- Feedback_ID
- Feedback_givenby
- Feedback_receivedby
- Feedback_Text

- Feedback_rating
- Feedback_date

- News_ID
- News_title
- News_Content
- News_AuthorID
- News_PostedDate
- NewsThumbnailImage
- News_Tags

- Forum_ID
- Forum_Name
- Forum_Description
- Forum_Createdby
- Forum_DateCreated

- Post_ID
- Post_ForumID
- Post_PostedBy
- Post_Content
- Post_AttachmentURL
- Post_Date
- Post_Status

- Review_ID
- Reviewer_ID
- Reviewee_ID
- Review_Text
- Review_Rating
- Review_Date

- Report_ID
- Report_UserID
- Report_GraduationYear
- Report_Industry
- Report_SalaryRange
- Report_Date

- Project_ID
- Project_Name
- Project_Description
- Project_CreatedBy
- Project_Collaborators
- Project_StartDate
- Project_Enddate
- Project_Status

- Help_ID

- Help_title
- Help_Content
- Help_Section
- Help_LastUpdated
- Help_Visibility
- Help_Author

Normalization Process

Normalizing to 1 NF:

1NF requires:

- Each column contains atomic values (no multivalued attributes).
- Each column contains values of a single type.
- Each row is unique, identified by a primary key.

Action for 1NF:

- Identify repeating groups and multivalued attributes.
- Split them into separate tables.

Analysis for 1NF:

From the provided attributes, no repeating groups or multivalued attributes are evident. However, we need to ensure every table has a primary key and atomic values.

Resultant Tables for 1NF:

User Table
| **User_ID (PK)** | User_CNIC | User_FullName | User_NuID | User_DoB | User_EmailID | User_MobileNo | User_PasswordHashed | User_Campus | User_CurrentCountry | User_CurrentCity | User_Profile_Picture | User_Role | User_IsActive | User_DateCreated | User_LastUpdated | User_Department |

Alumni Table
| **Alumni_ProfileID (PK)** | Alumni_UserID (FK) | Alumni_Organization | Alumni_CurrentPosition | Alumni_LinkedInProfile | Alumni_Industry | Alumni_ProfessionalSummary | Alumni_PrivacySettings |

Student Table
| **Student_UserID (PK, FK)** | Student_CurrentYear | Student_CurrentSemester | Student_GPA |

Event Table
| **Event_ID (PK)** | Event_Name | Event_Description | Event_Date | Event_Time | Event_Location | Event_OrganizerID | Event_MaxParticipants | Event_RegistrationDeadline | Event_Banner | Event_ParticipantCount | Event_CreatedDate | Event_Type |

Registration Table
| **Registration_ID (PK)** | Registration_EventID (FK) | Registration_UserID (FK) | Registration_Date | Registration_TicketNumber | Registration_Status |

Donation Table
| **Donation_ID (PK)** | Donation_UserID (FK) | Donation_Amount | Donation_PaymentMethod | Donation_TransactionID | Donation_Date | Donation_Note |

Job Table
| **Job_ID (PK)** | Job_Type | Job_SalaryRange | Job_ExperienceLevel | Job_Title | Job_Description | Job_CompanyName | Job_Location | Job_Applicationlink | Job_PostedBy | Job_DatePosted | Job_Deadline |

Mentorship Table
| **Mentorship_ID (PK)** | Mentor_ID (FK) | Mentee_ID (FK) | Mentorship_StartDate | Mentorship_EndDate | Mentorship_Status |

Feedback Table
| **Feedback_ID (PK)** | Feedback_GivenBy (FK) | Feedback_ReceivedBy (FK) | Feedback_Text | Feedback_Rating | Feedback_Date |

News Table
| **News_ID (PK)** | News_Title | News_Content | News_AuthorID (FK) | News_PostedDate | News_ThumbnailImage | News_Tags |

Forum Table
| **Forum_ID (PK)** | Forum_Name | Forum_Description | Forum_CreatedBy (FK) | Forum_DateCreated |

Post Table
| **Post_ID (PK)** | Post_ForumID (FK) | Post_PostedBy (FK) | Post_Content | Post_AttachmentURL | Post_Date | Post_Status |

Review Table
| **Review_ID (PK)** | Reviewer_ID (FK) | Reviewee_ID (FK) | Review_Text | Review_Rating | Review_Date |

Report Table
| **Report_ID (PK)** | Report_UserID (FK) | Report_GraduationYear | Report_Industry | Report_SalaryRange | Report_Date |

Project Table
| **Project_ID (PK)** | Project_Name | Project_Description | Project_CreatedBy (FK) | Project_Collaborators | Project_StartDate | Project_EndDate | Project_Status |

Help Table
| **Help_ID (PK)** | Help_Title | Help_Content | Help_Section | Help_LastUpdated | Help_Visibility | Help_Author (FK) |

Normalizing 2 NF:

2NF requires:

- The table is in 1NF.
- All non-key attributes are fully functionally dependent on the primary key (no partial dependency).

Action for 2NF:

- Identify and remove partial dependencies by creating new tables for attributes only partially dependent on composite keys.

Analysis for 2NF:

- TableFeedback have partial dependencies on composite keys.

Resultant Tables for 2NF:

Feedback Table
| **Feedback_ID (PK)** | Feedback_GivenBy (FK) | Feedback_ReceivedBy (FK) |

FeedbackDetails Table
| **Feedback_ID (PK, FK)** | Feedback_Text | Feedback_Rating | Feedback_Date |

Normalizing to 3 NF:

3NF requires:

- The table is in **2NF**.
- No **transitive dependency** (non-key attributes depend only on the primary key).

Action for 3NF:

- Remove transitive dependencies by creating new tables.

Analysis for 3NF:

- In the **User Table**, attributes like User_CurrentCountry and User_CurrentCity are directly related to the user and not dependent on any other attribute. These are not transitive dependencies and can remain in the **User Table**.
- However, in the **Project Table**, the attribute Project_Collaborators introduced a **transitive dependency**. The collaborators' details (e.g., roles, statuses) depend not only on the Project_ID but also on individual collaborators themselves.

Resolution:

- To remove this transitive dependency, the Project_Collaborators attribute was normalized into a separate **Collaboration Table**.
    - The **Project Table** now only contains project-specific details.
    - The new **Collaboration Table** represents the many-to-many relationship between Projects and Users and includes attributes like Collaboration_Role, Collaboration_JoinDate, and Collaboration_Status.

Project Table

| **Project_ID (PK)** | Project_Name | Project_Description | Project_CreatedBy (FK) | Project_Collaborators | Project_StartDate | Project_EndDate | Project_Status |

Collaboration Table

| **Collaboration_ID (PK)** | Collaboration_ProjectID | Collaboration_UserID | Collaboration_Role | Collaboration_JoinDate | Collaboration_Status |

BCNF:

BCNF requires:

- The table is in 3NF.
- Every determinant is a candidate key.

Action for BCNF:

- Further split tables if any determinant exists that isn't a candidate key.

Analysis for BCNF:

Most tables already satisfy BCNF as primary keys are clearly defined. If any anomalies arise, further decomposition might be necessary.

Resultant Tables for BCNF:

No additional changes beyond 3NF are required.

FINAL ENTITIES AND ATTRIBUTES:

1. User Table

| **User_ID (PK)** | User_CNIC | User_FullName | User_NuID | User_DoB | User_EmailID | User_MobileNo | User_PasswordHashed | User_Campus | User_Profile_Picture | User_Role | User_IsActive | User_DateCreated | User_LastUpdated | User_Department | User_CurrentCountry | User_CurrentCity |

2. Alumni Table

| **Alumni_ProfileID (PK)** | Alumni_UserID (FK) | Alumni_Organization | Alumni_CurrentPosition | Alumni_LinkedInProfile | Alumni_Industry | Alumni_ProfessionalSummary | Alumni_PrivacySettings |

3. Student Table

| **Student_UserID (PK, FK)** | Student_CurrentYear | Student_CurrentSemester | Student_GPA |

4. Event Table

| **Event_ID (PK)** | Event_Name | Event_Description | Event_Date | Event_Time | Event_Location | Event_OrganizerID (FK) | Event_MaxParticipants | Event_RegistrationDeadline | Event_Banner | Event_ParticipantCount | Event_CreatedDate | Event_Type |

5. Registration Table

| **Registration_ID (PK)** | Registration_EventID (FK) | Registration_UserID (FK) | Registration_Status | Registration_Date

6. Donation Table

| **Donation_ID (PK)** | Donation_UserID (FK) | Donation_Title | Donation_description | Donation_TargetedAmount | Donation_AmountRaised | Donation_Date | Donation_deadline | Donation_Status | Donation_Category

7. Job Table

| **Job_ID (PK)** | Job_Type | Job_SalaryRange | Job_ExperienceLevel | Job_Title | Job_Description | Job_CompanyName | Job_Location | Job_ApplicationLink | Job_PostedBy (FK) | Job_DatePosted | Job_Deadline | Job_Tags

8. Mentorship Table

| **Mentorship_ID (PK)** | Mentor_ID (FK) | Mentee_ID (FK) | Mentorship_StartDate | Mentorship_EndDate | Mentorship_Status | Mentorship_Objectives | Mentorship Notes

9. Feedback Table

| **Feedback_ID (PK)** | Feedback_GivenBy (FK) | Feedback_ReceivedBy (FK) |

10. FeedbackDetails Table

| **Feedback_ID (PK, FK)** | Feedback_Text | Feedback_Rating | Feedback_Date |

11. News Table

| **News_ID (PK)** | News_Title | News_Content | News_AuthorID (FK) | News_PostedDate | News_ThumbnailImage | News_Tags |

12. Forum Table

| **Forum_ID (PK)** | Forum_Name | Forum_Description | Forum_CreatedBy (FK) | Forum_DateCreated |

13. Post Table

| **Post_ID (PK)** | Post_ForumID (FK) | Post_PostedBy (FK) | Post_Content | Post_AttachmentURL | Post_Date | Post_Status | Post_bannerImage |

14. Review Table

| **Review_ID (PK)** | Reviewer_ID (FK) | Reviewee_ID (FK) | Review_Text | Review_Rating | Review_Date |

15. Report Table

| **Report_ID (PK)** | Report_UserID (FK) | Report_GraduationYear | Report_Industry | Report_SalaryRange | Report_Date | Report_Image |

16. Project Table

| **Project_ID (PK)** | Project_Name | Project_Description | Project_CreatedBy (FK) | Project_StartDate | Project_EndDate | Project_Status |

17. Collaboration Table

| **Collaboration_ID (PK)** | Collaboration_ProjectID | Collaboration_UserID | Collaboration_Role | Collaboration_JoinDate | Collaboration_Status |

18. Help Table

| **Help_ID (PK)** | Help_Title | Help_Content | Help_Section | Help_LastUpdated | Help_Visibility | Help_Author (FK) |

RELATIONSHIPS AND MULTIPLICITY:

1. User ↔ Alumni

- **Role:** Alumni extends User.
- **Description:** Each alumni is a user, but not every user is an alumni. Alumni-specific data is stored in the Alumni table.
- **Foreign Key:** Alumni_UserID
- **References From:** User.User_ID
- **Multiplicity:**
  - **1 User : 0 or 1 Alumni**
  - A user can have at most one alumni profile.

2. User ↔ Student

- **Role:** Student extends User.
- **Description:** Students are users with specific academic attributes like year, semester, and GPA.
- **Foreign Key:** Student_UserID
- **References From:** User.User_ID
- **Multiplicity:**
  - **1 User : 0 or 1 Student**
  - A user can be either a student or an alumni, but not both simultaneously.

3. User ↔ Event (Organizer)

- **Role:** User as an event organizer.
- **Description:** A user can organize multiple events, but an event can have only one organizer.
- **Foreign Key:** Event_OrganizerID
- **References From:** User.User_ID
- **Multiplicity:**
  - **1 User : 0 or Many Events**
  - A user can organize multiple events, but each event has only one organizer.

4. Event ↔ Registration

- **Role:** Registration links users to events.
- **Description:** A user can register for multiple events, and each event can have many registrations.
- **Foreign Key:**
  - Registration_EventID → Event.Event_ID
  - Registration_UserID → User.User_ID
- **Multiplicity:**
  - **1 Event : 0 or Many Registrations**
  - **1 User : 0 or Many Registrations**

5. User ↔ Donation

- **Role:** User as a donor.
- **Description:** A user can make multiple donations.

- **Foreign Key:** Donation_UserID
- **References From:** User.User_ID
- **Multiplicity:**
  - **1 User : 0 or Many Donations**

6. User ↔ Job (PostedBy)

- **Role:** User as a job poster.
- **Description:** A user can post multiple jobs, and each job is posted by one user.
- **Foreign Key:** Job_PostedBy
- **References From:** User.User_ID
- **Multiplicity:**
  - **1 User : 0 or Many Jobs**

7. User ↔ Mentorship (Mentor/Mentee)

- **Role:** Mentor and Mentee in mentorship.
- **Description:** A mentorship involves one mentor and one mentee, both of whom are users.
- **Foreign Keys:**
  - Mentor_ID → User.User_ID
  - Mentee_ID → User.User_ID
- **Multiplicity:**
  - **1 Mentor (User) : 0 or Many Mentorships**
  - **1 Mentee (User) : 0 or Many Mentorships**

8. User ↔ Feedback (GivenBy/ReceivedBy)

- **Role:** Feedback links users who provide and receive feedback.
- **Description:** Users can exchange feedback with each other.
- **Foreign Keys:**
  - Feedback_GivenBy → User.User_ID
  - Feedback_ReceivedBy → User.User_ID
- **Multiplicity:**
  - **1 User : 0 or Many Feedbacks Given**
  - **1 User : 0 or Many Feedbacks Received**

9. Feedback ↔ FeedbackDetails

- **Role:** Details enrich feedback.
- **Description:** Each feedback has additional details like text and rating stored in a separate table.
- **Foreign Key:** Feedback_ID
- **References From:** Feedback.Feedback_ID
- **Multiplicity:**
  - **1 Feedback : 1 FeedbackDetails**

10. User ↔ News

- **Role:** User as the author of news.
- **Description:** A user can post multiple news items, and each news item is authored by one user.
- **Foreign Key:** News_AuthorID
- **References From:** User.User_ID
- **Multiplicity:**
  - **1 User : 0 or Many News Items**

11. Forum ↔ Post

- **Role:** Forum as the container of posts.
- **Description:** A forum contains multiple posts, and each post belongs to one forum.
- **Foreign Key:** Post_ForumID
- **References From:** Forum.Forum_ID
- **Multiplicity:**
  - **1 Forum : 0 or Many Posts**

12. User ↔ Post

- **Role:** User as the creator of posts.
- **Description:** A user can create multiple posts, and each post is authored by one user.
- **Foreign Key:** Post_PostedBy
- **References From:** User.User_ID
- **Multiplicity:**
  - **1 User : 0 or Many Posts**

13. User ↔ Peer Review

- **Role:** Reviewer and Reviewee in peer reviews.
- **Description:** Users can review each other.
- **Foreign Keys:**
  - Reviewer_ID → User.User_ID
  - Reviewee_ID → User.User_ID
- **Multiplicity:**
  - **1 Reviewer (User) : 0 or Many Peer Reviews**
  - **1 Reviewee (User) : 0 or Many Peer Reviews**

14. User ↔ Salary Report

- **Role:** User as the reporter.
- **Description:** A user can submit multiple salary reports.
- **Foreign Key:** Report_UserID
- **References From:** User.User_ID

- **Multiplicity:**
    - **1 User : 0 or Many Salary Reports**

15. Help ↔ User

- **Role:** User as the author of help content.
- **Description:** A user can author help entries.
- **Foreign Key:** Help_Author
- **References From:** User.User_ID
- **Multiplicity:**
    - **1 User : 0 or Many Help Entries**

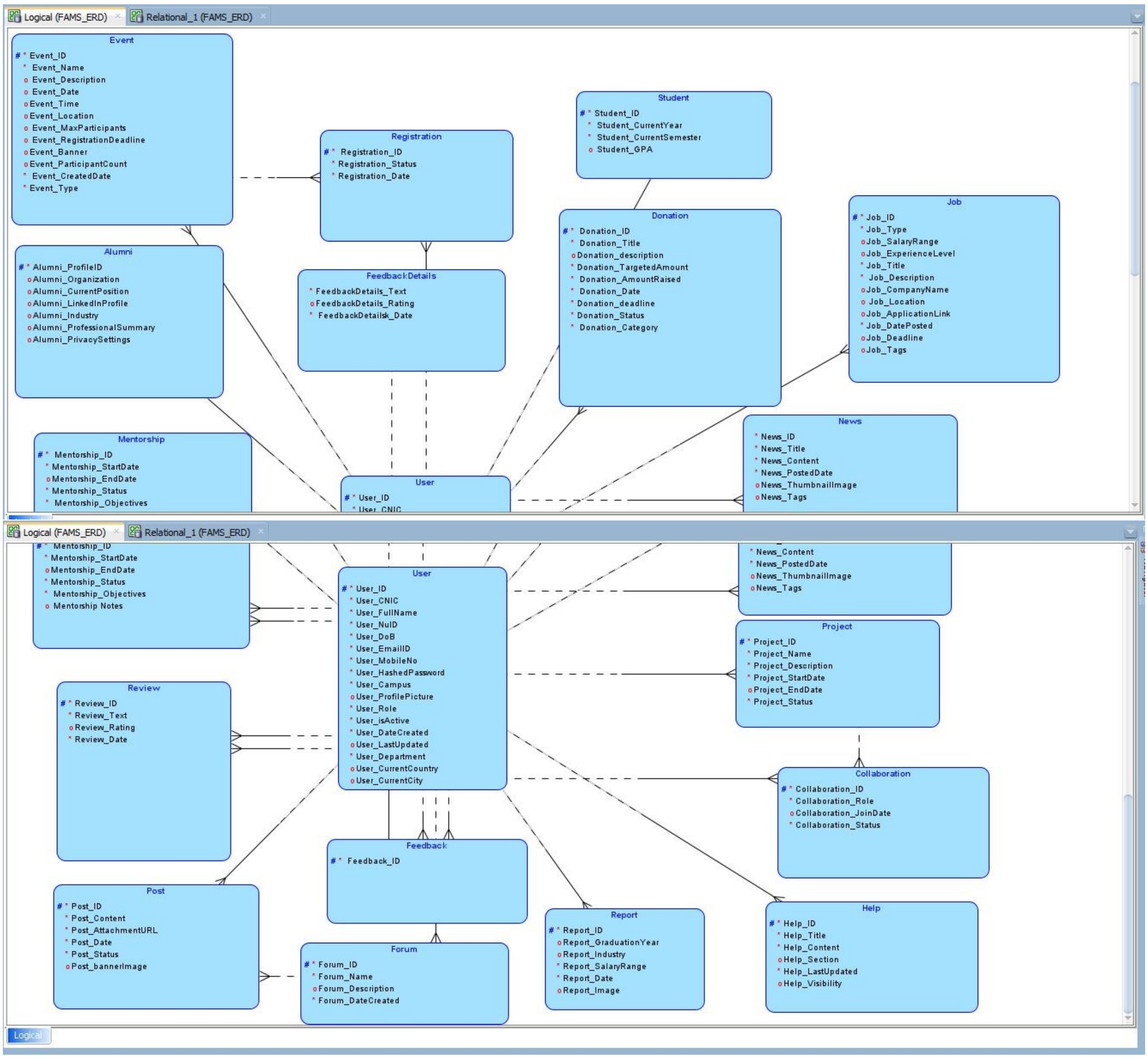
16. Project ↔ Collaboration

- **Role:** Collaboration links projects to users as collaborators.
- **Description:** A project can have multiple collaborators, and a user can collaborate on multiple projects.
- **Foreign Key:**
    - Collaboration_ProjectID → Project.Project_ID
- **Multiplicity:**
    - **1 Project : 0 or Many Collaborations**

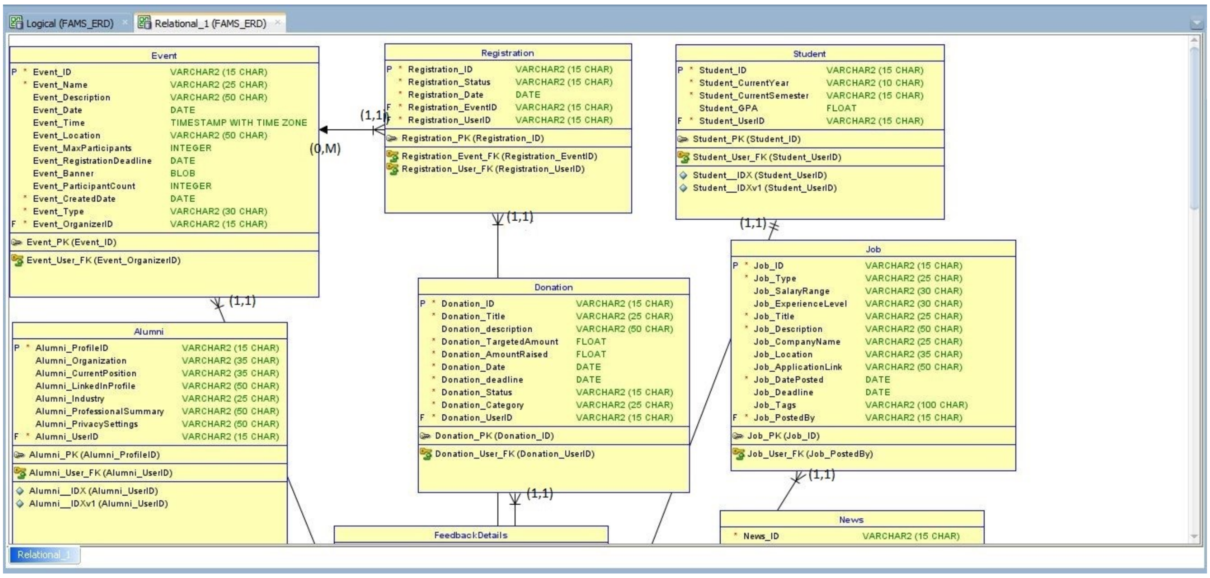
17. User ↔ Collaboration

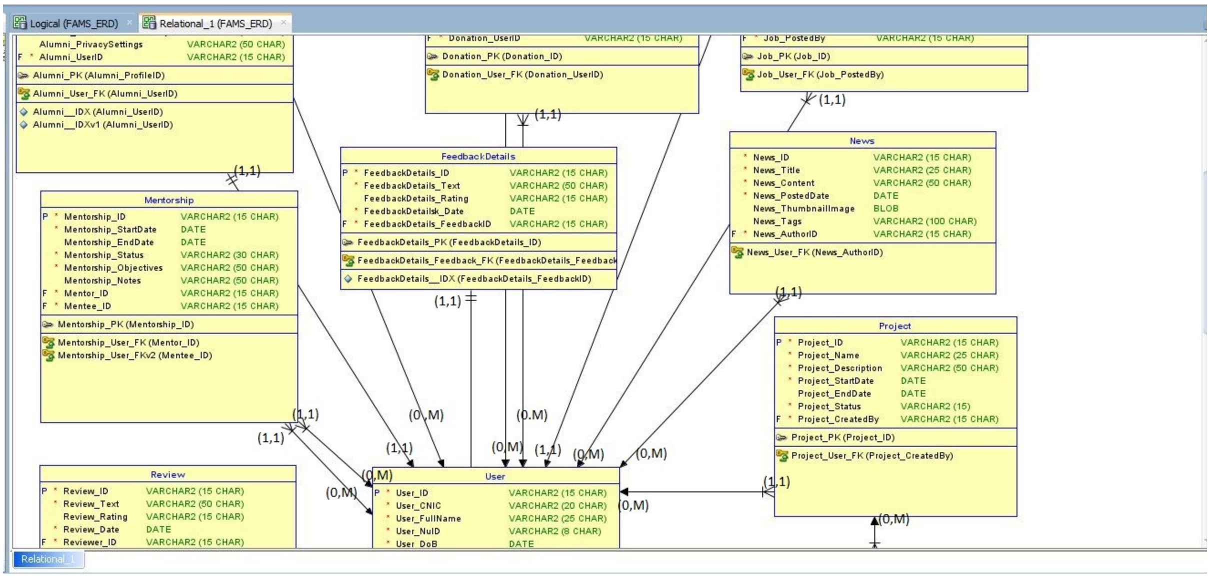
- **Role:** Users are collaborators in projects.
- **Description:** A user can have multiple collaborations across projects.
- **Foreign Key:**
    - Collaboration_UserID → User.User_ID
- **Multiplicity:**
    - **1 User : 0 or Many Collaborations**

ENTITIES BLUE PRINT IN DATA MODELER:

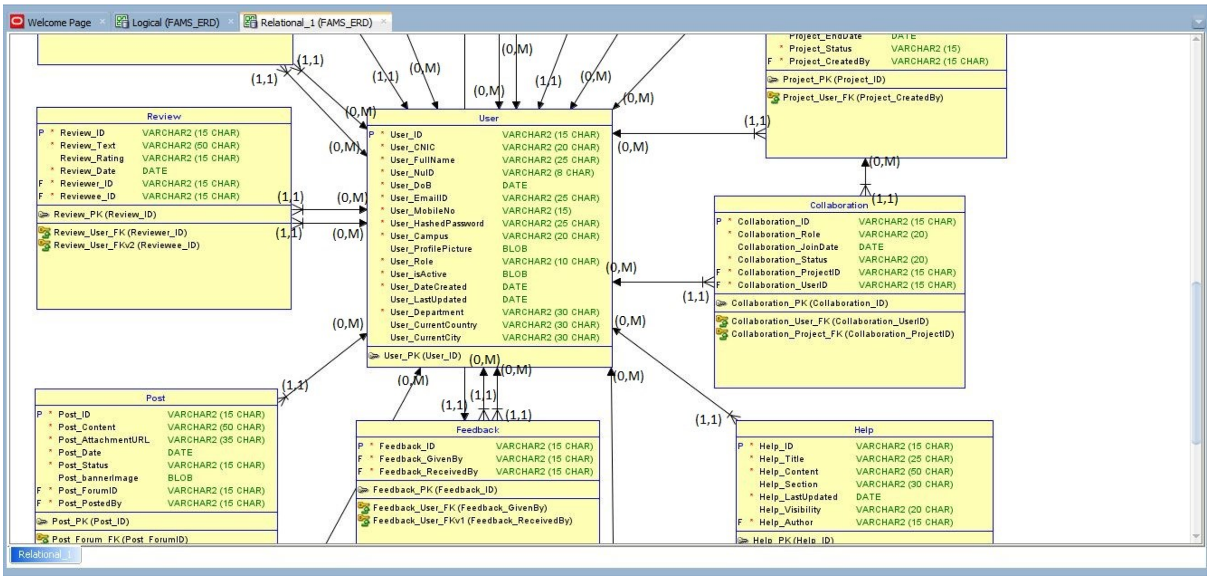


ER-DIAGRAM WITH CARDINALITY CONSTRAINT:


Logical (FAMS_ERD)
Relational_1 (FAMS_ERD)
Event
P * Event_ID VARCHAR2 (15 CHAR)
* Event_Name VARCHAR2 (25 CHAR)
Event_Description VARCHAR2 (50 CHAR)
Event_Date DATE
Event_Time TIMESTAMP WITH TIME ZONE
Event_Location VARCHAR2 (50 CHAR)
Event_MaxParticipants INTEGER
Event_RegistrationDeadline DATE
Event_Banner BLOB
Event_ParticipantCount INTEGER
* Event_CreatedDate DATE
* Event_Type VARCHAR2 (30 CHAR)
F * Event_OrganizerID VARCHAR2 (15 CHAR)
Event_PK (Event_ID)
Event_User_FK (Event_OrganizerID)
Registration
P * Registration_ID VARCHAR2 (15 CHAR)
* Registration_Status VARCHAR2 (15 CHAR)
* Registration_Date DATE
F * Registration_EventID VARCHAR2 (15 CHAR)
F * Registration_UserID VARCHAR2 (15 CHAR)
Registration_PK (Registration_ID)
Registration_Event_FK (Registration_EventID)
Registration_User_FK (Registration_UserID)
Student
P * Student_ID VARCHAR2 (15 CHAR)
* Student_CurrentYear VARCHAR2 (10 CHAR)
* Student_CurrentSemester VARCHAR2 (15 CHAR)
Student_GPA FLOAT
F * Student_UserID VARCHAR2 (15 CHAR)
Student_PK (Student_ID)
Student_User_FK (Student_UserID)
Student__IDX (Student_UserID)
Student__IDXv1 (Student_UserID)
(1,1)
(0,M)
(1,1)
(1,1)
(1,1)
Alumni
P * Alumni_ProfileID VARCHAR2 (15 CHAR)
Alumni_Organization VARCHAR2 (35 CHAR)
Alumni_CurrentPosition VARCHAR2 (35 CHAR)
Alumni_LinkedInProfile VARCHAR2 (50 CHAR)
Alumni_Industry VARCHAR2 (25 CHAR)
Alumni_ProfessionalSummary VARCHAR2 (50 CHAR)
Alumni_PrivacySettings VARCHAR2 (50 CHAR)
F * Alumni_UserID VARCHAR2 (15 CHAR)
Alumni_PK (Alumni_ProfileID)
Alumni_User_FK (Alumni_UserID)
Alumni__IDX (Alumni_UserID)
Alumni__IDXv1 (Alumni_UserID)
Donation
P * Donation_ID VARCHAR2 (15 CHAR)
* Donation_Title VARCHAR2 (25 CHAR)
Donation_description VARCHAR2 (50 CHAR)
* Donation_TargetedAmount FLOAT
* Donation_AmountRaised FLOAT
* Donation_Date DATE
* Donation_deadline DATE
* Donation_Status VARCHAR2 (15 CHAR)
* Donation_Category VARCHAR2 (25 CHAR)
F * Donation_UserID VARCHAR2 (15 CHAR)
Donation_PK (Donation_ID)
Donation_User_FK (Donation_UserID)
(1,1)
Job
P * Job_ID VARCHAR2 (15 CHAR)
* Job_Type VARCHAR2 (25 CHAR)
Job_SalaryRange VARCHAR2 (30 CHAR)
Job_ExperienceLevel VARCHAR2 (30 CHAR)
* Job_Title VARCHAR2 (25 CHAR)
* Job_Description VARCHAR2 (50 CHAR)
Job_CompanyName VARCHAR2 (25 CHAR)
Job_Location VARCHAR2 (35 CHAR)
Job_ApplicationLink VARCHAR2 (50 CHAR)
* Job_DatePosted DATE
Job_Deadline DATE
Job_Tags VARCHAR2 (100 CHAR)
F * Job_PostedBy VARCHAR2 (15 CHAR)
Job_PK (Job_ID)
Job_User_FK (Job_PostedBy)
(1,1)
FeedbackDetails
News
* News_ID VARCHAR2 (15 CHAR)
Relational_1


Logical (FAMS_ERD)
Relational_1 (FAMS_ERD)
Alumni_PrivacySettings VARCHAR2 (50 CHAR)
F * Alumni_UserID VARCHAR2 (15 CHAR)
Alumni_PK (Alumni_ProfileID)
Alumni_User_FK (Alumni_UserID)
Alumni__IDX (Alumni_UserID)
Alumni__IDXv1 (Alumni_UserID)
F * Donation_UserID VARCHAR2 (15 CHAR)
Donation_PK (Donation_ID)
Donation_User_FK (Donation_UserID)
F * Job_PostedBy VARCHAR2 (15 CHAR)
Job_PK (Job_ID)
Job_User_FK (Job_PostedBy)
(1,1)
(1,1)
(1,1)
FeedbackDetails
P * FeedbackDetails_ID VARCHAR2 (15 CHAR)
* FeedbackDetails_Text VARCHAR2 (50 CHAR)
FeedbackDetails_Rating VARCHAR2 (15 CHAR)
* FeedbackDetailsk_Date DATE
F * FeedbackDetails_FeedbackID VARCHAR2 (15 CHAR)
FeedbackDetails_PK (FeedbackDetails_ID)
FeedbackDetails_Feedback_FK (FeedbackDetails_Feedback
FeedbackDetails__IDX (FeedbackDetails_FeedbackID)
News
* News_ID VARCHAR2 (15 CHAR)
* News_Title VARCHAR2 (25 CHAR)
* News_Content VARCHAR2 (50 CHAR)
* News_PostedDate DATE
News_ThumbnailImage BLOB
News_Tags VARCHAR2 (100 CHAR)
F * News_AuthorID VARCHAR2 (15 CHAR)
News_User_FK (News_AuthorID)
Mentorship
P * Mentorship_ID VARCHAR2 (15 CHAR)
* Mentorship_StartDate DATE
Mentorship_EndDate DATE
* Mentorship_Status VARCHAR2 (30 CHAR)
* Mentorship_Objectives VARCHAR2 (50 CHAR)
Mentorship_Notes VARCHAR2 (50 CHAR)
F * Mentor_ID VARCHAR2 (15 CHAR)
F * Mentee_ID VARCHAR2 (15 CHAR)
Mentorship_PK (Mentorship_ID)
Mentorship_User_FK (Mentor_ID)
Mentorship_User_FKv2 (Mentee_ID)
(1,1)
(1,1)
(1,1)
(0,M)
(0.M)
(1,1)
(0,M)
(1,1)
(0,M)
(0,M)
(1,1)
Project
P * Project_ID VARCHAR2 (15 CHAR)
* Project_Name VARCHAR2 (25 CHAR)
* Project_Description VARCHAR2 (50 CHAR)
* Project_StartDate DATE
Project_EndDate DATE
* Project_Status VARCHAR2 (15)
F * Project_CreatedBy VARCHAR2 (15 CHAR)
Project_PK (Project_ID)
Project_User_FK (Project_CreatedBy)
(1,1)
(0,M)
(0,M)
(0,M)
Review
P * Review_ID VARCHAR2 (15 CHAR)
* Review_Text VARCHAR2 (50 CHAR)
Review_Rating VARCHAR2 (15 CHAR)
* Review_Date DATE
F * Reviewer_ID VARCHAR2 (15 CHAR)
User
P * User_ID VARCHAR2 (15 CHAR)
* User_CNIC VARCHAR2 (20 CHAR)
* User_FullName VARCHAR2 (25 CHAR)
* User_NuID VARCHAR2 (8 CHAR)
* User_DoB DATE
Relational_1

ER-DIAGRAM WITH CARDINALITY CONSTRAINT



	Project_EndDate	DATE
*	Project_Status	VARCHAR2 (15)
F *	Project_CreatedBy	VARCHAR2 (15 CHAR)
	Project_PK (Project_ID)	
	Project_User_FK (Project_CreatedBy)	

Review

P *	Review_ID	VARCHAR2 (15 CHAR)
*	Review_Text	VARCHAR2 (50 CHAR)
	Review_Rating	VARCHAR2 (15 CHAR)
*	Review_Date	DATE
F *	Reviewer_ID	VARCHAR2 (15 CHAR)
F *	Reviewee_ID	VARCHAR2 (15 CHAR)
	Review_PK (Review_ID)	
	Review_User_FK (Reviewer_ID)	
	Review_User_FKv2 (Reviewee_ID)	

User

P *	User_ID	VARCHAR2 (15 CHAR)
*	User_CNIC	VARCHAR2 (20 CHAR)
*	User_FullName	VARCHAR2 (25 CHAR)
*	User_NuID	VARCHAR2 (8 CHAR)
*	User_DoB	DATE
*	User_EmailID	VARCHAR2 (25 CHAR)
*	User_MobileNo	VARCHAR2 (15)
*	User_HashedPassword	VARCHAR2 (25 CHAR)
*	User_Campus	VARCHAR2 (20 CHAR)
	User_ProfilePicture	BLOB
*	User_Role	VARCHAR2 (10 CHAR)
*	User_isActive	BLOB
*	User_DateCreated	DATE
	User_LastUpdated	DATE
*	User_Department	VARCHAR2 (30 CHAR)
	User_CurrentCountry	VARCHAR2 (30 CHAR)
	User_CurrentCity	VARCHAR2 (30 CHAR)
	User_PK (User_ID)	

Collaboration

P *	Collaboration_ID	VARCHAR2 (15 CHAR)
*	Collaboration_Role	VARCHAR2 (20)
	Collaboration_JoinDate	DATE
*	Collaboration_Status	VARCHAR2 (20)
F *	Collaboration_ProjectID	VARCHAR2 (15 CHAR)
F *	Collaboration_UserID	VARCHAR2 (15 CHAR)
	Collaboration_PK (Collaboration_ID)	
	Collaboration_User_FK (Collaboration_UserID)	
	Collaboration_Project_FK (Collaboration_ProjectID)	

Post

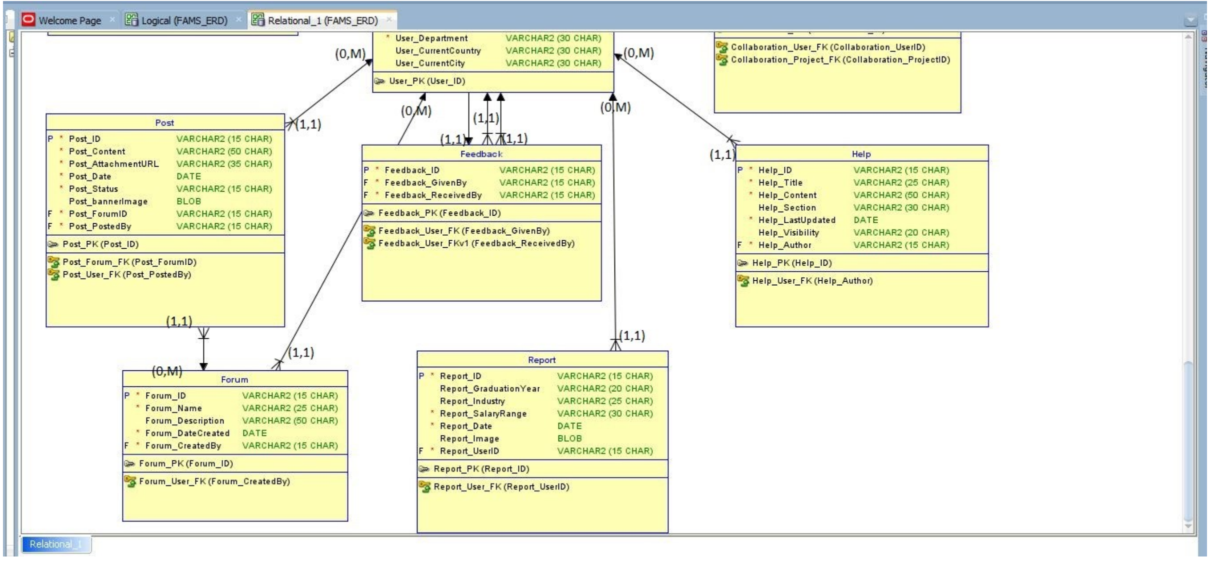
P *	Post_ID	VARCHAR2 (15 CHAR)
*	Post_Content	VARCHAR2 (50 CHAR)
*	Post_AttachmentURL	VARCHAR2 (35 CHAR)
*	Post_Date	DATE
*	Post_Status	VARCHAR2 (15 CHAR)
	Post_bannerImage	BLOB
F *	Post_ForumID	VARCHAR2 (15 CHAR)
F *	Post_PostedBy	VARCHAR2 (15 CHAR)
	Post_PK (Post_ID)	
	Post_Forum_FK (Post_ForumID)	

Feedback

P *	Feedback_ID	VARCHAR2 (15 CHAR)
F *	Feedback_GivenBy	VARCHAR2 (15 CHAR)
F *	Feedback_ReceivedBy	VARCHAR2 (15 CHAR)
	Feedback_PK (Feedback_ID)	
	Feedback_User_FK (Feedback_GivenBy)	
	Feedback_User_FKv1 (Feedback_ReceivedBy)	

Help

P *	Help_ID	VARCHAR2 (15 CHAR)
*	Help_Title	VARCHAR2 (25 CHAR)
*	Help_Content	VARCHAR2 (50 CHAR)
	Help_Section	VARCHAR2 (30 CHAR)
*	Help_LastUpdated	DATE
	Help_Visibility	VARCHAR2 (20 CHAR)
F *	Help_Author	VARCHAR2 (15 CHAR)
	Help_PK (Help_ID)	



*	User_Department	VARCHAR2 (30 CHAR)
	User_CurrentCountry	VARCHAR2 (30 CHAR)
	User_CurrentCity	VARCHAR2 (30 CHAR)
	User_PK (User_ID)	

Collaboration_User_FK (Collaboration_UserID)
Collaboration_Project_FK (Collaboration_ProjectID)

Post

P *	Post_ID	VARCHAR2 (15 CHAR)
*	Post_Content	VARCHAR2 (50 CHAR)
*	Post_AttachmentURL	VARCHAR2 (35 CHAR)
*	Post_Date	DATE
*	Post_Status	VARCHAR2 (15 CHAR)
	Post_bannerImage	BLOB
F *	Post_ForumID	VARCHAR2 (15 CHAR)
F *	Post_PostedBy	VARCHAR2 (15 CHAR)
	Post_PK (Post_ID)	
	Post_Forum_FK (Post_ForumID)	
	Post_User_FK (Post_PostedBy)	

Feedback

P *	Feedback_ID	VARCHAR2 (15 CHAR)
F *	Feedback_GivenBy	VARCHAR2 (15 CHAR)
F *	Feedback_ReceivedBy	VARCHAR2 (15 CHAR)
	Feedback_PK (Feedback_ID)	
	Feedback_User_FK (Feedback_GivenBy)	
	Feedback_User_FKv1 (Feedback_ReceivedBy)	

Help

P *	Help_ID	VARCHAR2 (15 CHAR)
*	Help_Title	VARCHAR2 (25 CHAR)
*	Help_Content	VARCHAR2 (50 CHAR)
	Help_Section	VARCHAR2 (30 CHAR)
*	Help_LastUpdated	DATE
	Help_Visibility	VARCHAR2 (20 CHAR)
F *	Help_Author	VARCHAR2 (15 CHAR)
	Help_PK (Help_ID)	
	Help_User_FK (Help_Author)	

Forum

P *	Forum_ID	VARCHAR2 (15 CHAR)
*	Forum_Name	VARCHAR2 (25 CHAR)
	Forum_Description	VARCHAR2 (50 CHAR)
*	Forum_DateCreated	DATE
F *	Forum_CreatedBy	VARCHAR2 (15 CHAR)
	Forum_PK (Forum_ID)	
	Forum_User_FK (Forum_CreatedBy)	

Report

P *	Report_ID	VARCHAR2 (15 CHAR)
	Report_GraduationYear	VARCHAR2 (20 CHAR)
	Report_Industry	VARCHAR2 (25 CHAR)
*	Report_SalaryRange	VARCHAR2 (30 CHAR)
*	Report_Date	DATE
	Report_Image	BLOB
F *	Report_UserID	VARCHAR2 (15 CHAR)
	Report_PK (Report_ID)	
	Report_User_FK (Report_UserID)	

GENERATED SCHEMA:

```
-- Generated by Oracle SQL Developer Data Modeler 23.1.0.087.0806
--   at:        2024-11-20 22:44:43 PKT
--   site:      Oracle Database 11g
--   type:      Oracle Database 11g



-- predefined type, no DDL - MDSYS.SDO_GEOMETRY

-- predefined type, no DDL - XMLTYPE

CREATE TABLE alumni (
    alumni_profileid          VARCHAR2(15 CHAR) NOT NULL,
    alumni_organization       VARCHAR2(35 CHAR),
    alumni_currentposition    VARCHAR2(35 CHAR),
    alumni_linkedinprofile    VARCHAR2(50 CHAR),
    alumni_industry           VARCHAR2(25 CHAR),
    alumni_professionalsummary VARCHAR2(50 CHAR),
    alumni_privacysettings    VARCHAR2(50 CHAR),
    alumni_userid             VARCHAR2(15 CHAR) NOT NULL
);

CREATE UNIQUE INDEX alumni__idx ON
    alumni (
        alumni_userid
    ASC );

CREATE UNIQUE INDEX alumni__idxv1 ON
    alumni (
        alumni_userid
    ASC );

ALTER TABLE alumni ADD CONSTRAINT alumni_pk PRIMARY KEY ( alumni_profileid );

CREATE TABLE collaboration (
    collaboration_id        VARCHAR2(15 CHAR) NOT NULL,
    collaboration_role      VARCHAR2(20) NOT NULL,
    collaboration_joindate  DATE,
    collaboration_status    VARCHAR2(20) NOT NULL,
    collaboration_projectid VARCHAR2(15 CHAR) NOT NULL,
    collaboration_userid    VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE collaboration ADD CONSTRAINT collaboration_pk PRIMARY KEY ( collaboration_id );

CREATE TABLE donation (
    donation_id             VARCHAR2(15 CHAR) NOT NULL,
    donation_title          VARCHAR2(25 CHAR) NOT NULL,
    donation_description    VARCHAR2(50 CHAR),
    donation_targetedamount FLOAT NOT NULL,
    donation_amountraised   FLOAT NOT NULL,
    donation_date           DATE NOT NULL,
    donation_deadline       DATE NOT NULL,
    donation_status         VARCHAR2(15 CHAR) NOT NULL,
    donation_category       VARCHAR2(25 CHAR) NOT NULL,
    donation_userid         VARCHAR2(15 CHAR) NOT NULL
);
```

```
ALTER TABLE donation ADD CONSTRAINT donation_pk PRIMARY KEY ( donation_id );

CREATE TABLE event (
 event_id                 VARCHAR2(15 CHAR) NOT NULL,
   event_name               VARCHAR2(25 CHAR) NOT NULL,
   event_description        VARCHAR2(50 CHAR),
   event_date               DATE,
   event_time               TIMESTAMP WITH TIME ZONE,
   event_location           VARCHAR2(50 CHAR),
   event_maxparticipants      INTEGER,
   event_registrationdeadline DATE,
   event_banner             BLOB,
   event_participantcount     INTEGER,
   event_createddate         DATE NOT NULL,
   event_type               VARCHAR2(30 CHAR) NOT NULL,
   event_organizerid        VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE event ADD CONSTRAINT event_pk PRIMARY KEY ( event_id );

CREATE TABLE feedback (
   feedback_id        VARCHAR2(15 CHAR) NOT NULL,
   feedback_givenby    VARCHAR2(15 CHAR) NOT NULL,
   feedback_receivedby VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE feedback ADD CONSTRAINT feedback_pk PRIMARY KEY ( feedback_id );

CREATE TABLE feedbackdetails (
   feedbackdetails_id        VARCHAR2(15 CHAR) NOT NULL,
   feedbackdetails_text       VARCHAR2(50 CHAR) NOT NULL,
   feedbackdetails_rating     VARCHAR2(15 CHAR),
   feedbackdetailsk_date      DATE NOT NULL,
   feedbackdetails_feedbackid VARCHAR2(15 CHAR) NOT NULL
);

CREATE UNIQUE INDEX feedbackdetails__idx ON
   feedbackdetails (
      feedbackdetails_feedbackid
   ASC );

ALTER TABLE feedbackdetails ADD CONSTRAINT feedbackdetails_pk PRIMARY KEY ( feedbackdetails_id );

CREATE TABLE forum (
   forum_id         VARCHAR2(15 CHAR) NOT NULL,
   forum_name       VARCHAR2(25 CHAR) NOT NULL,
   forum_description VARCHAR2(50 CHAR),
   forum_datecreated DATE NOT NULL,
   forum_createdby   VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE forum ADD CONSTRAINT forum_pk PRIMARY KEY ( forum_id );

CREATE TABLE help (
   help_id         VARCHAR2(15 CHAR) NOT NULL,
   help_title       VARCHAR2(25 CHAR) NOT NULL,
   help_content     VARCHAR2(50 CHAR) NOT NULL,
   help_section     VARCHAR2(30 CHAR),
   help_lastupdated DATE NOT NULL,
   help_visibility  VARCHAR2(20 CHAR),
   help_author      VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE help ADD CONSTRAINT help_pk PRIMARY KEY ( help_id );

CREATE TABLE job (
   job_id            VARCHAR2(15 CHAR) NOT NULL,
  job_type          VARCHAR2(25 CHAR) NOT NULL,
   job_salaryrange     VARCHAR2(30 CHAR),
   job_experiencelevel VARCHAR2(30 CHAR),
  job_title          VARCHAR2(25 CHAR) NOT NULL,
   job_description     VARCHAR2(50 CHAR) NOT NULL,
   job_companyname     VARCHAR2(25 CHAR),
   job_location       VARCHAR2(35 CHAR),
 job_applicationlink VARCHAR2(50 CHAR),
   job_dateposted      DATE NOT NULL,
   job_deadline       DATE,
   job_tags          VARCHAR2(100 CHAR),
   job_postedby       VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE job ADD CONSTRAINT job_pk PRIMARY KEY ( job_id );

CREATE TABLE mentorship (
   mentorship_id         VARCHAR2(15 CHAR) NOT NULL,
   mentorship_startdate  DATE NOT NULL,
   mentorship_enddate    DATE,
   mentorship_status     VARCHAR2(30 CHAR) NOT NULL,
   mentorship_objectives VARCHAR2(50 CHAR) NOT NULL,
   mentorship_notes      VARCHAR2(50 CHAR),
   mentor_id            VARCHAR2(15 CHAR) NOT NULL,
   mentee_id            VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE mentorship ADD CONSTRAINT mentorship_pk PRIMARY KEY ( mentorship_id );

CREATE TABLE news (
   news_id           VARCHAR2(15 CHAR) NOT NULL,
   news_title         VARCHAR2(25 CHAR) NOT NULL,
   news_content       VARCHAR2(50 CHAR) NOT NULL,
   news_posteddate     DATE NOT NULL,
   news_thumbnailimage BLOB,
   news_tags          VARCHAR2(100 CHAR),
   news_authorid       VARCHAR2(15 CHAR) NOT NULL
);

CREATE TABLE post (
   post_id           VARCHAR2(15 CHAR) NOT NULL,
   post_content       VARCHAR2(50 CHAR) NOT NULL,
   post_attachmenturl VARCHAR2(35 CHAR) NOT NULL,
   post_date          DATE NOT NULL,
   post_status        VARCHAR2(15 CHAR) NOT NULL,
   post_bannerimage   BLOB,
   post_forumid       VARCHAR2(15 CHAR) NOT NULL,
   post_postedby      VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE post ADD CONSTRAINT post_pk PRIMARY KEY ( post_id );

CREATE TABLE project (
   project_id         VARCHAR2(15 CHAR) NOT NULL,
   project_name       VARCHAR2(25 CHAR) NOT NULL,
   project_description VARCHAR2(50 CHAR) NOT NULL,
   project_startdate   DATE NOT NULL,
   project_enddate     DATE,
   project_status      VARCHAR2(15) NOT NULL,
   project_createdby   VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE project ADD CONSTRAINT project_pk PRIMARY KEY ( project_id );

CREATE TABLE registration (
   registration_id      VARCHAR2(15 CHAR) NOT NULL,
   registration_status  VARCHAR2(15 CHAR) NOT NULL,
registration_date    DATE NOT NULL,
   registration_eventid VARCHAR2(15 CHAR) NOT NULL,
   registration_userid  VARCHAR2(15 CHAR) NOT NULL

);

ALTER TABLE registration ADD CONSTRAINT registration_pk PRIMARY KEY ( registration_id );

CREATE TABLE report (
   report_id             VARCHAR2(15 CHAR) NOT NULL,
   report_graduationyear VARCHAR2(20 CHAR),
   report_industry       VARCHAR2(25 CHAR),
   report_salaryrange    VARCHAR2(30 CHAR) NOT NULL,
   report_date           DATE NOT NULL,
   report_image          BLOB,
   report_userid         VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE report ADD CONSTRAINT report_pk PRIMARY KEY ( report_id );

CREATE TABLE review (
   review_id     VARCHAR2(15 CHAR) NOT NULL,
   review_text   VARCHAR2(50 CHAR) NOT NULL,
   review_rating VARCHAR2(15 CHAR),
   review_date   DATE NOT NULL,
   reviewer_id   VARCHAR2(15 CHAR) NOT NULL,
   reviewee_id   VARCHAR2(15 CHAR) NOT NULL
);

ALTER TABLE review ADD CONSTRAINT review_pk PRIMARY KEY ( review_id );

CREATE TABLE student (
   student_id              VARCHAR2(15 CHAR) NOT NULL,
   student_currentyear     VARCHAR2(10 CHAR) NOT NULL,
   student_currentsemester VARCHAR2(15 CHAR) NOT NULL,
   student_gpa             FLOAT,
   student_userid          VARCHAR2(15 CHAR) NOT NULL
);

CREATE UNIQUE INDEX student__idx ON
   student (
       student_userid
   ASC );

CREATE UNIQUE INDEX student__idxv1 ON
   student (
       student_userid
   ASC );

ALTER TABLE student ADD CONSTRAINT student_pk PRIMARY KEY ( student_id );

CREATE TABLE "User" (
   user_id             VARCHAR2(15 CHAR) NOT NULL,
   user_cnic           VARCHAR2(20 CHAR) NOT NULL,
   user_fullname       VARCHAR2(25 CHAR) NOT NULL,
   user_nuid           VARCHAR2(8 CHAR) NOT NULL,
   user_dob            DATE NOT NULL,
   user_emailid        VARCHAR2(25 CHAR) NOT NULL,
   user_mobileno       VARCHAR2(15) NOT NULL,
   user_hashedpassword VARCHAR2(25 CHAR) NOT NULL,
   user_campus         VARCHAR2(20 CHAR) NOT NULL,
   user_profilepicture BLOB,
   user_role           VARCHAR2(10 CHAR) NOT NULL,
   user_isactive       BLOB NOT NULL,
   user_datecreated    DATE NOT NULL,
   user_lastupdated    DATE,
   user_department     VARCHAR2(30 CHAR) NOT NULL,
user_currentcountry VARCHAR2(30 CHAR),
   user_currentcity    VARCHAR2(30 CHAR)
);


ALTER TABLE "User" ADD CONSTRAINT user_pk PRIMARY KEY ( user_id );

ALTER TABLE alumni
   ADD CONSTRAINT alumni_user_fk FOREIGN KEY ( alumni_userid )
      REFERENCES "User" ( user_id );

ALTER TABLE collaboration
   ADD CONSTRAINT collaboration_project_fk FOREIGN KEY ( collaboration_projectid )
      REFERENCES project ( project_id );

ALTER TABLE collaboration
   ADD CONSTRAINT collaboration_user_fk FOREIGN KEY ( collaboration_userid )
      REFERENCES "User" ( user_id );

ALTER TABLE donation
   ADD CONSTRAINT donation_user_fk FOREIGN KEY ( donation_userid )
      REFERENCES "User" ( user_id );

ALTER TABLE event
   ADD CONSTRAINT event_user_fk FOREIGN KEY ( event_organizerid )
      REFERENCES "User" ( user_id );

ALTER TABLE feedback
   ADD CONSTRAINT feedback_user_fk FOREIGN KEY ( feedback_givenby )
      REFERENCES "User" ( user_id );

ALTER TABLE feedback
   ADD CONSTRAINT feedback_user_fkv1 FOREIGN KEY ( feedback_receivedby )
      REFERENCES "User" ( user_id );

ALTER TABLE feedbackdetails
   ADD CONSTRAINT feedbackdetails_feedback_fk FOREIGN KEY ( feedbackdetails_feedbackid )
      REFERENCES feedback ( feedback_id );

ALTER TABLE forum
   ADD CONSTRAINT forum_user_fk FOREIGN KEY ( forum_createdby )
      REFERENCES "User" ( user_id );

ALTER TABLE help
   ADD CONSTRAINT help_user_fk FOREIGN KEY ( help_author )
      REFERENCES "User" ( user_id );

ALTER TABLE job
   ADD CONSTRAINT job_user_fk FOREIGN KEY ( job_postedby )
      REFERENCES "User" ( user_id );

ALTER TABLE mentorship
   ADD CONSTRAINT mentorship_user_fk FOREIGN KEY ( mentor_id )
      REFERENCES "User" ( user_id );

ALTER TABLE mentorship
   ADD CONSTRAINT mentorship_user_fkv2 FOREIGN KEY ( mentee_id )
      REFERENCES "User" ( user_id );

ALTER TABLE news
   ADD CONSTRAINT news_user_fk FOREIGN KEY ( news_authorid )
      REFERENCES "User" ( user_id );

ALTER TABLE post
   ADD CONSTRAINT post_forum_fk FOREIGN KEY ( post_forumid )
      REFERENCES forum ( forum_id );
```

```
ALTER TABLE post
   ADD CONSTRAINT post_user_fk FOREIGN KEY ( post_postedby )
      REFERENCES "User" ( user_id );

ALTER TABLE project

 ADD CONSTRAINT project_user_fk FOREIGN KEY ( project_createdby )
      REFERENCES "User" ( user_id );

ALTER TABLE registration
   ADD CONSTRAINT registration_event_fk FOREIGN KEY ( registration_eventid )
      REFERENCES event ( event_id );

ALTER TABLE registration
   ADD CONSTRAINT registration_user_fk FOREIGN KEY ( registration_userid )
      REFERENCES "User" ( user_id );

ALTER TABLE report
   ADD CONSTRAINT report_user_fk FOREIGN KEY ( report_userid )
      REFERENCES "User" ( user_id );

ALTER TABLE review
   ADD CONSTRAINT review_user_fk FOREIGN KEY ( reviewer_id )
      REFERENCES "User" ( user_id );

ALTER TABLE review
   ADD CONSTRAINT review_user_fkv2 FOREIGN KEY ( reviewee_id )
      REFERENCES "User" ( user_id );

ALTER TABLE student
   ADD CONSTRAINT student_user_fk FOREIGN KEY ( student_userid )
      REFERENCES "User" ( user_id );



-- Oracle SQL Developer Data Modeler Summary Report:
--
-- CREATE TABLE                            18
-- CREATE INDEX                             5
-- ALTER TABLE                             40
-- CREATE VIEW                              0
-- ALTER VIEW                               0
-- CREATE PACKAGE                           0
-- CREATE PACKAGE BODY                      0
-- CREATE PROCEDURE                         0
-- CREATE FUNCTION                          0
-- CREATE TRIGGER                           0
-- ALTER TRIGGER                            0
-- CREATE COLLECTION TYPE                   0
-- CREATE STRUCTURED TYPE                   0
-- CREATE STRUCTURED TYPE BODY              0
-- CREATE CLUSTER                           0
-- CREATE CONTEXT                           0
-- CREATE DATABASE                          0
-- CREATE DIMENSION                         0
-- CREATE DIRECTORY                         0
-- CREATE DISK GROUP                        0
-- CREATE ROLE                              0
-- CREATE ROLLBACK SEGMENT                  0
-- CREATE SEQUENCE                          0
-- CREATE MATERIALIZED VIEW                 0
-- CREATE MATERIALIZED VIEW LOG             0
-- CREATE SYNONYM                           0
-- CREATE TABLESPACE                        0
-- CREATE USER                              0
--
-- DROP TABLESPACE                          0
-- DROP DATABASE                  0
--
-- REDACTION POLICY                0
--
-- ORDS DROP SCHEMA                 0
-- ORDS ENABLE SCHEMA                0

-- ORDS ENABLE OBJECT                0
--
-- ERRORS                        0
-- WARNINGS                        0
```