Database Design Project Journal

Contributors

Name	Email
Sunil Kuruba	skuru@uic.edu
Niharika Belavadi Shekar	nbela@uic.edu

Project Description

Help Session Activity app

The Help Session Activity App is an innovative platform designed to enhance academic support by connecting students with Teaching Assistants (TAs) for personalized assistance. It allows students to book sessions tailored to their specific needs, whether it's help with grading, assignments, or general questions. TAs can manage their availability through organized shifts, ensuring efficient time management and accessibility for students. This application not only simplifies the scheduling process but also fosters effective communication between students and TAs, creating a collaborative learning environment.

In addition to session management, the app incorporates a feedback mechanism where students can rate their experiences, providing valuable insights into the performance of TAs and the overall quality of assistance provided. With reporting features, the app generates data on student attendance, TA effectiveness, and session ratings, which can be utilized for continuous improvement of academic support services. Ultimately, the Help Session Activity App aims to streamline academic assistance, making it more efficient and user-friendly, while empowering students to take charge of their learning journey.

Requirements

Questions to be covered

1. How many TAs are working Friday afternoon?
2. How many students did each TA see each shift?
3. What did students most need assistance with?
4. What was the average session time?
5. Which TA helped the most students last month?
6. What was the longest wait time to start a session?
7. How many students didn't show up for sessions last week?
8. Which TA has the highest rating?
9. Are ratings different based on session time or day of week?

The Help Session Activity App facilitates the organization and management of student help sessions run by Teaching Assistants (TAs). The app tracks various aspects, including:

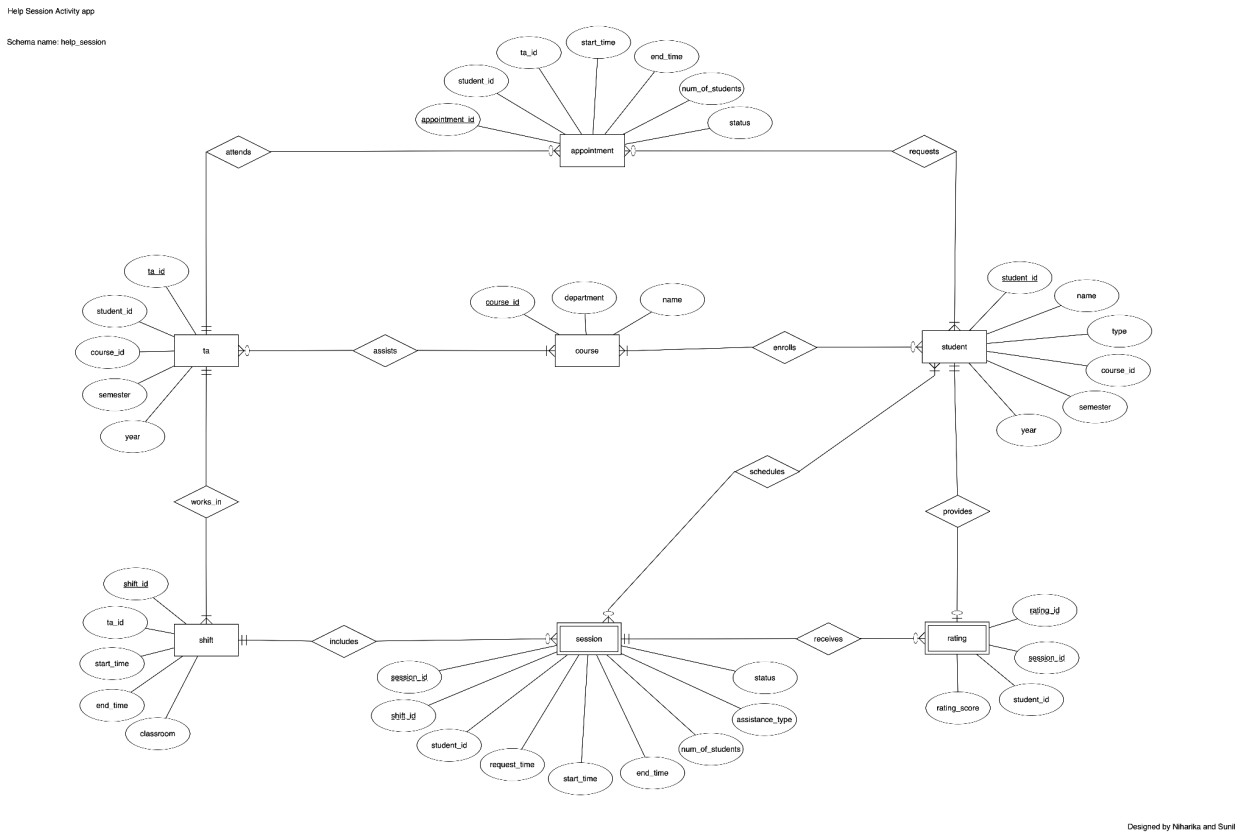
1. **Courses** offered and managed by TAs.
2. **Shifts** during which TAs are available for student sessions.
3. **Sessions** where students can book time with a TA.
4. **Appointments** for other student activities like consultation and guidance.
5. **Ratings** given by students after each session.
6. **Student** information and their related courses and sessions.

The system supports:

- Managing course and session schedules.
- Allowing students to book sessions and appointments.
- Enabling students to rate the TAs.
- Tracking attendance and session status.

Entity-relationship diagram

Drive Link: ERD_v2.png (Use Drive link for high quality and comfortable view)
ERD Plus Link: [ERD plus link] (This might not open for you, checkout the above link)



Open questions/issues

1. Wait time in app is the difference between request_time and start_time?

Process notes

Step	Activity
Discover entities, relationships, and attributes	
1A	Identify entities, relationships, and attributes in interviews.

Entities: TA, Student, Shift, Session, Rating, Appointment, Course
Relationships:

- TA teaches courses.
- TA works multiple shifts.
- Shifts have multiple sessions.
- Students book sessions/appointments.
- TAs attend appointments.
- Students give ratings to TAs.
- Students take one or more courses.

Attributes:

1. Course

A. `Course_ID`: Unique identifier for the course.
B. `Name`: Name of the course.
C. `Department`: The department offering the course (e.g., Computer Science, Math).

2. TA (Teaching Assistant)

A. `TA_ID`: Unique identifier for the TA.
B. `Student_ID`: Identifier for the TA if they are also a student.
C. `Semester`: The semester during which the TA is working (e.g., Fall 2024).
D. `Year`: The academic year.
E. `Course_ID`: Foreign key linking the TA to the course(s) they are assigned to assist with.

3. Shift

A. `Shift_ID`: Unique identifier for the shift.
B. `TA_ID`: Foreign key linking the shift to the TA who is working that shift.
C. `Start_Time`: The start time of the shift.

D. `End_Time`: The end time of the shift.
E. `Classroom`: The location or virtual room where the help session takes place.

4. Session

A. `Session_ID`: Unique identifier for the session.
B. `Shift_ID`: Foreign key linking the session to the TA's shift.
C. `Student_ID`: Foreign key linking the session to the student who booked it.
D. `Request_Time`: Time when the student requested the session.
E. `Start_Time`: Start time of the session.
F. `End_Time`: End time of the session.
G. `Number_Of_Students`: Number of students attending the Session
H. `Status`: Indicates whether the session was completed or marked as a no-show.
I. `Assistance_Type`: The type of help provided during the session (e.g., grading, assignment assistance, Q&A).

5. Appointment

A. `Appointment_ID`: Unique identifier for the appointment.
B. `Student_ID`: Foreign key linking the appointment to the student.
C. `TA_ID`: Foreign key linking the appointment to the TA.
D. `Start_Time`: The start time of the appointment.
E. `End_Time`: The end time of the appointment.
F. `Number_Of_Students`: Number of students attending the Appointment
G. `Status`: Indicates whether the appointment was completed or marked as a no-show.

6. Rating

A. `Rating_ID`: Unique identifier for the rating.
B. `Session_ID`: Foreign key linking the rating to the specific session.
C. `Student_ID`: Foreign key linking the rating to the student who provided it.
D. `Rating_Score`: A numeric score (e.g., 1-5) representing the student's feedback on the session.

7. Student

A. `Student_ID`: Unique identifier for the student.
B. `Name`: Full name of the student.

C. `Type`: The type of student (e.g., undergraduate, graduate).
D. `Course_ID`: Foreign key linking the student to the course(s) they are enrolled in.

1B	Draw ER diagram

Drive Link: ERD_v2.png
ERD Plus Link: ERD plus link [This might not open for you, checkout the above link]

1C	List standard attribute types in glossary.

Done

1D	Document names, synonyms, and descriptions in glossary.

Checkout **CS480-project**

Determine cardinality

Step	Activity
2A	Determine relationship maxima and minima.

- A **TA** can assist **many** courses and must assist at least **one** course.
- Each **Course** can have **many TAs** or **zero TAs**.
- A **TA** works in **many** shifts and must be part of at least **one** shift.
- A **Shift** can have at least and at most **one TA**.
- A **Shift** can have **many** sessions or **zero** sessions.
- A **Session** is part of at least and at most **one Shift**.
- A **Session** can receive **many** ratings or **zero** ratings.
- A **Rating** belongs to at least and at most **one Session**.
- A **Session** consists of **many** students or at least **one** student.
- A **Student** can be part of **many** sessions or **zero** sessions.
- A **Student** gives **many** ratings or **zero** ratings.
- A **Student** can request **many** appointments or **zero** appointments.
- An **Appointment** can be scheduled by **many** students or at least **one** student.
- A **Student** enrolls in **many** courses but at least **one** course.
- A **Course** can be taken by **many** students or **zero** students.

2B	Determine attribute maxima and minima.

Course

- Course_ID **R U**
- Name **R**
- Department **R**

TA (Teaching Assistant)

- TA_ID **R U**
- Student_ID **R**
- Semester **R**
- Year **R**
- Course_ID **R**
- **(**Student_ID, Semester, Year, Course_ID**) U**

Shift

- Shift_ID **R U**
- TA_ID **R**
- Start_Time **R**
- End_Time **R**
- Classroom **R**

Session

- Session_ID **R**
- Shift_ID **R**
- Student_ID **P R**
- Request_Time **R**
- Start_Time **R**
- End_Time **R**
- Num_Of_Students **R**
- Status **R**
- Assistance_Type **R**
- (Session_ID, Shift_ID) **U**

Appointment

- Appointment_ID **R U**
- Student_ID **P R**
- TA_ID **R**

- Start_Time **R**
- End_Time **R**
- Num_Of_Students **R**
- Status **R**

Rating

- Rating_ID **R**
- Session_ID **R**
- Student_ID **R**
- Rating_Score **R**
- (Session_ID, Student_ID) **U**

Student

- Student_ID **R U**
- Name **R**
- Type **R**
- Course_ID **P R**

2C	Identify unique attributes.
Refer above	
2D	Document cardinality in glossary and, optionally, on ER diagram.
Done	

Distinguish strong and weak entities

3A	Identify strong and weak entities.

Session is a weak entity, depends on **Shift**

Rating is a weak entity, depends on **Session**

3B	Determine the identifying relationship(s) for each weak entity.
Mentioned above	
3C	Document weak entities and identifying relationships

	in glossary and ER diagram.
Done	
Create supertype and subtype entities	
4A	Identify supertype and subtype entities.
Not applicable	
4B	Replace similar entities and optional attributes with supertype and subtype entities.
Not applicable	
4C	Identify partitions and partition attributes.
Not applicable	
4D	Document supertypes, subtypes, and partitions in glossary and ER diagram.
Not applicable	
Implement entities	
5A	Implement strong entities as tables.
Done. Schema name is **help_session** Shema Dump Drive link**:** **Niharika_Belavadi_Shekar and Sunil_Kuruba - Help_SessionDump20241019**	
5B	Create an artificial key when no suitable primary key exists.
Not applicable	
5C	Implement weak entities as tables.
Session and Rating entities are the weak entities	
5D	Implement supertype and subtype entities as tables.
Not applicable	

Implement relationships	
6A	Implement many-one relationships as a foreign key on the 'many' side.
Not applicable	
6B	Implement one-one relationships as a foreign key in the table with fewer rows.
Not applicable	
6C	Implement many-many relationships as new weak tables.
Not applicable	
Implement relationships	
7A	Implement plural attributes as new weak tables.
Not applicable	
7B	Specify cascade and restrict rules on new foreign keys in weak tables.
Done. Using ON DELETE CASCADE and ON UPDATE RESTRICT rules	
7C	Specify column data types corresponding to attribute types.
Done as per the data dictionary	
7D	Enforce relationship and attribute cardinality with UNIQUE and NOT NULL keywords.
Done as per the data dictionary	
Applying normal form	
8A	Identify dependencies on non-unique columns.
Not applicable	
8B	Eliminate redundancy by decomposing tables.
Not applicable	

8C	Consider denormalizing tables in reporting databases.
Not applicable	