OSC
Open Source Club

CLIENT NAME:

Open Source Club

COURSE NAME:

ISA301: Fundamentals of Database Design

TERM:

Spring 2023

American University of Sharjah

Final Report : OSC

11 May, 2023

UNDER THE GUIDANCE OF

Dr. Norita Ahmad

PREPRARED BY:

Bhavika Vohra	G00089132
Colin Sunny	B00085300
Mohammed Hossain	B00091128
Muhammed Dar	B00084081
Nabeel Naheem	B00084483

CONTENTS

(Pages are hyperlinked)

ACKNOWLEDGMENTS

Our team would like to thank Dr. Norita Ahmad for her continuous support, guidance, and encouragement during the timeline of this project.

We would also like to thank the OSC team for their valuable insight into the day-to-day operations of the organization. We also appreciate their support in providing us with details regarding the ongoing problems in the organization, one of the main aspects of our project.

Lastly, we would also like to thank Prem Rajendran, the Teaching Assistant for ISA301, for motivating us to do our best in the project and for providing us with his input every step of the way.

EXECUTIVE SUMMARY

The Open Source Club (OSC) is a student-led organization at the American University of Sharjah. In this project, we have created a database management system (DBMS) for OSC to use for internal operations.

We have created tables such as Member, Activities_Team, Executive_Team, Media_Team, Bills, Budget, Media_Content, Collaborators, and so on.

We have also analyzed the problems faced by OSC and solved them using DBMS and SQL. One problem faced by OSC is the inability to store media and event ideas. However, we solve this problem by creating an Idea_Repository table.

To conclude this project, we have also created a few business queries that we believe OSC would like to analyze to help them grow as a student organization.

As the project is based upon the Open Source Club, we have added the source code of this project to GitHub, to support the Open Source Community.

Link : https://github.com/bhavikavohra/ISA-301-Proj-OSC.git

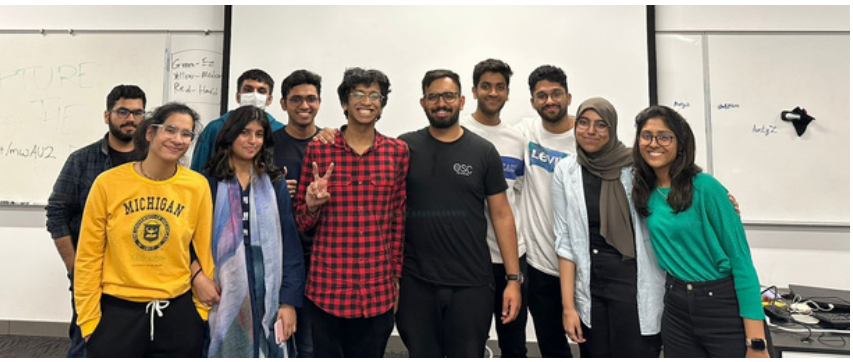
PART 1
INTRODUCTION AND CONCEPTUAL DESIGN

INTRODUCTION

01 About OSC

The Open Source Club (OSC) (est. 2020) is a student-led organization at the American University of Sharjah that aims to unite open-source enthusiasts across all majors through the promotion of open-source platforms. Currently holding 5 - 10 events per semester such as Capture the Flag, hackathons, and workshops/seminars, they wish to increase awareness of open-source software. The Open Source Club currently also posts biweekly reels on their Instagram account, titled 'Techy Tuesday' where they educate users about open source software that they can use in their daily lives.

Open source software is code that is designed to be publicly accessible (anyone can see, modify, and distribute the code as they see fit). Open-source software is developed in a decentralized and collaborative way, relying on peer review and community production. Examples of open-source software include GIMP, Audacity, WordPress, and so on.



02 Objective of the Project

For our project, we aim to create an internal database management system for OSC to use for internal affairs. We wish to create a system by which they have the ability to track events, bills and budgets, team members, and media content. We also aim to understand the current problems in the club to use this database to provide them with accurate solutions.

DID YOU KNOW?

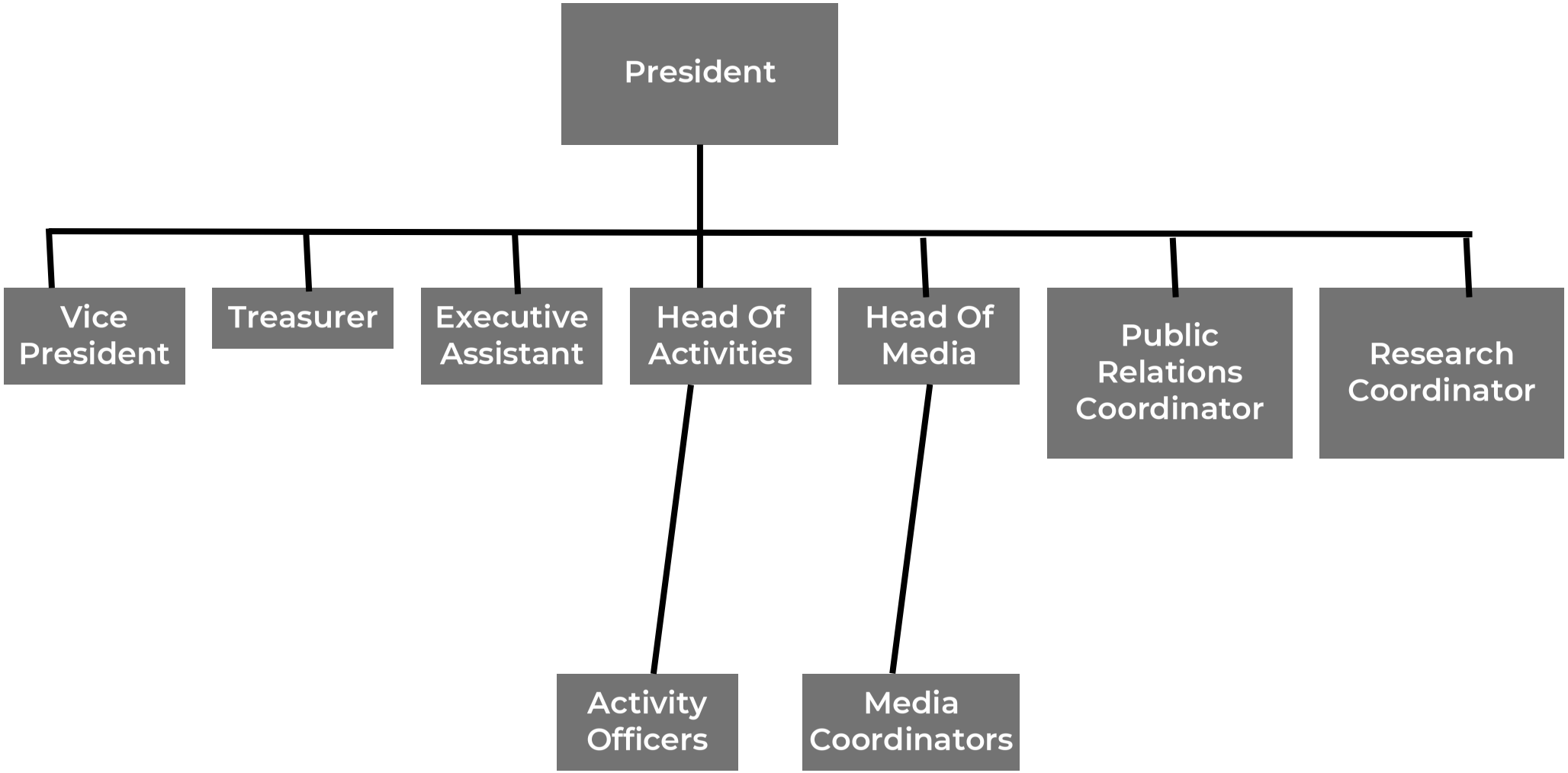
OSC was founded in 2020, by a group of friends who wished to promote the concept of Open-Source amongst the AUS community

CAPTURE THE FLAG

Capture the Flag, an annual OSC programming event interlaced with a treasure hunt attracted 50 participants from majors such as business and computer science

INTRODUCTION

03 Organization Structure of OSC



04 Motivation

The reasons we chose OSC for our project are:

1) The rise and increasing popularity of Open-Source Platforms in recent years due to their many benefits such as code customization and improved reliability. Multiple open-source projects are being developed daily for different fields such as EnRecipes for meal planning, Element for video conferencing, Photopea for video editing, and so on

2) One of our team members, Bhavika has been a part of OSC for the last 2 years as the Public Relations Coordinator. She is a first-hand witness to the inefficiencies of the systems currently used by OSC and can provide us with insider information regarding the working of the club.

3) Some members of our team are registered for ISA383 (Python for Business). As Python is open-source software, their interest in open-source platforms and OSC has broadened, pushing us to use OSC as an organization for our project.

05 Points of Contact

Our main points of contact for this project include:
1) Prem Rajendran - President of the Open Source Club

Prem has provided us with information regarding the current working of the club. He has also expressed the problems and inefficiencies of the club and the current management system. Other than that, he has also helped us to answer some questions which we have used to determine some business rules of the organization. Some questions include :

a) What kind of events does OSC host?
b) What are the different teams in OSC? (such as activities, media, etc)
c) Where does OSC store event ideas?
We aim to have conversations with him to further understand the working of OSC.

2) Bhavika Vohra - Public Relations Coordinator for the Open Source Club

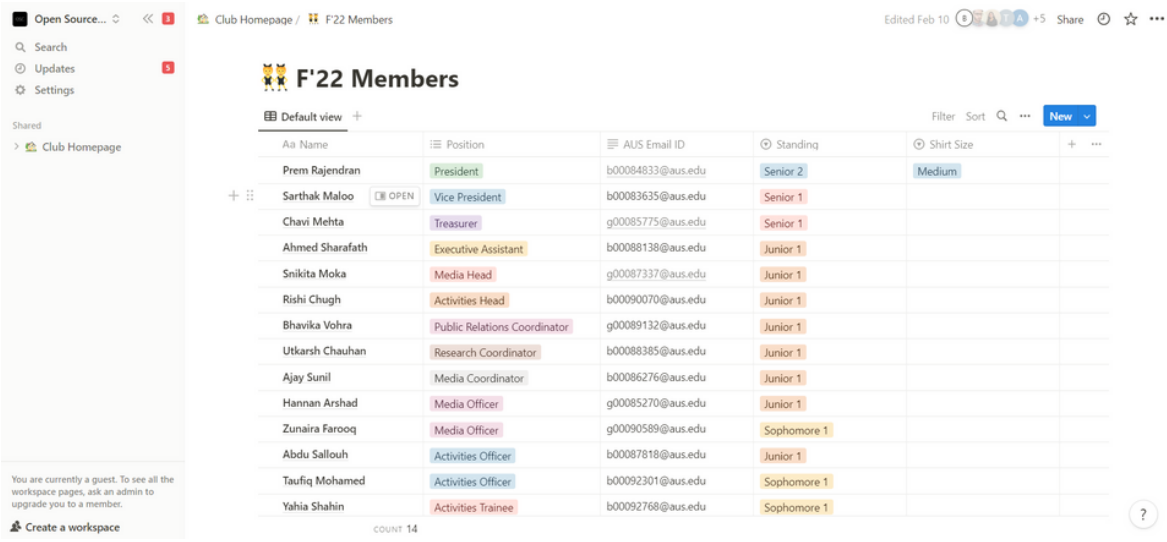
Bhavika can provide us with problems and inefficiencies that she has faced using the current management system. She can also contact OSC members to know the problems they encounter while using the current OSC management system. This will help to further understand the opportunities we can address in our project.

06 Background - Current Operations

Open Source Club is a club that engages in the collaboration and sharing of code. It accepts members of all standing regardless of their technical skills or media skills.

To store the data regarding members and events, OSC uses Notion, a free productivity and note-taking web application developed by Notion Labs Inc. It offers organizational tools including task management, project tracking, to-do lists, bookmarking, and more. OSC stores various data like student information, meeting logs, event calendars, and their idea repository.

When the club was initially established, it was easy to track and store data using Notion. However, due to the increased scalability of the club (in regard to team members and events), we are proposing to build a database management system in SQL Oracle for the club.



07 Problems and Opportunities

1)

PROBLEM:

The current system does not have a place to track bills and keep track of financial transactions related to the budget of the event. Committee members are required to make purchases before events and are reimbursed after the event. Currently, the president keeps manual track of pending receipts and reimbursements, which is very impractical, as it is a very time-consuming and tedious task.

SOLUTION:

Our project aims to include a Budget and Bills table in our database to record and track financial transactions that have been made for OSC. We plan to record data relating to financial transactions such as Date of Purchase, Purchaser, Amount of Purchase, Reason of Purchase, Merchant, and so forth. We find this problem of storing financial records as an opportunity to create and record financial transactions to increase the efficiency of the reimbursement process. We also aim to handle reimbursed bills in a more effective manner by marking them reimbursed in the database management system, rather than relying on traditional methods such as paper/memory.

2)

PROBLEM:

OSC wants to find the correlation between Event Attendance and the Likes/Views of Media Content. They wish to know if more likes/views on media content lead to more attendance at an event

SOLUTION:

To find a solution to this problem, we will create tables such as event (with attributes such as attendance) and media content (with attributes such as number of likes/views). We will then join these 2 tables and find the correlation between these two attributes.

07 Problems and Opportunities

3)

PROBLEM:

Another flaw is that the present system does not keep track of data from past collaborations with other internal or external organizations. OSC would like to store this data, as in the future it would help them recognize their collaborators.

SOLUTION:

We can use this problem as an opportunity to build a collaborators table, where we would store data of internal and external collaborators. We would store data such as NameOfCollaborator, Organization of Collaborator, Email-ID, PhoneNumber, Level of Education, and so on. This table would be helpful to recognize our collaborators and to get insight into the collaborators we can collaborate with on future events.

4)

PROBLEM:

OSC currently has an Idea_Repository set up in Notion. However, this Idea_Repository is redundant as it tracks incomplete information about the Idea, such as the IdeaName and Brief Description of the Idea.

SOLUTION:

We plan to create a table called Idea_Repository where we would be able to track additional details about the Idea, including but not limited to IdeaName, IdeaDescription, IdeaCategory (media idea or event idea), the member who suggested the idea, special resources needed for the Idea, and so on.

5)

PROBLEM:

OSC wants to start a flagship event, OSCDataScienceSeries, an event series where they promote the use of data science in different fields. They want to know whether they should host this event in Fall or Spring, in order to get the highest attendance.

SOLUTION:

We can do so by finding the average attendance of events hosted in the Fall and Spring semesters, using our newly created database management system.

EERD and BUSINESS RULES

Business Rules of Open Source Club ERD

1) Committee Members have to and can only be in 1 of 3 teams (total and disjoint constraint): Executive Team (including President, Vice_President, Executive Assistant, and Treasurer), Activities Team (Including Head_Of_Activities, Activities_Coordinator, Public_Relations_Coordinator, and Research_Coordinator), and Media Team (Media_Coordinator and Media_Officer)

2) A committee member must belong to one of the teams and can only be a part of one team.

3) An event can be a collaborated event. However, an event does not have to be a collaborated event (ie. it can be an event without any collaboration (or standalone)). Thus, the subtype relationship is disjoint and partial. An event cannot have a collaboration and no collaboration at the same time.

4) A Collaborated Event can have one to many Event_Collaborated but an Event Collaborated can have one and only one Collaborated Event. Event_Collaborated is an associative entity, that helps solve the many-to-many relationship between Collaborated Events and Collaborators.

5) An Event_Collaborated can have only one Collaborator but a Collaborator can have one to many Event_Collaborated.

6) A Collaborated event can have one to many Collaborators and a Collaborator can take part in one to many Collaborated Event

7) Collaborators can be of 2 types, internal and external. (collaborators are non-OSC individuals who can come to participate, speak in, or host OSC events.)

8) Collaborators are either internal or external (disjoint, no overlapping) and must belong to either category (total constraint).

9) An Activities Team member can manage zero to many Events, and each Event is managed by one and only one Activities Team member.

10) An Event must have 1 to many Media Content associated with it, but each Media Content is associated with zero to one Event. (for example, Techy Tuesday (an initiative where an open-source platform is promoted on a bi-weekly basis) is not an Event but still has Media Content associated with it).

11) A Media Team member can make 0 to many Media Content, but a Media Content is made by one and only one Media Team member.

12) The Executive Board members manage zero to many Bills, but each Bill is handled by only one Executive Team member.

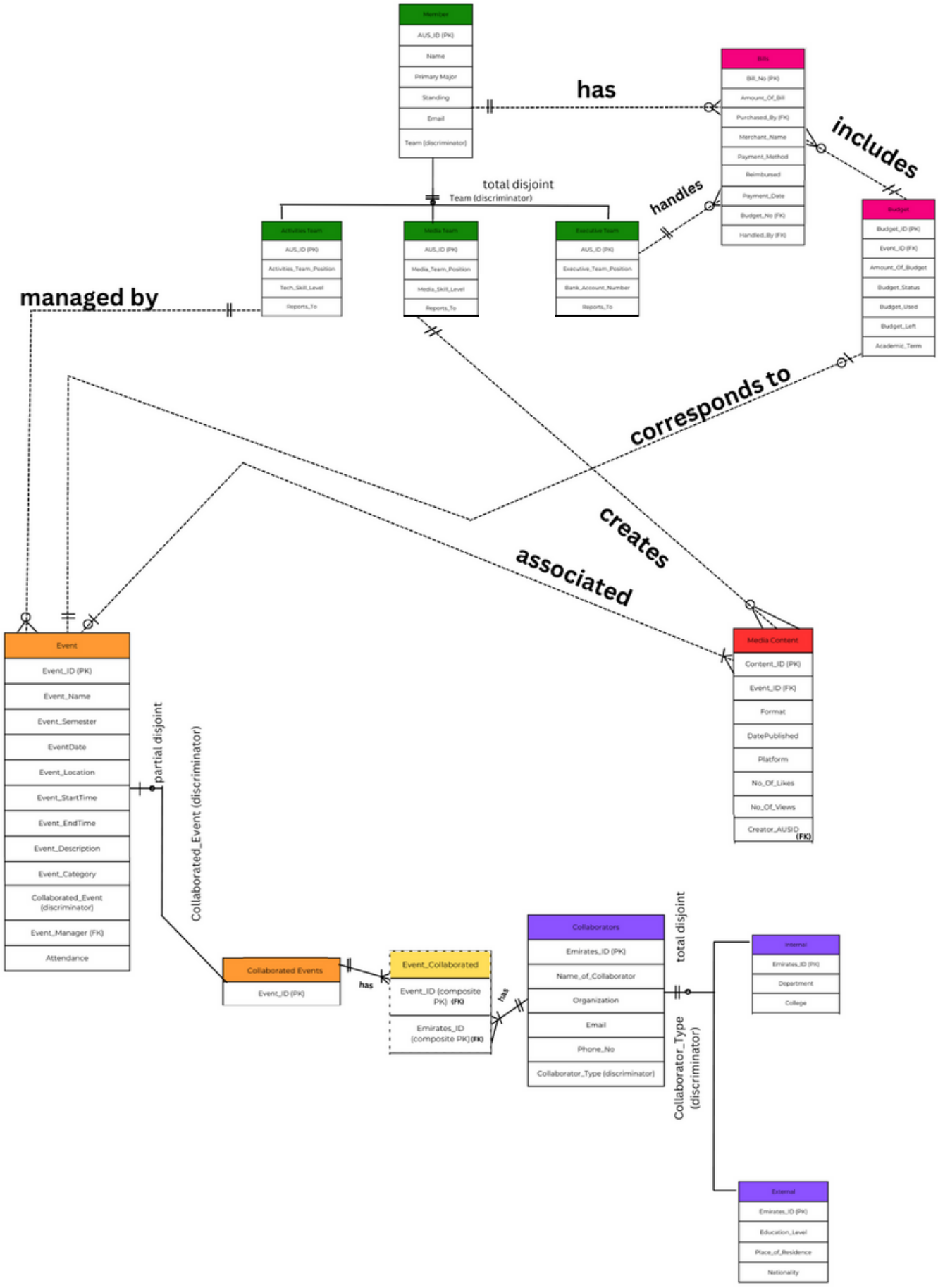
13) A Budget has zero to many Bills (some events have a budget, but none of it is used, thereby generating zero bills), but a Bill has one and only one Budget.

14) An Event has zero to only one Budget (some events do not have budgets) and a Budget corresponds to one and only one Event.

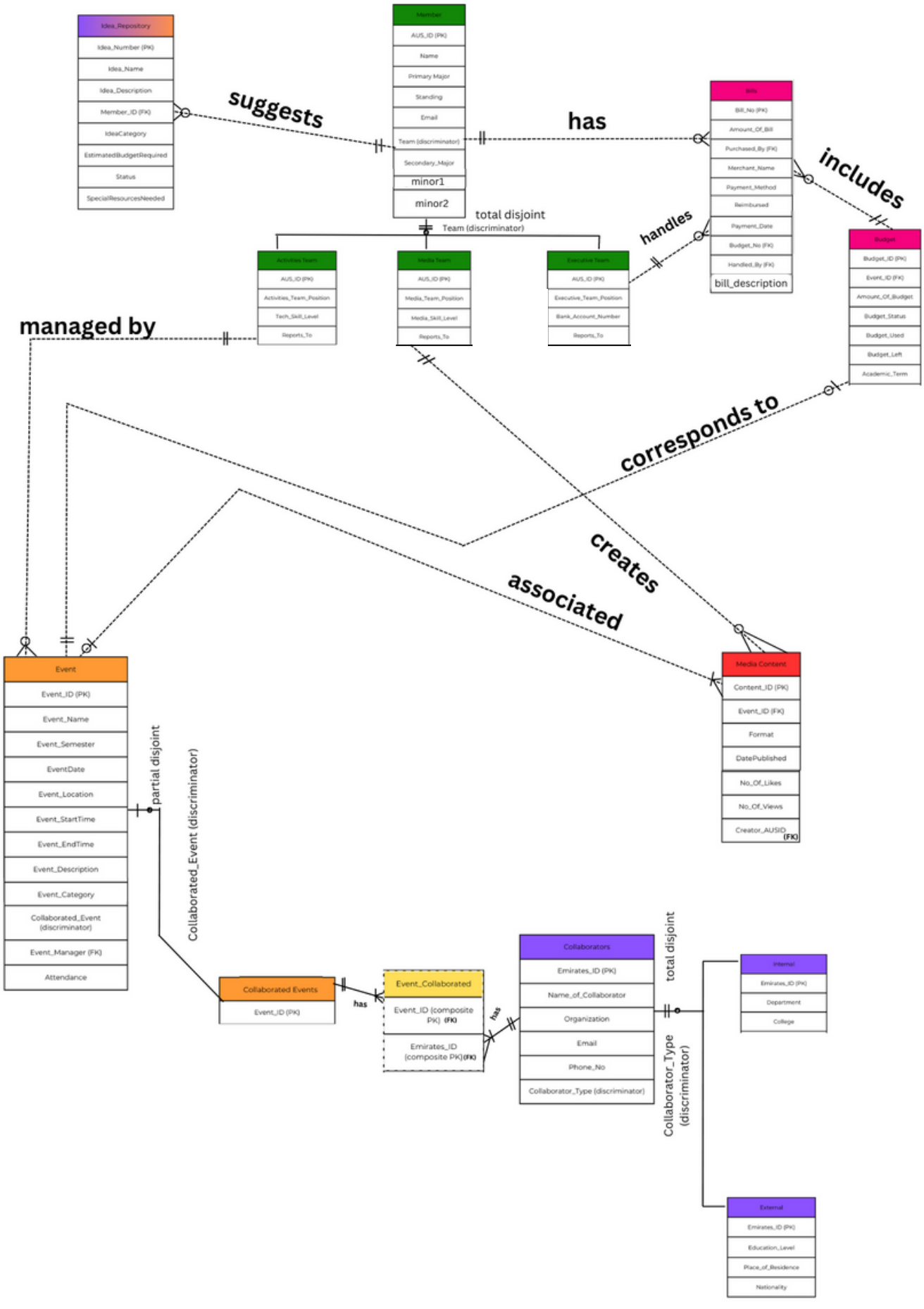
15) A Bill is paid for by one and only one Member, but a Member can pay for zero to many Bills (some members do not pay for any bills).

EERD

BEFORE (without new table, added and dropped columns)



AFTER (with the new table, added and dropped columns)



EERD and BUSINESS RULES

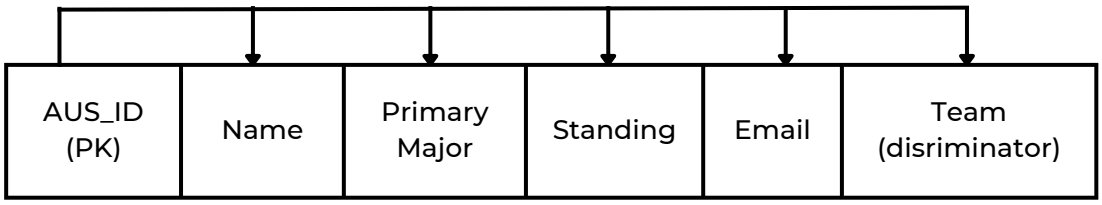
Our database contains the following entities (15 entities) with the following attributes :

1) Member

AUS_ID (PK)	the AUS_ID of the Team Member
Name	the name of the Team Member
Primary Major	the Primary Major of the Team Member
Standing	the standing of the Team Member
Email	the email of the team member, should be a unique value
Team (discriminator)	the team that the team member belongs to (ie. activities team, media team, or executive team)

The Member table is currently in 3rd normal form as we have no repeating groups (1NF), no partial dependencies (2NF), and no transitive dependencies (3NF). The PK (AUS_ID) determines all the other attributes of the table.

Email is derived from AUS_ID, however as AUS_ID is a PK, this does not lead to a transitive dependency.
An AUS student can have 2 majors and this can cause the problem of repeating groups. However, OSC only stores the primary major of the student, thereby avoiding this problem.



EERD and BUSINESS RULES

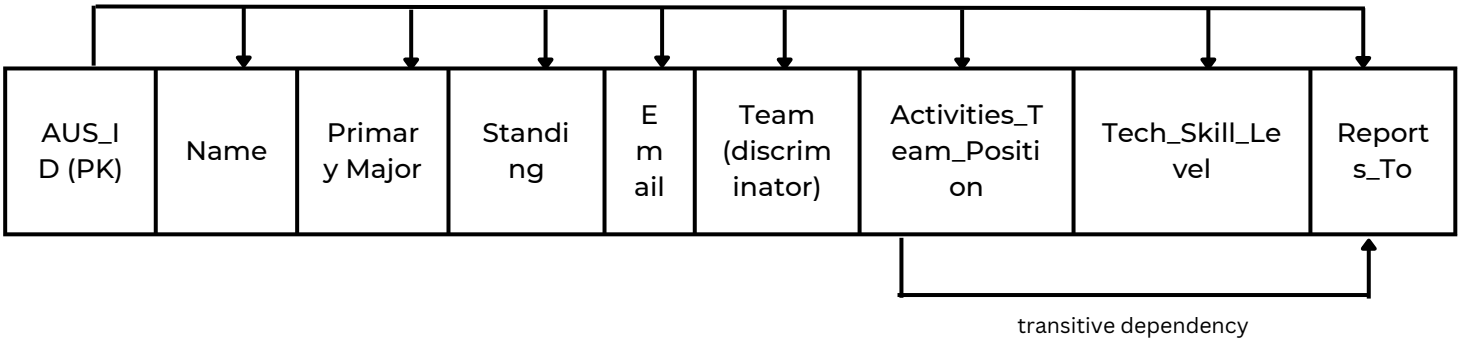
Our database contains the following entities with the following attributes :

2) Activities Team

AUS_ID (PK)	the AUS_ID of the Team Member
Name	the name of the Team Member
Primary Major	the Primary Major of the Team Member
Standing	the standing of the Team Member
Email	the email of the team member, should be a unique value
Team (discriminator)	the team that the team member belongs to (ie. activities team, media team, or executive team)
Activities_Team_Position	the position of the team member in the activities team
Tech_Skill_Level	the tech skill level of the member in the activities team (scale of 1 (lowest) -10 (highest))
Reports_To	who the team member reports to (ie. Head_Of_Activities or President)

The Activities Team table is currently in 2nd normal form as we have no repeating groups (1NF) and no partial dependencies (2NF). However, we have a transitive dependency as Activities_Team_Position alone can determine Reports_To. However, we have decided to keep it in 2NF and not convert it further to 3NF as removing the transitive dependency and creating new tables will be extremely time-consuming and need a lot of data storage space. The implementation of another table will also increase the monetary costs for OSC, which is unideal for a student organization.

The transitive dependency arises because Activities_Team_Position can determine Reports_To. Head_Of_Activities, Research_Coordinator, and Public_Relations_Coordinator will report to President but Activities_Coordinator will always report to Head_Of_Activities.



EERD and BUSINESS RULES

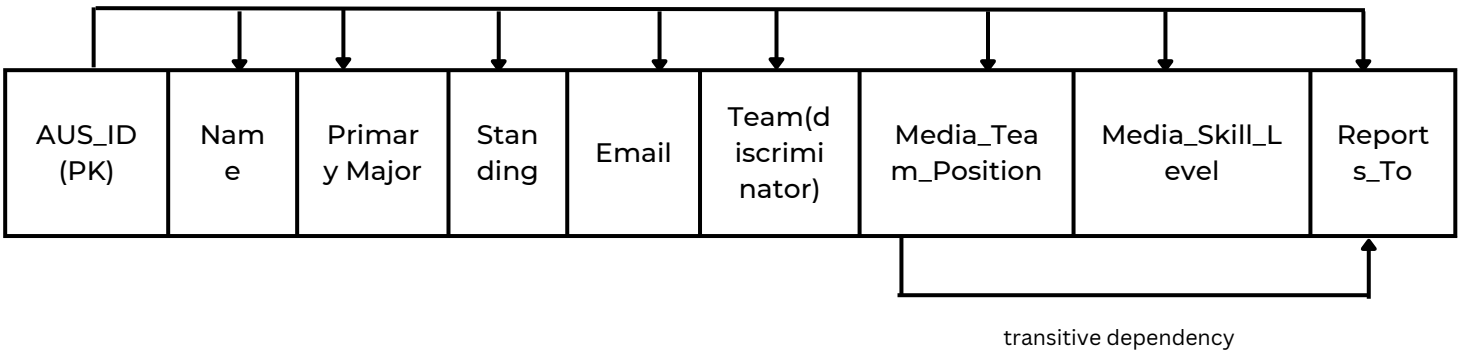
Our database contains the following entities with the following attributes :

3) Media Team

AUS_ID (PK)	the AUS_ID of the Team Member
Name	the name of the Team Member
Primary Major	the Primary Major of the Team Member
Standing	the standing of the Team Member
Email	the email of the team member, should be a unique value
Team (discriminator)	the team that the team member belongs to (ie. activities team, media team, or executive team)
Media_Team_Position	the position of the team member in the media team
Media_Skill_Level	the media skill level of the media team member (scale of 1 (lowest) -10 (highest))
Reports_To	who the media team member reports to (Head_Of_Media or Presdient)

The Media Team table is currently in 2nd normal form as we have no repeating groups (1NF) and no partial dependencies (2NF). However, we have a transitive dependency as Media_Team_Position alone can determine Reports_To. However, we have decided to keep it in 2NF and not convert it further to 3NF as removing the transitive dependency and creating new tables will be extremely time-consuming and need a lot of data storage space. The implementation of another table will also increase the monetary costs for OSC, which is unideal for a student organization.

The transitive dependency arises because Media_Team_Position can determine Reports_To. Head_Of_Media will report to President but Media_Officer will always report to Head_Of_Media.



EERD and BUSINESS RULES

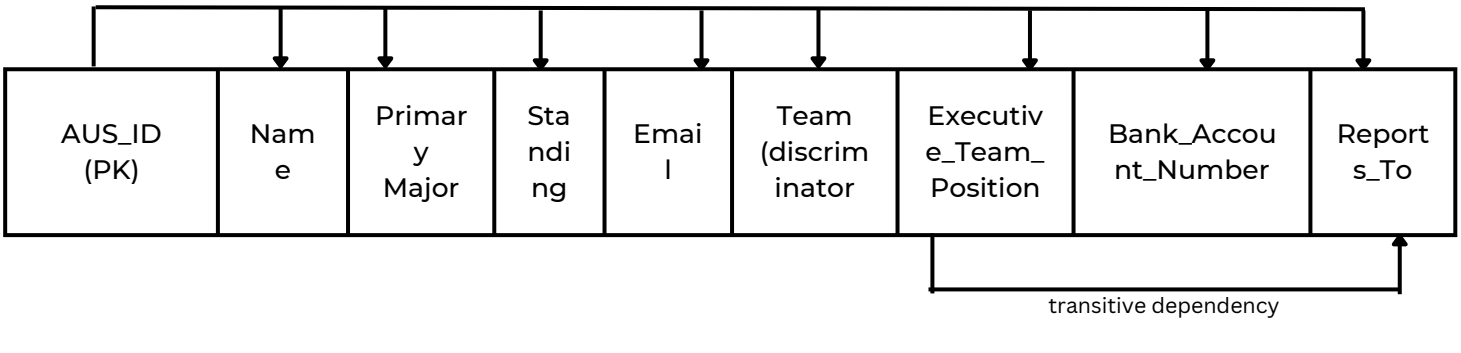
Our database contains the following entities with the following attributes :

4) Executive Team

AUS_ID (PK)	the AUS_ID of the Team Member
Name	the name of the Team Member
Primary Major	the Primary Major of the Team Member
Standing	the standing of the Team Member
Email	the email of the team member, should be a unique value
Team (discriminator)	the team that the team member belongs to (ie. activities team, media team, or executive team)
Executive_Team_Position	the position of the executive team member in the executive team
Bank_Account_Number	the bank account number of the executive team member (can be NULL as we do not have to store the bank account number of any one other than the President and Treasurer)
Reports_To	the position of whom the executive team member reports to (NULL for president a he does not report to anyone (internal to the club)

The Executive Team table is currently in 2nd normal form as we have no repeating groups (1NF) and no partial dependencies (2NF). However, we have a transitive dependency as Executive_Team_Position alone can determine Reports_To. However, we have decided to keep it in 2NF and not convert it further to 3NF as removing the transitive dependency and creating new tables will be extremely time-consuming and need a lot of data storage space. The implementation of another table will also increase the monetary costs for OSC, which is unideal for a student organization.

The transitive dependency arises because Executive_Team_Position can determine Reports_To. Vice-President, Executive_Assistant, and Treasurer will report to President, and President does not report to anyone (internal to the club).



EERD and BUSINESS RULES

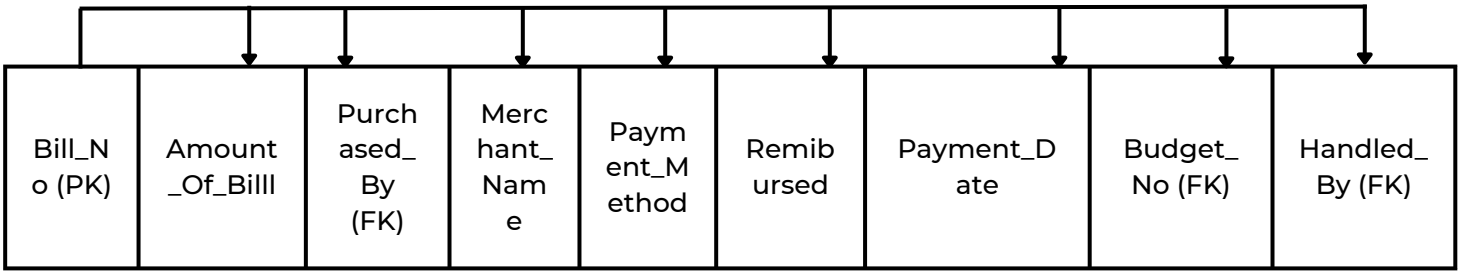
Our database contains the following entities with the following attributes :

5) Bills

Bill_No (PK)	the bill number (created by OSC, as xxx-yyy-zzz-NUM, where xxx: Event, yyy: last 3 digits of budgetNo, zzz: last 3 digits of ID of purchaser, NUM: serial number of the bill. for eg: CTF-351-833-001)
Amount_Of_Bill	the amount of the bill (ie. AED 75.5)
Purchased_By (FK)	the AUS_ID number of the team member who purchased (paid for) the bill (can be from any team)
Merchant_Name	the merchant whom the payment was made to (Amazon, Talabat)
Payment_Method	if the payment was made in cash or card
Reimbursed	if they payment was reimbursed to the team member ('Y' for reimbursed, 'N' for not reimbursed or pending)
Payment_Date	when they payment was made (this is important to help track the payments)
Budget_No (FK)	helps to find out which budget the bill belongs to (one budget can have many bills, one bill belongs to one budget)
Handled_By (FK)	the AUS_ID of the current president or treasurer (according to if the bill is handled by the treasurer or president - whomever is handling the bill has the responsibility of reimbursing it)

(

The Bills table is currently in 3rd normal form as we have no repeating groups (1NF), no partial dependencies (2NF), and no transitive dependencies (3NF). The PK (Bill_No) determines all the other attributes of the table.



EERD and BUSINESS RULES

Our database contains the following entities with the following attributes :

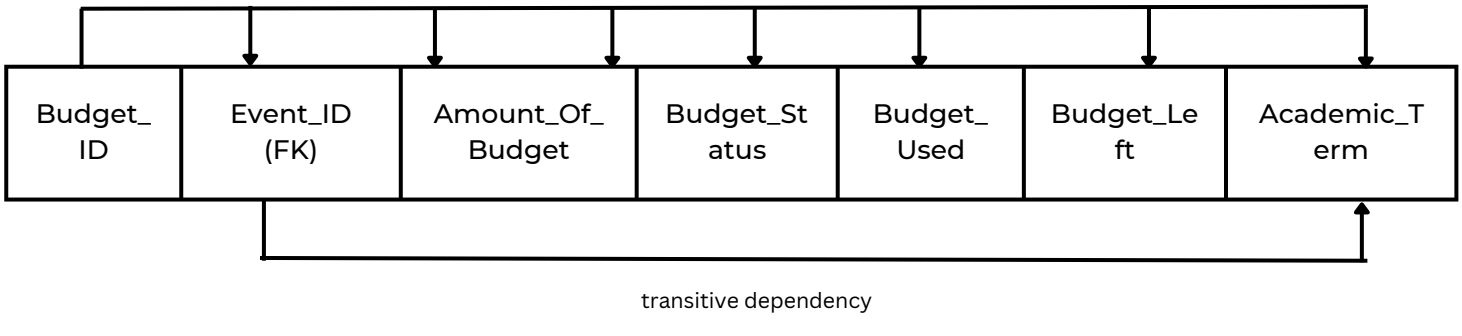
6) Budget

Budget_ID (PK)	**the ID of the budget (we use the same budget number from engage in our database, to help improve accessibility)**
Event_ID (FK)	the Event_ID of which the budget corresponds to (xxxTTT, where xxx - 3 letter acronym for the event, TTT - term, for eg, CTFS23) (a budget can only correspond to one event)
Amount_Of_Budget	the total amount of budget allocated for a specific event
Budget_Status	if the budget is approved, rejected or pending from Office of Student Affairs
Budget_Used	how much of the allocated budget has been used up till now (summation of the bills corresponding to the event)
Budget_Left	how much budget is left for a specific event (Amount_Of_Budget - Budget_Left)
Academic_Term	the academic term the budget corresponds to (ie. Spring 2023 or Fall 2023)

The Budget table is currently in 2nd normal form as we have no repeating groups (1NF) and no partial dependencies (2NF). The PK (Budget_ID) determines all the other attributes of the table.

However, we have a transitive dependency as Event_ID alone can determine Academic_Term. However, we have decided to keep it in 2NF and not convert it further to 3NF as removing the transitive dependency and creating new tables will be extremely time-consuming and need a lot of data storage space. The implementation of another table will also increase the monetary costs for OSC, which is unideal for a student organization.

The transitive dependency arises because the last 3 characters of Event_ID such as F20 or S21 can determine Academic_Term such as Fall 2020 or Spring 2021.



EERD and BUSINESS RULES

Our database contains the following entities with the following attributes :

7) Events

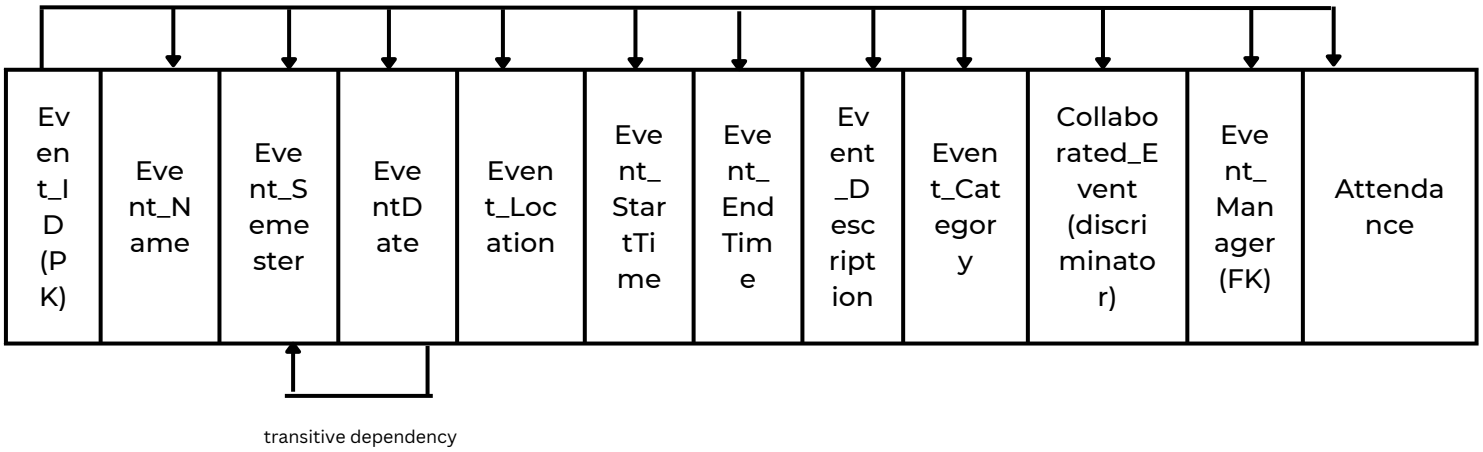
Event_ID (PK)	the Event_ID for the event (xxxTTT, where xxx - 3 letter acronym for the event, TTT - term, for eg, CTFS23)
Event_Name	the name of the Event
Event_Semester	the semester of the Event
EventDate	the date of the event (if an event is hosted over 2 days, we keep the value as NULL)
Event_Location	the location of the event (can be zoom or google meets as well)
Event_StartTime	the start time of an event
Event_EndTime	the end time of an event
Event_Description	short description of the event
Event_Category	if the event was educational, recreational or informational
Collaborated_Event (discriminator)	if the event had a collaborator ('Y' for yes, 'N' for no)
Event_Manager (FK)	the AUS_ID of the activities team member who is managng the event (only one event_manager per event)
Attendance	the attendance of the event (number of participants)

The Events table is currently in 2nd normal form as we have no repeating groups (1NF) and no partial dependencies (2NF) The PK (Event_ID) determines all the other attributes of the table.

However, we have a transitive dependency as EventDate alone can determine Event_Semester. However, we have decided to keep it in 2NF and not convert it further to 3NF as removing the transitive dependency and creating new tables will be extremely time-consuming and need a lot of data storage space. The implementation of another table will also increase the monetary costs for OSC, which is unideal for a student organization.
The transitive dependency arises because EventDate can determine Event_Semester such as Fall 2020 or Spring 2021.

For the Budget table, we argued that Event_ID can determine Event_Semester, but since Event_ID is the PK, no transitive dependency arises.

EERD and BUSINESS RULES



EERD and BUSINESS RULES

Our database contains the following entities with the following attributes :

8) Collaborated Events

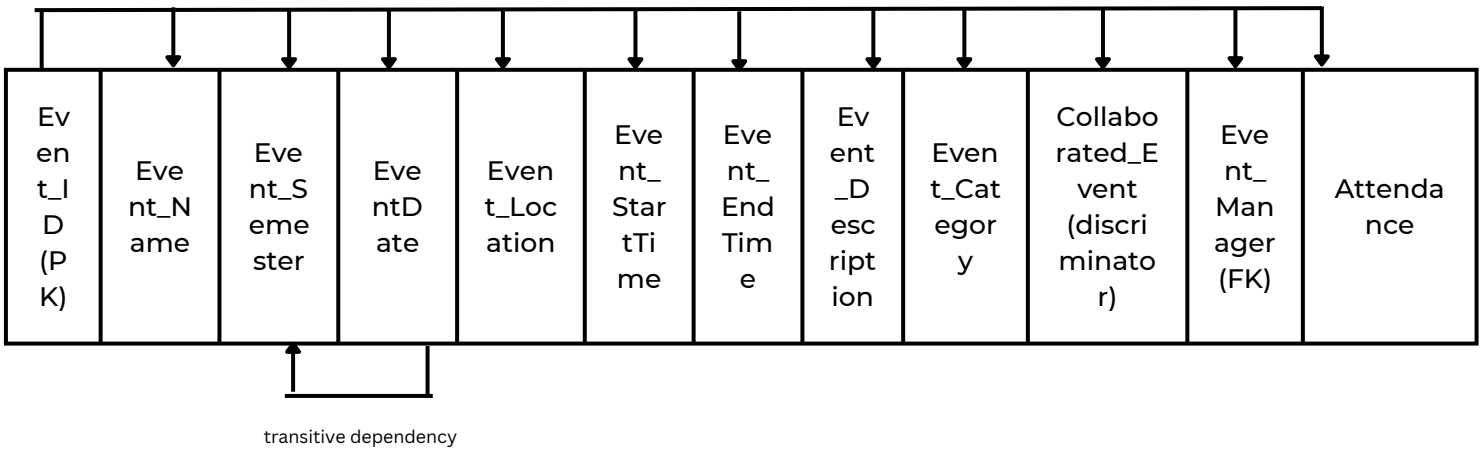
Event_ID (PK)	the Event_ID for the event (xxxTTT, where xxx - 3 letter acronym for the event, TTT - term, for eg, CTFS23)
Event_Name	the name of the Event
Event_Semester	the semester of the Event
EventDate	the date of the event (if an event is hosted over 2 days, we keep the value as NULL)
Event_Location	the location of the event (can be zoom or google meets as well)
Event_StartTime	the start time of an event
Event_EndTime	the end time of an event
Event_Description	short description of the event
Event_Category	if the event was educational, recreational or informational
Collaborated_Event (discriminator)	if the event had a collaborator ('Y' for yes, 'N' for no) (in this case, all would be 'Y' as this table stores all collaborated events)
Event_Manager (FK)	the AUS_ID of the activities team member who is managng the event (only one event_manager per event)
Attendance	the attendance of the event (number of participants)

The Collaborated Events is currently in 2nd normal form as we have no repeating groups (1NF) and no partial dependencies (2NF) The PK (Event_ID) determines all the other attributes of the table.

However, we have a transitive dependency as EventDate alone can determine Event_Semester. However, we have decided to keep it in 2NF and not convert it further to 3NF as removing the transitive dependency and creating new tables will be extremely time-consuming and need a lot of data storage space. The implementation of another table will also increase the monetary costs for OSC, which is unideal for a student organization.

For the Budget table, we argued that Event_ID can determine Event_Semester, but since Event_ID is the PK, no transitive dependency arises.

EERD and BUSINESS RULES



EERD and BUSINESS RULES

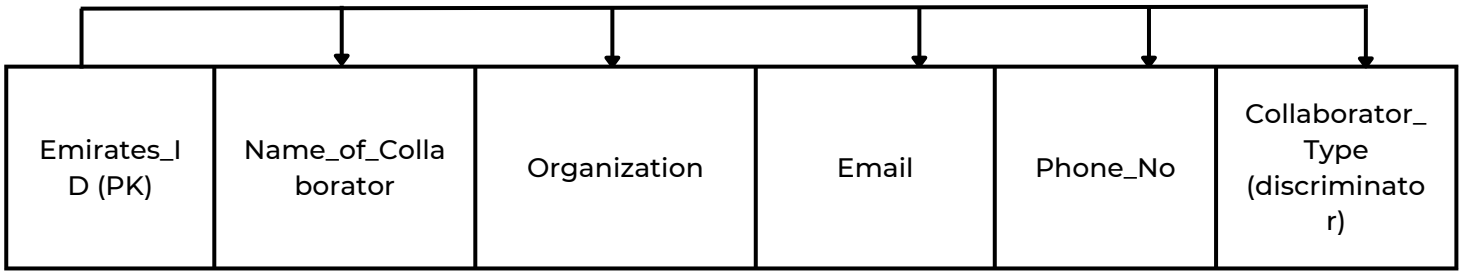
Our database contains the following entities with the following attributes :

9) Collaborators

Emirates_ID (PK)	the Emirates_ID of the collaborator (if the collaborator is a specific organization, the POC's details would be used)
Name_of_Collaborator	the name of the collaborator
Organization	the organization of the collaborator
Email	the email ID of the collaborator (should be unique)
Phone_No	the phone number of the collaborator (should be unique)
Collaborator_Type (discriminator)	if the collaborator is internal or external (internal - belongs to AUS / external - does not belong to AUS)

The Collaborators table is currently in 3rd normal form as we have no repeating groups (1NF), no partial dependencies (2NF), and no transitive dependencies (3NF). The PK (Emirates_ID) determines all the other attributes of the table.

For this table, Email does not determine Organization as some Collaborators may use email web domains of Gmail and Yahoo.



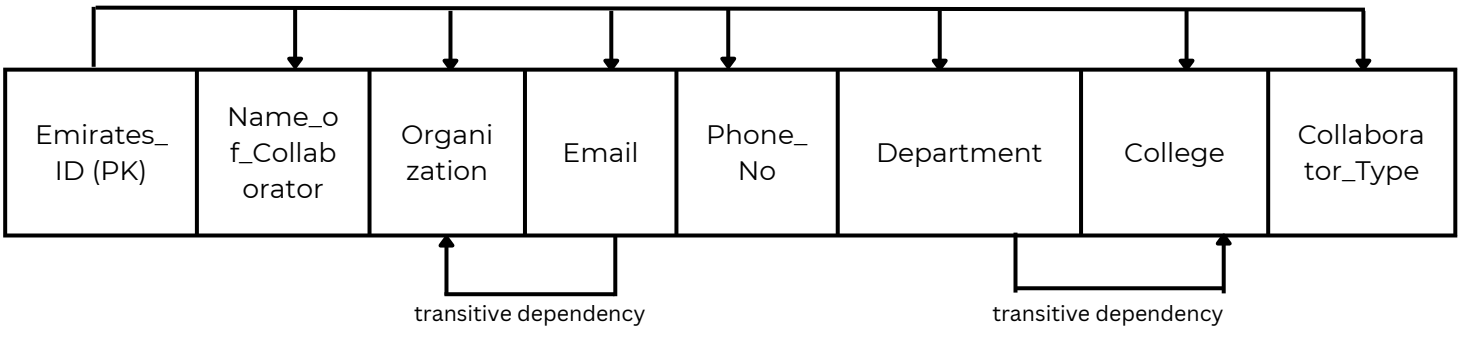
EERD and BUSINESS RULES

Our database contains the following entities with the following attributes :

10) Internal

Emirates_ID (PK)	the Emirates_ID of the collaborator (if the collaborator is a specific organization, the POC's details would be used)
Name_of_Collaborator	the name of the collaborator
Organization	the organization of the collaborator
Email	the email ID of the collaborator (should be unique)
Phone_No	the phone number of the collaborator (should be unique)
Collaborator_Type	if the collaborator is internal or external (internal - belongs to AUS / external - does not belong to AUS), in this table, only internal collaborators are stored, so the value would be internal
Department	the department the collaborator belongs to / falls under
College	the college the collaborator belongs to / falls under

The Internal table is currently in 2nd normal form as we have no repeating groups (1NF) and no partial dependencies (2NF). However, we have transitive dependencies as Email (a non-prime attribute) can determine Organization, and Department (a non-prime attribute) can determine College. However, we have decided to keep it in 2NF and not convert it further to 3NF as removing the transitive dependency and creating new tables will be extremely time-consuming and need a lot of data storage space. The implementation of another table will also increase the monetary costs for OSC, which is unideal for a student organization.
The transitive dependency arises because Email can determine Organization as Internal collaborators use the aus.edu email domain signifying AUS as their organization. Another transitive dependency arises as Department can determine the College.



EERD and BUSINESS RULES

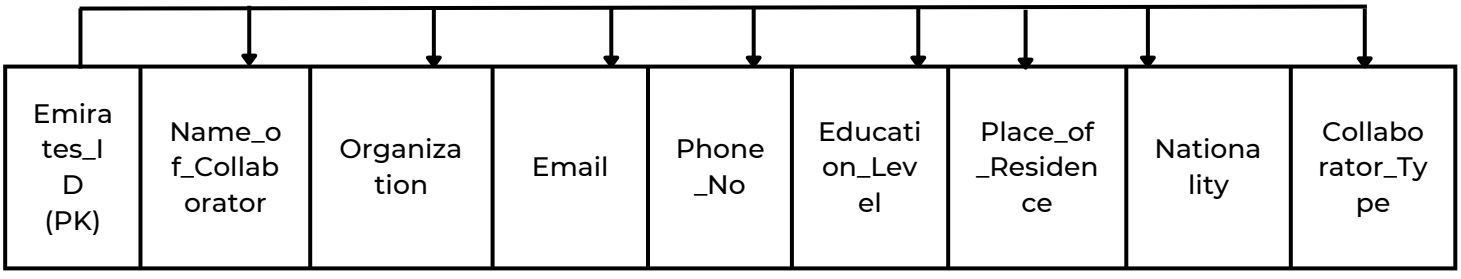
Our database contains the following entities with the following attributes :

11) External

Emirates_ID (PK)	the Emirates_ID of the collaborator (if the collaborator is a specific organization, the POC's details would be used)
Name_of_Collaborator	the name of the collaborator
Organization	the organization of the collaborator
Email	the email ID of the collaborator (should be unique)
Phone_No	the phone number of the collaborator (should be unique)
Collaborator_Type	if the collaborator is internal or external (internal - belongs to AUS / external - does not belong to AUS), in this table, only external collaborators are stored, so the value would be external
Education_Level	Office of Student Affairs requires us to store the education level of external collaborators
Place_of_Residence	Office of Student Affairs requires us to store the place of residence of external collaborators
Nationality	Office of Student Affairs requires us to store the nationality of external collaborators

The External table is currently in 3rd normal form as we have no repeating groups (1NF), no partial dependencies (2NF), and no transitive dependencies (3NF). The PK (Emirates_ID) determines all the other attributes of the table.

Email does not determine the organization, as many external organizations use web domains of Gmail and Yahoo.



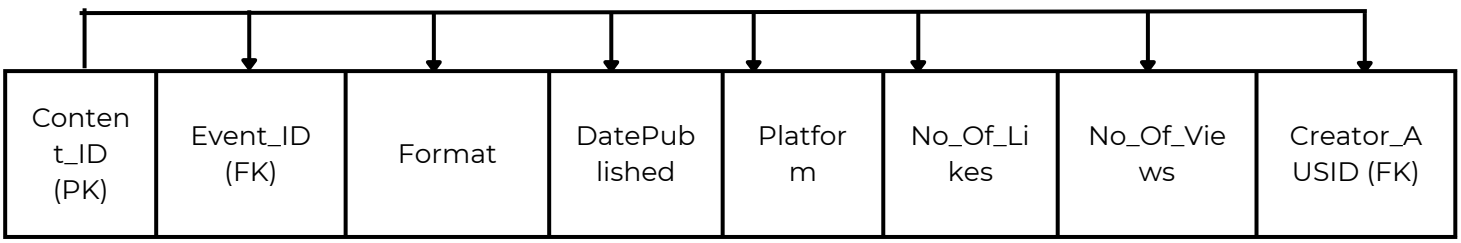
EERD and BUSINESS RULES

Our database contains the following entities with the following attributes :

12) Media Content

Content_ID (PK)	**the content_ID of the media content (xxx-TTT-yy - xxx - acronym for the media post, TTT - term, yy - number, for eg: TT-S23-01)**
Event_ID (FK)	the event_ID of which the media content was created for (some media content can exist independant from an event)
Format	the format of the media content (post, video or reel)
DatePublished	the date the media content was posted/published
Platform	the platform the media content was posted on
No_Of_Likes	the number of likes the media content received
No_Of_Views	the number of likes the media content received
Creator_AUSID (FK)	the AUS_ID of the media team member who created the media content

The Media Content table is currently in 3rd normal form as we have no repeating groups (1NF), no partial dependencies (2NF), and no transitive dependencies (3NF). The PK (Content_ID) determines all the other attributes of the table.



EERD and BUSINESS RULES

Our database contains the following entities with the following attributes :

13) Event_Collaborated (associate entity between Collaborators and Collaborated Events)

Event_ID (composite PK) (FK)	the Event_ID from the collaborated events table
Emirates_ID (composite PK) (FK)	the Emirates_ID from the collaborators table

The Event_Collaborated table is currently in 3rd normal form as we have no repeating groups (1NF), no partial dependencies (2NF), and no transitive dependencies (3NF).

PART 2
DESIGN

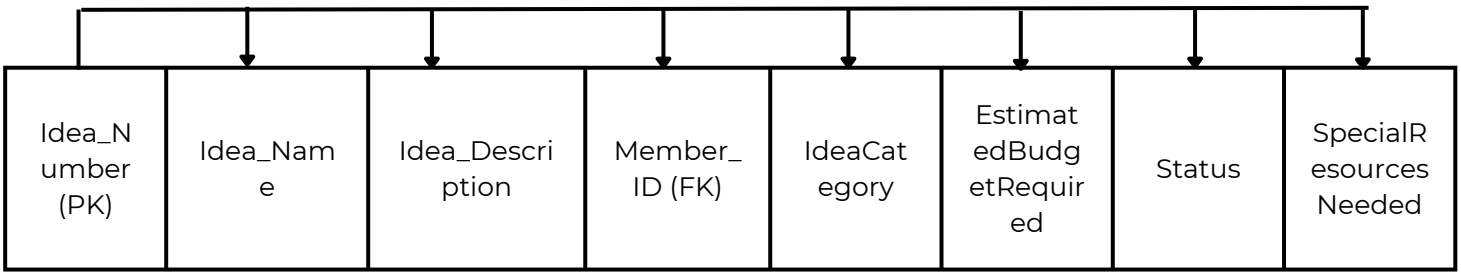
INSERTING AN ADDITIONAL TABLE - IDEA_REPOSITORY

We have decided to add a table called Idea_Repository to provide OSC with the opportunity to store event and media ideas from members. This Idea_Repository would have attributes as:

Idea_Number (PK)	the unique idea number associated with each and every idea
Idea_Name	the name of the idea (ie. the name of the event / media content)
Idea_Description	a brief description of the idea
Member_ID (FK)	the AUS_ID of the member who suggested the idea (an idea is suggested by only one member)
IdeaCategory	if the idea is an event idea or a media content idea
EstimatedBudgetRequired	the estimated budget required to implement idea (can be NULL as some ideas do not require budget)
Status	the status of the idea (can be 'Completed', 'To be Executed', 'Unapproved' or 'Proposed'
SpecialResourcesNeeded	if any special resources are needed for the event (such as projector, customized taboo cards)

The Idea_Repository table is currently in the 3rd normal form as we have no repeating groups (1NF), no partial dependencies (2NF), and no transitive dependencies (3NF). The PK (IdeaNumber) determines all the other attributes of the table.

We have connected Idea_Repository to Members, with the FK as Member_ID in the Idea_Repository table. The business rule is that a member can suggest 0 to many ideas, and an idea can be suggested by one and only one member.

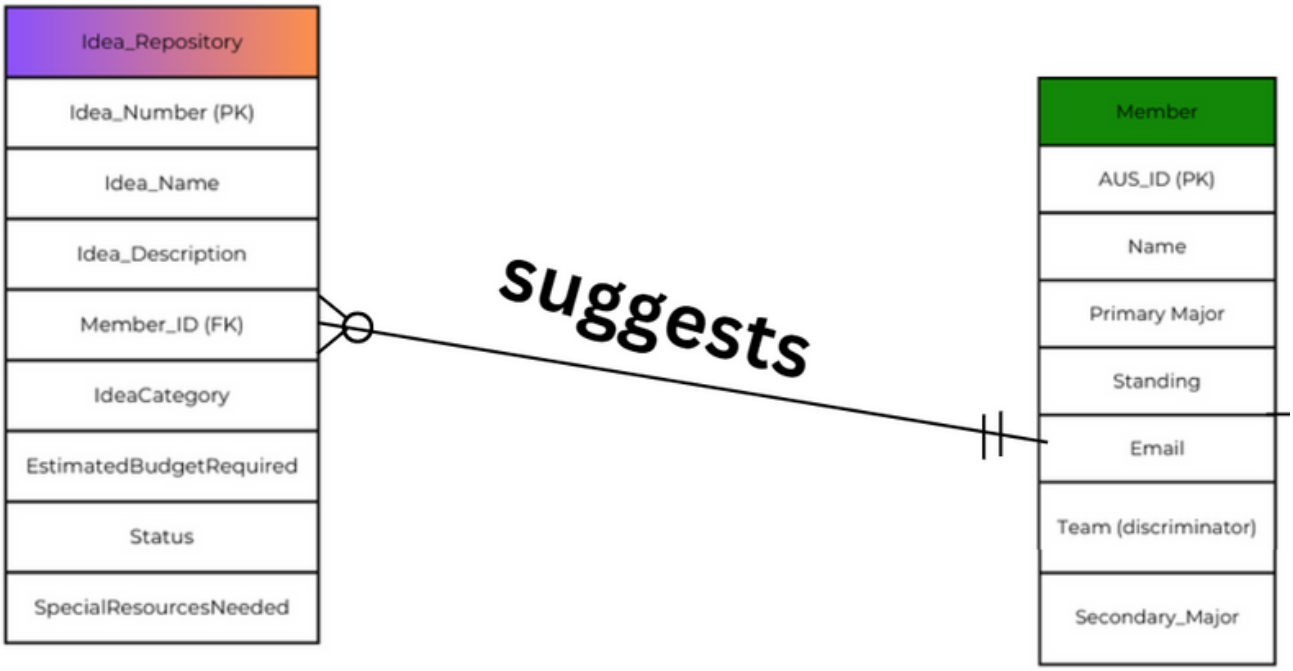


INSERTING AN ADDITIONAL TABLE - IDEA_REPOSITORY

```
Create table Idea_Repository(
 Idea_Number VARCHAR(9) Primary Key,
 Idea_Name VARCHAR(50) NOT NULL,
 Idea_Description VARCHAR(100) NOT NULL,
 Member_ID VARCHAR(9) NOT NULL,
 IdeaCategory VARCHAR(15) NOT NULL,
 Estimated_Budget_Required NUMBER(6,2),
 Status VARCHAR(30) NOT NULL CHECK (Status='Completed' or Status='To be Executed' or
Status= 'Unapproved' or Status = 'Proposed'),
 SpecialResoucesNeeded VARCHAR (50)
 );

 Alter table Idea_Repository
Add Foreign Key (Member_ID) References Member (AUS_ID);

insert into Idea_Repository
values('001', 'OSCMovieNight2.0', 'A movie night with OSC to watch the movie Snowden ',
'b00084833', 'EventIdea', '100', 'Proposed', 'projector' );

INSERT INTO Idea_Repository VALUES ('002', 'GuesstheOSSoftware', 'A Instagram Story
Series with OS software hints and polls to guess the OSS.', 'g00087337', 'MediaIdea', '50',
'Proposed', null);
INSERT INTO Idea_Repository VALUES ('003', 'MeetTheTeam', 'Day in the Life Instagram
story series with different members once a week.', 'g00087337', 'MediaIdea', '0', 'Proposed',
null);
INSERT INTO Idea_Repository VALUES ('004', 'TechnoBytes', 'Instagram Story series with
OSS related news. ', 'g00090589', 'MediaIdea', '0', 'Proposed', null);
INSERT INTO Idea_Repository VALUES ('005', 'OSCDataScienceSeries', 'An event series
about data science in different fields such as business, engineering, architecture.',
'g00089132', 'EventIdea', null, 'To be Executed', null);
INSERT INTO Idea_Repository VALUES ('006', 'WhoWantsToBeAMillionare?', 'An event
wherein we play the game Who Wants to Be A Millionaire with questions about OSS.',
'b00090070', 'EventIdea', null, 'To be Executed', null);
INSERT INTO Idea_Repository VALUES ('007', 'Taboo2.0', 'A part 2 for the Taboo event.',
'b00088385', 'EventIdea', '200', 'Proposed', 'customized OSC Taboo cards');
INSERT INTO Idea_Repository VALUES ('008', 'Aren''tWeOSome?', 'A Who''s Most Likely to
Instagram Story Series with OSC members.', 'g00089132', 'MediaIdea', '0', 'To be Executed',
'OSC Member pictures');
INSERT INTO Idea_Repository VALUES ('009', 'BoardGameNightwithOSC', 'A board game
night with OSC, playing games like chess, UNO and so on.', 'g00089132', 'EventIdea', '300',
'Proposed', 'board games');
INSERT INTO Idea_Repository VALUES ('010', '7Days7OSS', 'An Instagram Story Series where
we educate members about OSS, one a day for a week.', 'g00085775', 'MediaIdea', '0',
'Unapproved', null);
```

The reason we have decided to add this table is to help OSC solve the problem of being unable to store great event and media content ideas of the team members. Through this table, we can store the idea name, idea description, idea type, estimated budget required for the idea and special resources needed for the ideas.

INSERTING AN ADDITIONAL TABLE - IDEA_REPOSITORY



The newly created Idea_Repository table is connected to Member and has a many-to-one relationship. The business rule is that a member can suggest zero to many ideas, but one specific idea can only be suggested by one and only one member.

The reason we have connected the Idea_Repository table to Member is that team members have lots of unique ideas for new events and media content. We have not restricted the ideas to a specific team, as sometimes an Activities team member can have a Media_Content idea, and a Media team member can have an Event Idea.

MULTIVALUED ATTRIBUTE

We have 2 multivalued attributes in our Member table.

1) Major - A student at AUS is allowed to enroll in up to 2 majors (ie. double major). This results in the major being a multivalued attribute in the member table. Initially, OSC only stored the primary_major of the student. However, since we want to provide complete information about a student, we altered the member table to add an attribute called secondary_major.

To avoid the problem of repeating groups, we have kept the primary_major and secondary_major of each student in different columns. Not all students are required to have a double major, but should have at least one major, to be currently enrolled at AUS.

As Activities_Team, Media_Team, and Executive_Team are subtypes of the Member table, we would also be required to alter these tables to ensure they have the same variables as the Member table.

Because the Activities_Team, Media_Team, and Executive_Team have values from the Member table, the Secondary_Major of the students should be consistent throughout the tables.

```
ALTER TABLE MEMBER
ADD SECONDARY_MAJOR VARCHAR (35);

UPDATE MEMBER
SET SECONDARY_MAJOR = 'ECONOMICS'
WHERE AUS_ID = 'G00085775';

UPDATE MEMBER
SET SECONDARY_MAJOR = 'MATHEMATICS'
WHERE AUS_ID = 'B00089801';

UPDATE MEMBER
SET SECONDARY_MAJOR = 'MUSIC'
WHERE AUS_ID = 'B00093124';

ALTER TABLE MEDIA_TEAM
ADD SECONDARY_MAJOR VARCHAR (35);

ALTER TABLE EXECUTIVE_TEAM
ADD SECONDARY_MAJOR VARCHAR (35);

ALTER TABLE ACTIVITIES_TEAM
ADD SECONDARY_MAJOR VARCHAR (35);

UPDATE EXECUTIVE_TEAM
SET SECONDARY_MAJOR = 'ECONOMICS'
WHERE AUS_ID = 'G00085775';
```

MULTIVALUED ATTRIBUTE

```
UPDATE ACTIVITIES_TEAM
SET SECONDARY_MAJOR = 'MATHEMATICS'
WHERE AUS_ID = 'B00089801';

UPDATE MEDIA_TEAM
SET SECONDARY_MAJOR = 'MUSIC'
WHERE AUS_ID = 'B00093124';
```

2) Minor - A student is allowed to enroll in up to 2 minors (ie. double minor) at AUS. This would result in the minor being a multivalued attribute in the member table. Initially, OSC did not store the minors of the student. However, since we want to provide complete information about a student, we altered the member table to add attributes called minor1 and minor2. Students are not required to minor.

To avoid the problem of repeating groups, we have kept the minor1 and minor2 of each student in different columns.

As Activities_Team, Media_Team, and Executive_Team are subtypes of the Member table, we would also be required to alter these tables to ensure they have the same variables as the Member table.

Because the Activities_Team, Media_Team, and Executive_Team have values from the Member table, the minor1 and minor2 attributes of the students should be consistent throughout the tables.

```
ALTER TABLE MEMBER
ADD MINOR1 VARCHAR (35);

ALTER TABLE MEMBER
ADD MINOR2 VARCHAR (35);

ALTER TABLE MEDIA_TEAM
ADD MINOR1 VARCHAR (35);

ALTER TABLE MEDIA_TEAM
ADD MINOR2 VARCHAR (35);

ALTER TABLE EXECUTIVE_TEAM
ADD MINOR1 VARCHAR (35);

ALTER TABLE EXECUTIVE_TEAM
ADD MINOR2 VARCHAR (35);

ALTER TABLE ACTIVITIES_TEAM
ADD MINOR1 VARCHAR (35);

ALTER TABLE ACTIVITIES_TEAM
ADD MINOR2 VARCHAR (35);
```

MULTIVALUED ATTRIBUTE

```
UPDATE MEMBER
SET MINOR1 = 'ECONOMICS'
WHERE AUS_ID = 'G00089132';

UPDATE MEMBER
SET MINOR1 = 'DATA SCIENCE'
WHERE AUS_ID = 'B00090070';

UPDATE MEMBER
SET MINOR1 = 'DATA SCIENCE'
WHERE AUS_ID = 'G00085775';

UPDATE MEMBER
SET MINOR2 = 'MUSIC'
WHERE AUS_ID = 'G00085775';

UPDATE MEMBER
SET MINOR1 = 'DATA SCIENCE'
WHERE AUS_ID = 'B00084833';

UPDATE MEMBER
SET MINOR2 = 'APPLIED COMPUTATIONAL MATHEMATICS'
WHERE AUS_ID = 'B00084833';

UPDATE ACTIVITIES_TEAM
SET MINOR1 = 'ECONOMICS'
WHERE AUS_ID = 'G00089132';

UPDATE ACTIVITIES_TEAM
SET MINOR1 = 'DATA SCIENCE'
WHERE AUS_ID = 'B00090070';

UPDATE EXECUTIVE_TEAM
SET MINOR1 = 'DATA SCIENCE'
WHERE AUS_ID = 'G00085775';

UPDATE EXECUTIVE_TEAM
SET MINOR2 = 'MUSIC'
WHERE AUS_ID = 'G00085775';

UPDATE EXECUTIVE_TEAM
SET MINOR1 = 'DATA SCIENCE'
WHERE AUS_ID = 'B00084833';

UPDATE EXECUTIVE_TEAM
SET MINOR2 = 'APPLIED COMPUTATIONAL MATHEMATICS'
WHERE AUS_ID = 'B00084833';
```

MULTIVALUED ATTRIBUTE

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM
1	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team
5	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team
6	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team
7	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team
8	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team
9	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team
10	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team
11	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team
12	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team
13	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team
14	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team
15	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team
16	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team
17	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team
18	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team
19	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team
20	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team
21	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team
22	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team
23	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team
24	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team
25	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team

Member Table - Before

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	EXECUTIVE_TEAM_POSITION	BANK_ACCOUNT_NUMBER	REPORTS_TO
1	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team	President	AE867280963400758960283	(null)
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team	Vice-President	(null)	President
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team	Treasurer	AE636789189562026613410	President
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team	Executive Assistant	(null)	President

Executive_Team Table - Before

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	ACTIVITIES_TEAM_POSITION	TECH_SKILL_LEVEL	REPORTS_TO
1	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team	Head Of Activities	7	President
2	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team	Public Relations Coordinator	5	President
3	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team	Research Coordinator	6	President
4	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team	Activities Officer	2	Head Of Activities
5	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team	Activities Officer	10	Head Of Activities
6	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team	Activities Officer	8	Head Of Activities
7	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team	Activities Officer	5	Head Of Activities
8	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team	Activities Officer	9	Head Of Activities
9	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team	Activities Officer	10	Head Of Activities
10	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team	Activities Officer	3	Head Of Activities

Activities_Team Table - Before

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	MEDIA_TEAM_POSITION	MEDIA_SKILL_LEVEL	REPORTS_TO
1	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team	Head of Media	10	President
2	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team	Media Coordinator	7	Head of Media
3	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team	Media Coordinator	8	Head of Media
4	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team	Media Coordinator	7	Head of Media
5	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team	Media Coordinator	10	Head of Media
6	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team	Media Coordinator	6	Head of Media
7	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team	Media Coordinator	7	Head of Media
8	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team	Media Coordinator	9	Head of Media
9	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team	Media Coordinator	6	Head of Media
10	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team	Media Coordinator	7	Head of Media
11	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team	Media Coordinator	2	Head of Media

Media_Team Table - Before

MULTIVALUED ATTRIBUTE

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	SECONDARY_MAJOR	MINOR1	MINOR2
1	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team	(null)	Data Science	Applied Computational Mathematics
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team	(null)	(null)	(null)
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team	Economics	Data Science	Music
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team	(null)	(null)	(null)
5	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team	(null)	Data Science	(null)
6	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team	(null)	Economics	(null)
7	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team	(null)	(null)	(null)
8	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team	(null)	(null)	(null)
9	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team	(null)	(null)	(null)
10	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team	(null)	(null)	(null)
11	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team	(null)	(null)	(null)
12	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team	(null)	(null)	(null)
13	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team	Mathematics	(null)	(null)
14	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team	(null)	(null)	(null)
15	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team	(null)	(null)	(null)
16	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team	Music	(null)	(null)
17	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team	(null)	(null)	(null)
18	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team	(null)	(null)	(null)
19	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team	(null)	(null)	(null)
20	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team	(null)	(null)	(null)
21	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team	(null)	(null)	(null)
22	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team	(null)	(null)	(null)
23	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team	(null)	(null)	(null)
24	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team	(null)	(null)	(null)
25	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team	(null)	(null)	(null)

Member Table - After

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	EXECUTIVE_TEAM_POSITION	BANK_ACCOUNT_NUMBER	REPORTS_TO	SECONDARY_MAJOR	MINOR1	MINOR2
1	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team	President	AE867280963400758960283	(null)	(null)	Data Science	Applied Computational Mathemat
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team	Vice-President	(null)	President	(null)	(null)	(null)
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team	Treasurer	AE636789189562026613410	President	Economics	Data Science	Music
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team	Executive Assistant	(null)	President	(null)	(null)	(null)

Executive_Team Table - After

	JS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	ACTIVITIES_TEAM_POSITION	TECH_SKILL_LEVEL	REPORTS_TO	SECONDARY_MAJOR	MINOR1	MINOR2
1	090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team	Head Of Activities	7	President	(null)	Data Science	(null)
2	089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team	Public Relations Coordinator	5	President	(null)	Economics	(null)
3	088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team	Research Coordinator	6	President	(null)	(null)	(null)
4	087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team	Activities Officer	2	Head Of Activities	(null)	(null)	(null)
5	092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team	Activities Officer	10	Head Of Activities	(null)	(null)	(null)
6	087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team	Activities Officer	8	Head Of Activities	(null)	(null)	(null)
7	089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team	Activities Officer	5	Head Of Activities	(null)	(null)	(null)
8	087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team	Activities Officer	9	Head Of Activities	(null)	(null)	(null)
9	089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team	Activities Officer	10	Head Of Activities	Mathematics	(null)	(null)
10	093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team	Activities Officer	3	Head Of Activities	(null)	(null)	(null)

Activities_Team Table - After

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	MEDIA_TEAM_POSITION	MEDIA_SKILL_LEVEL	REPORTS_TO	SECONDARY_MAJOR	MINOR1	MINOR2
1	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team	Head of Media	10	President	(null)	(null)	(null)
2	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team	Media Coordinator	7	Head of Media	Music	(null)	(null)
3	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team	Media Coordinator	8	Head of Media	(null)	(null)	(null)
4	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team	Media Coordinator	7	Head of Media	(null)	(null)	(null)
5	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team	Media Coordinator	10	Head of Media	(null)	(null)	(null)
6	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team	Media Coordinator	6	Head of Media	(null)	(null)	(null)
7	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team	Media Coordinator	7	Head of Media	(null)	(null)	(null)
8	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team	Media Coordinator	9	Head of Media	(null)	(null)	(null)
9	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team	Media Coordinator	6	Head of Media	(null)	(null)	(null)
10	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team	Media Coordinator	7	Head of Media	(null)	(null)	(null)
11	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team	Media Coordinator	2	Head of Media	(null)	(null)	(null)

Media_Team Table - After

DERIVED ATTRIBUTE

Our budget table has an attribute called Budget_Left, which is equal to (Amount_of_Budget - Budget_Used). To calculate the budget left over for each event, we can subtract the total budget used per event, from the total allocated budget of the event.

Initially, we manually inputed the Budget_Left by calculating the difference between the Amount_of_Budget and Budget_Used. However, as this is a derived attribute, we can set a formula to automatically calculate (derive) Budget_Left using Amount_of_Budget and Budget_Used.

First, we updated the existing entries of Budget_Left, to set them to be calculated as Amount_of_Budget - Budget_Used. However, since this will only update the existing entries, we have further dropped the Budget_Left column and altered the budget table to create a new column called Budget_Left (a new column with the same name). This new column is a derived attribute, and we have set its value of it as (amount_of_budget - budget_used).

	BUDGET_ID	EVENT_ID	AMOUNT_OF_BUDGET	BUDGET_STATUS	BUDGET_USED	ACADEMIC_TERM	BUDGET_LEFT
1	859043	CFF20	100	Approved	97.5	Fall 2020	2.5
2	295710	PALOALTF20	150	Approved	150	Fall 2020	0
3	876502	MNF20	50	Approved	50	Fall 2020	0
4	140793	AMNGUSF20	100	Approved	100.75	Fall 2020	-0.75
5	572819	AWSF20	250	Approved	0	Fall 2020	250
6	493628	ONLSECS21	100	Approved	78.2	Spring 2021	21.8
7	926517	IOTS21	100	Approved	100	Spring 2021	0
8	731984	ONWOSCS21	100	Approved	100	Spring 2021	0
9	251768	CFF21	100	Approved	100	Fall 2021	0
10	504219	TABOOS22	200	Approved	221	Spring 2022	-21
11	821076	CTFS22	100	Approved	94.3	Spring 2022	5.7
12	165397	GITHUBS22	100	Approved	99.2	Spring 2022	0.8
13	708263	WEB3.0F22	100	Approved	75	Fall 2022	25
14	205150	CARNF22	600	Approved	600	Fall 2022	0
15	437501	LINUXF22	500	Approved	458	Fall 2022	42
16	954638	CFS23	100	Approved	100	Spring 2023	0
17	320197	CTF2.0S23	500	Approved	450	Spring 2023	50

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BUDGET_ID	VARCHAR2(10 BYTE)	No	(null)	1	(null)
2	EVENT_ID	VARCHAR2(10 BYTE)	No	(null)	2	(null)
3	AMOUNT_OF_BUDGET	NUMBER(6,2)	No	(null)	3	(null)
4	BUDGET_STATUS	VARCHAR2(10 BYTE)	Yes	(null)	4	(null)
5	BUDGET_USED	NUMBER(6,2)	No	0.00	5	(null)
6	ACADEMIC_TERM	VARCHAR2(15 BYTE)	No	(null)	6	(null)
7	BUDGET_LEFT	NUMBER	Yes	"AMOUNT_OF_BUDGET"-"BUDGET_USED"	7	(null)

After

DERIVED ATTRIBUTE

```
UPDATE BUDGET
SET BUDGET_LEFT = AMOUNT_OF_BUDGET - BUDGET_USED ;

ALTER TABLE BUDGET
DROP COLUMN BUDGET_LEFT;

ALTER TABLE BUDGET
 ADD BUDGET_LEFT AS (AMOUNT_OF_BUDGET - BUDGET_USED);
```

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BUDGET_ID	VARCHAR2(10 BYTE)	No	(null)	1	(null)
2	EVENT_ID	VARCHAR2(10 BYTE)	No	(null)	2	(null)
3	AMOUNT_OF_BUDGET	NUMBER(6,2)	No	(null)	3	(null)
4	BUDGET_STATUS	VARCHAR2(10 BYTE)	Yes	(null)	4	(null)
5	BUDGET_USED	NUMBER(6,2)	No	0.00	5	(null)
6	BUDGET_LEFT	NUMBER(6,2)	No	(null)	6	(null)
7	ACADEMIC_TERM	VARCHAR2(15 BYTE)	No	(null)	7	(null)

	BUDGET_ID	EVENT_ID	AMOUNT_OF_BUDGET	BUDGET_STATUS	BUDGET_USED	BUDGET_LEFT	ACADEMIC_TERM
1	859043	CFF20	100	Approved	97.5	2.5	Fall 2020
2	295710	PALOALTF20	150	Approved	150	0	Fall 2020
3	876502	MNF20	50	Approved	50	0	Fall 2020
4	140793	AMNGUSF20	100	Approved	100.75	-0.75	Fall 2020
5	572819	AWSF20	250	Approved	0	250	Fall 2020
6	493628	ONLSECS21	100	Approved	78.2	21.8	Spring 2021
7	926517	IOTS21	100	Approved	100	0	Spring 2021
8	731984	ONWOSCS21	100	Approved	100	0	Spring 2021
9	251768	CFF21	100	Approved	100	0	Fall 2021
10	504219	TABOOS22	200	Approved	221	-21	Spring 2022
11	821076	CTFS22	100	Approved	94.3	5.7	Spring 2022
12	165397	GITHUBS22	100	Approved	99.2	0.8	Spring 2022
13	708263	WEB3.0F22	100	Approved	75	25	Fall 2022
14	205150	CARNF22	600	Approved	600	0	Fall 2022
15	437501	LINUXF22	500	Approved	458	42	Fall 2022
16	954638	CFS23	100	Approved	100	0	Spring 2023
17	320197	CTF2.0S23	500	Approved	450	50	Spring 2023

Before

INSERTING A COLUMN INTO A PRE-EXISTING TABLE

Apart from adding the Secondary_Major and minor1/minor2 attributes in the member, activities_team, executive_team, and media_team table, OSC also wishes to store the description of each bill in the bills table. To help them do so, we have added a Bill_Description column in the Bills table, where a description of the bill can be added (up to 100 characters).

This column would be helpful to know how and where each bill was used, such as for food. drinks, printing, and so on. Using this, we could also track the items OSC spends on for their events.

```
ALTER TABLE BILLS
ADD BILL_DESCRIPTION VARCHAR (100);

UPDATE BILLS
SET BILL_DESCRIPTION = 'GIFT CARD'
WHERE BILL_NO = 'PALOALTF207101322';

UPDATE BILLS
SET BILL_DESCRIPTION = 'PRINTING CUSTOMIZED TABOO CARDS'
WHERE BILL_NO = 'TABOOS2221938513';
```

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BILL_NO	VARCHAR2(20 BYTE)	No	(null)	1	(null)
2	AMOUNT_OF_BILL	NUMBER(6,2)	No	(null)	2	(null)
3	PURCHASED_BY	VARCHAR2(9 BYTE)	No	(null)	3	(null)
4	MERCHANT_NAME	VARCHAR2(25 BYTE)	No	(null)	4	(null)
5	PAYMENT_METHOD	VARCHAR2(15 BYTE)	No	(null)	5	(null)
6	REIMBURSED	CHAR(1 BYTE)	No	(null)	6	(null)
7	PAYMENT_DATE	DATE	No	(null)	7	(null)
8	BUDGET_NO	VARCHAR2(10 BYTE)	No	(null)	8	(null)
9	HANDLED_BY	VARCHAR2(9 BYTE)	No	(null)	9	(null)

	BILL_NO	AMOUNT_OF_BILL	PURCHASED_BY	MERCHANT_NAME	PAYMENT_METHOD	REIMBURSED	PAYMENT_DATE	BUDGET_NO	HANDLED_BY
1	CFF2004313201	97.5	g00089132	Noon	Card	Y	21/09/20	859043	b00084833
2	PALOALTF207101322	150	g00089132	Amazon	Card	Y	28/09/20	295710	g00085775
3	MNF205021324	50	g00089132	Amazon	Card	Y	14/10/20	876502	g00085775
4	AMNGUSF207931325	150.75	g00089132	Amazon	Card	Y	14/10/20	140793	b00084833
5	ONLSECS216281327	78.2	g00089132	Talabat	Card	Y	08/03/21	493628	g00085775
6	IOTS215171328	100	g00089132	Amazon	Card	Y	14/04/21	926517	b00084833
7	ONWOSCS219843859	100	b00088385	Amazon	Card	Y	05/05/21	731984	g00085775
8	CFF2176838510	100	b00088385	Amazon	Card	Y	20/09/21	251768	b00084833
9	TABOOS2221938513	221	b00090070	Amazon	Cash	Y	03/03/22	504219	b00084833
10	CTFS2207638514	94.3	b00090070	Amazon	Cash	Y	24/03/22	821076	g00085775
11	GITHUBS2239713215	99.2	g00089132	Amazon	Cash	Y	13/04/22	165397	b00084833
12	WEB3.0F2226313216	75	g00089132	Amazon	Cash	Y	11/10/22	708263	b00084833
13	CARNF2215013221	600	g00089132	Amazon	Cash	Y	20/10/22	205150	b00084833
14	LINUXF2250113217	458	g00089132	Amazon	Cash	Y	29/10/22	437501	g00085775
15	CFS2363813218	100	g00089132	Amazon	Cash	Y	14/02/23	954638	b00084833
16	CTF2.0S2319713219	150	g00089132	Talabat	Cash	N	21/10/23	320197	g00085775
17	CTF2.0S2319738520	300	g00089132	Amazon	Cash	N	21/10/23	320197	g00085775

Before

INSERTING A COLUMN INTO A PRE-EXISTING TABLE

	BILL_NO	AMOUNT_OF_BILL	PURCHASED_BY	MERCHANT_NAME	PAYMENT_METHOD	REIMBURSED	PAYMENT_DATE	BUDGET_NO	HANDLED_BY	BILL_DESCRIPTION
1	CFF2004313201	97.5	g00089132	Noon	Card	Y	21/09/20	859043	b00084833	(null)
2	PALOALTF207101322	150	g00089132	Amazon	Card	Y	28/09/20	295710	g00085775	gift card
3	MNF205021324	50	g00089132	Amazon	Card	Y	14/10/20	876502	g00085775	(null)
4	AMNGUSF207931325	150.75	g00089132	Amazon	Card	Y	14/10/20	140793	b00084833	(null)
5	ONLSECS216281327	78.2	g00089132	Talabat	Card	Y	08/03/21	493628	g00085775	(null)
6	IOTS215171328	100	g00089132	Amazon	Card	Y	14/04/21	926517	b00084833	(null)
7	ONWOSCS219843859	100	b00088385	Amazon	Card	Y	05/05/21	731984	g00085775	(null)
8	CFF2176838510	100	b00088385	Amazon	Card	Y	20/09/21	251768	b00084833	(null)
9	TABOOS2221938513	221	b00090070	Amazon	Cash	Y	03/03/22	504219	b00084833	printing customized taboo cards
10	CTFS2207638514	94.3	b00090070	Amazon	Cash	Y	24/03/22	821076	g00085775	(null)
11	GITHUBS2239713215	99.2	g00089132	Amazon	Cash	Y	13/04/22	165397	b00084833	(null)
12	WEB3.0F2226313216	75	g00089132	Amazon	Cash	Y	11/10/22	708263	b00084833	(null)
13	CARNF2215013221	600	g00089132	Amazon	Cash	Y	20/10/22	205150	b00084833	(null)
14	LINUXF2250113217	458	g00089132	Amazon	Cash	Y	29/10/22	437501	g00085775	(null)
15	CFS2363813218	100	g00089132	Amazon	Cash	Y	14/02/23	954638	b00084833	(null)
16	CTF2.0S2319713219	150	g00089132	Talabat	Cash	N	21/10/23	320197	g00085775	(null)
17	CTF2.0S2319738520	300	g00089132	Amazon	Cash	N	21/10/23	320197	g00085775	(null)

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	BILL_NO	VARCHAR2(20 BYTE)	No	(null)	1	(null)
2	AMOUNT_OF_BILL	NUMBER(6,2)	No	(null)	2	(null)
3	PURCHASED_BY	VARCHAR2(9 BYTE)	No	(null)	3	(null)
4	MERCHANT_NAME	VARCHAR2(25 BYTE)	No	(null)	4	(null)
5	PAYMENT_METHOD	VARCHAR2(15 BYTE)	No	(null)	5	(null)
6	REIMBURSED	CHAR(1 BYTE)	No	(null)	6	(null)
7	PAYMENT_DATE	DATE	No	(null)	7	(null)
8	BUDGET_NO	VARCHAR2(10 BYTE)	No	(null)	8	(null)
9	HANDLED_BY	VARCHAR2(9 BYTE)	No	(null)	9	(null)
10	BILL_DESCRIPTION	VARCHAR2(100 BYTE)	Yes	(null)	10	(null)

After

DROP A COLUMN FROM A PRE-EXISTING TABLE

Currently, our media_content table has an attribute called Platform, which stores the platform where OSC posts the media content that they produce. This column seemed necessary for the OSC members to include in their database, as they have social media on Instagram and Facebook.

However, since OSC only posts media content on Instagram, the platform content seems redundant and outdated. The platform column currently only stores 'Instagram' as the value for all media content. All media content that is made by OSC is made and customized to post on their Instagram page.

Thus, we have decided to drop the Platform column from the media_content table, as it does not provide us with any additional or useful information for further analysis

```
ALTER TABLE MEDIA_CONTENT
DROP COLUMN PLATFORM;
```

	COLUMN_N...	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	CONTENT_ID	VARCHAR2(15 BYTE)	No	(null)	1	(null)
2	CREATOR_AUS_ID	VARCHAR2(9 BYTE)	Yes	(null)	2	(null)
3	EVENT_ID	VARCHAR2(10 BYTE)	Yes	(null)	3	(null)
4	FORMAT	VARCHAR2(15 BYTE)	No	(null)	4	(null)
5	DATEPUBLISHED	DATE	No	(null)	5	(null)
6	PLATFORM	VARCHAR2(15 BYTE)	No	(null)	6	(null)
7	NO_OF_LIKES	NUMBER(38,0)	No	(null)	7	(null)
8	NO_OF_VIEWS	NUMBER(38,0)	No	(null)	8	(null)

	CONTENT_ID	CREATOR_AUS_ID	EVENT_ID	FORMAT	DATEPUBLISHED	PLATFORM	NO_OF_LIKES	NO_OF_VIEWS
1	CFF20-01	g00090589	CFF20	Post	22/09/20	Instagram	100	172
2	PALOALTF20-01	b00093124	PALOALTF20	Post	28/09/20	Instagram	27	500
3	NNF20-01	g00085270	NNF20	Post	06/10/20	Instagram	31	200
4	MNF20-01	g00088567	MNF20	Post	14/10/20	Instagram	50	150
5	AMNGUSF20-01	b00086276	AMNGUSF20	Post	26/10/20	Instagram	46	100
6	AWSF20-01	g00088575	AWSF20	Post	27/11/20	Instagram	34	650
7	ONLSECS21-01	b00091171	ONLSECS21	Post	07/03/21	Instagram	47	1500
8	IOTS21-01	b00091171	IOTS21	Post	13/04/21	Instagram	70	700
9	ONWOSCS21-01	g00094956	ONWOSCS21	Video	04/05/21	Instagram	44	219
10	CFF21-01	b00094999	CFF21	Video	19/09/21	Instagram	40	210
11	CTBTWPYF21-01	b00094999	CTBTWPYF21	Video	12/10/21	Instagram	33	210
12	MUSEDITF21-01	b00088589	MUSEDITF21	Post	15/11/21	Instagram	39	210
13	TABOOS22-01	g00090589	TABOOS22	Post	02/02/22	Instagram	37	88
14	TABOOS22-02	b00093124	TABOOS22	Reel	27/02/22	Instagram	35	1566
15	CTFS22-00	g00085270	CTFS22	Reel	22/03/22	Instagram	33	700
16	CTFS22-01	g00088567	CTFS22	Reel	23/03/22	Instagram	31	219
17	CTFS22-02	b00086276	CTFS22	Reel	31/03/22	Instagram	100	210
18	GITHUBS22-01	g00088575	GITHUBS22	Reel	12/04/22	Instagram	27	210
19	WEB3.0F22-01	b00091171	WEB3.0F22	Reel	08/10/22	Instagram	31	210
20	CARNF22-01	g00085270	CARNF22	Reel	18/10/22	Instagram	100	600
21	CARNF22-02	b00093124	CARNF22	Reel	21/10/22	Instagram	300	1200
22	LINUXF22-01	g00094956	LINUXF22	Reel	04/12/22	Instagram	50	88
23	CFS23-01	g00088575	CFS23	Reel	12/12/22	Instagram	46	999
24	CTF2.0S23-00	b00094999	CTF2.0S23	Reel	14/03/23	Instagram	50	2067
25	CTF2.0S23-01	g00094956	CTF2.0S23	Reel	19/03/23	Instagram	46	988
26	CTF2.0S23-02	b00088589	CTF2.0S23	Reel	30/03/23	Instagram	100	700
27	TTS23-01	g00090589	(null)	Reel	08/03/23	Instagram	55	3000
28	TTS23-02	b00093124	(null)	Reel	22/03/23	Instagram	31	2100
29	TTS23-03	g00085270	(null)	Reel	05/04/23	Instagram	52	5699

Before

DROP A COLUMN FROM A PRE-EXISTING TABLE

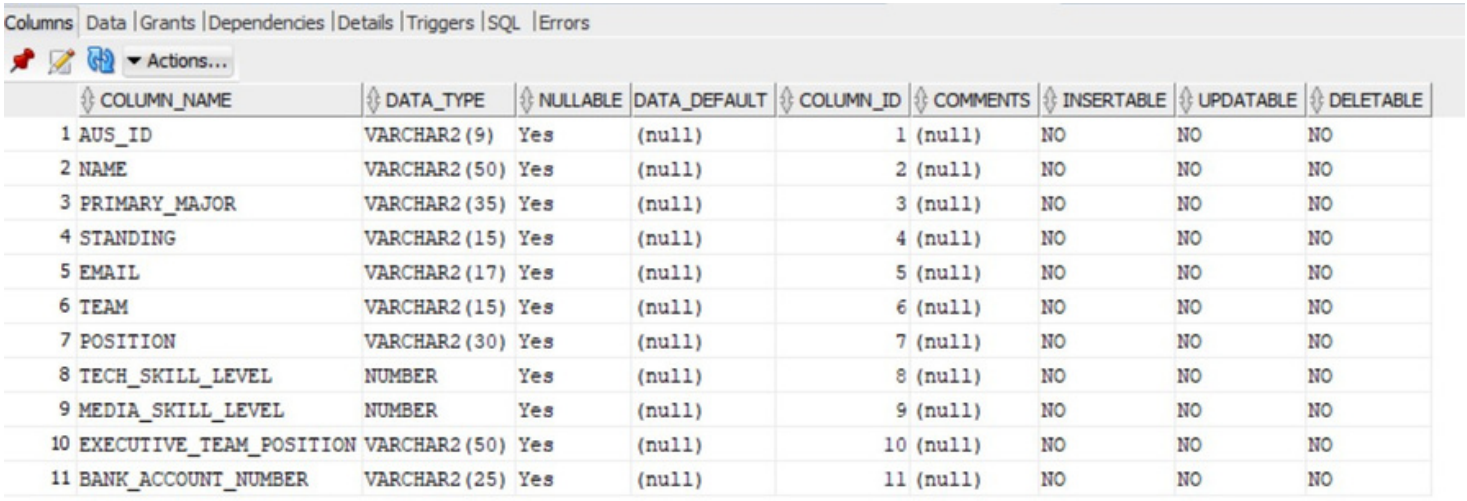
	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	CONTENT_ID	VARCHAR2(15 BYTE)	No	(null)	1	(null)
2	CREATOR_AUS_ID	VARCHAR2(9 BYTE)	Yes	(null)	2	(null)
3	EVENT_ID	VARCHAR2(10 BYTE)	Yes	(null)	3	(null)
4	FORMAT	VARCHAR2(15 BYTE)	No	(null)	4	(null)
5	DATEPUBLISHED	DATE	No	(null)	5	(null)
6	NO_OF_LIKES	NUMBER(38,0)	No	(null)	6	(null)
7	NO_OF_VIEWS	NUMBER(38,0)	No	(null)	7	(null)

	CONTENT_ID	CREATOR_AUS_ID	EVENT_ID	FORMAT	DATEPUBLISHED	NO_OF_LIKES	NO_OF_VIEWS
1	CFF20-01	g00090589	CFF20	Post	22/09/20	100	172
2	PALOALTF20-01	b00093124	PALOALTF20	Post	28/09/20	27	500
3	NNF20-01	g00085270	NNF20	Post	06/10/20	31	200
4	MNF20-01	g00088567	MNF20	Post	14/10/20	50	150
5	AMNGUSF20-01	b00086276	AMNGUSF20	Post	26/10/20	46	100
6	AWSF20-01	g00088575	AWSF20	Post	27/11/20	34	650
7	ONLSECS21-01	b00091171	ONLSECS21	Post	07/03/21	47	1500
8	IOTS21-01	b00091171	IOTS21	Post	13/04/21	70	700
9	ONWOSCS21-01	g00094956	ONWOSCS21	Video	04/05/21	44	219
10	CFF21-01	b00094999	CFF21	Video	19/09/21	40	210
11	CTBTWPYF21-01	b00094999	CTBTWPYF21	Video	12/10/21	33	210
12	MUSEDITF21-01	b00088589	MUSEDITF21	Post	15/11/21	39	210
13	TABOOS22-01	g00090589	TABOOS22	Post	02/02/22	37	88
14	TABOOS22-02	b00093124	TABOOS22	Reel	27/02/22	35	1566
15	CTFS22-00	g00085270	CTFS22	Reel	22/03/22	33	700
16	CTFS22-01	g00088567	CTFS22	Reel	23/03/22	31	219
17	CTFS22-02	b00086276	CTFS22	Reel	31/03/22	100	210
18	GITHUBS22-01	g00088575	GITHUBS22	Reel	12/04/22	27	210
19	WEB3.0F22-01	b00091171	WEB3.0F22	Reel	08/10/22	31	210
20	CARNF22-01	g00085270	CARNF22	Reel	18/10/22	100	600
21	CARNF22-02	b00093124	CARNF22	Reel	21/10/22	300	1200
22	LINUXF22-01	g00094956	LINUXF22	Reel	04/12/22	50	88
23	CFS23-01	g00088575	CFS23	Reel	12/12/22	46	999
24	CTF2.0S23-00	b00094999	CTF2.0S23	Reel	14/03/23	50	2067
25	CTF2.0S23-01	g00094956	CTF2.0S23	Reel	19/03/23	46	988
26	CTF2.0S23-02	b00088589	CTF2.0S23	Reel	30/03/23	100	700
27	TTS23-01	g00090589	(null)	Reel	08/03/23	55	3000
28	TTS23-02	b00093124	(null)	Reel	22/03/23	31	2100
29	TTS23-03	g00085270	(null)	Reel	05/04/23	52	5699

After

UPDATABLE VIEW: Rafid

```
CREATE VIEW ALL_TEAMS  AS
SELECT AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM, NULL AS POSITION,
NULL AS TECH_SKILL_LEVEL, NULL AS MEDIA_SKILL_LEVEL, NULL AS
EXECUTIVE_TEAM_POSITION, NULL AS BANK_ACCOUNT_NUMBER
FROM MEMBER
UNION
SELECT AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM,
ACTIVITIES_TEAM_POSITION AS POSITION, TECH_SKILL_LEVEL, NULL AS
MEDIA_SKILL_LEVEL, NULL AS EXECUTIVE_TEAM_POSITION, NULL AS
BANK_ACCOUNT_NUMBER
FROM ACTIVITIES_TEAM
UNION
SELECT AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM,
MEDIA_TEAM_POSITION AS POSITION, NULL AS TECH_SKILL_LEVEL, MEDIA_SKILL_LEVEL,
NULL AS EXECUTIVE_TEAM_POSITION, NULL AS BANK_ACCOUNT_NUMBER
FROM MEDIA_TEAM
UNION
SELECT AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM,
EXECUTIVE_TEAM_POSITION AS POSITION, NULL AS TECH_SKILL_LEVEL, NULL AS
MEDIA_SKILL_LEVEL, BANK_ACCOUNT_NUMBER, EXECUTIVE_TEAM_POSITION
```



Columns | Data | Grants | Dependencies | Details | Triggers | SQL | Errors

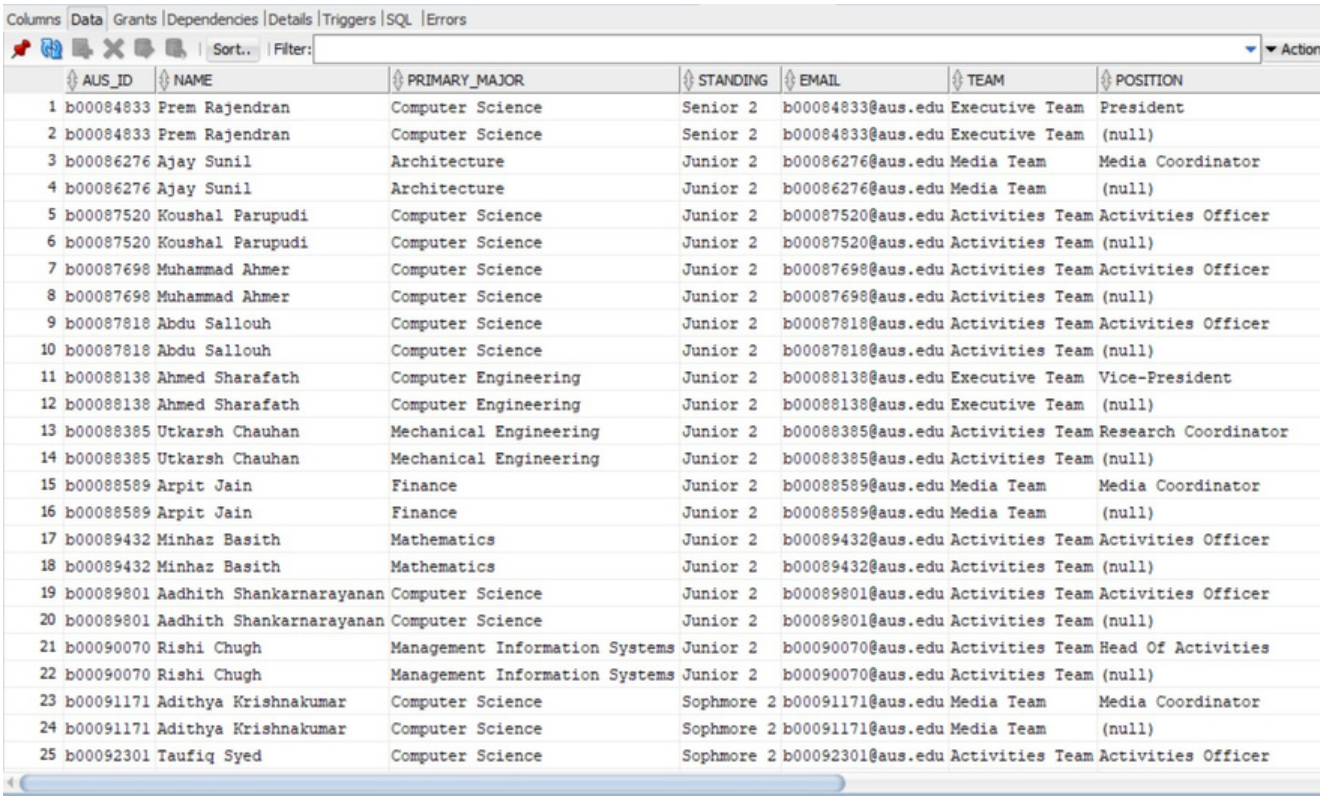
Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS	INSERTABLE	UPDATABLE	DELETABLE
1	AUS_ID	VARCHAR2(9)	Yes	(null)	1	(null)	NO	NO	NO
2	NAME	VARCHAR2(50)	Yes	(null)	2	(null)	NO	NO	NO
3	PRIMARY_MAJOR	VARCHAR2(35)	Yes	(null)	3	(null)	NO	NO	NO
4	STANDING	VARCHAR2(15)	Yes	(null)	4	(null)	NO	NO	NO
5	EMAIL	VARCHAR2(17)	Yes	(null)	5	(null)	NO	NO	NO
6	TEAM	VARCHAR2(15)	Yes	(null)	6	(null)	NO	NO	NO
7	POSITION	VARCHAR2(30)	Yes	(null)	7	(null)	NO	NO	NO
8	TECH_SKILL_LEVEL	NUMBER	Yes	(null)	8	(null)	NO	NO	NO
9	MEDIA_SKILL_LEVEL	NUMBER	Yes	(null)	9	(null)	NO	NO	NO
10	EXECUTIVE_TEAM_POSITION	VARCHAR2(50)	Yes	(null)	10	(null)	NO	NO	NO
11	BANK_ACCOUNT_NUMBER	VARCHAR2(25)	Yes	(null)	11	(null)	NO	NO	NO

UPDATABLE VIEW: Rafid

Continuation;

.



Columns | Data | Grants | Dependencies | Details | Triggers | SQL | Errors

Sort.. | Filter: | Action

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	POSITION
1	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team	President
2	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team	(null)
3	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team	Media Coordinator
4	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team	(null)
5	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team	Activities Officer
6	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team	(null)
7	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team	Activities Officer
8	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team	(null)
9	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team	Activities Officer
10	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team	(null)
11	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team	Vice-President
12	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team	(null)
13	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team	Research Coordinator
14	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team	(null)
15	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team	Media Coordinator
16	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team	(null)
17	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team	Activities Officer
18	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team	(null)
19	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team	Activities Officer
20	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team	(null)
21	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team	Head Of Activities
22	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team	(null)
23	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team	Media Coordinator
24	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team	(null)
25	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team	Activities Officer

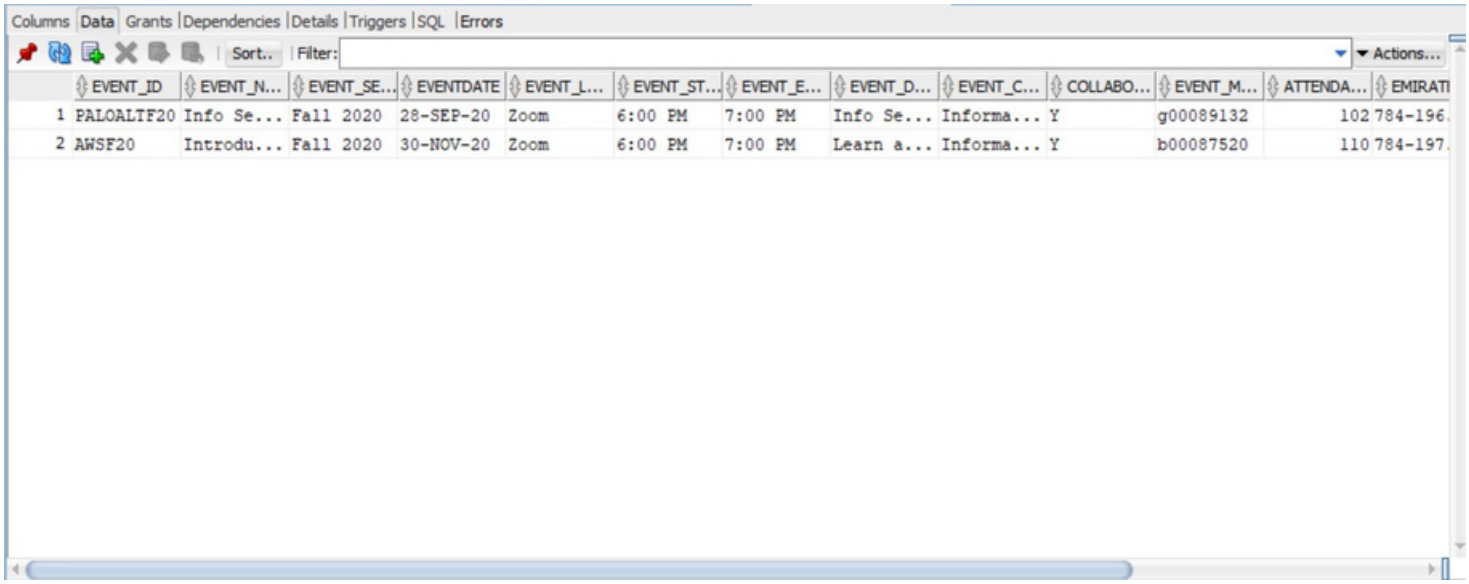
This view is a solution to avoid joining multiple member tables together. Essentially, we combine all the member tables i.e., member, activities, media, and executive team tables together, by creating a union between all four tables and selecting all attributes from each. It combines the common attributes, creating a large table that contains all the data about all members of the OSC.

When performing queries using the member table, more often than not, we had to join it with other member tables such as the activities team. Hence, this view brings back to the 2NF for ease of computing

Note: The output is too large to put in a single screenshot

UPDATABLE VIEW: Colin

```
CREATE VIEW EXTERNAL_EVENTS  AS
SELECT EVENT.EVENT_ID, EVENT.EVENT_NAME, EVENT.EVENT_SEMESTER,
EVENT.EVENTDATE, EVENT.EVENT_LOCATION, EVENT.EVENT_STARTTIME,
EVENT.EVENT_ENDTIME, EVENT.EVENT_DESCRIPTION, EVENT.EVENT_CATEGORY,
EVENT.COLLABORATED_EVENT, EVENT.EVENT_MANAGER, EVENT.ATTENDANCE,
EVENT_COLLABORATED.EMIRATES_ID, COLLABORATORS.NAME_OF_COLLABORATOR,
COLLABORATORS.ORGANIZATION, COLLABORATORS.COLLABORATOR_TYPE
FROM (EVENT JOIN EVENT_COLLABORATED ON EVENT.EVENT_ID =
EVENT_COLLABORATED.EVENT_ID)JOIN COLLABORATORS ON
EVENT_COLLABORATED.EMIRATES_ID = COLLABORATORS.EMIRATES_ID
WHERE EVENT.COLLABORATED_EVENT = 'Y' AND COLLABORATOR_TYPE = 'EXTERNAL';
```



	EVENT_ID	EVENT_N...	EVENT_SE...	EVENTDATE	EVENT_L...	EVENT_ST...	EVENT_E...	EVENT_D...	EVENT_C...	COLLABO...	EVENT_M...	ATTENDA...	EMIRATI
1	PALOALTF20	Info Se...	Fall 2020	28-SEP-20	Zoom	6:00 PM	7:00 PM	Info Se...	Informa...	Y	g00089132	102	784-196.
2	AWSF20	Introdu...	Fall 2020	30-NOV-20	Zoom	6:00 PM	7:00 PM	Learn a...	Informa...	Y	b00087520	110	784-197.

Created a view called "External_Events". A view is like a virtual table that doesn't actually store data but shows data from other tables based on certain criteria.

The view includes data from three tables: "Event", "EVENT_COLLABORATED", and "COLLABORATORS". It selects specific columns from these tables and filters them based on certain conditions.

The conditions are that the event must be a collaborated event ('Y' in the "Collaborated_Event" column) and the collaborator type must be external ('External' in the "Collaborator_Type" column).

The columns that are included in the view are event ID, event name, event semester, event date, event location, event start time, event end time, event description, event category, event manager, attendance, collaborator Emirates ID, name of collaborator, organization, and collaborator type.

So basically, the view will show all the events that are collaborated with external organizations and the details of those events along with the external organization and collaborator's information.

Note: The full table cannot be viewed in the screenshot as the table is large

UPDATABLE VIEW: Bhavika

We have created an updatable view called IdeaRepositoryView. This updatable view helps us to insert new rows, update rows, and delete rows to the Idea_Repository table directly, through the view. Any insert, delete, or updation functions that are performed to the IdeaRepositoryView, are directly propagated to the Idea_Repository table.

We have created a view for the Idea_Repository table so it becomes easier to insert, update and delete rows for this table. Every time a member has a new event or media content idea, they can easily update the Idea_Respository table through the IdeaRepositoryView. They can also very easily update and delete these ideas from the Idea_Repository table.

```
CREATE VIEW IDEAREPOSITORYVIEW AS
SELECT *
FROM IDEA_REPOSITORY
WITH CHECK OPTION;

INSERT INTO IDEAREPOSITORYVIEW VALUES ('011', 'TECHTALKS', 'A SERIES OF TALKS
ABOUT OSS IN DIFFERENT FIELDS.', 'G00089132', 'EVENTIDEA', '200', 'PROPOSED',
'SPEAKER, PROJECTOR');

UPDATE IDEAREPOSITORYVIEW SET IDEA_NAME = 'OSCTECHTALKS',
ESTIMATED_BUDGET_REQUIRED = '300' WHERE IDEA_NUMBER = '011';

DELETE FROM IDEAREPOSITORYVIEW WHERE STATUS = 'UNAPPROVED'
```

We have created an updatable view called IdeaRepositoryView, where we selected all rows from the Idea_Repository table. The WITH CHECK OPTION clause ensures that any update, insert, or delete operation performed on the view satisfies the condition specified in the WHERE clause of the view definition.

For example, in the IdeaRepositoryView view definition you provided, any update, insert or delete operation on the view will be checked to ensure that the WHERE condition is satisfied. In this case, the WHERE condition is not specified, so any operation on the view is allowed. However, if you add a WHERE condition to the view definition, such as WHERE Idea_Status = 'Approved', then any update, insert or delete operation on the view will be checked to ensure that the WHERE condition is satisfied.

If an operation violates the WHERE condition, it will fail and the database will return an error message. This helps to ensure data integrity and prevent unintended changes to the underlying tables.

To test the view, we have inserted a row into the Idea_Repository table, with the Idea_Name of 'TechTalks'. Further, we have updated the same column, to change the budget from AED 200 to AED 300. Lastly, we have deleted all those rows where the status is 'Unapproved'.

UPDATABLE VIEW: Bhavika

	IDEA_NUMBER	IDEA_NAME	IDEA_DESCRIPTION	MEMBER_ID	IDEACATEGORY	ESTIMATED_BUDGET_REQUIRED	STATUS	SPEC
1	001	OSCMovieNight2.0	A movie night with OSC to watch the movie Snowden	b00084833	EventIdea	100	Proposed	proje
2	002	GuesstheOSSoftware	A Instagram Story Series with OS software hints and polls to guess the OSS.	g00087337	MediaIdea	50	Proposed	(null
3	003	MeetTheTeam	Day in the Life Instagram story series with different members once a week.	g00087337	MediaIdea	0	Proposed	(null
4	004	TechnoBytes	Instagram Story series with OSS related news.	g00090589	MediaIdea	0	Proposed	(null
5	005	OSCDataScienceSeries	An event series about data science in different fields such as business, engineering, architecture.	g00089132	EventIdea	(null)	To be Executed	(null
6	006	WhoWantsToBeAMillionare?	An event wherein we play the game Who Wants to Be A Millionaire with questions about OSS.	b00090070	EventIdea	(null)	To be Executed	(null
7	007	Taboo2.0	A part 2 for the Taboo event.	b00088385	EventIdea	200	Proposed	custo
8	008	Aren'tWeOSome?	A Who's Most Likely to Instagram Story Series with OSC members.	g00089132	MediaIdea	0	To be Executed	OSC M
9	009	BoardGameNightwithOSC	A board game night with OSC, playing games like chess, UNO and so on.	g00089132	EventIdea	300	Proposed	board
10	010	7Days7OSS	An Instagram Story Series where we educate members about OSS, one a day for a week.	g00085775	MediaIdea	0	Unapproved	(null

Before

	IDEA_NUMBER	IDEA_NAME	IDEA_DESCRIPTION	MEMBER_ID	IDEACATEGORY	ESTIMATED_BUDGET_REQUIRED	STATUS	SPEC
1	001	OSCMovieNight2.0	A movie night with OSC to watch the movie Snowden	b00084833	EventIdea	100	Proposed	proje
2	002	GuesstheOSSoftware	A Instagram Story Series with OS software hints and polls to guess the OSS.	g00087337	MediaIdea	50	Proposed	(null
3	003	MeetTheTeam	Day in the Life Instagram story series with different members once a week.	g00087337	MediaIdea	0	Proposed	(null
4	004	TechnoBytes	Instagram Story series with OSS related news.	g00090589	MediaIdea	0	Proposed	(null
5	005	OSCDataScienceSeries	An event series about data science in different fields such as business, engineering, architecture.	g00089132	EventIdea	(null)	To be Executed	(null
6	006	WhoWantsToBeAMillionare?	An event wherein we play the game Who Wants to Be A Millionaire with questions about OSS.	b00090070	EventIdea	(null)	To be Executed	(null
7	007	Taboo2.0	A part 2 for the Taboo event.	b00088385	EventIdea	200	Proposed	custo
8	008	Aren'tWeOSome?	A Who's Most Likely to Instagram Story Series with OSC members.	g00089132	MediaIdea	0	To be Executed	OSC M
9	009	BoardGameNightwithOSC	A board game night with OSC, playing games like chess, UNO and so on.	g00089132	EventIdea	300	Proposed	board
10	011	OSCTechTalks	A series of talks about OSS in different fields.	g00089132	EventIdea	300	Proposed	speak

After

PROCEDURES: Bhavika

We have created a procedure to help insert new members into the Member table. As the new academic semester will begin soon, OSC will start recruiting new members to join the team. Thus, we have created a procedure to help make the process of inserting members and their data into the club database easier.

We have created a procedure to insert new members into the club database, as this is something that is done on a semester basis. By creating a procedure, we can reuse the procedure multiple times to help insert new members into the teams.

Ultimately, we would also have to insert the new member into the corresponding team, which we will do with the help of a trigger.

```
CREATE OR REPLACE PROCEDURE INSERTMEMBER(
    AUS_ID IN VARCHAR,
    NAME IN VARCHAR,
    PRIMARY_MAJOR IN VARCHAR,
    STANDING IN VARCHAR,
    EMAIL IN VARCHAR,
    TEAM IN VARCHAR
)
AS
BEGIN
    INSERT INTO MEMBER (AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL,
TEAM)
    VALUES (AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM);
    DBMS_OUTPUT.PUT_LINE('MEMBER ' || NAME || ' ADDED.');
END;
/

SELECT * FROM MEMBER;

EXEC INSERTMEMBER ('B00095219', 'NEERAJ VOHRA', 'ACCOUNTING', 'JUNIOR 1',
'B00095219@AUS.EDU', 'ACTIVITIES TEAM');

SELECT * FROM MEMBER;
```

PROCEDURES: Bhavika

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM
1	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team
5	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team
6	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team
7	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team
8	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team
9	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team
10	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team
11	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team
12	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team
13	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team
14	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team
15	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team
16	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team
17	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team
18	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team
19	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team
20	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team
21	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team
22	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team
23	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team
24	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team
25	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team

Before

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team
5	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team
6	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team
7	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team
8	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team
9	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team
10	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team
11	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team
12	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team
13	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team
14	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team
15	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team
16	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team
17	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team
18	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team
19	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team
20	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team
21	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team
22	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team
23	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team
24	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team
25	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team
26	b00095219	Neeraj Vohra	Accounting	Junior 1	b00095219@aus.edu	Activities Team

After

PROCEDURES: Bhavika

We have created another procedure to help us insert the event and media ideas of the members into the Idea_Repository table. A lot of team members have innovative and unique event and media ideas, which we would like to store in our Idea_Repository table. Due to the large inflow of ideas on a daily basis, we have created a procedure to help us easily add ideas to the Idea_Repository. Since this is a repetitive process, we have decided to make a procedure.

```
CREATE OR REPLACE PROCEDURE INSERT_IDEA_REPOSITORY (
  I_IDEA_NUMBER IN VARCHAR2,
  I_IDEA_NAME IN VARCHAR2,
  I_IDEA_DESCRIPTION IN VARCHAR2,
  I_MEMBER_ID IN VARCHAR2,
  I_IDEA_CATEGORY IN VARCHAR2,
  I_ESTIMATED_BUDGET_REQUIRED IN NUMBER,
  I_STATUS IN VARCHAR2,
  I_SPECIAL_RESOURCES_NEEDED IN VARCHAR2
)
AS
BEGIN
  INSERT INTO Idea_Repository(Idea_Number, Idea_Name, Idea_Description,
Member_ID, IdeaCategory, Estimated_Budget_Required, Status,
SpecialResoucesNeeded)
  VALUES (I_IDEA_NUMBER, I_IDEA_NAME, I_IDEA_DESCRIPTION, I_MEMBER_ID,
I_IDEA_CATEGORY, I_ESTIMATED_BUDGET_REQUIRED, I_STATUS,
I_SPECIAL_RESOURCES_NEEDED);
  DBMS_OUTPUT.PUT_LINE('Idea ' || I_IDEA_NAME || ' added successfully');
END;
/


Select * from IDEA_repository;

EXEC INSERT_IDEA_REPOSITORY  ('012', 'EatwithOSC', 'Have lunch with OSC
members', 'g00089132', 'EventIdea',200, 'Proposed',NULL);

Select * from IDEA_repository;
```

PROCEDURES: Bhavika

	IDEA_NUMBER	IDEA_NAME	IDEA_DESCRIPTION	MEMBER_ID	IDEACATEGORY	ESTIMATED_BUDGET_REQUIRED	STATUS	SPEC
1	001	OSCMovieNight2.0	A movie night with OSC to watch the movie Snowden	b00084833	EventIdea	100	Proposed	proje
2	002	GuesstheOSSoftware	A Instagram Story Series with OS software hints and polls to guess the OSS.	g00087337	MediaIdea	50	Proposed	(null
3	003	MeetTheTeam	Day in the Life Instagram story series with different members once a week.	g00087337	MediaIdea	0	Proposed	(null
4	004	TechnoBytes	Instagram Story series with OSS related news.	g00090589	MediaIdea	0	Proposed	(null
5	005	OSCDataScienceSeries	An event series about data science in different fields such as business, engineering, architecture.	g00089132	EventIdea	(null)	To be Executed	(null
6	006	WhoWantsToBeAMillionare?	An event wherein we play the game Who Wants to Be A Millionaire with questions about OSS.	b00090070	EventIdea	(null)	To be Executed	(null
7	007	Taboo2.0	A part 2 for the Taboo event.	b00088385	EventIdea	200	Proposed	custo
8	008	Aren'tWeOSome?	A Who's Most Likely to Instagram Story Series with OSC members.	g00089132	MediaIdea	0	To be Executed	OSC M
9	009	BoardGameNightwithOSC	A board game night with OSC, playing games like chess, UNO and so on.	g00089132	EventIdea	300	Proposed	board
10	011	OSCTechTalks	A series of talks about OSS in different fields.	g00089132	EventIdea	300	Proposed	speak

Before

	IDEA_NUMBER	IDEA_NAME	IDEA_DESCRIPTION	MEMBER_ID	IDEACATEGORY	ESTIMATED_BUDGET_REQUIRED	STATUS	SPEC
1	001	OSCMovieNight2.0	A movie night with OSC to watch the movie Snowden	b00084833	EventIdea	100	Proposed	proje
2	002	GuesstheOSSoftware	A Instagram Story Series with OS software hints and polls to guess the OSS.	g00087337	MediaIdea	50	Proposed	(null
3	003	MeetTheTeam	Day in the Life Instagram story series with different members once a week.	g00087337	MediaIdea	0	Proposed	(null
4	004	TechnoBytes	Instagram Story series with OSS related news.	g00090589	MediaIdea	0	Proposed	(null
5	005	OSCDataScienceSeries	An event series about data science in different fields such as business, engineering, architecture.	g00089132	EventIdea	(null)	To be Executed	(null
6	006	WhoWantsToBeAMillionare?	An event wherein we play the game Who Wants to Be A Millionaire with questions about OSS.	b00090070	EventIdea	(null)	To be Executed	(null
7	007	Taboo2.0	A part 2 for the Taboo event.	b00088385	EventIdea	200	Proposed	custo
8	008	Aren'tWeOSome?	A Who's Most Likely to Instagram Story Series with OSC members.	g00089132	MediaIdea	0	To be Executed	OSC M
9	009	BoardGameNightwithOSC	A board game night with OSC, playing games like chess, UNO and so on.	g00089132	EventIdea	300	Proposed	board
10	011	OSCTechTalks	A series of talks about OSS in different fields.	g00089132	EventIdea	300	Proposed	speak
11	012	EatwithOSC	Have lunch with OSC members	g00089132	EventIdea	200	Proposed	(null

After

PROCEDURES: Bhavika

We have created another procedure to help us mark previously unreimbursed bills, as reimbursed after the money has been reimbursed to the member who paid for the bill. As the president and treasurer are graduating, they would like to make sure to complete and reimburse all the pending reimbursements. In order to help them to do so in a quicker, easier, and more efficient manner, we have created a procedure called mark_bill_as_reimbursed, through which they can convert non-reimbursed bills into reimbursed bills. Essentially, the procedure would help the treasurer/president to change the reimbursement status of bills from 'N' to 'Y'.

```
CREATE OR REPLACE PROCEDURE MARK_BILL_AS_REIMBURSED(P_BILL_NO IN
VARCHAR2)
AS
BEGIN
  UPDATE BILLS
  SET REIMBURSED = 'Y'
  WHERE BILL_NO = P_BILL_NO;

  DBMS_OUTPUT.PUT_LINE('BILL ' || P_BILL_NO || ' HAS BEEN MARKED AS
REIMBURSED.');
END;
/

SELECT *
FROM BILLS;

EXEC MARK_BILL_AS_REIMBURSED('CTF2.0S2319738520');

SELECT *
FROM BILLS;

EXEC MARK_BILL_AS_REIMBURSED('CTF2.0S2319713219');

SELECT *
FROM BILLS;
```

PROCEDURES: Bhavika

	BILL_NO	AMOUNT_OF_BILL	PURCHASED_BY	MERCHANT_NAME	PAYMENT_METHOD	REIMBURSED	PAYMENT_DATE	BUDGET_NO	HANDLED_BY
1	CFF2004313201	97.5	g00089132	Noon	Card	Y	21/09/20	859043	b00084833
2	PALOALTF207101322	150	g00089132	Amazon	Card	Y	28/09/20	295710	g00085775
3	MNF205021324	50	g00089132	Amazon	Card	Y	14/10/20	876502	g00085775
4	AMNGUSF207931325	150.75	g00089132	Amazon	Card	Y	14/10/20	140793	b00084833
5	ONLSECS216281327	78.2	g00089132	Talabat	Card	Y	08/03/21	493628	g00085775
6	IOTS215171328	100	g00089132	Amazon	Card	Y	14/04/21	926517	b00084833
7	ONWOSCS219843859	100	b00088385	Amazon	Card	Y	05/05/21	731984	g00085775
8	CFF2176838510	100	b00088385	Amazon	Card	Y	20/09/21	251768	b00084833
9	TABOOS2221938513	221	b00090070	Amazon	Cash	Y	03/03/22	504219	b00084833
10	CTFS2207638514	94.3	b00090070	Amazon	Cash	Y	24/03/22	821076	g00085775
11	GITHUBS2239713215	99.2	g00089132	Amazon	Cash	Y	13/04/22	165397	b00084833
12	WEB3.0F2226313216	75	g00089132	Amazon	Cash	Y	11/10/22	708263	b00084833
13	CARNF2215013221	600	g00089132	Amazon	Cash	Y	20/10/22	205150	b00084833
14	LINUXF2250113217	458	g00089132	Amazon	Cash	Y	29/10/22	437501	g00085775
15	CFS2363813218	100	g00089132	Amazon	Cash	Y	14/02/23	954638	b00084833
16	CTF2.0S2319713219	150	g00089132	Talabat	Cash	N	21/10/23	320197	g00085775
17	CTF2.0S2319738520	300	g00089132	Amazon	Cash	N	21/10/23	320197	g00085775

Before

	BILL_NO	AMOUNT_OF_BILL	PURCHASED_BY	MERCHANT_NAME	PAYMENT_METHOD	REIMBURSED	PAYMENT_DATE	BUDGET_NO	HANDLED_BY
1	CFF2004313201	97.5	g00089132	Noon	Card	Y	21/09/20	859043	b00084833
2	PALOALTF207101322	150	g00089132	Amazon	Card	Y	28/09/20	295710	g00085775
3	MNF205021324	50	g00089132	Amazon	Card	Y	14/10/20	876502	g00085775
4	AMNGUSF207931325	150.75	g00089132	Amazon	Card	Y	14/10/20	140793	b00084833
5	ONLSECS216281327	78.2	g00089132	Talabat	Card	Y	08/03/21	493628	g00085775
6	IOTS215171328	100	g00089132	Amazon	Card	Y	14/04/21	926517	b00084833
7	ONWOSCS219843859	100	b00088385	Amazon	Card	Y	05/05/21	731984	g00085775
8	CFF2176838510	100	b00088385	Amazon	Card	Y	20/09/21	251768	b00084833
9	TABOOS2221938513	221	b00090070	Amazon	Cash	Y	03/03/22	504219	b00084833
10	CTFS2207638514	94.3	b00090070	Amazon	Cash	Y	24/03/22	821076	g00085775
11	GITHUBS2239713215	99.2	g00089132	Amazon	Cash	Y	13/04/22	165397	b00084833
12	WEB3.0F2226313216	75	g00089132	Amazon	Cash	Y	11/10/22	708263	b00084833
13	CARNF2215013221	600	g00089132	Amazon	Cash	Y	20/10/22	205150	b00084833
14	LINUXF2250113217	458	g00089132	Amazon	Cash	Y	29/10/22	437501	g00085775
15	CFS2363813218	100	g00089132	Amazon	Cash	Y	14/02/23	954638	b00084833
16	CTF2.0S2319713219	150	g00089132	Talabat	Cash	N	21/10/23	320197	g00085775
17	CTF2.0S2319738520	300	g00089132	Amazon	Cash	Y	21/10/23	320197	g00085775

After

	BILL_NO	AMOUNT_OF_BILL	PURCHASED_BY	MERCHANT_NAME	PAYMENT_METHOD	REIMBURSED	PAYMENT_DATE	BUDGET_NO	HANDLED_BY
1	CFF2004313201	97.5	g00089132	Noon	Card	Y	21/09/20	859043	b00084833
2	PALOALTF207101322	150	g00089132	Amazon	Card	Y	28/09/20	295710	g00085775
3	MNF205021324	50	g00089132	Amazon	Card	Y	14/10/20	876502	g00085775
4	AMNGUSF207931325	150.75	g00089132	Amazon	Card	Y	14/10/20	140793	b00084833
5	ONLSECS216281327	78.2	g00089132	Talabat	Card	Y	08/03/21	493628	g00085775
6	IOTS215171328	100	g00089132	Amazon	Card	Y	14/04/21	926517	b00084833
7	ONWOSCS219843859	100	b00088385	Amazon	Card	Y	05/05/21	731984	g00085775
8	CFF2176838510	100	b00088385	Amazon	Card	Y	20/09/21	251768	b00084833
9	TABOOS2221938513	221	b00090070	Amazon	Cash	Y	03/03/22	504219	b00084833
10	CTFS2207638514	94.3	b00090070	Amazon	Cash	Y	24/03/22	821076	g00085775
11	GITHUBS2239713215	99.2	g00089132	Amazon	Cash	Y	13/04/22	165397	b00084833
12	WEB3.0F2226313216	75	g00089132	Amazon	Cash	Y	11/10/22	708263	b00084833
13	CARNF2215013221	600	g00089132	Amazon	Cash	Y	20/10/22	205150	b00084833
14	LINUXF2250113217	458	g00089132	Amazon	Cash	Y	29/10/22	437501	g00085775
15	CFS2363813218	100	g00089132	Amazon	Cash	Y	14/02/23	954638	b00084833
16	CTF2.0S2319713219	150	g00089132	Talabat	Cash	Y	21/10/23	320197	g00085775
17	CTF2.0S2319738520	300	g00089132	Amazon	Cash	Y	21/10/23	320197	g00085775

After

TRIGGERS: Nabeel

In the budget table there is an attribute called budget status which shows if the budget for an event has been approved or not, this trigger is designed to reject budgets if the budget amount used is greater than the amount allocated by the OSC for a particular event

-- Trigger to update the status of the budget. The budget is rejected if the amount used is greater than the amount allocated by the club.

```
CREATE OR REPLACE TRIGGER UPDATE_BUDGET_STATUS
AFTER INSERT OR UPDATE OF BUDGET_USED ON BUDGET
BEGIN
   UPDATE BUDGET
   SET BUDGET_STATUS = 'REJECTED'
   WHERE BUDGET_USED > AMOUNT_OF_BUDGET ;
END;
/

-- UPDATING A SPECIFIC BUDGET_USED ATTRIBUTE FOR A BUDGET_ID
UPDATE BUDGET
SET BUDGET_USED = 200
WHERE BUDGET_ID = '731984';

SELECT *
FROM BUDGET;
```

We can check if the trigger works by updating the budget_used column and increasing it past the allocated amount.

TRIGGERS: Nabeel

	BUDGET_ID	EVENT_ID	AMOUNT_OF_BUDGET	BUDGET_STATUS	BUDGET_USED	BUDGET_LEFT	ACADEMIC_TERM
1	859043	CFF20	100	Approved	97.5	2.5	Fall 2020
2	295710	PALOALTF20	150	Approved	150	0	Fall 2020
3	876502	MNF20	50	Approved	50	0	Fall 2020
4	140793	AMNGUSF20	100	Approved	100.75	-0.75	Fall 2020
5	572819	AWSF20	250	Approved	0	250	Fall 2020
6	493628	ONLSECS21	100	Approved	78.2	21.8	Spring 2021
7	926517	IOTS21	100	Approved	100	0	Spring 2021
8	731984	ONWOSCS21	100	Approved	100	0	Spring 2021
9	251768	CFF21	100	Approved	100	0	Fall 2021
10	504219	TABOOS22	200	Approved	221	-21	Spring 2022
11	821076	CTFS22	100	Approved	94.3	5.7	Spring 2022
12	165397	GITHUBS22	100	Approved	99.2	0.8	Spring 2022
13	708263	WEB3.0F22	100	Approved	75	25	Fall 2022
14	205150	CARNF22	600	Approved	600	0	Fall 2022
15	437501	LINUXF22	500	Approved	458	42	Fall 2022
16	84483	PALOALTF20	1000	Approved	500	50	Spring 2023
17	954638	CFS23	100	Approved	100	0	Spring 2023
18	320197	CTF2.0S23	500	Approved	450	50	Spring 2023

Before creating the trigger all statuses show as approved regarless of the budget used

TRIGGERS: Nabeel

umns | Data | Model | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Sort.. | Filter:

	BUDGET_ID	EVENT_ID	AMOUNT_OF_BUDGET	BUDGET_STATUS	BUDGET_USED	BUDGET_LEFT	ACADEMIC_TERM
1	859043	CFF20	100	Approved	97.5	2.5	Fall 2020
2	295710	PALOALTF20	150	Approved	150	0	Fall 2020
3	876502	MNF20	50	Approved	50	0	Fall 2020
4	140793	AMNGUSF20	100	Rejected	100.75	-0.75	Fall 2020
5	572819	AWSF20	250	Approved	0	250	Fall 2020
6	493628	ONLSECS21	100	Approved	78.2	21.8	Spring 2021
7	926517	IOTS21	100	Approved	100	0	Spring 2021
8	731984	ONWOSCS21	100	Rejected	200	0	Spring 2021
9	251768	CFF21	100	Approved	100	0	Fall 2021
10	504219	TABOOS22	200	Rejected	221	-21	Spring 2022
11	821076	CTFS22	100	Approved	94.3	5.7	Spring 2022
12	165397	GITHUBS22	100	Approved	99.2	0.8	Spring 2022
13	708263	WEB3.0F22	100	Approved	75	25	Fall 2022
14	205150	CARNF22	600	Approved	600	0	Fall 2022
15	437501	LINUXF22	500	Approved	458	42	Fall 2022
16	84483	PALOALTF20	1000	Approved	500	50	Spring 2023
17	954638	CFS23	100	Approved	100	0	Spring 2023
18	320197	CTF2.0S23	500	Approved	450	50	Spring 2023

After creating the trigger statuses show as approved or rejected depending on the budget used. If the budget used is higher than allocated it is regected.

TRIGGERS: Bhavika

The media_team wants to find out the Likes to Views Ratio for the media content that they post. To help them address this opportunity, we have created a trigger to automatically calculate the ratio between the likes and views of each media content created and posted by OSC.

First, we altered the Media_Content table and added a new attribute (column) called Like_View_Ratio. Then we create a trigger, that upon the insert or updation of any row in the media_content table calculated the ratio between the No_Of_Likes and the No_Of_Views. If the No_Of_Views is greater than 0, then the ratio is calculated by dividing the No_Of_Likes and No_Of_Views columns, otherwise, the ratio is calculated as 0.

To test out this trigger, we have inserted a new row in the Media_Content table. As seen in the output, the Like_View_Ratio is automatically calculated for this row.

We have also tried to update a row (where Content_ID = 'TTS23-01), and upon updating the row, the Like_View_Rtio was also automatically calculated and updated for this row.

Lastly, to maintain uniformity, we have also updated the Like_View_Ratio of the pre-existing rows, by setting them as No_Of_Likes / No_Of_Views. We did this to update the ratio in the pre-existing rows, but any new rows will be automatically updated with the calculated ratio due to the trigger.

```
ALTER TABLE MEDIA_CONTENT ADD (LIKE_VIEW_RATIO NUMBER(10, 2));

CREATE OR REPLACE TRIGGER TRG_CALCULATE_LIKE_VIEW_RATIO
BEFORE INSERT OR UPDATE ON MEDIA_CONTENT
FOR EACH ROW
BEGIN
   IF :NEW.NO_OF_VIEWS > 0 THEN
      :NEW.LIKE_VIEW_RATIO := :NEW.NO_OF_LIKES / :NEW.NO_OF_VIEWS;
   ELSE
      :NEW.LIKE_VIEW_RATIO := 0;
   END IF;
END;
/

INSERT INTO MEDIA_CONTENT (CONTENT_ID, CREATOR_AUS_ID, EVENT_ID, FORMAT,
DATEPUBLISHED, NO_OF_LIKES, NO_OF_VIEWS)
VALUES ('TT-S23-04', 'G00085270', NULL, 'REEL','7-APR-2023', 1000, 5000);

UPDATE MEDIA_CONTENT
SET NO_OF_LIKES = 150, NO_OF_VIEWS = 6000
WHERE CONTENT_ID = 'TTS23-01';


UPDATE MEDIA_CONTENT
SET LIKE_VIEW_RATIO = CASE
   WHEN NO_OF_VIEWS = 0 THEN 0
   ELSE NO_OF_LIKES / NO_OF_VIEWS
  END;
```

TRIGGERS: Bhavika

To further interpret the Likes to Views ratio, we can assume that a ratio closer to 1 shows that most viewers of the media content are liking the media_content.

If the ratio is further away from 1, shows that most of the viewers are not liking the media_content.

For example, the Likes to Views ratio for TTS23-04 is 0.2. Thus, we can conclude, that only 20% of the viewers of that media_content are liking the media_content reel as well.

Every Media_Content created by OSC has a Likes to Views ratio below 1. This implies that all of OSC Media_Content gets more views than likes. People prefer to view media_content without liking it. OSC should try to get the Likes to Views ratio above or close to 1, as it shows good engagement for the media (content) of the club.

TRIGGERS: Bhavika

	CONTENT_ID	CREATOR_AUS_ID	EVENT_ID	FORMAT	DATEPUBLISHED	NO_OF_LIKES	NO_OF_VIEWS	LIKE_VIEW_RATIO
1	CFF20-01	g00090589	CFF20	Post	22/09/20	100	172	(null)
2	PALOALTF20-01	b00093124	PALOALTF20	Post	28/09/20	27	500	(null)
3	NNF20-01	g00085270	NNF20	Post	06/10/20	31	200	(null)
4	MNF20-01	g00088567	MNF20	Post	14/10/20	50	150	(null)
5	AMNGUSF20-01	b00086276	AMNGUSF20	Post	26/10/20	46	100	(null)
6	AWSF20-01	g00088575	AWSF20	Post	27/11/20	34	650	(null)
7	ONLSECS21-01	b00091171	ONLSECS21	Post	07/03/21	47	1500	(null)
8	IOTS21-01	b00091171	IOTS21	Post	13/04/21	70	700	(null)
9	ONWOSCS21-01	g00094956	ONWOSCS21	Video	04/05/21	44	219	(null)
10	CFF21-01	b00094999	CFF21	Video	19/09/21	40	210	(null)
11	CTBTWPYF21-01	b00094999	CTBTWPYF21	Video	12/10/21	33	210	(null)
12	MUSEDITF21-01	b00088589	MUSEDITF21	Post	15/11/21	39	210	(null)
13	TABOOS22-01	g00090589	TABOOS22	Post	02/02/22	37	88	(null)
14	TABOOS22-02	b00093124	TABOOS22	Reel	27/02/22	35	1566	(null)
15	CTFS22-00	g00085270	CTFS22	Reel	22/03/22	33	700	(null)
16	CTFS22-01	g00088567	CTFS22	Reel	23/03/22	31	219	(null)
17	CTFS22-02	b00086276	CTFS22	Reel	31/03/22	100	210	(null)
18	GITHUBS22-01	g00088575	GITHUBS22	Reel	12/04/22	27	210	(null)
19	WEB3.0F22-01	b00091171	WEB3.0F22	Reel	08/10/22	31	210	(null)
20	CARNF22-01	g00085270	CARNF22	Reel	18/10/22	100	600	(null)
21	CARNF22-02	b00093124	CARNF22	Reel	21/10/22	300	1200	(null)
22	LINUXF22-01	g00094956	LINUXF22	Reel	04/12/22	50	88	(null)
23	CFS23-01	g00088575	CFS23	Reel	12/12/22	46	999	(null)
24	CTF2.0S23-00	b00094999	CTF2.0S23	Reel	14/03/23	50	2067	(null)
25	CTF2.0S23-01	g00094956	CTF2.0S23	Reel	19/03/23	46	988	(null)
26	CTF2.0S23-02	b00088589	CTF2.0S23	Reel	30/03/23	100	700	(null)
27	TTS23-01	g00090589	(null)	Reel	08/03/23	55	3000	(null)
28	TTS23-02	b00093124	(null)	Reel	22/03/23	31	2100	(null)

Before creating the trigger - Adding a new column called Like_View Ratio

	CONTENT_ID	CREATOR_AUS_ID	EVENT_ID	FORMAT	DATEPUBLISHED	NO_OF_LIKES	NO_OF_VIEWS	LIKE_VIEW_RATIO
1	TT-S23-04	g00085270	(null)	Reel	07/04/23	1000	5000	0.2
2	CFF20-01	g00090589	CFF20	Post	22/09/20	100	172	(null)
3	PALOALTF20-01	b00093124	PALOALTF20	Post	28/09/20	27	500	(null)
4	NNF20-01	g00085270	NNF20	Post	06/10/20	31	200	(null)
5	MNF20-01	g00088567	MNF20	Post	14/10/20	50	150	(null)
6	AMNGUSF20-01	b00086276	AMNGUSF20	Post	26/10/20	46	100	(null)
7	AWSF20-01	g00088575	AWSF20	Post	27/11/20	34	650	(null)
8	ONLSECS21-01	b00091171	ONLSECS21	Post	07/03/21	47	1500	(null)
9	IOTS21-01	b00091171	IOTS21	Post	13/04/21	70	700	(null)
10	ONWOSCS21-01	g00094956	ONWOSCS21	Video	04/05/21	44	219	(null)
11	CFF21-01	b00094999	CFF21	Video	19/09/21	40	210	(null)
12	CTBTWPYF21-01	b00094999	CTBTWPYF21	Video	12/10/21	33	210	(null)
13	MUSEDITF21-01	b00088589	MUSEDITF21	Post	15/11/21	39	210	(null)
14	TABOOS22-01	g00090589	TABOOS22	Post	02/02/22	37	88	(null)
15	TABOOS22-02	b00093124	TABOOS22	Reel	27/02/22	35	1566	(null)
16	CTFS22-00	g00085270	CTFS22	Reel	22/03/22	33	700	(null)
17	CTFS22-01	g00088567	CTFS22	Reel	23/03/22	31	219	(null)
18	CTFS22-02	b00086276	CTFS22	Reel	31/03/22	100	210	(null)
19	GITHUBS22-01	g00088575	GITHUBS22	Reel	12/04/22	27	210	(null)
20	WEB3.0F22-01	b00091171	WEB3.0F22	Reel	08/10/22	31	210	(null)
21	CARNF22-01	g00085270	CARNF22	Reel	18/10/22	100	600	(null)
22	CARNF22-02	b00093124	CARNF22	Reel	21/10/22	300	1200	(null)
23	LINUXF22-01	g00094956	LINUXF22	Reel	04/12/22	50	88	(null)
24	CFS23-01	g00088575	CFS23	Reel	12/12/22	46	999	(null)
25	CTF2.0S23-00	b00094999	CTF2.0S23	Reel	14/03/23	50	2067	(null)
26	CTF2.0S23-01	g00094956	CTF2.0S23	Reel	19/03/23	46	988	(null)
27	CTF2.0S23-02	b00088589	CTF2.0S23	Reel	30/03/23	100	700	(null)
28	TTS23-01	g00090589	(null)	Reel	08/03/23	55	3000	(null)

Creating the trigger, and checking if the trigger calculates the Like_View_Ratio by inserting a new row

TRIGGERS: Bhavika

	CONTENT_ID	CREATOR_AUS_ID	EVENT_ID	FORMAT	DATEPUBLISHED	NO_OF_LIKES	NO_OF_VIEWS	LIKE_VIEW_RATIO
3	NNF20-01	g00085270	NNF20	Post	06/10/20	31	200	(null)
4	MNF20-01	g00088567	MNF20	Post	14/10/20	50	150	(null)
5	AMNGUSF20-01	b00086276	AMNGUSF20	Post	26/10/20	46	100	(null)
6	AWSF20-01	g00088575	AWSF20	Post	27/11/20	34	650	(null)
7	ONLSECS21-01	b00091171	ONLSECS21	Post	07/03/21	47	1500	(null)
8	IOTS21-01	b00091171	IOTS21	Post	13/04/21	70	700	(null)
9	ONWOSCS21-01	g00094956	ONWOSCS21	Video	04/05/21	44	219	(null)
10	CFF21-01	b00094999	CFF21	Video	19/09/21	40	210	(null)
11	CTBTWPYF21-01	b00094999	CTBTWPYF21	Video	12/10/21	33	210	(null)
12	MUSEDITF21-01	b00088589	MUSEDITF21	Post	15/11/21	39	210	(null)
13	TABOOS22-01	g00090589	TABOOS22	Post	02/02/22	37	88	(null)
14	TABOOS22-02	b00093124	TABOOS22	Reel	27/02/22	35	1566	(null)
15	CTFS22-00	g00085270	CTFS22	Reel	22/03/22	33	700	(null)
16	CTFS22-01	g00088567	CTFS22	Reel	23/03/22	31	219	(null)
17	CTFS22-02	b00086276	CTFS22	Reel	31/03/22	100	210	(null)
18	GITHUBS22-01	g00088575	GITHUBS22	Reel	12/04/22	27	210	(null)
19	WEB3.0F22-01	b00091171	WEB3.0F22	Reel	08/10/22	31	210	(null)
20	CARNF22-01	g00085270	CARNF22	Reel	18/10/22	100	600	(null)
21	CARNF22-02	b00093124	CARNF22	Reel	21/10/22	300	1200	(null)
22	LINUXF22-01	g00094956	LINUXF22	Reel	04/12/22	50	88	(null)
23	CFS23-01	g00088575	CFS23	Reel	12/12/22	46	999	(null)
24	CTF2.0S23-00	b00094999	CTF2.0S23	Reel	14/03/23	50	2067	(null)
25	CTF2.0S23-01	g00094956	CTF2.0S23	Reel	19/03/23	46	988	(null)
26	CTF2.0S23-02	b00088589	CTF2.0S23	Reel	30/03/23	100	700	(null)
27	TTS23-01	g00090589	(null)	Reel	08/03/23	150	6000	0.03
28	TTS23-02	b00093124	(null)	Reel	22/03/23	31	2100	(null)
29	TTS23-03	g00085270	(null)	Reel	05/04/23	52	5699	(null)
30	TT-S23-04	g00085270	(null)	Reel	07/04/23	1000	5000	0.2

Checking if the trigger calculates the Like_View_Ratio by updating a previously existing row

TRIGGERS: Bhavika

	CONTENT_ID	CREATOR_AUS_ID	EVENT_ID	FORMAT	DATEPUBLISHED	NO_OF_LIKES	NO_OF_VIEWS	LIKE_VIEW_RATIO
1	CFF20-01	g00090589	CFF20	Post	22/09/20	100	172	0.58
2	PALOALTF20-01	b00093124	PALOALTF20	Post	28/09/20	27	500	0.05
3	NNF20-01	g00085270	NNF20	Post	06/10/20	31	200	0.16
4	MNF20-01	g00088567	MNF20	Post	14/10/20	50	150	0.33
5	AMNGUSF20-01	b00086276	AMNGUSF20	Post	26/10/20	46	100	0.46
6	AWSF20-01	g00088575	AWSF20	Post	27/11/20	34	650	0.05
7	ONLSECS21-01	b00091171	ONLSECS21	Post	07/03/21	47	1500	0.03
8	IOTS21-01	b00091171	IOTS21	Post	13/04/21	70	700	0.1
9	ONWOSCS21-01	g00094956	ONWOSCS21	Video	04/05/21	44	219	0.2
10	CFF21-01	b00094999	CFF21	Video	19/09/21	40	210	0.19
11	CTBTWPYF21-01	b00094999	CTBTWPYF21	Video	12/10/21	33	210	0.16
12	MUSEDITF21-01	b00088589	MUSEDITF21	Post	15/11/21	39	210	0.19
13	TABOOS22-01	g00090589	TABOOS22	Post	02/02/22	37	88	0.42
14	TABOOS22-02	b00093124	TABOOS22	Reel	27/02/22	35	1566	0.02
15	CTFS22-00	g00085270	CTFS22	Reel	22/03/22	33	700	0.05
16	CTFS22-01	g00088567	CTFS22	Reel	23/03/22	31	219	0.14
17	CTFS22-02	b00086276	CTFS22	Reel	31/03/22	100	210	0.48
18	GITHUBS22-01	g00088575	GITHUBS22	Reel	12/04/22	27	210	0.13
19	WEB3.0F22-01	b00091171	WEB3.0F22	Reel	08/10/22	31	210	0.15
20	CARNF22-01	g00085270	CARNF22	Reel	18/10/22	100	600	0.17
21	CARNF22-02	b00093124	CARNF22	Reel	21/10/22	300	1200	0.25
22	LINUXF22-01	g00094956	LINUXF22	Reel	04/12/22	50	88	0.57
23	CFS23-01	g00088575	CFS23	Reel	12/12/22	46	999	0.05
24	CTF2.0S23-00	b00094999	CTF2.0S23	Reel	14/03/23	50	2067	0.02
25	CTF2.0S23-01	g00094956	CTF2.0S23	Reel	19/03/23	46	988	0.05
26	CTF2.0S23-02	b00088589	CTF2.0S23	Reel	30/03/23	100	700	0.14
27	TTS23-01	g00090589	(null)	Reel	08/03/23	150	6000	0.03
28	TTS23-02	b00093124	(null)	Reel	22/03/23	31	2100	0.01

Updating the Like_View_Ratio of pre-existing rows, to maintain uniformity

TRIGGERS: Bhavika

As previously mentioned, OSC is going to recruit new team members in the upcoming semesters. They wish to recruit team members for their executive, activities, and media team.

Previously, if a member is recruited, they had to be manually added to the member table as well as the respective team table that they belong to. This can be a tedious process. To solve this issue, we have created a trigger, that upon the insertion of a new row in the member table, duplicates the same row and inserts it into the respective team table. If we specify that the new member belongs to the Activities Team, then the same member will also be added to the Activities_Team table. The specific team tables have some more additional attributes as compared to the Member table, such as Reports_To, Activities_Team_Position, Media_Team_Position, and so on. In our initial code, these attributes cannot have NULL values, thus we have placed placeholders in those columns, which can be manually edited later. In the Executive_Team table, the Bank_Account_Number and Reports_To can be NULL, thus we have let those additional values remain as NULL when the member is added from the member table to the Executive_Team table.

If the team of the new member is a part of the 'Activities Team', they will be added to the Activities_Team table. If the team of the new member is a part of the 'Media Team', the member will be added to the Media_Team table. Similarly, if the new member is a part of the 'Executive Team', they will be added to the Executive_Team table.

To test this trigger, we have inserted 3 new rows. Firstly, we have inserted John Doe, a member that belongs to the Activities Team. As seen in the output, John is (simultaneously) added to the member table as well as the Activities_Team table.

Secondly, we have added Blanket Set, a member belonging to the Media Team. As seen in the output, Blanket is (simultaneously) added to the member table as well as the Media_Team table.

Lastly, we have added Gorgina Ilk, a member belonging to the Executive Team. As seen in the output, Gorgina is (simultaneously) added to the member table as well as the Executive_Team table.

OSC wishes to add new recruits to the members and specific team table, simultaneously (through a single insert function). Thus (to solve this problem), this trigger only works when new members are recruited into OSC and inserted (added) to the member table (database). This trigger would not be activated if the team of the current members are updated.

While trying this trigger, we have used the previously built procedure as well. We have created a procedure to help OSC add members to the member table. While testing the trigger, we used the procedure to insert values in the member table.

TRIGGERS: Bhavika

```
CREATE OR REPLACE TRIGGER TRG_ADD_MEMBER_TO_TEAM
AFTER INSERT ON MEMBER
FOR EACH ROW
BEGIN
  IF :NEW.TEAM = 'ACTIVITIES TEAM' THEN
    INSERT INTO ACTIVITIES_TEAM (AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL,
TEAM, ACTIVITIES_TEAM_POSITION, TECH_SKILL_LEVEL, REPORTS_TO)
    VALUES (:NEW.AUS_ID, :NEW.NAME, :NEW.PRIMARY_MAJOR, :NEW.STANDING,
:NEW.EMAIL, :NEW.TEAM, 'POSITION', 1, 'REPORTS TO');
  ELSIF :NEW.TEAM = 'MEDIA TEAM' THEN
    INSERT INTO MEDIA_TEAM (AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM,
MEDIA_TEAM_POSITION, MEDIA_SKILL_LEVEL, REPORTS_TO)
    VALUES (:NEW.AUS_ID, :NEW.NAME, :NEW.PRIMARY_MAJOR, :NEW.STANDING,
:NEW.EMAIL, :NEW.TEAM, 'POSITION', 1, 'REPORTS TO');
  ELSIF :NEW.TEAM = 'EXECUTIVE TEAM' THEN
    INSERT INTO EXECUTIVE_TEAM (AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL,
TEAM, EXECUTIVE_TEAM_POSITION, BANK_ACCOUNT_NUMBER, REPORTS_TO)
    VALUES (:NEW.AUS_ID, :NEW.NAME, :NEW.PRIMARY_MAJOR, :NEW.STANDING,
:NEW.EMAIL, :NEW.TEAM, 'POSITION', NULL, NULL);
  END IF;
END;

EXEC INSERTMEMBER ('G00089133', 'JOHN DOE', 'COMPUTER SCIENCE', 'JUNIOR 2',
'G00089133@AUS.EDU', 'ACTIVITIES TEAM');

INSERT INTO MEMBER (AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM)
VALUES ('G00089138', 'BLANKET SET', 'COMPUTER SCIENCE', 'JUNIOR 2',
'G00089138@AUS.EDU', 'MEDIA TEAM');

INSERT INTO MEMBER (AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM)
VALUES ('G00166666', 'GORGINA ILK', 'DESIGN MANAGEMENT', 'JUNIOR 1',
'G00166666@AUS.EDU', 'EXECUTIVE TEAM');

SELECT * FROM MEDIA_TEAM;

SELECT * FROM MEDIA_TEAM;

SELECT * FROM EXECUTIVE_TEAM;
```

TRIGGERS: Bhavika

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team
5	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team
6	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team
7	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team
8	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team
9	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team
10	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team
11	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team
12	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team
13	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team
14	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team
15	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team
16	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team
17	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team
18	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team
19	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team
20	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team
21	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team
22	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team
23	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team
24	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team
25	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team
26	b00095219	Neeraj Vohra	Accounting	Junior 1	b00095219@aus.edu	Activities Team

The initial Member table before creating the procedure. Note: 'Neeraj Vohra' is not assigned into a particular team table as his record was added to the member table, before the trigger was created (activated).

TRIGGERS: Bhavika

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team
5	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team
6	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team
7	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team
8	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team
9	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team
10	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team
11	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team
12	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team
13	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team
14	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team
15	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team
16	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team
17	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team
18	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team
19	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team
20	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team
21	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team
22	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team
23	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team
24	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team
25	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team
26	b00095219	Neeraj Vohra	Accounting	Junior 1	b00095219@aus.edu	Activities Team
27	g00089133	John Doe	Computer Science	Junior 2	g00089133@aus.edu	Activities Team
28	g00166666	Gorgina Ilk	Design Management	Junior 1	g00166666@aus.edu	Executive Team
29	g00089138	Blanket Set	Computer Science	Junior 2	g00089138@aus.edu	Media Team

The updated Member table after inserting the new rows

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	ACTIVITIES_TEAM_POSITION	TECH_SKILL_LEVEL	REPORTS_TO
1	b00090070	Rishi Chugh	Management Information Systems	Junior 2	b00090070@aus.edu	Activities Team	Head Of Activities	7	President
2	g00089132	Bhavika Vohra	Accounting	Junior 2	g00089132@aus.edu	Activities Team	Public Relations Coordinator	5	President
3	b00088385	Utkarsh Chauhan	Mechanical Engineering	Junior 2	b00088385@aus.edu	Activities Team	Research Coordinator	6	President
4	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team	Activities Officer	2	Head Of Activiti
5	b00092301	Taufiq Syed	Computer Science	Sophmore 2	b00092301@aus.edu	Activities Team	Activities Officer	10	Head Of Activiti
6	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team	Activities Officer	8	Head Of Activiti
7	b00089432	Minhaz Basith	Mathematics	Junior 2	b00089432@aus.edu	Activities Team	Activities Officer	5	Head Of Activiti
8	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team	Activities Officer	9	Head Of Activiti
9	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team	Activities Officer	10	Head Of Activiti
10	b00093718	Chirag Khanchandani	Finance	Sophmore 2	b00093718@aus.edu	Activities Team	Activities Officer	3	Head Of Activiti
11	g00089133	John Doe	Computer Science	Junior 2	g00089133@aus.edu	Activities Team	Position	1	Reports To

The updated Activities_Team table after inserting 'John Doe'

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	MEDIA_TEAM_POSITION	MEDIA_SKILL_LEVEL	REPORTS_TO
1	g00090589	Zunaira Farooq	Computer Engineering	Sophmore 2	g00090589@aus.edu	Media Team	Head of Media	10	President
2	b00093124	Mohamed Raiyan	Computer Science	Sophmore 2	b00093124@aus.edu	Media Team	Media Coordinator	7	Head of Media
3	g00085270	Hannan Arshad	Visual Communication	Junior 2	g00085270@aus.edu	Media Team	Media Coordinator	8	Head of Media
4	g00088567	Rhea Maria Jacob	Electrical Engineering	Junior 2	g00088567@aus.edu	Media Team	Media Coordinator	7	Head of Media
5	b00086276	Ajay Sunil	Architecture	Junior 2	b00086276@aus.edu	Media Team	Media Coordinator	10	Head of Media
6	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team	Media Coordinator	6	Head of Media
7	b00091171	Adithya Krishnakumar	Computer Science	Sophmore 2	b00091171@aus.edu	Media Team	Media Coordinator	7	Head of Media
8	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team	Media Coordinator	9	Head of Media
9	b00094999	Madhav Vohra	Accounting	Freshman 1	b00094999@aus.edu	Media Team	Media Coordinator	6	Head of Media
10	g00087588	Pritika Tilokani	Marketing	Junior 2	g00087588@aus.edu	Media Team	Media Coordinator	7	Head of Media
11	b00088589	Arpit Jain	Finance	Junior 2	b00088589@aus.edu	Media Team	Media Coordinator	2	Head of Media
12	g00089138	Blanket Set	Computer Science	Junior 2	g00089138@aus.edu	Media Team	Position	1	Reports To

The updated Media_Team table after inserting 'Blanket Set'

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	EXECUTIVE_TEAM_POSITION	BANK_ACCOUNT_NUMBER	REPORTS_TO
1	b00084833	Prem Rajendran	Computer Science	Senior 2	b00084833@aus.edu	Executive Team	President	AE867280963400758960283	(null)
2	b00088138	Ahmed Sharafath	Computer Engineering	Junior 2	b00088138@aus.edu	Executive Team	Vice-President	(null)	President
3	g00085775	Chavi Mehta	Finance	Senior 2	g00085775@aus.edu	Executive Team	Treasurer	AE636789189562026613410	President
4	g00087337	Snikita Moka	Architecture	Junior 2	g00087337@aus.edu	Executive Team	Executive Assistant	(null)	President
5	g00166666	Gorgina Ilk	Design Management	Junior 1	g00166666@aus.edu	Executive Team	Position	(null)	(null)

The updated Executive_Team table after inserting 'Georgina Ilk'

PART 3
USAGE

BUSINESS QUERIES: Bhavika

1) As the current President is graduating, he needs to appoint a new president. The new President should be an Activities Team member and has to have managed a minimum of 3 events. Out of these candidates, the member with the highest attendance average at the events that they have managed will be chosen.

```
SELECT E.EVENT_MANAGER, A.NAME, AVG(E.ATTENDANCE) AS AVG_ATTENDANCE, COUNT(E.EVENT_ID) AS EVENTS_MANAGED
FROM EVENT E JOIN ACTIVITIES_TEAM A ON E.EVENT_MANAGER=A.AUS_ID
GROUP BY E.EVENT_MANAGER, A.NAME
HAVING COUNT(E.EVENT_ID)>2
ORDER BY AVG_ATTENDANCE DESC;
```

Here, we have selected the ID of the student from the event table, the name of the student from the activities team table, the number of events they have managed from the events table, and the average attendance at the events that they have managed from the events table.

We have joined the events table and activities team table on the event_manager and aus_id columns, as these are both the AUS ID of the students in the activities team. We have put a condition that we only showcase those students who have managed a minimum of 3 events. As some of the students have managed more than 1 event, we grouped them by their name and ID. Lastly, we have also filtered the results in descending order of average attendance, to see which member has the highest average attendance at the events they manage.

Through the results, we can conclude that Taufiq Syed should be the next President of OSC, as he has managed 4 events (more than 3) and has the highest average attendance of 98 students.

For further interpretation, we can interpret that Aadhith Shankarnarayanan has the lowest average attendance rate of 22.5 students (22 students), amongst all those individuals who have managed a minimum of 3 events.

We could have performed this query without the use of a join, as the event table has an attribute called event_manager, but we have joined the events table and the activities team table, as we wanted to display the name of the activities team member as well.

	EVENT_MANAGER	NAME	AVG_ATTENDANCE	EVENTS_MANAGED
1	b00092301	Taufiq Syed	98	4
2	g00089132	Bhavika Vohra	93	4
3	b00087520	Koushal Parupudi	72.333333333333333333333333333333333333	3
4	b00090070	Rishi Chugh	68.333333333333333333333333333333333333	3
5	b00089801	Aadhith Shankarnarayanan	22.5	3

BUSINESS QUERIES: Bhavika

We can also use a subquery to perform the same query :

```
SELECT ATM.AUS_ID, ATM.NAME, COUNT(E.EVENT_ID) AS EVENTS_MANAGED,
AVG(E.ATTENDANCE) AS AVG_ATTENDANCE
FROM ACTIVITIES_TEAM ATM
FULL OUTER JOIN EVENT E ON E.EVENT_MANAGER = ATM.AUS_ID
WHERE ATM.AUS_ID IN (
 SELECT EVENT_MANAGER
 FROM EVENT
 GROUP BY EVENT_MANAGER
 HAVING COUNT(*) > 2
)
GROUP BY ATM.AUS_ID, ATM.NAME
ORDER BY AVG(E.ATTENDANCE) DESC;
```

BUSINESS QUERIES: Bhavika

2) As the current Head_Of_Media is leaving, she needs to appoint a new Head_Of_Media. The new Head_Of_Media should be a Media Team member and has to have created a minimum of 4 media content. Out of these candidates, the member with the highest attendance likes on the media content that they have created will be chosen.

```
SELECT C.CREATOR_AUS_ID, M.NAME, AVG(C.NO_OF_LIKES) AS AVG_LIKES,
COUNT(C.CONTENT_ID) AS CONTENT_CREATED
FROM MEDIA_CONTENT C JOIN MEDIA_TEAM M ON C.CREATOR_AUS_ID=M.AUS_ID
GROUP BY C.CREATOR_AUS_ID, M.NAME
HAVING COUNT(C.CONTENT_ID)>3
ORDER BY AVG_LIKES DESC;
```

Here, we have selected the ID of the student from the media content table, the name of the student from the media_team table, the number of media content they have created from the media content table, and the average likes of the media content they have created from the media content table. We have joined the media content table and media team table on the creator_ausid and aus_id columns, as these are both the AUS ID of the students in the media team. We have put a condition that we only showcase those students who have created a minimum of 4 media content. As some of the students have created more than 1 media content, we grouped them by their name and ID. Lastly, we have also filtered the results in descending order of average likes, to see which member has the highest average likes on the media content they create.

Through the results, we can conclude that Mohamed Raiyan should be the next Head_Of_Media at OSC, as he has created 4 media content and has the highest average likes of 98.25 likes.

For further interpretation, we can interpret that Hannan Arshad, is the only other student in the media team, who has created minimum 4 media content, and she has an average like count of 54 likes.

We could have performed this query without the use of a join, as the media_content table has an attribute called creator_aus_id, but we have joined the media team table and the media content table, as we wanted to display the name of the media team member as well.

	CREATOR_AUS_ID	NAME	AVG_LIKES	CONTENT_CREATED
1	b00093124	Mohamed Raiyan	98.25	4
2	g00085270	Hannan Arshad	54	4

BUSINESS QUERIES: Bhavika

3) OSC wants to find out the collaborators who have collaborated in more than one event. OSC wishes to appreciate these collaborators.

```
SELECT COLLABORATORS.EMIRATES_ID, COLLABORATORS.NAME_OF_COLLABORATOR,COLLABORATORS.ORGANIZATION, COLLABORATORS.COLLABORATOR_TYPE, COUNT(EVENT_COLLABORATED.EVENT_ID) AS NUM_OF_EVENTS_COLLABORATED_ON FROM COLLABORATORS
INNER JOIN EVENT_COLLABORATED ON COLLABORATORS.EMIRATES_ID = EVENT_COLLABORATED.EMIRATES_ID
GROUP BY COLLABORATORS.EMIRATES_ID, COLLABORATORS.NAME_OF_COLLABORATOR, COLLABORATORS.ORGANIZATION, COLLABORATORS.COLLABORATOR_TYPE
HAVING COUNT(EVENT_COLLABORATED.EVENT_ID) > 1;
```

Here, we have selected the collaborators' Emirates ID, collaborators' name, collaborators' organization, and collaborator type from the collaborators' table. We have also selected the count of the number of events the collaborator has taken part in. We have inner-joined the event_collaborated table and collaborators table on the Emirates ID of the collaborators (Collaborators.Emirates_ID = Event_Collaborated.Emirates_ID), as we want to showcase the name and collaborator type of the collaborator. We have put the condition to showcase only those collaborators who have collaborated with OSC for more than 1 event. We have grouped the results by Emirates_ID, Name_of_Collaborator, Organization, and Collaborator_Type from the Collaborators table.

Through the results, we can conclude that there is only one collaborator Meriam Mkadmi, from IEECS - AUS Chapter (internal to AUS), and her organization has collaborated with the OSC for 2 events. OSC wishes to appreciate Meriam as well as her organization IEEECS - AUS Chapter.

	EMIRATES_ID	NAME_OF_COLLABORATOR	ORGANIZATION	COLLABORATOR_TYPE	NUM_OF_EVENTS_COLLABORATED_ON
1	784-2001-5675567-1	Meriam Mkadmi	IEEECS - AUS Chapter	Internal	2

We want to further see the events IEECS - AUS chapter has collaborated on OSC and the attendance of these events, to see if collab events with IEEECS - AUS Chapter are doing well. To do so, we can do the following:

```
SELECT CE.EVENT_ID, CE.EVENT_NAME, CE. ATTENDANCE
FROM COLLABORATED_EVENTS CE
INNER JOIN EVENT_COLLABORATED EC ON CE.EVENT_ID = EC.EVENT_ID
WHERE EC.EMIRATES_ID = '784-2001-5675567-1';
```

	EVENT_ID	EVENT_NAME	ATTENDANCE
1	GITHUBS22	Introducing Github	113
2	PALOALTF20	Info Session with Palo Alto	102

As we can see, OSC has collaborated with IEEECS - AUS Chapter on 2 events, namely 'Introducing Github' and 'Info Session with Palo Alto' and these events had an attendance of 113 and 102 participants respectively.

BUSINESS QUERIES: Bhavika

4) The treasurer, Chavi Mehta has a few questions regarding the budget. She would like to know which events spent more than the allocated budget. She would like to know this, so she is able to understand how often, does OSC go over budget for events.

```
SELECT EVENT.EVENT_ID, EVENT.EVENT_NAME, BUDGET.AMOUNT_OF_BUDGET AS ALLOCATED_BUDGET, BUDGET.BUDGET_USED AS SPENT_AMOUNT, BUDGET.BUDGET_LEFT AS REMAINING_FUNDS
FROM EVENT
JOIN BUDGET ON EVENT.EVENT_ID = BUDGET.EVENT_ID
WHERE BUDGET_USED>AMOUNT_OF_BUDGET
ORDER BY SPENT_AMOUNT DESC;
```

Here, we have selected the Event_ID, Event_Name from the Event table, and the Amount_of_Budget, Budget_Used, Budget_Left from the Budget table. We have also joined the Budget and Event table, on the Event_ID, as Event_ID is a foreign key in the Budget table. We have put a condition to only display those events, where the budget used is greater than the amount of budget allocated, thereby displaying events only for those events where Budget_Used > Amount_of_Budget (they have overspent the allocated budget). We have sorted the results in descending order based on the SPENT_AMOUNT column (which is an alias for BUDGET_USED). This means that the events that have spent the most money over their allocated budgets will be listed first in the result set.

	EVENT_ID	EVENT_NAME	ALLOCATED_BUDGET	SPENT_AMOUNT	REMAINING_FUNDS
1	TABOOS22	Game Show - Taboo	200	221	-21
2	AMNGUSF20	Data Analysis Reported	100	100.75	-0.75

After running the query, we find out that OSC has gone over budget for 2 events (since it incorporated), namely Game Show - Taboo (by AED 21) and Data Analysis (by AED 0.75). However, Chavi concludes that since OSC has only gone over budget for 2 events till now, this is not a threat to the operations of the organization.

BUSINESS QUERIES: Bhavika

5) Chavi also wants to know how much budget was spent according to the different semesters such as Fall 2022, Spring 2023, and so on. To do so, we will help her perform a query:

```
SELECT ACADEMIC_TERM, SUM(BUDGET.AMOUNT_OF_BUDGET) AS ALLOCATED_AMOUNT, SUM(BUDGET_USED) AS TOTAL_SPENT
FROM BUDGET
GROUP BY ACADEMIC_TERM
ORDER BY TOTAL_SPENT DESC;
```

Here, we have selected Academic Term, the sum of the Amount_of_Budget as the Allocated_Amount, and the sum of the Budget_Used as the Total_Spent. We did not find the need to join any tables in this, as the budget table itself has academic term as an attribute, which we have used to group the findings by. We need to use the group by function, as some academic terms have more than 1 budget (which we need to sum). Ultimately, we have ordered the results in a descending manner according to the Total_Spent (sum of the budget_used).

	ACADEMIC_TERM	ALLOCATED_AMOUNT	TOTAL_SPENT
1	Fall 2022	1200	1133
2	Spring 2023	600	550
3	Spring 2022	400	414.5
4	Fall 2020	650	398.25
5	Spring 2021	300	278.2
6	Fall 2021	100	100

Upon looking at the results, we can infer that Fall 2022 had the highest budget spent, followed by Spring 2023, Spring 2022, Fall 2020, and Spring 2021. Fall 2021 had the lowest budget spent, as well as the lowest allocated amount of budget.

Upon analyzing, we can conclude that Fall 2022 had the highest budget spent of AED 1133, as they had events such as AUS Carnival, Linux, and Introduction to Web 3.0 where OSC spent amounts of AED 600, AED 458, and AED 75.

BUSINESS QUERIES: Bhavika

6) OSC wanted an idea repository to store event and media content ideas of team members. We created an Idea_Repository table and inserted a few media_content and event ideas.

```
SELECT *
FROM IDEA_REPOSITORY;
```

Here, we have selected all the rows from the Idea_Repository table, that we have newly created to store the media content and event ideas from the team members. This table has Idea_Number, Idea_Name, Idea_Description, Member_ID, Idea_Category, Estimated_Budget_Required, Status and SpecialResoucesNeeded.

If OSC wishes to have events/media content in the upcoming semesters, they can use the Idea_Repository table to think of ideas.

	IDEA_NUMBER	IDEA_NAME	IDEA_DESCRIPTION	MEMBER_ID	IDEACATEGORY	ESTIMATED_BUDGET_REQUIRED	STATUS	SPEC
1	001	OSCMovieNight2.0	A movie night with OSC to watch the movie Snowden	b00084833	EventIdea	100	Proposed	proje
2	002	GuesstheOSSoftware	A Instagram Story Series with OS software hints and polls to guess the OSS.	g00087337	MediaIdea	50	Proposed	(null
3	003	MeetTheTeam	Day in the Life Instagram story series with different members once a week.	g00087337	MediaIdea	0	Proposed	(null
4	004	TechnoBytes	Instagram Story series with OSS related news.	g00090589	MediaIdea	0	Proposed	(null
5	005	OSCDataScienceSeries	An event series about data science in different fields such as business, engineering, architecture.	g00089132	EventIdea	(null)	To be Executed	(null
6	006	WhoWantsToBeAMillionare?	An event wherein we play the game Who Wants to Be A Millionaire with questions about OSS.	b00090070	EventIdea	(null)	To be Executed	(null
7	007	Taboo2.0	A part 2 for the Taboo event.	b00088385	EventIdea	200	Proposed	custo
8	008	Aren'tWeOSome?	A Who's Most Likely to Instagram Story Series with OSC members.	g00089132	MediaIdea	0	To be Executed	OSC M
9	009	BoardGameNightwithOSC	A board game night with OSC, playing games like chess, UNO and so on.	g00089132	EventIdea	300	Proposed	board
10	011	OSCTechTalks	A series of talks about OSS in different fields.	g00089132	EventIdea	300	Proposed	speak

BUSINESS QUERIES: Bhavika

7) Before the end of the academic year 2022-2023, the president and treasurer of OSC want to see how much money is left to be reimbursed by OSC and to whom, as they wish to close the books of accounts for their term of service.

```
SELECT M.NAME, B.PURCHASED_BY AS AUS_ID, SUM(B.AMOUNT_OF_BILL) AS MONEY_OWED, B.HANDLED_BY AS PAYER, E.NAME AS PAYER_NAME
FROM MEMBER M
RIGHT JOIN BILLS B ON M.AUS_ID = B.PURCHASED_BY JOIN EXECUTIVE_TEAM E ON B.HANDLED_BY=E.AUS_ID
WHERE REIMBURSED = 'N'
GROUP BY M.NAME, B.PURCHASED_BY, B.HANDLED_BY, E.NAME;
```

Here, we have selected the name of the member who purchased the bill from the member table, purchased_by (AUS_ID of the purchaser), the sum of the different bills purchased by them, and the person who is handling the bill from the bills table. We also project the name of the person who is handling the bill (from the executive team (table)).

For this query, we have performed a right join between the member table and the bills table, on the AUS_ID of the member (from the member table) and the AUS_ID of the purchaser (from the bills table). We have further performed a join between the Executive Team and Bills table, on Handled_By and AUS_ID, so we can showcase the name and AUS_ID of the executive team member who owes the money to the payee.

We have limited the query to showcase only those payees who are yet to be reimbursed (Reimbursed = 'N"). We have grouped the results by the Member name, the purchaser_ID (AUS_ID of the purchaser from the bills table), the AUS_ID of the payee (the executive team member who should make the reimbursement from the bills table), as well as the name of the reimburser (name of the executive team member who should make the reimbursement from the executive team member table), as sometimes more than one bill could be pending for reimbursement for a specific purchaser, to be reimbursed from the same payer (executive team member).

	NAME	AUS_ID	MONEY_OWED	PAYER	PAYER_NAME
1	Bhavika Vohra	g00089132	450	g00085775	Chavi Mehta

As we can see, by the end of this academic year, Bhavika Vohra should be reimbursed an amount of AED 450, by Chavi Mehta, the treasurer of OSC. Bhavika is the only person left to be reimbursed currently.

BUSINESS QUERIES: Bhavika

8) OSC wants to introduce a new flagship event, OSCDataScienceSeries, from the idea repository. They want to know whether they should host this event in the upcoming fall or spring semester, as they want to maximize their event attendance. To do so, they want to see the average attendance of fall and spring semester events in general.

```
SELECT
  CASE
    WHEN EVENT_SEMESTER LIKE 'FALL%' THEN 'FALL'
    WHEN EVENT_SEMESTER LIKE 'SPRING%' THEN 'SPRING'
  END AS EVENT_SEMESTER,
  AVG(ATTENDANCE) AS AVG_ATTENDANCE
FROM EVENT
WHERE EVENT_SEMESTER LIKE 'FALL%' OR EVENT_SEMESTER LIKE 'SPRING%'
GROUP BY
  CASE
    WHEN EVENT_SEMESTER LIKE 'FALL%' THEN 'FALL'
    WHEN EVENT_SEMESTER LIKE 'SPRING%' THEN 'SPRING'
  END
ORDER BY AVG(ATTENDANCE) DESC;
```

In this query, we are grouping all Fall and all Spring semesters together, using the CASE function, inside the select function as well as the group by function. We use the END function to end the CASE statement. Semesters that start with Fall, will be grouped together as Fall, and semesters that start with Spring, will be grouped together as Spring

We are also projecting the Event_Semester and the Average Attendance of all the events in these 2 semesters together, from the events table, to showcase the average attendance of all Spring events as well as Fall events. Then, we are ordering the results, in a descending manner of average attendance, to understand which semesters (Fall or Spring), have higher average attendance.

Some events have a NULL value for attendance, as attendance was not recorded for these events, due to reasons such as lack of ability to track attendance. However, these account for only 0.1% of the total records, and we choose to keep this data in our query, as it does not create much impact.

As we can see, fall semester events have a higher average attendance of 72 participants, as compared to spring semester events, which have an average attendance of 64 participants. A reason for this could be, that Fall semesters usually have a higher intake of students. Another reason for this could be that Fall semesters are usually longer, so students are more incentivized to attend events, as it does not impact their hectic schedules.

	EVENT_SEMESTER	AVG_ATTENDANCE
1	Fall	72.636363636363636363636363636363636364
2	Spring	64.571428571428571428571428571428571428

Thus, according to the results of this query, OSC would be advised to host the new OSCDataScienceSeries, an event series about the application of Data Science in different fields in the upcoming Fall semester.

BUSINESS QUERIES: Bhavika

9) The media team wishes to understand the correlation between the likes and views on a post and the attendance of an event. They want to understand if more likes/views on media content for events, leads to an increase in event attendance.

```
SELECT
CORR(MC.NO_OF_LIKES, E.ATTENDANCE) AS CORRELATION_LIKES_ATT,
CORR(MC.NO_OF_VIEWS, E.ATTENDANCE) AS CORRELATION_VIEWS_ATT
FROM
  MEDIA_CONTENT MC
  INNER JOIN EVENT E ON MC.EVENT_ID = E.EVENT_ID
WHERE
   E.ATTENDANCE IS NOT NULL;
```

Here, we have selected (projected) the correlation between no_of_likes from the media content table and Attendance from the event table as Correlation_Likes_Att, and the correlation between no_of_views from the media content table and Attendance from the event table as Correlation_Views_att, to help us see if the correlation between the media content likes and event attendance or media content views and event attendance is higher.

We have performed an inner join between the Media_Content table and the Event table on the Event_ID in the media content and event table, as we want to use event attendance as a variable to perform the correlation. Event_ID is a foreign key in the media content table.

Some events have a NULL value for attendance, as attendance was not recorded for these events, due to reasons such as lack of ability to track attendance. We have decided to not include these values in our correlation, to increase the accuracy of our results.

	CORRELATION_LIKES_ATT	CORRELATION_VIEWS_ATT
1	0.64992156946085656248139759916939622653 97	0.27817420443960596308359935742980411777 65

As we can see, the correlation between likes on media content and attendance is greater than the correlation between views on media content and attendance. This implies that people who have liked media content are more likely to attend an event, as compared to someone who viewed media content.

The correlation between likes of a media_content and attendance of an event is 0.64, whereas the correlation between views of a media_content and attendance of an event is 0.27. The correlation between likes and attendance is higher, which implies that people who like a media_content are more likely to attend an event.

BUSINESS QUERIES: Mhd. Dar

10) Query to show information about members who have a name starting with 'Ra'

```
SELECT *
FROM MEMBER
WHERE NAME LIKE 'RA%';
```

If we need to look up information about a specific member but only know that their name starts with 'Ra' then this query helps us narrow down the list of members we would need to search through to only two.

	AUS_ID	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM
1	g00088575	Ramziyya Abdul Rahman	Electrical Engineering	Junior 2	g00088575@aus.edu	Media Team
2	g00094956	Raagini Vohra	Accounting	Freshman 1	g00094956@aus.edu	Media Team

11) Query to gather information about activities team member whose major is Computer Science and the Standing is Junior 2

```
SELECT AUS_ID, NAME, PRIMARY_MAJOR, STANDING, EMAIL, TEAM,
ACTIVITIES_TEAM_POSITION, TECH_SKILL_LEVEL, REPORTS_TO
FROM ACTIVITIES_TEAM
WHERE PRIMARY_MAJOR = 'COMPUTER SCIENCE' AND STANDING = 'JUNIOR 2';
```

By doing so, we can exactly know how many Activities team members that are majoring in Computer Science and are Junior 2 in academic standing are participating. Similarly, we can run the same query with the same major and a different standing. This query can help identify the number of students from a specific major in a specific academic standing.

	AUS...	NAME	PRIMARY_MAJOR	STANDING	EMAIL	TEAM	ACTIVITIES_TEAM_POSITION	TECH_SKILL_LEVEL	REPORTS_TO
1	b00087818	Abdu Sallouh	Computer Science	Junior 2	b00087818@aus.edu	Activities Team	Activities Officer	2	Head Of Activities
2	b00087520	Koushal Parupudi	Computer Science	Junior 2	b00087520@aus.edu	Activities Team	Activities Officer	8	Head Of Activities
3	b00087698	Muhammad Ahmer	Computer Science	Junior 2	b00087698@aus.edu	Activities Team	Activities Officer	9	Head Of Activities
4	b00089801	Aadhith Shankarnarayanan	Computer Science	Junior 2	b00089801@aus.edu	Activities Team	Activities Officer	10	Head Of Activities

BUSINESS QUERIES: Colin

12) List the names of events and their corresponding collaborators

```
SELECT E.EVENT_NAME, E.EVENT_ID, E.ATTENDANCE, EC.EMIRATES_ID, C.ORGANIZATION
FROM EVENT E
JOIN EVENT_COLLABORATED EC ON E.EVENT_ID = EC.EVENT_ID
JOIN COLLABORATORS C ON EC.EMIRATES_ID = C.EMIRATES_ID
ORDER BY C.ORGANIZATION, EC.EVENT_ID;
```

The goal is to retrieve specific data related to events and their collaborators.

SELECT: We are specifying the columns we want to retrieve from the tables: E.EVENT_NAME, E.EVENT_ID, E.ATTENDANCE, EC.EMIRATES_ID, and C.ORGANIZATION.

FROM: We are specifying the tables from which we want to retrieve the data: EVENT, EVENT_COLLABORATED, and COLLABORATORS.

JOIN: We are joining the tables together based on specific conditions. The first join is between the EVENT and EVENT_COLLABORATED tables, matching their EVENT_ID columns. The second join is between the EVENT_COLLABORATED and COLLABORATORS tables, matching their EMIRATES_ID columns.

ORDER BY: We are specifying the sorting order for the result set. In this case, we are sorting the rows by the organization (C.ORGANIZATION) and then by the event ID (EC.EVENT_ID).

SQL | All Rows Fetched: 10 in 0.007 seconds

	EVENT_NAME	EVENT_ID	ATTENDANCE	EMIRATES_ID	ORGANIZATION
1	Introduction to Amazon Web Services	AWSF20	110	784-1979-8864567-1	Amazon Web Sevices
2	Linux Workshop	LINUXF22	55	784-1971-2683533-1	American Univesity of Sharjah
3	Music Editing Workshop	MUSEDITF21	17	784-1985-4398539-1	American Univesity of Sharjah
4	AUS Carnival	CARNF22	213	784-2002-9842592-1	Entrepreneurship Club AUS
5	AUS Carnival	CARNF22	213	784-2001-3759259-1	Gastronomy Club AUS
6	Introduction to Neural Networks	NNF20	77	784-2000-8885567-1	IEEE - AUS Chapter
7	Introducing Github	GITHUBS22	113	784-2001-5675567-1	IEEECS - AUS Chapter
8	Info Session with Palo Alto	PALOALTF20	102	784-2001-5675567-1	IEEECS - AUS Chapter
9	Introduction to Internet of Things	IOTS21	46	784-2002-9985567-1	IEEESIGHT - AUS Chapter
10	Info Session with Palo Alto	PALOALTF20	102	784-1969-1234567-1	Palo Alto

13) List the name of the student in the media team that has done the most number of media content

```
SELECT MT.AUS_ID, M.NAME, COUNT(*) AS NUM_OF_CONTENT
FROM MEMBER M
JOIN MEDIA_TEAM MT ON M.AUS_ID = MT.AUS_ID
JOIN MEDIA_CONTENT MC ON MT.AUS_ID = MC.CREATOR_AUS_ID
GROUP BY MT.AUS_ID, M.NAME
ORDER BY NUM_OF_CONTENT DESC
FETCH FIRST 1 ROW ONLY;
```

BUSINESS QUERIES: Colin

The goal is to find the member who has created the highest number of media content.

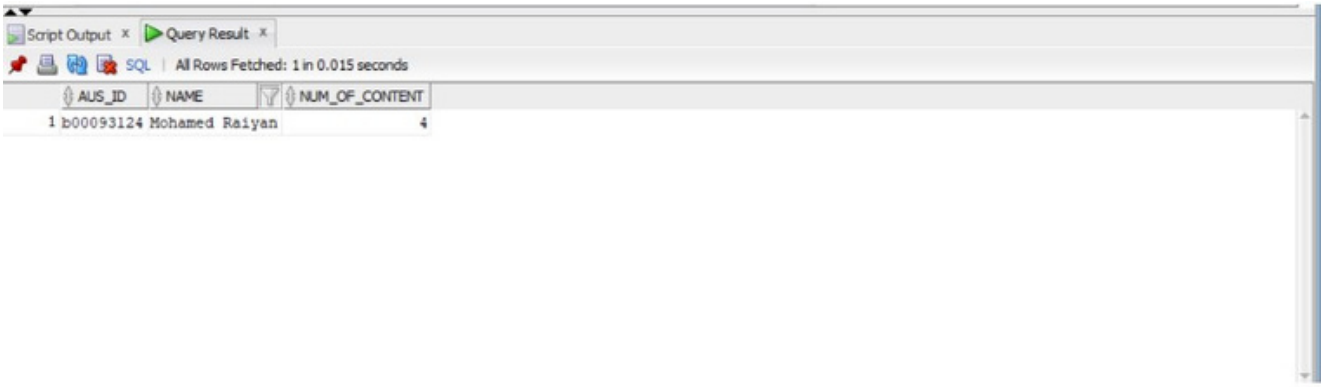
SELECT: We are specifying the columns we want to retrieve from the tables: MT.AUS_ID, M.NAME, and COUNT(*) AS NUM_OF_CONTENT.

FROM: We are specifying the tables from which we want to retrieve the data: MEMBER, MEDIA_TEAM, and MEDIA_CONTENT.

JOIN: We are joining the tables together based on specific conditions. The first join is between the MEMBER and MEDIA_TEAM tables, matching their AUS_ID columns. The second join is between the MEDIA_TEAM and MEDIA_CONTENT tables, matching their AUS_ID and CREATOR_AUS_ID columns, respectively.

GROUP BY: We are grouping the result set by MT.AUS_ID and M.NAME. This means that the COUNT(*) function will calculate the number of media content for each unique combination of MT.AUS_ID and M.NAME.

ORDER BY: We are specifying the sorting order for the result set. In this case, we are sorting the rows by the number of content (NUM_OF_CONTENT) in descending order.
FETCH FIRST 1 ROW ONLY: We are limiting the result set to only retrieve the first row, which will be the member with the highest number of media content.



BUSINESS QUERIES: Nabeel

14) Query to select the event with the least attendance for every semester

```
SELECT E.EVENT_SEMESTER, E.EVENT_NAME, E.ATTENDANCE
FROM EVENT E
WHERE E.ATTENDANCE = (SELECT MIN(ATTENDANCE) FROM EVENT WHERE EVENT_SEMESTER = E.EVENT_SEMESTER)
ORDER BY E.EVENT_SEMESTER, E.EVENT_NAME;
```

	EVENT_SEMESTER	EVENT_NAME	ATTENDANCE
1	Fall 2020	Movie Night with OSC	30
2	Fall 2021	Build your own chatbot with Python!	12
3	Fall 2022	Introduction to Web 3.0	55
4	Fall 2022	Linux Workshop	55
5	Spring 2021	Open Night with Open Source	34
6	Spring 2022	Game Show - Taboo	23
7	Spring 2023	Capture the Flag 2.0	102

15) Query to find the top 5 events with the highest budget, excluding events with null budgets.

```
SELECT E.EVENT_ID, E.EVENT_NAME, SUM(B.AMOUNT_OF_BUDGET) AS TOTAL_BUDGET
FROM EVENT E LEFT JOIN BUDGET B ON E.EVENT_ID = B.EVENT_ID
WHERE B.AMOUNT_OF_BUDGET IS NOT NULL
GROUP BY E.EVENT_ID, E.EVENT_NAME
ORDER BY TOTAL_BUDGET DESC
FETCH FIRST 5 ROWS ONLY;
```

	EVENT_ID	EVENT_NAME	TOTAL_BUDGET
1	CARNF22	AUS Carnival	600
2	CTF2.0S23	Capture the Flag 2.0	500
3	LINUXF22	Linux Workshop	500
4	AWSF20	Introduction to Amazon Web Services	250
5	TABOOS22	Game Show - Taboo	200

BUSINESS QUERIES: Rafid

16) Find the top 5 Contents that have the highest amount of likes

```
SELECT CONTENT_ID, EVENT_ID, NO_OF_LIKES FROM MEDIA_CONTENT
ORDER BY NO_OF_LIKES DESC
FETCH FIRST 5 ROWS ONLY;
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 2.137 seconds

	CONTENT_ID	EVENT_ID	NO_OF_LIKES
1	CARNF22-02	CARNF22	300
2	CFF20-01	CFF20	100
3	CTF2.0S23-02	CTF2.0S23	100
4	CARNF22-01	CARNF22	100
5	CTFS22-02	CTFS22	100

This SQL query is trying to retrieve the Content_ID, Event_ID, and No_of_likes columns from the Media_Content table, and then sort the results in descending order based on the number of likes.
Finally, the query limits the number of results to only the top 5 rows, which are the media content items with the highest number of likes.
In summary, this query is attempting to retrieve the top 5 most popular media content items, based on the number of likes, along with their respective content and event IDs.
Therefore, This query is to judge the best content created by our media team, as this will help us assign who will be the leader to guide the rest of the media team

17) Show the number of bills that needs to be reimbursed

```
SELECT BILL_NO, AMOUNT_OF_BILL, PURCHASED_BY, MERCHANT_NAME,
PAYMENT_METHOD, REIMBURSED, PAYMENT_DATE, BUDGET_NO, HANDLED_BY
FROM BILLS
WHERE REIMBURSED = 'N';
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.004 seconds

	BILL_NO	AMOUNT_OF_BILL	PURCHASED_BY	MERCHANT_NAME	PAYMENT_METHOD	REIMBURSED	PAYMENT_DATE	BUDGET_NO	HANDLED_BY
1	CTF2.0S2319713219	150	g00089132	Talabat	Cash	N	21-OCT-23	320197	g00085775
2	CTF2.0S2319738520	300	g00089132	Amazon	Cash	N	21-OCT-23	320197	g00085775

This SQL query is trying to retrieve the columns Bill_No, Amount_of_Bill, Purchased_By, Merchant_Name, Payment_Method, Reimbursed, Payment_Date, Budget_No, and Handled_By from the Bills table, where the value of the Reimbursed column is equal to 'N'.
In simple words, the query is trying to identify all the bills that have not been reimbursed yet and retrieve the relevant information associated with those bills.
Therefore, this query would help the club find out which bills are not paid back yet.

Previously, we found out that Bhavika has to be reimbursed Dhs 450 by OSC, this query helps us to know which bills Bhavika has to be reimbursed with. Through the bill_no, we can also understand which event Bhavika needs to be reimbursed for.

CONCLUSION

Through this project, we have been able to help OSC solve the problems they were facing such as the inability to track financial transactions and the ability to recognize collaborators, through building a new Database Management System using Oracle SQL, with tables such as Bills, Budget, and Collaborators.

We have been able to achieve a more efficient and effective database management system for the members, events, media content, etc of the club.

Toward the end of the project, we also developed a few business queries, to help OSC analyze their database and answer a few questions, such as 'When to host a flagship event?', 'Does More Likes/Views on a Media Content Lead to a Higher Attendance at Events?'

This project has also helped us practically apply our in-class knowledge by developing a new database management system from scratch. We were able to create tables, insert values, drop and add columns, create business queries, and join tables (using right, left, inner, outer, etc joins). Engaging in this project has proven to be an invaluable experience, as it not only allowed us to apply the theoretical knowledge we gained in class but also solidified our understanding of the subject matter through practical implementation, critical thinking, and collaborative problem-solving.