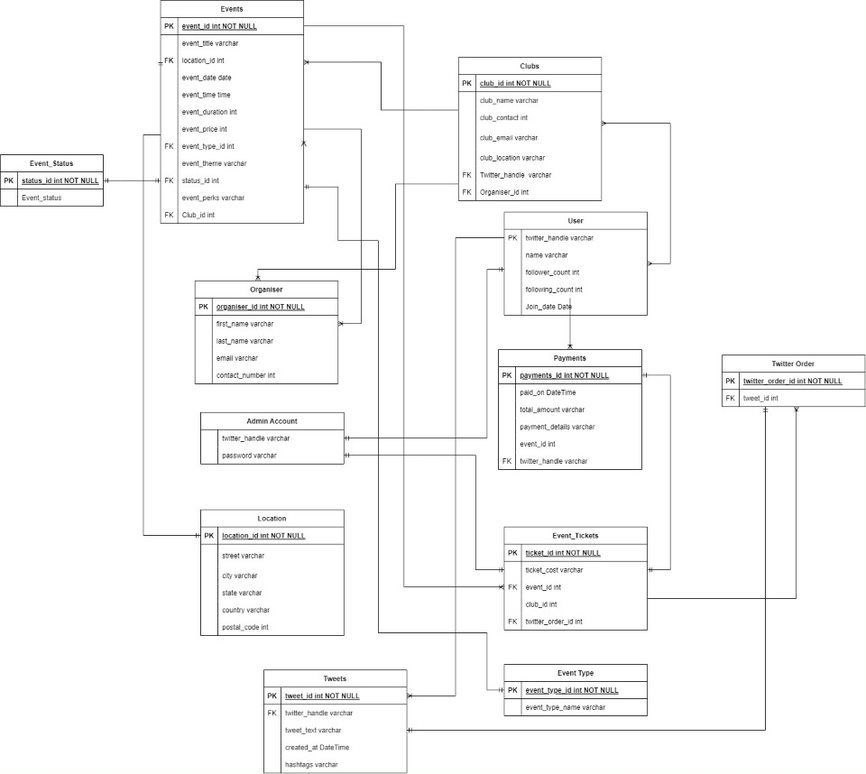
ASSIGNMENT 2

**Team Name**–Nik's Favorite Team

**Members**

1. Raaga Spandana Parisi (NUID 002768462)
2. Nikhil kumar Bavandla (NUID 002658875)
3. Poornachandra Kongara (NUID 002778509)

**ER Diagram for the Database:**



**USE CASES**

1. **Use Case:** Register for an account in Northeastern Events Portal (NEP)

**Description:** User registers for an account in NEP

**Actor:** User

**Precondition:** When a user wants to register for an event happening on campus, firstly they will be registered in the events portal.

**Steps:**

**Actor action:** User request for registration

**System Responses:** If user information is correct then user is registered and use case ends.

**Post Condition:** user successfully registered

**Alternate Path:** The user request is not correct, and system throws an error

**Error:** User information is incorrect

1. **Use Case:** Register for an event

**Description:** User selects events to register

**Actors:** User

**Precondition:** User must have a unique Twitter handle to tweet

**Steps:**

**Actor action –** User tweets about an event along with the event ID

**System Responses –** An order is made for the event that matches the event ID

**Post Condition:** An order is added to Twitter\_Order table for the event the user tweeted.

**Alternate Path:** The event not currently available in the events database

**Error:** No such event is Available

1. **Use Case:** View an event already registered through Twitter by a user

**Description:** User views an event that they registered for already

**Actors:** User

**Precondition:** User must have made an order

**Steps:**

**Actor action –** User views an event from the event database using the events ID.

**System Responses –** events are fetched from the events database using the events ID.

**Post Condition:** system displays all the events the user registered for.

1. **Use Case:** View the events happening on a particular date.

**Description:** User views the events that are taking place on the specified date.

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User views the events that are happening on a particular date.

**System Responses:** the list of events happening on that date is displayed

**Post Condition:** system displays the list of events for that condition

1. **Use Case:** View the events that are free to attend

**Description:** User views the events that require no payment to attend the event

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User views the events that are happening for free on the campus

**System Responses:** the list of events happening for free are displayed

**Post Condition:** system displays the list of events for that condition

1. **Use Case:** View the events that are happening during a particular time slot

**Description:** User views the events that are happening from a particular start time to a particular end time.

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies the start time and the end time to look for events happening in that time

**System Responses:** the list of events happening in that time slot is displayed

**Post Condition:** system displays the list of events for that condition

1. **Use Case:** View the events that are happening in a location

**Description:** Use views the events that are happening in a particular location

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies the location they want to look for events

**System Responses:** the list of events happening in the specified location

**Post Condition:** system displays the list of events for that condition

1. **Use Case:** View the events that have free perks

**Description:** User views the events that are giving free perks for attending like free food, free swag, etc.

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies the type of free perks they want

**System Responses:** the list of events that are providing such specified perks

**Post Condition:** system displays the list of events for that condition

1. **Use Case:** View the events that have a particular theme

**Description:** User views the events that are dare distinguished by the theme - is the event about sustainability, is the event geared towards a particular topic, or is the event for networking

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies a keyword. The keyword will be the type of event they want to attend.

**System Responses:** the list of events that are related to that keyword

**Post Condition:** system displays the list of events for that condition

1. **Use Case:** View the events that are being organized by a club

**Description:** User views the events that are being organized by a particular club or organization

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies the name of the club or organization that is organizing the event

**System Responses:** the list of events that are related to that club or organization name

**Post Condition:** system displays the list of events for that condition

1. **Use Case:** View the payments made by the user

**Description:** User views the payments made by them

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User views the payments made by thems

**System Responses:** the list of payments associated with that user ID is displayed

**Post Condition:** system displays the list of payments for that condition from the payments table

1. **Use Case:** View the events that are below a particular price ($20)

**Description:** User views the products below a particular price

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User views the events below a particular price

**System Responses:** the list of events below the price is displayed

**Post Condition:** system displays the list of events for the condition

1. **Use Case:** View the event status

**Description:** User views the events by their status of completion, if the events are completed or scheduled for a future date.

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies the type of status

**System Responses:** the list of events related to that type of status are displayed

**Post Condition:** system displays the list of events for the condition

1. **Use Case:** View the details of the organizer

**Description:** User views the details of the organizer of a particular club.

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies the club ID for which they want to view the organizer details for

**System Responses:** the details of the club organizer for that Club are displayed

**Post Condition:** system displays the details of the organizer

1. **Use Case:** View the location of a club

**Description:** User views the location of a particular club

**Actor:** User

**Precondition:**

**Steps:**

**Actor action:** User specifies the club using the club ID

**System Responses:** the location of that club is displayed using the club ID

**Post Condition:** system displays the details of the club location

**Relational Algebra**

1. **Use Case:** Register for an account in Northeastern Events Portal (NEP)

Relational algebra is not possible for this use case as there is no register operation.

1. **Use Case:** Register for an event

Relational algebra is not possible for this use case as there is no register operation.

1. **Use Case:** View an event already registered through Twitter by a user

Πname,twitter\_handle ( σticket\_id is not null and twitter\_handle = “@Nikhil” (event ticket))

1. **Use Case:** View the events happening on a particular date.

σ{event\_date = “2022/10/21”} (Events)

1. **Use Case:** View the events that are free to attend

σ (event\_type.event\_type = “free”)(Events⋈Event\_TypeEvent.event\_type\_id=Event\_type.event\_type\_id )

1. **Use Case:** View the events that are happening during a particular time slot

σ{event\_time = 16:00:00 ∩ event\_duration = 2} (Events)

1. **Use Case:** View the events that are happening in a location

σ (Location.street = “huntington ave”)(Events⋈LocationEvent.location\_id=Location.location\_id )

1. **Use Case:** View the events that have free perks

σ{event\_perks = “free tshirts”}(Events)

1. **Use Case:** View the events that have a particular theme

σ{event\_theme = “fall themed”} (Events)

1. **Use Case:** View the events that are being organized by a club

σ (Club.club\_name = “husky competitive programming club”)(Events⋈ClubsEvent.club\_id=Club.club\_id )

1. **Use Case:** View the payments made by the user

σ (User.twitter\_handle = “@Nikhil”)(User⋈PaymentsUser.twitter\_handle=Payments.twitter\_handle )

1. **Use Case:** View the events that are below a particular price ($20)

Π {Events.event\_title, Event\_Ticket.Ticket\_Cost}σ( Event\_ticket.ticket\_cost < 20)(Events⋈Event\_TicketsEvent.Event\_id=Event\_ticket.Event\_id)

1. **Use Case:** View the event status

Π{Events.event\_title, Event\_Status.Event\_status}σ( Event\_status.Event\_Status = “Registration Open”)(Events⋈Event\_StatusEvent.status\_id=Event\_Status.status\_id)

1. **Use Case:** View the details of the organizer

σ{club\_name = “AI Skunkworks”} (Clubs)

1. **Use Case:** View the location of a club

σ (Club.Club\_name = “Badminton Graduate Club”)(Clubs⋈LocationClub.club\_location\_id=Location.Location\_id)

**SQL for the physical model that represents the entire conceptual model:**

1. **Use Case:** Register for an account in Northeastern Events Portal (NEP)

Insert into admin (twitter\_handle, password) values (‘@john’, ‘john#21’);

Insert into admin (twitter\_handle, password) values (‘@ramu’, ‘maru$#12’);

1. **Use Case:** Register for an event

Insert into tweets (tweet\_id, twitter\_handle, tweet\_text, created\_at) values (‘T34343’, ‘@john’, ‘I want to buy a ticket for Diwali event’, ‘10-5-222’);

Insert into twitter\_order (twitter\_order\_id, tweet\_id) values (‘#Ord4345’, ‘T34343’);

Insert into event\_tickets (ticket\_id, ticket\_cost, event\_id, club\_id, twitter\_order\_id) values (‘I45’, ‘50’, ‘E3539’, ‘Cl34’, ‘Ord4345’);

1. **Use Case:** View an event already registered through Twitter by a user

Select e.event\_id from events e, event\_tickets t, club c, user u where e.event\_id=t.event\_id and c.club\_id=e.club\_id and u.twitter\_handle=c.twitter\_handle and u.twitter\_handle=’@john’;

1. **Use Case:** View the events happening on a particular date.

Select \* from events where event\_date=’10-5-22’;

1. **Use Case:** View the events that are free to attend

Select \* from events e, event type t where t.event\_type\_id = e.event\_type\_id and t.event\_type=’free’;

1. **Use Case:** View the events that are happening during a particular time slot

Select \* from events where event\_time=’3 PM’;

1. **Use Case:** View the events that are happening in a location

Select street, city, state from events e, location l where l.location\_d = e.location\_id and l.city= ’boston’;

1. **Use Case:** View the events that have free perks

Select event\_id, event\_name, event\_perks from events where events\_perks like ‘%Free%’;

1. **Use Case:** View the events that have a particular theme

Select event\_id, event\_name, event\_theme from events where event\_theme=’Halloween’;

1. **Use Case:** View the events that are being organized by a club

Select e.event\_Id, e.event\_title from events e, club c where c.club\_id=e.club\_id and club\_name = ’nu\_sanskriti’;

1. **Use Case:** View the payments made by the user

Select p.payment\_id, p.paid\_on, p.total\_amount from payments p, user u where u.twitter\_handle = p.twitter\_handle and u.name=’ram’;

1. **Use Case:** View the events that are below a particular price ($20)

Select event\_id, event\_title from events where event\_price<20;

1. **Use Case:** View the event status

Select e.event\_id, e.event\_title, s.event\_status from events e, event\_status s where s.status\_id = e.status\_id;

1. **Use Case:** View the details of the organizer for an event;

Select o.organizer\_id, o.first\_name from events e, clubs c, organizer o where c.club\_id=e.club\_id and o.oraganizer\_id = c.organizer\_id;

1. **Use Case:** View the location of a club

Select club\_name, club\_location from clubs where club\_name = ’apmc’;

**SQL Queries and Relational Algebra answering physical model**

1. What user posted this tweet?

**Query:** Select twitter\_handle, tweet\_text from tweets where tweet\_id=’T1233’;

**Relational Notation:** Πtwitter\_handle, tweet\_text(σtweet\_id = ‘T1233’ (tweets))

1. When did the user post this tweet?

**Query:** Select twitter\_handle, tweet\_text, created\_at from tweets where created\_at =’11-4-2022’;

**Relational Notation:** Πtwitter\_handle, tweet\_text, created\_at(σcreated\_at = ‘11-4-2022’ (tweets))

1. What tweets have this user posted in the past 24 hours?

**Query:** Select tweet\_id, tweet\_text from tweets where twitter\_handle=’@ram’ and (created\_at between (SYSDATE-1) and SYSDATE ());

**Relational Notation:** Πtwitter\_id, tweet\_text(σtwitter\_handle = ‘@ram’ ∩ created\_at between (SYSDATE-1) and SYSDATE ()) (Tweets)

1. How many tweets have this user posted in the past 24 hours?

**Query:** Select count(tweet\_id) as NoOfTweets from tweets where twitter\_handle=’@ram’ and (created\_at between (SYSDATE-1) and SYSDATE ());

**Relational Notation:** *ρ* NoOfTweets (∏ count(tweet\_id) (σ twitter\_handle=’@ram’ ∩ created\_at between (SYSDATE-1) and SYSDATE ()) (Tweets))

1. When did this user join twitter?

**Query:** Select u.twitter\_handle, u.join\_date from tweets t, user u where u.twitter\_handle=t.twitter\_handle and u.twitter\_handle=’@john’;

**Relational Notation:** ∏ (u.twitter\_handle, u.join\_date)σ (u.twitter\_handle = “@john”)(User⋈Tweetsu.twitter\_handle=t.twitter\_handle )

1. What keywords/hashtags are popular?

**Query:** Select hashtags from tweets group by hashtags having count(tweet\_id)>100;

**Relational Notation: Not possible as Group By function does not exist in relational algebra.**

1. What tweets are popular?

**Query:** Select tweet\_id from tweets where retweet>1000;

**Relational Notation:** ∏ (tweet\_id)( σ retweet > 1000) (tweets)