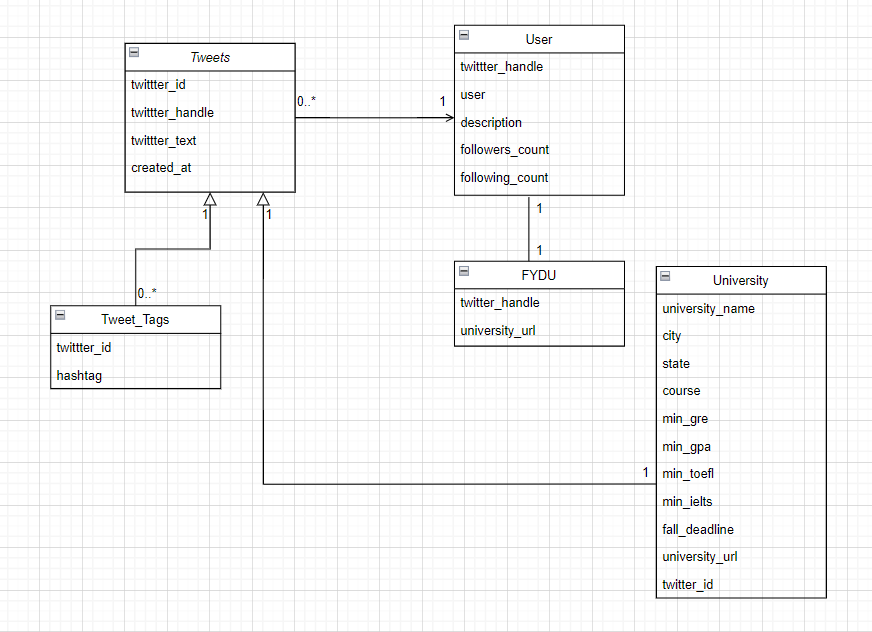
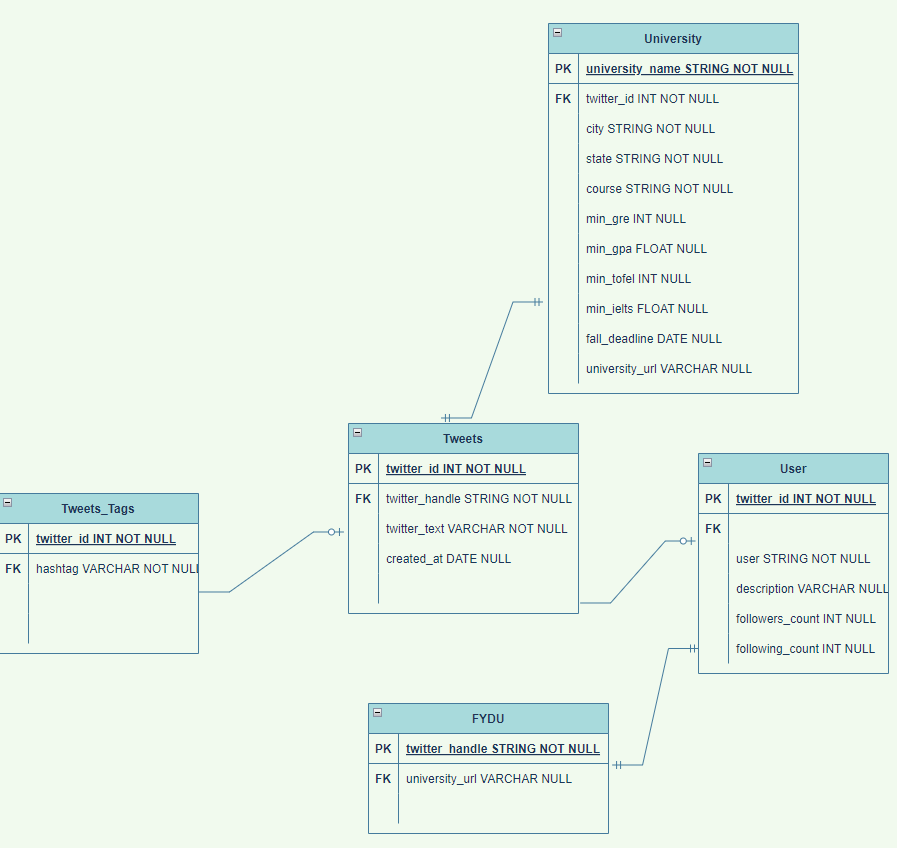
**A Model on Find your Dream University using Twitter**

The university model has been updated to be more specific for a particular university. (University Warehouse). The university model also incorporates Twitter database schema. In this model, the user can find details about the university by tweeting the university name along with the university URL. The FYDU account is the main admin who can handle the university and user database.

UML Diagram of the FYDU model



Physical ER Diagram of the FYDU model



**Explanation on some of the design decisions:**

* The FYDU account has a login and password. This login is the same as a user’s Twitter handle. The Twitter handle is unique – hence it can also be treated as the primary key of the table.
* ‘User’ table stores data such as user’s twitter handle, twitter description about each user can tweet any number of tweets. The FYDU-user (admin user) is also one of the users and this information can be stored in the user table itself.
* A user(student) can ask for university recommendation through Twitter by tweeting about his/her requirement and profile.
* Any tag in the tweet will be saved in the ‘Tweet\_Tags’ table.
* The ‘University’ table contains essential details about the minimum requirement and few other details about the university.

**SQL Statements for the conceptual model:**

use assignment2\_schema;

**USERS TABLE**

**CREATE TABLE `users` (**

**`followers\_count` int NOT NULL,**

**`name` varchar(45) NOT NULL,**

**`description` varchar(45) DEFAULT NULL,**

**`following\_count` int NOT NULL,**

**`tweet\_handle` varchar(45) NOT NULL,**

**PRIMARY KEY (`tweet\_handle`)**

**);**

**TWEET\_TAGS TABLE**

**CREATE TABLE `tweet\_tags` (**

**`tweet\_id` int NOT NULL,**

**`tag` varchar(45) DEFAULT NULL,**

**PRIMARY KEY (`tweet\_id`)**

**) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;**

**UNIVERSITY TABLE**

CREATE TABLE `university` (

`tweet\_id` int NOT NULL,

`uni\_name` varchar(45) NOT NULL,

`city` varchar(45) DEFAULT NULL,

`state` varchar(45) DEFAULT NULL,

`course` varchar(45) DEFAULT NULL,

`min\_gpa` varchar(45) DEFAULT NULL,

`min\_gre` varchar(45) DEFAULT NULL,

`min\_cgpa` varchar(45) DEFAULT NULL,

`min\_toefl` varchar(45) DEFAULT NULL,

`min\_ielts` varchar(45) DEFAULT NULL,

`fall\_deadline` varchar(45) DEFAULT NULL,

`university\_url` varchar(45) DEFAULT NULL,

PRIMARY KEY (`tweet\_id`),

CONSTRAINT `tweet\_fk3` FOREIGN KEY (`tweet\_id`) REFERENCES `tweet\_table` (`tweet\_ids`),

CONSTRAINT `tweet\_id\_fk2` FOREIGN KEY (`tweet\_id`) REFERENCES `tweet\_table` (`tweet\_ids`),

CONSTRAINT `tweet\_id\_fk3` FOREIGN KEY (`tweet\_id`) REFERENCES `tweet\_table` (`tweet\_ids`),

CONSTRAINT `tweet\_ids\_fk3` FOREIGN KEY (`tweet\_id`) REFERENCES `tweet\_table` (`tweet\_ids`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

**TWEET\_TABLE TABLE**

CREATE TABLE `tweet\_table` (  
  `tweet\_ids` int NOT NULL,  
  `tweet\_handle` varchar(45) NOT NULL,  
  `tweet\_text` varchar(45) NOT NULL,  
  `created\_at` date NOT NULL,  
  PRIMARY KEY (`tweet\_ids`),  
  UNIQUE KEY `tweet\_handle\_UNIQUE` (`tweet\_handle`),  
  UNIQUE KEY `tweet\_text\_UNIQUE` (`tweet\_text`),  
  UNIQUE KEY `created\_at\_UNIQUE` (`created\_at`),  
  CONSTRAINT `tweet\_handle\_fk1` FOREIGN KEY (`tweet\_handle`) REFERENCES `users` (`tweet\_handle`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

**FIND\_YOUR\_DREAM\_UNIVERSITY TABLE**

CREATE TABLE `find\_your\_dream\_university` (  
  `tweet\_handle` varchar(45) NOT NULL,  
  `password` varchar(45) NOT NULL,  
  PRIMARY KEY (`tweet\_handle`),  
  CONSTRAINT `tweet\_handle\_fk2` FOREIGN KEY (`tweet\_handle`) REFERENCES `users` (`tweet\_handle`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

**CONSTRAINT FOR TWEET\_TABLE**

ALTER TABLE `tweet\_table`  
ADD CONSTRAINT `tweet\_handle\_fk1` FOREIGN KEY (`tweet\_handle`)  
REFERENCES users(`tweet\_handle`);

**CONSTRAINT FOR TWEET\_TAGS**

ALTER TABLE `tweet\_tags`  
ADD CONSTRAINT `tweet\_id\_fk1` FOREIGN KEY (`tweet\_id`)  
REFERENCES tweet\_table (`tweet\_ids`);

**CONSTRAINT FOR FIND\_YOUR\_DREAM\_UNIVERSITY**

ALTER TABLE `find\_your\_dream\_university`  
ADD CONSTRAINT `tweet\_handle\_fk2` FOREIGN KEY (`tweet\_handle`)  
REFERENCES users (`tweet\_handle`);

**CONSTRAINT FOR UNIVERSITY**

ALTER TABLE `university`  
ADD CONSTRAINT `tweet\_fk3` FOREIGN KEY (`tweet\_id`)  
REFERENCES tweet\_table (`tweet\_ids`);

ALTER TABLE `university`  
DROP FOREIGN KEY tweet\_id\_fk3;  
  
ALTER TABLE `university`  
DROP FOREIGN KEY tweet\_id\_fk2;

**INSERT QUERIES**

Insert queries:

INSERT INTO `assignment2\_schema`.`tweet\_tags`

(`tweet\_id`,

`tag`)

VALUES

(15913234,

'#universityofvirgina #FYDU');

INSERT INTO `assignment2\_schema`.`university`

(`tweet\_id`,

`uni\_name`,

`city`,

`state`,

`course`,

`min\_gpa`,

`min\_gre`,

`min\_cgpa`,

`min\_toefl`,

`min\_ielts`,

`fall\_deadline`,

`university\_url`)

VALUES

(159127255,'California Institute of Technology','Pasadena','California',

'Computer Science',

3.5,

310,

6.5,

75,

6.5,

'2023-03-31','https://t.co/J6dItz9LBS');

INSERT INTO `assignment2\_schema`.`university`

(`tweet\_id`,

`uni\_name`,

`city`,

`state`,

`course`,

`min\_gpa`,

`min\_gre`,

`min\_cgpa`,

`min\_toefl`,

`min\_ielts`,

`fall\_deadline`,

`university\_url`)

VALUES

(1591334,

'Northeastern University',

'Boston',

'Massachusetts',

'Information Systems, Engineering Management',

3,300,

7,

80,6.5,

'2023-01-15',

'https://t.co/uzYtrgyw7q');

INSERT INTO `assignment2\_schema`.`university`

(`tweet\_id`,

`uni\_name`,

`city`,

`state`,

`course`,

`min\_gpa`,

`min\_gre`,

`min\_cgpa`,

`min\_toefl`,

`min\_ielts`,

`fall\_deadline`,

`university\_url`)

VALUES

(15913234,

'University of Virginia',

'Charlottesville',

'Virginia',

'Computer Science, Supply Chain Management',

3,

300,

7,

80,

6.5,

'2023-05-30',

'https://t.co/uzYtrgyw7q');

**Use-Cases**

1. Register for an account in FYDU

Description: User registers for an account in FYDU

Actor: User

Precondition: When a user(student) wants to search for university, firstly he will be registered

Steps:

Actor action: User request for registration

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

Post Condition: Student successfully registered

Alternate Path: The student request is not correct and system throws an error

Error: User information is incorrect

**SQL: INSERT INTO find\_your\_dream\_university (tweet\_handle, password) VALUES (vidish\_kale, vidish08)**

1. View the University which is present in particular city (e.g., Boston)

Description: User views the university which is present in particular city

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university for that specific city

System Responses – The list of university in that specific city is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university in that particular city, it’ll display zero rows

**SQL: Select uni\_name from university where city like ‘%Boston%’**

**Relational algebra:**

**πuni\_name  
 σcity LIKE "%Boston%"university**

1. View the University which is present in particular State (e.g., Massachusetts)

Description: User views the university which is present in particular State

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university for that specific State

System Responses – The list of university in that specific state is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university in that particular state, it’ll display zero rows

**SQL: Select uni\_name from university where state like ‘%Massachusetts%’**

**Relational algebra:**

**πuni\_name  
 σstate LIKE "%Massachusetts%"university**

1. View the University which is offering particular course (e.g., Information Systems)

Description: User views the university which offers that course

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university offering that course

System Responses – The list of university offering that course is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university offering that particular course, it’ll display zero rows

**SQL: Select uni\_name from university where course like ‘%Computer *Science* %’**

**Relational algebra:**

**πuni\_name  
 σcourse LIKE "%Computer Science%"university**

1. View the University based on the minimum gre-score requirement (e.g., 300)

Description: User views the university which needs that much gre

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university accepting that much gre score.

System Responses – The list of university offering that course is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university offering that particular course, it’ll display zero rows

**SQL: Select uni\_name from university where min\_gre >= 300**

**Relational algebra:**

**πuni\_name  
 σmin\_gre >= 300university**

1. View the University based on the minimum GPA requirement (e.g., 3)

Description: User views the university which needs that much GPA

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university accepting that much GPA score.

System Responses – The list of university with that much GPA requirement is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university with that much minimum GPA, it’ll display zero rows

**SQL: Select uni\_name from university where min\_gpa >= 3**

**Relational algebra:**

**πuni\_name  
 σmin\_gpa >= 3university**

1. View the University based on the minimum CGPA requirement (e.g., 7.2)

Description: User views the university which needs that much GPA

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university accepting that much CGPA score.

System Responses – The list of university with that much CGPA requirement is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university with that much minimum CGPA, it’ll display zero rows

**SQL: Select uni\_name from university where min\_cgpa >= 7**

**Relational algebra:**

**πuni\_name  
 σmin\_cgpa >= 7university**

1. View the University based on the minimum Toefl requirement (e.g., 80)

Description: User views the university which needs that much toefl

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university accepting that much toefl score.

System Responses – The list of university with that much toefl requirement is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university with that much minimum toefl, it’ll display zero rows

**SQL: Select uni\_name from university where min\_toefl >= 80**

**Relational algebra:**

**πuni\_name  
 σmin\_toefl >= 80university**

1. View the University based on the minimum Ielts requirement (e.g., 6.5)

Description: User views the university which needs that much Ielts

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university accepting that much Ielts score.

System Responses – The list of university with that much Ielts requirement is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university with that much minimum Ielts, it’ll display zero rows

**SQL: Select uni\_name from university where min\_ielts >= 6.5**

**Relational algebra:**

**πuni\_name  
 σmin\_ielts >= 6.5university**

1. Search the University based on name (e.g., Northeastern)

Description: User views the university which has the same name as search criteria

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university which has the same name as search criteria

System Responses – The university with the same name is displayed

Post Condition: User can choose to view the displayed university

Alternate Path: If there is no university with same name, it’ll display zero rows

**SQL: Select uni\_name from university where uni\_name like ‘%NEU%’ or uni\_name like ‘%Northeastern%’**

**Relational algebra:**

**πuni\_name  
 σuni\_name LIKE "%NEU%" OR uni\_name LIKE "%Northeastern%"university**

1. View the University URL for the particular university

Description: User views the university URL of the university

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university link

System Responses – The list of academic details will be displayed

Post Condition: User can choose to view all the required criteria (pre-requisite)

Alternate Path: If there is no university link, it’ll display zero rows

**SQL: Select university\_url from university where uni\_name like ‘%NEU%’ or uni\_name like ‘%Northeastern%’**

**Relational algebra:**

**πuniversity\_url  
 σuni\_name LIKE "%NEU%" OR uni\_name LIKE "%Northeastern%"university**

1. View the University which is offering particular course in specified state (e.g., Information Systems in MA)

Description: User views the university which offers that course in particular state

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university offering that course in that state

System Responses – The list of universities offering that course is displayed

Post Condition: User can choose to view all the displayed universities

Alternate Path: If there is no university offering that particular course, it’ll display zero rows

**SQL: Select uni\_name from university where course like ‘%Computer Science%’ and state like ‘%California%’**

**Relational algebra:**

**πuni\_name  
 σcourse LIKE "%Computer Science%" AND state LIKE "%California%"university**

1. View the University deadline for the particular university

Description: User views the university deadline of the university

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the university link for the deadlines

System Responses – The deadlines for that intake will be displayed

Post Condition: User can choose to view all the deadlines(pre-requisite)

Alternate Path: If there is no university deadline, it’ll display zero rows

**SQL: Select fall\_deadline from university where uni\_name like ‘%NEU%’ or uni\_name like ‘%Northeastern%’**

**Relational algebra:**

**πfall\_deadline  
 σuni\_name LIKE "%NEU%" OR uni\_name LIKE "%Northeastern%"university**

1. View the University having the specified CGPA in particular course (e.g., 3.5, Data Science)

Description: User views the university that offer the course which needs that much GPA

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the courses accepting that much CGPA score.

System Responses – The list of courses with that much CGPA requirement is displayed

Post Condition: User can choose to view all the displayed university courses

Alternate Path: If there is no university with that specified CGPA, it’ll display zero rows

**SQL: Select uni\_name from university where min\_cgpa >= 7 and course like ‘%Information Systems%’**

**Relational algebra:**

**πuni\_name  
 σmin\_cgpa >= 7 AND course LIKE "%Information Systems%"university**

1. View the Universities fall deadlines for desired course

Description: User views the all the universities deadlines for the desired course

Actors: User

Precondition: User should be registered in FYDU account

Steps:

Actor action – User views the universities link for the deadlines

System Responses – The deadlines for that intake will be displayed

Post Condition: User can choose to view all the deadlines(pre-requisite)

Alternate Path: If there is no university link, it’ll display zero rows

**SQL: Select fall\_deadline, uni\_name from university where course like ‘%Information Systems%’**

**Relational algebra:**

**πfall\_deadline, uni\_name  
 σcourse LIKE "%Engineering Management%"university**