

W

O

R

L

T

—

Create the backend for Facebook for superheroes

Keep track of supervillans

MVP

Perform Full CRUD Operations
Prompt user for input to show
the list/table requested.

Add a character

Update character information

Remove character

Add friend

remove friend

add enemy

remove enemy

heros

id generated always as identity PK

name varchar(256) Unique NN

about_me varchar(512) Not Null

biography varchar(2048) Not Null

image_url varchar(64)

relationship_types

id int generated always as Identity PK

name varchar(64) Unique Not Null

heroes

id generated
name varchar(256)
about_me
biography
image

relationship_types

id int
label varchar(64)

Relationship

id int PK
hero1_id int
FK (hero1_id) References hero(id)
hero2_id int
FK (hero2_id) ref hero(id)
relationship_type_id int
FK (relationship_type_id) ref relationship_types(id)

ability_types

id int generated
label varchar(64)

Abilities

id int generated
hero_ID
FK (hero_ID) References heroes(id)
ability_type_id int
FK (ability_type_id) ref ability_types(id)

heroes
 id PK
 name
 about
 biography
 image

relationship
 id PK
 hero1_id FK (heroes.id)
 hero2_id FK (heroes.id)
 relationship_type FK

abilities
 id PK
 hero_id FK (heroes.id)
 ability_types_id FK (ability_types.id)

ability_types
 id PK
 label

relationship_types
 id PK
 label
 friend
 enemy

Queries to Test CRUD

✓ Add new hero ✓

Change the new heroes About me

✓ As a hero I want to add a friend to the relationships list ✓

✓ I want to look at a list of my relationships:
- Friends ✓
- villains ✓

✓ As a hero I want to add a villain to the relationships list ✓

✓ As a hero I want to change a hero relation to a villain ✓

✓ I want to remove myself from the database ✓

Add new hero

SQL Statement +

INSERT INTO heroes(

id GENERATED ALWAYS AS IDENTITY PK,
name VARCHAR(256) UNIQUE NOT NULL,
about_me VARCHAR(512) NOT NULL
biography VARCHAR(2048) NOT NULL
image_url VARCHAR(64)

)

name = input("What's your hero's name?")
about_me = input("What's your story?")
bio = input("What's your biography?")

~~etc.~~

eg (query, (name, about, bio))

query =

{INSERT INTO heroes (name, about_me, biography)
{VALUES ('Ryno Gardin', 'ipsum', 'longer ipsum')}}}

%s

%s

%s

1

\$1

\$2

\$3

eg query, (name, about, bio)

Change the new heroes About me

id = (what hero do we want to update)

new = input('What is new about you?')

def eq(query, (id, new))

(UPDATE users
SET about-me = new
WHERE id = %s)

id = input('Which id?')

execute_query(↓, (id,))

eq(query, (s,))

I want to look at a list of my relationships:

- Friends
- villains

~~hero~~ = user_id that is inquiring about relationships.

```
SELECT relationships.hero2_id,  
relationships_type_id
```

FROM Relationships

INNER JOIN relationship_type
ON heroes.id = relationship_type.hero1

WHERE hero1 = 95

[illegible]

I am (the hero) and I want a table
of my friends

I want a table of my enemies

hero = id# for which hero we
are using for hero1_id in Relationships
Table.

def eq(select

(name of Friend/enemy) (type for E)

heroes TABLE
name

Relationship_types
label

FROM ?

WHERE Relationships

hero1_id = hero

using (FK) hero2_id and (FK) rs-type_id
as references somehow

SELECT

```
FROM relationships r
JOIN heroes h1
  ON r.hero1_id = h1.id
JOIN heroes h2
  ON r.hero2_id = h2.id
JOIN relationship_types rt
  ON r.relationship_type_id = rt.id
WHERE h1.id = (what hero?)
or
WHERE h1.name = (what hero?)
```

✓


As a hero I want to add a
villain to the relationships
list

INSERT INTO relationships (

hero1_id,
hero2_id,
relationship_type_id)

VALUES(1,1,2)



Change  to a 1 and
add a friend.

As a hero I want to change
a hero relation to a villain

UPDATE relationships

SET relationship_type_id = 2

WHERE hero2_id = (Who do you dislike?)

I want to remove myself
from the database

```
DELETE  
FROM heroes  
WHERE id = (what hero?)
```

this removes all the hero from
the other tables as well.

