Homework #3
CS486/586
Prof. Charles Winstead
CRN: 40876

Jiacheng Zhao
Jordan Witte
Resheet Schultz
Due: 01-30-2019

Part I – Table Creation, Population, and Constraints (50 pts) (5 pts for (a), 3 pts (b)-(p))

Exercise 1

For the following exercises, you will be creating, modifying and querying SQL tables. For each item, show the SQL you used and the resulting state (all rows) of your table(s) (or the error message that SQL returns). Do all these tasks using SQL statements (not a GUI). You will be using data in 'Avengers src w19' posted on Piazza (adapted from superherodb.com).

a) *Create a Heroes table with columns for Character Name, Intelligence, Strength, Speed, Durability, Combat, and Species.*

(a) With Character Name as the primary key

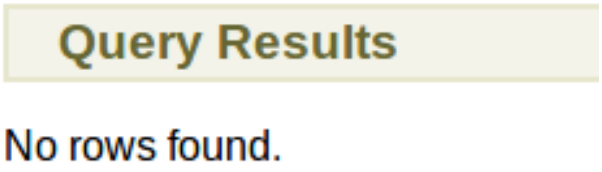
(b) Intelligence/Speed/Durability/Combat limited in value from 0-100

(c) Species limited to "Human, Mutant, Android, or Asgardian"

```
CREATE TABLE Hero (
  character_name VARCHAR(60) PRIMARY KEY,

  intelligence integer CONSTRAINT intelligence_range CHECK (intelligence >= 0 AND
      intelligence <= 100),
  strength integer,
  speed integer CONSTRAINT speed_range CHECK (speed >= 0 AND speed <= 100),
  durability integer CONSTRAINT durability_range CHECK (durability >= 0 AND
      durability <= 100),
  combat integer CONSTRAINT combat_range CHECK (combat >= 0 AND combat <= 100),
  species varchar(60) CONSTRAINT species_values CHECK (species in ('Human', 'Mutant',
      'Android', 'Asgardian'))
);
```

Result:



b) *Insert rows for all characters with Durability greater than 70*

```
INSERT INTO Hero VALUES ('Iron Man', 95, 500000, 60, 80, 80, 'Human');
INSERT INTO Hero VALUES ('Thor', 70, 10000000, 75, 100, 95, 'Asgardian');
INSERT INTO Hero VALUES ('Hulk', 20, 20000000, 50, 95, 100, 'Mutant');
INSERT INTO Hero VALUES ('Vision', 85, 165000, 55, 90, 90, 'Android');
```

character_name	intelligence	strength	speed	durability	combat	species
Iron Man	95	500000	60	80	80	Human
Thor	70	10000000	75	100	95	Asgardian
Hulk	20	20000000	50	95	100	Mutant
Vision	85	165000	55	90	90	Android

c) *Modify your table to add columns for* ***Real Name*** *and* ***Joined In****.*

```
alter table Hero
  add real_name varchar(60),
  add joined_in date;
```

character_name	intelligence	strength	speed	durability	combat	species	real_name	joined_in
Iron Man	95	500000	60	80	80	Human	*NULL*	*NULL*
Thor	70	10000000	75	100	95	Asgardian	*NULL*	*NULL*
Hulk	20	20000000	50	95	100	Mutant	*NULL*	*NULL*
Vision	85	165000	55	90	90	Android	*NULL*	*NULL*

d) *Update the existing rows in the table to add* ***Real Name*** *and* ***Joined In*** *information.*

```
UPDATE Hero SET real_name = 'Anthony Edward "Tony" Stark',
    joined_in = date('9/1/1963')
    WHERE character_name = 'Iron Man';
UPDATE Hero SET real_name = 'Thor Odinson/Dr. Donald Blake',
    joined_in = date('9/1/1963')
    WHERE character_name = 'Thor';
UPDATE Hero SET real_name = 'Dr. Robert Bruce Banner',
    joined_in = date('9/1/1963')
    WHERE character_name = 'Hulk';
UPDATE Hero SET real_name = 'Victor Shade',
```

```
    joined_in = date('11/1/1968')
    WHERE character_name = 'Vision';
```

character_name	intelligence	strength	speed	durability	combat	species	real_name	joined_in
Iron Man	95	500000	60	80	80	Human	Anthony Edward "Tony" Stark	1963-09-01
Thor	70	10000000	75	100	95	Asgardian	Thor Odinson/Dr. Donald Blake	1963-09-01
Hulk	20	20000000	50	95	100	Mutant	Dr. Robert Bruce Banner	1963-09-01
Vision	85	165000	55	90	90	Android	Victor Shade	1968-11-01

e) *Insert rows for characters with Combat equal to 60.*

```
INSERT INTO Hero VALUES ('Captain America', 60, 1500, 40, 55, 60, 'Human', 'Steven
    "Steve" Rogers', date('3/1/1964'));
INSERT INTO Hero VALUES ('Black Panther', 90, 800, 70, 70, 60, 'Human', 'T''Challa',
    date('5/1/1968'));
```

character_name	intelligence	strength	speed	durability	combat	species	real_name	joined_in
Iron Man	95	500000	60	80	80	Human	Anthony Edward "Tony" Stark	1963-09-01
Thor	70	10000000	75	100	95	Asgardian	Thor Odinson/Dr. Donald Blake	1963-09-01
Hulk	20	20000000	50	95	100	Mutant	Dr. Robert Bruce Banner	1963-09-01
Vision	85	165000	55	90	90	Android	Victor Shade	1968-11-01
Captain America	60	1500	40	55	60	Human	Steven "Steve" Rogers	1964-03-01
Black Panther	90	800	70	70	60	Human	T'Challa	1968-05-01

f) *What happens if you try to insert **Hulk** a second time?*

The command

```
INSERT INTO Hero VALUES ('Hulk', 20, 20000000, 50, 95, 100, 'Mutant');
```

fails because the new tuple shares a primary key value with an existing tuple, and returns this error message:

SQL error:

```
ERROR:  duplicate key value violates unique constraint "hero_pkey"
DETAIL:  Key (character_name)=(Hulk) already exists.
```

In statement:

```
INSERT INTO Hero VALUES ('Hulk', 20, 20000000, 50, 95,  100, 'Mutant');
```

g) *Write a query to find all characters with a space in their Character Name and how long they've been in the Avengers*

```
  SELECT character_name, years - 1*incomplete_year::INT years , months
      +12*incomplete_year::INT months, days FROM
(SELECT character_name,
DATE_PART('year', CURRENT_DATE) - date_part('year' , joined_in) -1 AS years,
DATE_PART('month', CURRENT_DATE) - date_part('month' , joined_in) AS months,
DATE_PART('day', CURRENT_DATE) - date_part('day' , joined_in) AS days,
DATE_PART('month', CURRENT_DATE) - date_part('month' , joined_in) < 0 AS incomplete_year
```

```
FROM hero WHERE character_name LIKE '\% \%') a;
```

character_name	years	months	days
Iron Man	54	4	28
Captain America	53	10	28
Black Panther	49	8	28

h) *Create a second table to hold the list of available powers, with a single column, called power, where power is unique.*

```
CREATE TABLE Powers (power varchar(60) unique);
```

Query Results

No rows found.

i) *Create a third table to hold each Character's Powers, with columns for Character Name and Character Power, with Character Name as a foreign key to the first table, and Character Power a foreign key to the second table.*

```
CREATE TABLE HeroPower(
  character_name varchar(60) REFERENCES Hero(character_name),
  power varchar(60) REFERENCES Powers(power));
```

Query Results

No rows found.

j) *Insert rows in the second table corresponding to all possible powers.*

```
INSERT INTO Powers VALUES ('Energy Blasts');
INSERT INTO Powers VALUES ('Flight');
INSERT INTO Powers VALUES ('Super Strength');
INSERT INTO Powers VALUES ('Accelerated Healing');
INSERT INTO Powers VALUES ('Martial Arts');
INSERT INTO Powers VALUES ('Weapons');
INSERT INTO Powers VALUES ('Leadership');
INSERT INTO Powers VALUES ('Marksmanship');
INSERT INTO Powers VALUES ('Stealth');
```

power
Energy Blasts
Flight
Super Strength
Accelerated Healing
Martial Arts
Weapons
Leadership
Marksmanship
Stealth

k) *Insert rows in the third table corresponding to all characters in the first table. For characters with multiple powers, each power should be listed separately.*

```
INSERT INTO HeroPower VALUES ('Iron Man', 'Energy Blasts');
INSERT INTO HeroPower VALUES ('Iron Man', 'Flight');
INSERT INTO HeroPower VALUES ('Iron Man', 'Super Strength');
INSERT INTO HeroPower VALUES ('Thor', 'Flight');
INSERT INTO HeroPower VALUES ('Thor', 'Energy Blasts');
INSERT INTO HeroPower VALUES ('Thor', 'Super Strength');
INSERT INTO HeroPower VALUES ('Hulk', 'Accelerated Healing');
INSERT INTO HeroPower VALUES ('Hulk', 'Super Strength');
INSERT INTO HeroPower VALUES ('Captain America', 'Martial Arts');
INSERT INTO HeroPower VALUES ('Captain America', 'Weapons');
INSERT INTO HeroPower VALUES ('Captain America', 'Leadership');
INSERT INTO HeroPower VALUES ('Vision', 'Energy Blasts');
INSERT INTO HeroPower VALUES ('Vision', 'Flight');
INSERT INTO HeroPower VALUES ('Vision', 'Super Strength');
INSERT INTO HeroPower VALUES ('Black Panther', 'Martial Arts');
INSERT INTO HeroPower VALUES ('Black Panther', 'Stealth');
INSERT INTO HeroPower VALUES ('Black Panther', 'Leadership');
```

character_name	power
Iron Man	Energy Blasts
Iron Man	Flight
Iron Man	Super Strength
Thor	Flight
Thor	Energy Blasts
Thor	Super Strength
Hulk	Accelerated Healing
Hulk	Super Strength
Captain America	Martial Arts
Captain America	Weapons
Captain America	Leadership
Vision	Energy Blasts
Vision	Flight
Vision	Super Strength
Black Panther	Martial Arts
Black Panther	Stealth
Black Panther	Leadership

l) *What happens if you try to insert* ***Stealth*** *as a power for* ***Black Widow****?*

The command

```
INSERT INTO HeroPower VALUES ('Black Widow', 'Stealth');
```

fails due to a missing foreign key value, with the error message:

SQL error:

```
ERROR:  insert or update on table "heropower" violates foreign key constraint "heropower_character_name_fkey"
DETAIL:  Key (character_name)=(Black Widow) is not present in table "hero".
```

In statement:
INSERT INTO HeroPower VALUES ('Black Widow', 'Stealth');

m) *What happens if you try to delete the row in first table for* ***Iron Man****?*

The command

```
DELETE FROM Hero WHERE character_name = 'Iron Man' ;
```

fails due to other entries referencing the entry with a foreign key, and returns the error message:

SQL error:

```
ERROR:  update or delete on table "hero" violates foreign key constraint "heropower_character_name_fkey" on table "heropower"
DETAIL:  Key (character_name)=(Iron Man) is still referenced from table "heropower".
```

In statement:

```
DELETE FROM Hero WHERE character_name = 'Iron Man' ;
```

n) *What happens if you try to modify change the value of* ***Combat*** *for* ***Thor*** *to 110?*

The command

```
UPDATE Hero SET combat=110 WHERE character_name = 'Thor' ;
```

fails due to the domain constraint on 'combat', and returns the error message:

SQL error:

```
ERROR:  new row for relation "hero" violates check constraint "combat_range"
DETAIL:  Failing row contains (Thor, 70, 10000000, 75, 100, 110, Asgardian, Thor Odinson/Dr. Donald Blake, 1963-09-01).
```

In statement:

```
UPDATE Hero SET combat=110 WHERE character_name = 'Thor' ;
```

o) *Write a query to find the total Combat amount for all characters who can fly.*

```
SELECT SUM(combat) FROM Hero NATURAL JOIN HeroPower WHERE power='Flight';
```

sum
265

p) *Write a query to find the character with the highest combination of* ***Speed*** *and* ***Intelligence****.*

```
SELECT character_name from Hero where speed+intelligence = (SELECT
    max(speed+intelligence) from Hero);
```

character_name
Black Panther

Part II Views (25 pts)

Exercise 2

Create an SQL view definition for a table:

```
PowerInfo(power, num_heros, max_intel, max_speed, max_dur)
```

that lists the number of different superhero having each power and the maximum intelligence, speed, and durability of the characters who have that power.

Show your CREATE VIEW statement and the full table that your view generates. (10 pts)

```
CREATE VIEW PowerInfo AS
  SELECT power,
        COUNT(*) AS num_heros,
        MAX(intelligence) AS max_intel,
        MAX(speed) AS max_speed,
        MAX(durability) as max_dur
    FROM Powers NATURAL JOIN HeroPower NATURAL JOIN Hero
    GROUP BY power;
```

power	num_heros	max_intel	max_speed	max_dur
Leadership	2	90	70	70
Super Strength	4	95	75	100
Flight	3	95	75	100
Martial Arts	2	90	70	70
Accelerated Healing	1	20	50	95
Energy Blasts	3	95	75	100
Weapons	1	60	40	55
Stealth	1	90	70	70

Exercise 3

Write an SQL query that finds the names of the Smartest, Fastest, and Toughest heroes that have each power. Use the PowerInfo view you defined in the previous question. Show your query and the result. (15 pts)

```
SELECT powers.power, smartest, toughest, fastest FROM
(SELECT power FROM powers) AS powers

JOIN
(SELECT powers.power, S.character_name as fastest
FROM Hero AS S
NATURAL JOIN powers
NATURAL JOIN heropower
NATURAL JOIN powerinfo
WHERE powerinfo.max_speed = S.speed ) AS best_at_speed
ON powers.power=best_at_speed.power

JOIN
(SELECT powers.power, T.character_name AS toughest
FROM Hero AS T
NATURAL JOIN powers
```

```
NATURAL JOIN heropower
NATURAL JOIN powerinfo
WHERE powerinfo.max_dur = T.durability ) AS best_at_durability
ON powers.power = best_at_durability.power

JOIN
(SELECT powers.power, S.character_name AS smartest
FROM Hero AS S
NATURAL JOIN powers
NATURAL JOIN heropower
NATURAL JOIN powerinfo
WHERE powerinfo.max_intel= S.intelligence) AS best_at_intel
ON powers.power = best_at_intel.power;
```

power	smartest	toughest	fastest
Energy Blasts	Iron Man	Thor	Thor
Flight	Iron Man	Thor	Thor
Super Strength	Iron Man	Thor	Thor
Accelerated Healing	Hulk	Hulk	Hulk
Weapons	Captain America	Captain America	Captain America
Martial Arts	Black Panther	Black Panther	Black Panther
Stealth	Black Panther	Black Panther	Black Panther
Leadership	Black Panther	Black Panther	Black Panther

Part III, using the Spy relational database (25 pts)

Exercise 4

List the number of agents and total salary for agents in each team with more than 5 agents. (5 pts)

```
SELECT team.team_id, team.name, COUNT(*) num_agents, SUM(salary) total_salary
FROM agent
NATURAL JOIN team
NATURAL JOIN teamrel
GROUP BY team_id
HAVING COUNT(*) > 5;
```

team_id	name	num_agents	total_salary
29	Ghost Hunters	9	669290
4	Oink	8	484011
34	Boat Team 4	9	709197
40	F Sharp	10	710818
32	Boat Team 2	10	999024
10	Blackout	10	1125828
9	BumbleBee	9	713755
7	FlyOnTheWall	10	757974
35	Boat Team 6	8	539849
38	Scorpion	9	506824
15	Failsafe	10	1094412
6	Timebomb	9	738654
26	Thunderbird	9	588656
12	SpecialForces	9	567882
39	Blue Dagger	8	664551
24	Blunt	10	1032435
19	ShowBiz	9	580350
36	Boat Team 7	8	1008056
25	Camaro	8	612684
31	Boat Team 1	10	868482
30	Widow Makers	9	900159
21	Rimspeed	9	609671
14	Beasties	10	614608
3	Roadkill	10	687112
17	Rolaids	10	759506
37	Jester	8	520069
28	Cha Cha Cha	10	857505
22	Leadphut	8	612848
20	Giraffe	9	680389
33	Boat Team 3	9	950638
13	Gypsies	10	631051
1	Renegade	10	1070201
5	Blaster	8	650351
18	SqueakyClean	8	967741
2	Haberdash	10	797287
16	Vikings	10	783381
27	Swing Voters	10	894397
23	Charley Hunter	10	649829
11	Cyclone	10	644527
8	Terminator	10	884585

40 row(s)

Exercise 5

List the agents (id, first, last) who have been on more than one 'Top Secret' mission that was a failure. Write this query two ways, once using EXISTS and once using IN. (10 pts)

Solution using **IN**:

```
SELECT first, last, agent_id FROM agent
WHERE agent_id IN
(SELECT agent.agent_id
FROM agent
JOIN teamrel ON agent.agent_id = teamrel.agent_id
JOIN team ON teamrel.team_id = team.team_id
JOIN mission ON team.team_id = mission.team_id
JOIN securityclearance sc ON mission.access_id = sc.sc_id
WHERE sc.sc_level = 'Top Secret' AND mission_status='failed'
GROUP BY agent.agent_id HAVING COUNT(*)>=2
) ;
```

Solution using **EXISTS**:

```
SELECT first, last, agent_id FROM agent a
WHERE EXISTS (SELECT agent.agent_id
FROM agent
JOIN teamrel ON agent.agent_id = teamrel.agent_id
JOIN team ON teamrel.team_id = team.team_id
JOIN mission ON team.team_id = mission.team_id
JOIN securityclearance sc ON mission.access_id = sc.sc_id
WHERE sc.sc_level = 'Top Secret' AND mission_status='failed'
AND a.agent_id=agent.agent_id
GROUP BY agent.agent_id having count(*)>=2
)
```

first	last	agent_id
David	Shapiro	59
George	Yang	70
Pete	Heinlein	81
Tim	Brock	87
Julien	Sibbett	101
Mary	Baker	113
Nick	House	141
Richard	Venkatesh	168
Travis	Balasubramanian	174
Bob	Ferguson	186
George	Williams	320
Charles	Morris	367
Ethan	Adkins	437
Tim	Ness	476
Bill	White	488
Tim	Divola	498
George	Welch	521
George	Flood	560
Nicholas	Larsen	581
Robert	Roberts	587
Crispin	Rios	604
Richard	Salazar	702
Mark	Lincoln	719
Ken	Kennedy	730
Max	Lugar	733
Jeff	Domenici	753
Chuck	Bunning	785
Barbara	Allen	788
Travis	Drabowsky	818
Ethan	Derrick	899
Aristidis	Matchick	911
Maric	Curtis	947
Chris	May	1007
Jonathan	May	1073
Chris	O'brien	1085

35 row(s)

The statement using **IN** runs faster (at about 6 ms as opposed to 60 ms on phpPgAdmin) because the subquery runs only once (is not a correlated subquery) for the whole execution. The **EXISTS** version runs the subquery for every entry in the agent table, and each call of the subquery returns a different value (this is a correlated subquery).

Exercise 6

Find the skill name and the number of agents with that skill for the skill(s) with the fewest agents. (10 pts)
Useful information: You can use subqueries in the FROM clause and the HAVING clause.
Example subquery in the HAVING clause:

```
SELECT skill, COUNT(*) AS num_agents_with_skill
FROM skill
NATURAL JOIN skillrel
GROUP BY skill HAVING COUNT(*) =
  (SELECT MIN(n.c)
    FROM (SELECT COUNT(*) AS c
          FROM skill
          NATURAL JOIN skillrel
          GROUP BY skill) AS n);
```

skill	num_agents_with_skill
Sniper	3

Extra credit (5 pts)

Exercise 7

Write one or more DELETE statements that remove all Heroes with the 'Accelerated Healing' Power. Note that removing these heroes will also mean removing their HeroPower information. Show your DELETE statement(s).

Deleting the entries from **hero** is initially impossible because entries in **heropower** reference the hero name(s) as a foreign key. But we need to use those entries in **heropower** to know which entries in **hero** need to be removed.

We can get around this by adjusting the foreign key constraint to CASCADE on delete. Then the relevant entries in **heropower** get removed automatically. After the delete let's assume that we want to restore the original settings on the foreign key constraint.

```
ALTER TABLE heropower
  DROP CONSTRAINT heropower_character_name_fkey;
ALTER TABLE heropower
  ADD CONSTRAINT heropower_character_name_fkey FOREIGN KEY (character_name) REFERENCES
      hero(character_name) ON DELETE CASCADE;

DELETE FROM hero WHERE character_name IN
  (SELECT character_name FROM heropower
    WHERE power = 'Accelerated Healing');

ALTER TABLE heropower
  DROP CONSTRAINT heropower_character_name_fkey;
ALTER TABLE heropower
  ADD CONSTRAINT heropower_character_name_fkey FOREIGN KEY (character_name) REFERENCES
      hero(character_name);
```