Mario Super Sluggers

The quest to rescue Baseball Kingdom!

Trevor Pirone
DBMS 308
12/04/17

Table of Contents

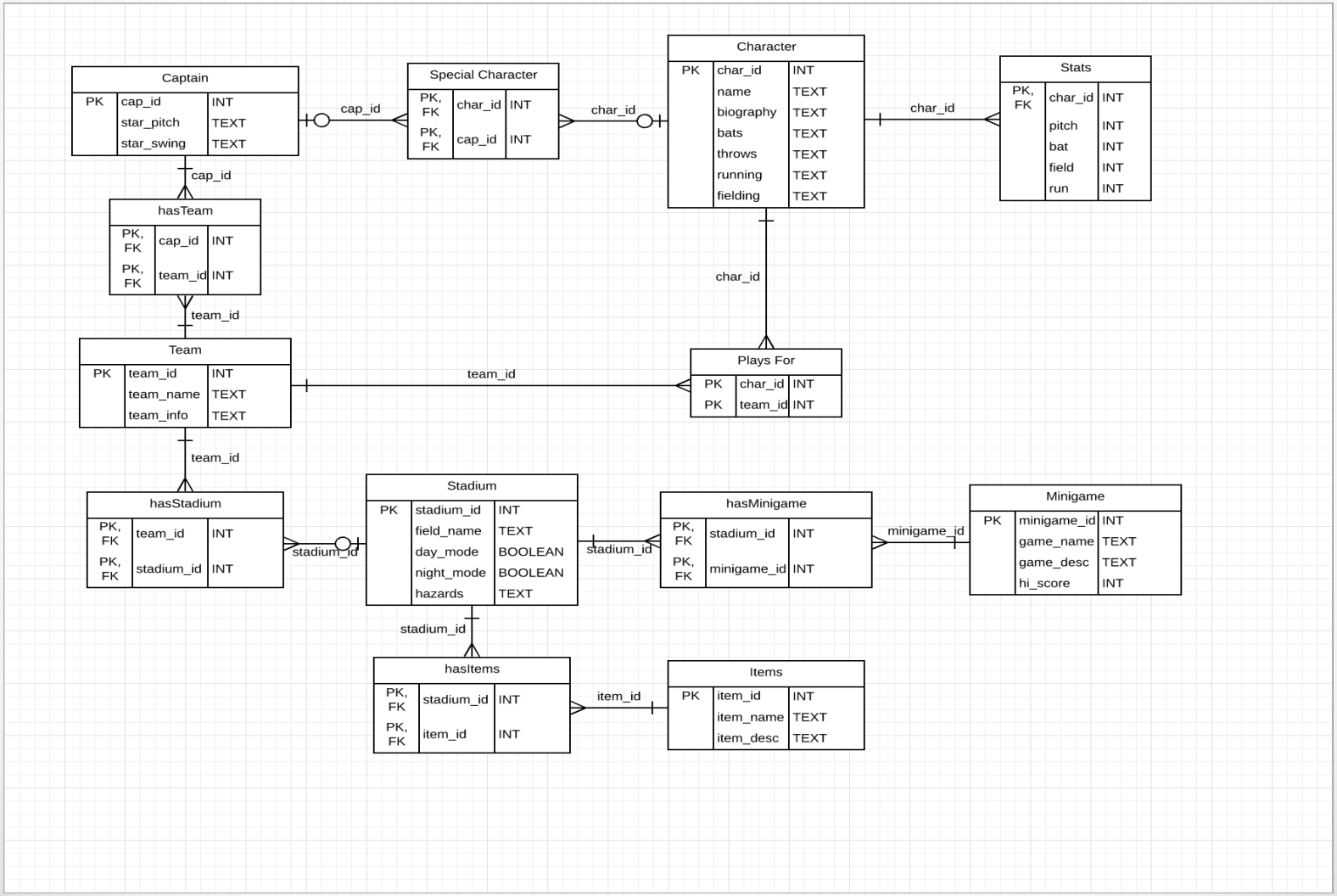
Executive Summary

Mario Super Sluggers is the Wii sequel to the GCN game Mario Superstar Baseball. Not only are Mario and his friends tasked to relinquish Baseball Kingdom back from the wrath of Bowser's evil claws, he was also tasked to create a database to keep track of all of the players, teams, items, stadiums, captains, and minigames on the island. Due to Mario's occupation (plumber), he is not technically competent enough to do so which is why he hired Trevor Pirone (that's me) to create it for him.

The database consists of create statements for all elements that appear in the game (i.e. Characters, Stats, Captains, Teams, Stadiums, etc.). From these tables, queries, views, triggers, and stored procedures are used to retrieve more in-depth information about the game (i.e. Return the characters that play for the Mario Fireballs.) Specific users have roles that can modify the database in such a way that is needed to accommodate any changes that occur in the kingdom (i.e. creating a new player, adding a new team, removing an item, etc.)

Due to some extra requirements that are needed at the request of Trevor's boss, he added an additional character and item in the game,
Alan "The Pleasant Valley Pulverizer" Labouseur and his magical staff!

ER Diagram


Captain
PK cap_id INT
star_pitch TEXT
star_swing TEXT
cap_id
Special Character
PK, FK char_id INT
PK, FK cap_id INT
char_id
Character
PK char_id INT
name TEXT
biography TEXT
bats TEXT
throws TEXT
running TEXT
fielding TEXT
char_id
Stats
PK, FK char_id INT
pitch INT
bat INT
field INT
run INT
cap_id
hasTeam
PK, FK cap_id INT
PK, FK team_id INT
team_id
char_id
Team
PK team_id INT
team_name TEXT
team_info TEXT
team_id
Plays For
PK char_id INT
PK team_id INT
team_id
hasStadium
PK, FK team_id INT
PK, FK stadium_id INT
stadium_id
Stadium
PK stadium_id INT
field_name TEXT
day_mode BOOLEAN
night_mode BOOLEAN
hazards TEXT
stadium_id
hasMinigame
PK, FK stadium_id INT
PK, FK minigame_id INT
minigame_id
Minigame
PK minigame_id INT
game_name TEXT
game_desc TEXT
hi_score INT
stadium_id
hasItems
PK, FK stadium_id INT
PK, FK item_id INT
item_id
Items
PK item_id INT
item_name TEXT
item_desc TEXT

Characters Tables

The characters tables includes all of the characters in the game.

```
CREATE TABLE Characters (
        char_id INT NOT NULL,
        name TEXT NOT NULL,
        biography TEXT NOT NULL,
        bats TEXT NOT NULL,
        throws TEXT NOT NULL,
        running TEXT NOT NULL,
        fielding TEXT NOT NULL,
        UNIQUE(char_id),
        PRIMARY KEY(char_id)
);
```

	char_id integer	name text	biography text	bats text	throws text	running text	fielding text
1	1	Mario	A 5-tool player who can hit, run, and pitch!	Right	Right	Enlarge	None
2	2	Luigi	A 5-tool player who can jump, pitch, and hit!	Left	Right	None	Super Jump
3	3	Peach	A slap hitter who can pitch the lights out.	Right	Right	None	Quick Throw
4	4	Daisy	Her amazing glove is her best feature.	Left	Right	None	Super Dive
5	5	Yoshi	Top-class speed in the field and on base.	Left	Right	None	Tongue Catch
6	6	Birdo	A good player who has trouble hitting curves.	Right	Right	None	Suction Catch
7	7	Wario	Good stamina and a great contact zone.	Right	Right	None	Laser Beam
8	8	Waluigi	Charge swings drop power, but increase hits.	Left	Left	None	Quick Throw
9	9	Donkey Kong	A strong hitter who can also climb walls.	Left	Right	None	Clamber
10	10	Diddy Kong	His hits and pitches move like crazy!	Left	Right	None	Clamber
11	11	Bowser	His speed and stamina baffles batters!	Left	Right	Spin Attack	None
12	12	Bowser Jr.	A pitcher with great ball and bat control!	Right	Left	Spin Attack	None

	char_id integer	name text	biography text	bats text	throws text	running text	fielding text
63	63	Fire Bro	A player with a mighty charge swing!	Right	Right	None	Fire Throw
64	64	Boomerang Bro	A player who throws a mean curveball!	Right	Right	None	Bommerang Throw
65	65	Green Dry Bones	A powerful Dry Bones with quick feet.	Left	Left	Scatter Dive	None
66	66	Dark Bones	A powerful Dry Bones with a good arm.	Left	Left	Scatter Dive	None
67	67	Blue Dry Bones	A powerful Dry Bones with a good glove.	Left	Left	Scatter Dive	None
68	68	Blue Shy Guy	A Shy Guy who loves to pitch!	Left	Right	None	Super Dive
69	69	Yellow Shy Guy	A Shy Guy who loves to steal!	Left	Right	None	Super Dive
70	70	Green Shy Guy	A Shy Guy who loves to hit and run!	Left	Right	None	Super Dive
71	71	Gray Shy Guy	A Shy Guy who loves to play the field!	Left	Right	None	Super Dive
72	72	Male Mii	A 5-tool player that can do it all!	Right	Right	None	Quick Throw
73	73	Female Mii	A 5-tool player that can do it all!	Left	Right	None	Quick Throw
74	74	Alan Labouseur	A straight up savage and boss!	Both	Both	Enlarge	Super Jump

Functional Dependencies

char_id → name
char_id → biography
char_id → bats
char_id → throws
char_id → running
char_id → fielding

Captains Table

The captains table includes all of the players that are captains of a team.

```
CREATE TABLE Captains (
        cap_id INT NOT NULL,
        star_pitch TEXT NOT NULL,
        star_swing TEXT NOT NULL,
        UNIQUE(cap_id),
        PRIMARY KEY(cap_id)
);
```

Functional Dependencies:

cap_id → star_pitch

cap_id → star_swing

	cap_id integer	star_pitch text	star_swing text
1	1	Fireball	Fire Swing
2	2	Tornado Ball	Tornado Swing
3	3	Heart Ball	Heart Swing
4	4	Flower Ball	Flower Swing
5	5	Rainbow Ball	Egg Swing
6	6	Suction Ball	Cannon Swing
7	7	Phony Ball	Phony Swing
8	8	Liar Ball	Liar Swing
9	9	Barrel Ball	Barrel Swing
10	10	Banana Ball	Banana Swing
11	11	Killer Ball	Breath Swing
12	12	Grafitti Ball	Grafitti Swing

Special Characters Table

This table acts as a link between the Characters and Captains tables. It represents the characters in the game that are also captains of a team.

```
CREATE TABLE Special_Characters (
        char_id INT REFERENCES Characters(char_id),
        cap_id INT REFERENCES Captains(cap_id),
        PRIMARY KEY(char_id, cap_id)
);
```

Functional Dependencies:

char_id, cap_id →

	char_id integer	cap_id integer
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12

Stats Table

The stats table keeps track of a player's skill level in four categories. Each stat is ranked on a score out of 10.

```
CREATE TABLE Stats (
        char_id INT REFERENCES Characters(char_id),
        pitch INT NOT NULL,
        CHECK (pitch >= 0),
        bat INT NOT NULL,
        CHECK (bat >= 0),
        field INT NOT NULL,
        CHECK (field >= 0),
        run INT NOT NULL,
        CHECK (run >= 0),
        UNIQUE(char_id),
        PRIMARY KEY(char_id)
);
```

Functional Dependencies:

char_id → pitch
char_id → bat
char_id → field
char_id → run

	char_id integer	pitch integer	bat integer	field integer	run integer
20	20	7	3	7	3
21	21	5	8	4	2
22	22	5	4	4	7
23	23	3	3	8	7
24	24	4	8	6	3
25	25	5	5	7	5
26	26	6	10	2	1
27	27	4	7	7	3
28	28	6	3	6	4
29	29	6	3	7	5
30	30	3	6	4	6
31	31	4	4	7	5

	char_id integer	pitch integer	bat integer	field integer	run integer
32	32	8	2	8	2
33	33	4	7	6	3
34	34	4	7	4	5
35	35	9	3	3	5
36	36	7	7	3	4
37	37	4	10	5	2
38	38	3	7	4	7
39	39	4	5	7	4
40	40	4	4	5	7
41	41	6	4	5	6
42	42	3	4	4	8
43	43	4	2	6	8

Teams Table

The teams table includes all of the teams that can be played with in the game.

```
CREATE TABLE Teams (
        team_id INT NOT NULL,
        team_name TEXT NOT NULL,
        team_info TEXT NOT NULL,
        UNIQUE(team_id),
        PRIMARY KEY(team_id)
);
```

Functional Dependencies:

team_id → team_name

team_id → team_info

	team_id integer	team_name text	team_info text
1	0	Free Agent	Free Agent
2	1	Mario Fireballs	The Mario Fireballs are well balanced.
3	2	Luigi Knights	No description exists.
4	3	Peach Monarchs	The Peach Monarchs excel at pitching and defense.
5	4	Daisy Flowers	No description exists.
6	5	Yoshi Eggs	The Yoshi Eggs rely on their great team speed.
7	6	Birdo Bows	No descriptiuon exists.
8	7	Wario Muscles	The Wario Muscles are an odd, tricky bunch.
9	8	Waluigi Spitballs	No description exists.
10	9	DK Wilds	The DK Wilds rely on power and defense.
11	10	Diddy Monkeys	No description exists.
12	11	Bowser Monsters	The Bowser Monsters are a balanced and mighty team.
13	12	Bowser Jr. Rookies	No description exists.

has Team Table

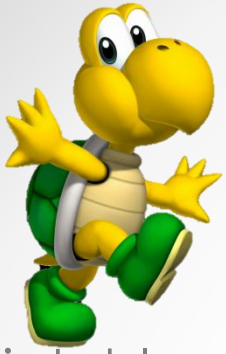
This table acts as a link between the Captains and Teams tables. It represents the teams that are owned by a captain.

```
CREATE TABLE hasTeam (
        cap_id INT REFERENCES Captains(cap_id),
        team_id INT REFERENCES Teams(team_id),
        PRIMARY KEY(cap_id, team_id)
);
```

Functional Dependencies:

cap_id, team_id →

	cap_id integer	team_id integer
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12



plays For Table

This table acts as a link between the Characters and Teams tables. It represents the rosters for each team or what characters play on which team.

```
CREATE TABLE playsFor (
        char_id INT REFERENCES Characters(char_id),
        team_id INT REFERENCES Teams(team_id),
        PRIMARY KEY(char_id, team_id)
);
```

Functional Dependencies:

char_id, team_id →

	char_id integer	team_id integer
46	46	5
47	47	3
48	48	3
49	49	3
50	50	3
51	51	1
52	52	1
53	53	1
54	54	1
55	55	9
56	56	9
57	57	9

	char_id integer	team_id integer
59	59	7
60	60	11
61	61	11
62	62	11
63	63	11
64	64	11
65	65	11
66	66	11
67	67	11
68	68	5
69	69	5
70	70	5

Stadiums Table

The stadiums tables includes all of the stadiums that teams can play within for games.

```
CREATE TABLE Stadiums (
        stadium_id INT NOT NULL,
        field_name TEXT NOT NULL,
        day_mode BOOLEAN NOT NULL,
        night_mode BOOLEAN NOT NULL,
        hazards TEXT NOT NULL,
        UNIQUE(stadium_id),
        PRIMARY KEY(stadium_id)
);
```

	stadium_id integer	field_name text	day_mode boolean	night_mode boolean	hazards text
1	0	None	f	f	None
2	1	Mario Stadium	t	t	None
3	2	Peach Ice Garden	t	t	Freezies/Snowflakes
4	3	Yoshi Park	t	t	Steam Train/Warp Pipes
5	4	Wario City	t	t	Manholes/Directional Arrows
6	5	DK Jungle	t	t	Barrels/Giant Flowers
7	6	Luigi's Mansion	f	t	Gravestones/Tall Grass
8	7	Daisy Cruiser	t	t	Dining Tables/Cheep Cheeps/Glooper Blooper
9	8	Bowser Jr. Playroom	t	f	Thwomps/ Chain Chomps/Bullet Bills
10	9	Bowser Castle	f	t	Lava Puddles/King Bob-omb/Bowser Statue

Functional Dependencies:

stadium_id → field_name

stadium_id → day_mode

stadium_id → night_mode

stadium_id → hazards

has Stadium Table

This table acts as a link between the Teams and Stadiums tables. It represents all of the teams that own their own stadium.

```
CREATE TABLE hasStadium (
        team_id INT REFERENCES Teams(team_id),
        stadium_id INT REFERENCES Stadiums(stadium_id),
        PRIMARY KEY(team_id, stadium_id)
);
```

Functional Dependencies:

team_id, stadium_id →

	team_id integer	stadium_id integer
1	1	1
2	2	6
3	3	2
4	4	7
5	5	3
6	6	0
7	7	4
8	8	0
9	9	5
10	10	0
11	11	9
12	12	8

Items Table

The items table includes all of the items players can use in the game on offense to attack the defense.

```
CREATE TABLE Items (
        item_id INT NOT NULL,
        item_name TEXT NOT NULL,
        item_desc TEXT NOT NULL,
        UNIQUE(item_id),
        PRIMARY KEY(item_id)
);
```

Functional Dependencies:

item_id → item_name

item_id → item_desc

	item_id integer	item_name text	item_desc text
1	1	Green Shell	Shoots at a player and dazes if it hits.
2	2	Fire Ball	Some fireballs bounce to burn someone.
3	3	Mini Boos	Makes the ball and its shadow invisible for about six seconds. The ball
4	4	Bob-omb	When a Bob-omb gets shot in the outfield, it sits and stay until it exp
5	5	POW-Ball	All fielders are stunned by a small earthquake for about four seconds w
6	6	Banana Peel	Five bananas are shot out to slip the outfielders.
7	7	Lightning Bolt	Will strike any characters on the screen.
8	8	Alan's Magic Staff	All max stats on every player for the rest of the inning. Very rare!

Minigames Table

The minigames tables includes all of the minigames that can be played in the game.


```
CREATE TABLE Minigames (
        minigame_id INT NOT NULL,
        game_name TEXT NOT NULL,
        game_desc TEXT NOT NULL,
        hi_score INT NOT NULL,
        CHECK (hi_score >= 0),
        UNIQUE(minigame_id),
        PRIMARY KEY(minigame_id)
);
```

	minigame_id integer	game_name text	game_desc text	hi_score integer
1	1	Bob-omb Derby	Hit bob-ombs into the sky and earn as many points as you can!	6000
2	2	Wall Ball	Thorw the ball at the walls to break them and earn the most points!	5000
3	3	Piranha Panic	Prevent the piranha plants from reaching your player by throwing baseballs at them!	4000
4	4	Gem Catch	Maneuver the player to catch gems to earn points! The more valuable gem, the more points can be earned!	3500
5	5	Barrel Basher	Fend off barrels and bob-ombs by hitting them with baseballs protecting your shield!	2500
6	6	Ghost K	Throw baseballs at the ghosts on the screen to make them disappear!	3000
7	7	Blooper Baserun	Collect the most coins by ruinning around the bases! Be careful! Don't hit the Glooper Blooper's tentacles!	4000
8	8	Grafitti Runner	Paint as much of the canvas as you can by running around with the paintbrush!	3500
9	9	Bowser Pinball	Keep a spiked ball within the area and hit coins and walls to earn points!	5000

Functional Dependencies:

minigame_id → game_name

minigame_id → game_desc

minigame_id → hi_score

has Items Table

This tables acts as a link between the Stadiums and Items tables. It represents all of the items a player can receive whilst playing a game at a stadium.

```
CREATE TABLE hasItems (
        stadium_id INT REFERENCES Stadiums(stadium_id),
        item_id INT REFERENCES Items(item_id),
        PRIMARY KEY(stadium_id, item_id)
);
```

Functional Dependencies:

stadium_id, item_id →

	stadium_id integer	item_id integer
11	2	3
12	2	4
13	2	5
14	2	6
15	2	7
16	2	8
17	3	1
18	3	2
19	3	3
20	3	4
21	3	5
22	3	6

	stadium_id integer	item_id integer
36	5	4
37	5	5
38	5	6
39	5	7
40	5	8
41	6	1
42	6	2
43	6	3
44	6	4
45	6	5
46	6	6
47	6	7

has Minigame Table

This table acts as a link between the Stadiums and Minigames tables. It represents all of the stadiums that have a minigame and where each one is held.

CREATE TABLE hasMinigame (

stadium_id INT REFERENCES Stadiums(stadium_id),

minigame_id INT REFERENCES Minigames(minigame_id),

PRIMARY KEY(stadium_id, minigame_id)

);

Functional Dependencies:

stadium_id, minigame_id →

	stadium_id integer	minigame_id integer
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9

Views

View 1: Create a view that returns all of the characters ids, names, name of their team, and their stats ordered by the team name and character name alphabetically, respectively.

```
CREATE VIEW view_all_characters_with_team_and_stats
AS
SELECT s.char_id, c.name, t.team_name, pitch, bat, field, run
FROM Teams t
INNER JOIN playsFor p ON t.team_id = p.team_id
INNER JOIN Characters c ON c.char_id = p.char_id
INNER JOIN Stats s ON s.char_id = c.char_id
GROUP BY s.char_id, c.name, t.team_name
ORDER BY t.team_name ASC, c.name ASC;

SELECT *
FROM view_all_characters_with_team_and_stats;
```

	char_id integer	name text	team_name text	pitch integer	bat integer	field integer	run integer
20	24	Funky Kong	DK Wilds	4	8	6	3
21	26	King K. Rool	DK Wilds	6	10	2	1
22	27	Kritter	DK Wilds	4	7	7	3
23	56	Red Kritter	DK Wilds	3	8	7	3
24	25	Tiny Kong	DK Wilds	5	5	7	5
25	74	Alan Labouseur	Free Agent	10	10	10	10
26	73	Female Mii	Free Agent	6	6	6	6
27	72	Male Mii	Free Agent	6	6	6	6
28	14	Baby Luigi	Mario Fireballs	5	2	5	8
29	13	Baby Mario	Mario Fireballs	5	3	4	8
30	41	Blooper	Mario Fireballs	6	4	5	6
31	22	Blue Noki	Mario Fireballs	5	4	4	7
32	21	Blue Pianta	Mario Fireballs	5	8	4	2
33	54	Green Noki	Mario Fireballs	4	5	4	7
34	2	Luigi	Mario Fireballs	6	6	7	7
35	1	Mario	Mario Fireballs	6	7	6	7
36	40	Monty Mole	Mario Fireballs	4	4	5	7
37	53	Red Noki	Mario Fireballs	4	4	5	7
38	51	Red Pianta	Mario Fireballs	4	8	4	2
39	52	Yellow Pianta	Mario Fireballs	4	8	4	2
40	16	Baby Daisy	Peach Monarchs	6	4	5	6
41	15	Baby Peach	Peach Monarchs	8	2	5	6
42	47	Blue Toad	Peach Monarchs	4	6	3	7
43	4	Daisy	Peach Monarchs	7	6	8	5
44	49	Green Toad	Peach Monarchs	4	5	4	7
45	3	Peach	Peach Monarchs	9	4	8	5
46	37	Petey Piranha	Peach Monarchs	4	10	5	2
47	50	Purple Toad	Peach Monarchs	5	6	2	7
48	18	Toad	Peach Monarchs	5	5	3	7
49	19	Toadette	Peach Monarchs	5	3	4	8
50	20	Toadsworth	Peach Monarchs	7	3	7	3
51	48	Yellow Toad	Peach Monarchs	3	6	4	7
52	35	Boo	Wario Muscles	9	3	3	5
53	28	Goomba	Wario Muscles	6	3	6	4
54	59	Green Paratroopa	Wario Muscles	3	5	7	5
55	36	King Boo	Wario Muscles	7	7	3	4
56	30	Koopa Troopa	Wario Muscles	3	6	4	6
57	29	Paragoomba	Wario Muscles	6	3	7	5
58	31	Paratroopa	Wario Muscles	4	4	7	5
59	58	Red Koopa Troopa	Wario Muscles	4	6	3	6
60	8	Waluigi	Wario Muscles	8	4	8	5
61	7	Wario	Wario Muscles	5	8	3	4
62	6	Birdo	Yoshi Eggs	4	8	7	5
63	68	Blue Shy Guy	Yoshi Eggs	5	4	7	4
64	43	Blue Yoshi	Yoshi Eggs	4	2	6	8
65	71	Gray Shy Guy	Yoshi Eggs	4	4	8	4
66	70	Green Shy Guy	Yoshi Eggs	3	5	7	5

Views

View 2: Create a view that returns all information about stadiums with their corresponding minigames.

```
CREATE VIEW view_all_stadiums_with_minigames
AS
SELECT s.stadium_id, s.field_name, s.day_mode, s.night_mode, s.hazards, m.minigame_id,
m.game_name , m.game_desc, m.hi_score
FROM Stadiums s
FULL OUTER JOIN hasMinigame h ON s.stadium_id = h.stadium_id
FULL OUTER JOIN Minigames m ON h.minigame_id = m.minigame_id
WHERE s.stadium_id != 0 OR m.minigame_id != 0
GROUP BY s.stadium_id, s.field_name, s.day_mode, s.night_mode, s.hazards, m.minigame_id,
m.game_name , m.game_desc, m.hi_score
ORDER BY s.stadium_id, m.minigame_id;


SELECT *
FROM view_all_stadiums_with_minigames;
```

	stadium_id integer	field_name text	day_mode boolean	night_mode boolean	hazards text	minigame_id integer	game_name text	game_desc text	hi_score integer
1	1	Mario Stadium	t	t	None	1	Bob-omb Derby	Hit bob-ombs into the sky and earn as many points as yo	6000
2	2	Peach Ice Garden	t	t	Freezies/Snowflakes	2	Wall Ball	Thorw the ball at the walls to break them and earn the	5000
3	3	Yoshi Park	t	t	Steam Train/Warp Pipes	3	Piranha Panic	Prevent the piranha plants from reaching your player by	4000
4	4	Wario City	t	t	Manholes/Directional Arrows	4	Gem Catch	Maneuver the player to catch gems to earn points! The m	3500
5	5	DK Jungle	t	t	Barrels/Giant Flowers	5	Barrel Basher	Fend off barrels and bob-ombs by hitting them with base	2500
6	6	Luigi's Mansion	f	t	Gravestones/Tall Grass	6	Ghost K	Throw baseballs at the ghosts on the screen to make the	3000
7	7	Daisy Cruiser	t	t	Dining Tables/Cheep Cheeps/Glooper Blooper	7	Blooper Baserun	Collect the most coins by ruinning around the bases! Be	4000
8	8	Bowser Jr. Playroom	t	f	Thwomps/ Chain Chomps/Bullet Bills	8	Grafitti Runner	Paint as much of the canvas as you can by running arour	3500
9	9	Bowser Castle	f	t	Lava Puddles/King Bob-omb/Bowser Statue	9	Bowser Pinball	Keep a spiked ball within the area and hit coins and we	5000

Views

View 3: Return the ids, star pitches, and star swings of captains that have a Banana Swing or Fireball pitch.

```
CREATE VIEW view_captain_banana_swing_or_fireball
AS
SELECT *
FROM Captains
WHERE (star_swing = 'Banana Swing'
OR star_pitch = 'Fireball')
AND cap_id IN(SELECT char_id
                FROM Special_Characters
                WHERE char_id IN (SELECT char_id
                                    FROM Characters
                                    WHERE char_id > 0 AND char_id <= 12
                                  )
              );
SELECT *
FROM view_captain_banana_swing_or_fireball;
```

	cap_id integer	star_pitch text	star_swing text
1	1	Fireball	Fire Swing
2	10	Banana Ball	Banana Swing

Views

View 4: Return the id, name, and score of all minigames with a high score greater than 4000.

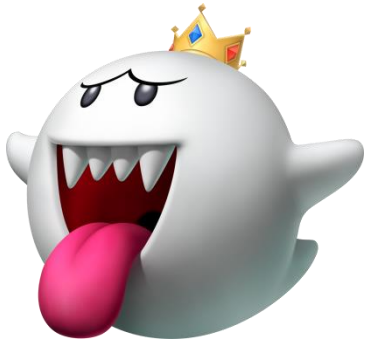
```
CREATE VIEW minigames_with_hi_score_greater_than_4000
AS
SELECT minigame_id, game_name, hi_score, COUNT(minigame_id)
FROM Minigames
WHERE hi_score > 4000
GROUP BY hi_score, minigame_id, game_name
HAVING COUNT(minigame_id) > 0
ORDER BY minigame_id;


SELECT *
FROM minigames_with_hi_score_greater_than_4000;
```

	minigame_id integer	game_name text	hi_score integer	count bigint
1	1	Bob-omb Derby	6000	1
2	2	Wall Ball	5000	1
3	9	Bowser Pinball	5000	1

Reports

Query 1: Get the names, ids, bat stats, and pitch stats of all players on the Wario Muscles with a batting stat greater than 6 or a pitching stat less than 4.

```
SELECT playsFor.char_id, Characters.name, playsFor.team_id, bat, pitch
FROM Characters, Teams, playsFor, Stats
WHERE Characters.char_id = playsFor.char_id
AND Teams.team_id = playsFor.team_id
AND playsFor.team_id = 7
AND Characters.char_id = Stats.char_id
AND (bat > 6 OR pitch < 4)
GROUP BY playsFor.char_id, Characters.name, playsFor.team_id, bat, pitch
ORDER BY Characters.name;
```



	char_id integer	name text	team_id integer	bat integer	pitch integer
1	59	Green Paratroopa	7	5	3
2	36	King Boo	7	7	7
3	30	Koopa Troopa	7	6	3
4	7	Wario	7	8	5

Reports

Query 2: Get the names, ids, and run stats of all captains that own a stadium where their run stat is greater than 5 in reverse alphabetical order.

```
SELECT DISTINCT Characters.name, Special_Characters.char_id, Captains.cap_id,
hasTeam.team_id, hasStadium.stadium_id, run
FROM Characters, Special_Characters, Captains, hasTeam, Teams, hasStadium,
Stadiums, Stats
WHERE Characters.char_id = Special_Characters.char_id
AND Special_Characters.cap_id = Captains.cap_id
AND Captains.cap_id = hasTeam.cap_id
AND hasTeam.team_id = Teams.team_id
AND Teams.team_id = hasStadium.team_id
AND hasStadium.stadium_id = Stadiums.stadium_id
AND Stadiums.stadium_id != 0
AND Characters.char_id = Stats.char_id
AND run > 5
ORDER BY Characters.name DESC;
```

	name text	char_id integer	cap_id integer	team_id integer	stadium_id integer	run integer
1	Yoshi	5	5	5	3	9
2	Mario	1	1	1	1	7
3	Luigi	2	2	2	6	7
4	Bowser Jr.	12	12	12	8	7

Reports

	name text	char_id integer	average numeric
1	Alan Labouseur	74	10.00

Query 3: Return the name and id of the character with the highest average of all stats.

```
SELECT c.name, s.char_id, ROUND(AVG((pitch + bat + field + run) / 4.0), 2) AS Average
FROM Characters c
INNER JOIN Stats s ON c.char_id = s.char_id
GROUP BY s.char_id, c.name
ORDER BY Average DESC
LIMIT 1;
```

Reports

Query 4: Return the name and ids of the teams that play in a stadium that is only available in day or night mode. (i.e. one or the other, NOT both)

	team_id integer	team_name text
1	2	Luigi Knights
2	11	Bowser Monsters
3	12	Bowser Jr. Rookies

```
SELECT t.team_id, t.team_name
FROM Teams t
FULL OUTER JOIN hasStadium h ON t.team_id = h.team_id
FULL OUTER JOIN Stadiums s ON h.stadium_id = s.stadium_id
WHERE (s.day_mode = TRUE AND s.night_mode = FALSE)
OR (s.day_mode = FALSE AND s.night_mode = TRUE)
GROUP BY t.team_id, t.team_name
ORDER BY t.team_id ASC;
```

Reports

Query 5: Return the names, character ids, and the team name of all characters with a fielding stat larger than or equal to 7. (NOT including free agents.)

```
SELECT Stats.char_id, name, team_name, field
FROM Characters, Teams, playsFor, Stats
WHERE Characters.char_id = playsFor.char_id
AND playsFor.team_id = Teams.team_id
AND Characters.char_id = Stats.char_id
AND field >= 7
AND Teams.team_id <> 0
ORDER BY Teams.team_name ASC, field DESC, Characters.name;
```

	char_id integer	name text	team_name text	field integer
1	61	Green Magikoopa	Bowser Monsters	8
2	32	Magikoopa	Bowser Monsters	8
3	60	Red Magikoopa	Bowser Monsters	8
4	62	Yellow Magikoopa	Bowser Monsters	8
5	17	Baby DK	DK Wilds	8
6	10	Diddy Kong	DK Wilds	8
7	23	Dixie Kong	DK Wilds	8
8	55	Blue Kritter	DK Wilds	7
9	57	Brown Kritter	DK Wilds	7
10	27	Kritter	DK Wilds	7
11	56	Red Kritter	DK Wilds	7
12	25	Tiny Kong	DK Wilds	7
13	2	Luigi	Mario Fireballs	7
14	4	Daisy	Peach Monarchs	8
15	3	Peach	Peach Monarchs	8
16	20	Toadsworth	Peach Monarchs	7
17	8	Waluigi	Wario Muscles	8
18	59	Green Paratroopa	Wario Muscles	7
19	29	Paragoomba	Wario Muscles	7
20	31	Paratroopa	Wario Muscles	7
21	71	Gray Shy Guy	Yoshi Eggs	8
22	6	Birdo	Yoshi Eggs	7
23	68	Blue Shy Guy	Yoshi Eggs	7
24	70	Green Shy Guy	Yoshi Eggs	7
25	39	Shy Guy	Yoshi Eggs	7
26	69	Yellow Shy Guy	Yoshi Eggs	7

Stored Procedures

Stored Procedure 1: Create a function that allows a user to input the name of the player and find the team they play for.

```
CREATE OR REPLACE FUNCTION locatePlayer(TEXT)
RETURNS TABLE(name TEXT, team_name TEXT) AS
$$
DECLARE
        findChar TEXT := $1;
BEGIN
        RETURN QUERY
        SELECT Characters.name, Teams.team_name
        FROM Characters, playsFor, Teams
        WHERE Characters.char_id = playsFor.char_id
        AND playsFor.team_id = Teams.team_id
        AND Characters.name = findChar;
END;
$$ LANGUAGE plpgsql;

SELECT locatePlayer('Dry Bones');
SELECT locatePlayer('Wiggler');
SELECT locatePlayer('Alan Labouseur');
```

	locateplayer record
1	("Dry Bones","Bowser Monsters")

	locateplayer record
1	(Wiggler,"Yoshi Eggs")

	locateplayer record
1	("Alan Labouseur","Free Agent")

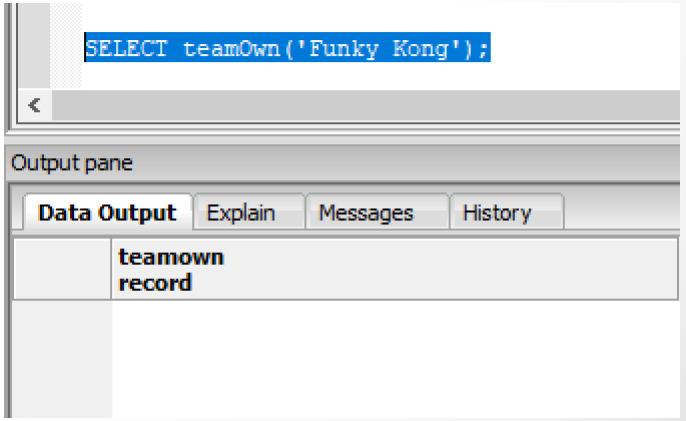
Stored Procedures

Stored Procedure 2: Create a function that returns the name, team name, and team stadium of a player. If a player is not a captain, then the result will be empty.

```
CREATE OR REPLACE FUNCTION teamOwn(TEXT)
RETURNS TABLE(name TEXT, team_name TEXT, field_name TEXT) AS
$$
DECLARE
        findOwner TEXT := $1;
BEGIN
        RETURN QUERY
        SELECT c.name, t.team_name, s.field_name
        FROM Characters c
        INNER JOIN Special_Characters sc ON c.char_id = sc.char_id
        INNER JOIN Captains a ON sc.cap_id = a.cap_id
        INNER JOIN hasTeam h ON a.cap_id = h.cap_id
        INNER JOIN Teams t ON h.team_id = t.team_id
        INNER JOIN hasStadium d ON t.team_id = d.team_id
        INNER JOIN Stadiums s ON d.stadium_id = s.stadium_id
        WHERE c.name = findOwner
        GROUP BY c.name, t.team_name, s.field_name
        LIMIT 1;
END;
$$ LANGUAGE plpgsql;

SELECT teamOwn('Daisy');
SELECT teamOwn('Waluigi');
SELECT teamOwn('Funky Kong');
```

	teamown record
1	(Daisy,"Daisy Flowers","Daisy Cruiser")

	teamown record
1	(Waluigi,"Waluigi Spitballs",None)



Stored Procedures

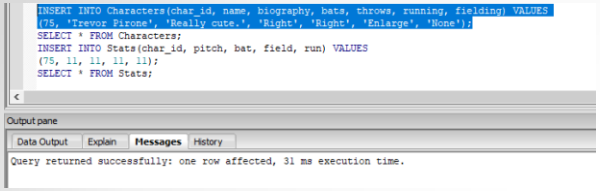
Stored Procedure 3: Create a function that allows the user to search for a team's roster by inputting the name of the team. (Also works for free agents!)

```
CREATE OR REPLACE FUNCTION showRoster(TEXT)
RETURNS TABLE(team_name TEXT, name TEXT, char_id INT) AS
$$
DECLARE
        teamName TEXT := $1;
BEGIN
        RETURN QUERY
        SELECT t.team_name, c.name, c.char_id
        FROM Characters c
        FULL OUTER JOIN playsFor p ON c.char_id = p.char_id
        FULL OUTER JOIN Teams t ON p.team_id = t.team_id
        WHERE t.team_name = teamName
        GROUP BY t.team_name, c.name, c.char_id
        ORDER BY c.name, c.char_id;
END
$$ LANGUAGE plpgsql;

SELECT showRoster('Free Agent');
SELECT showRoster('DK Wilds');
SELECT showRoster('Peach Monarchs');
```

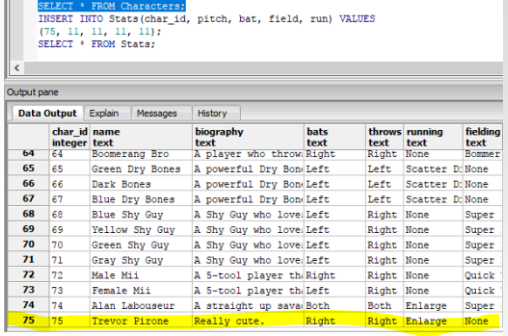
	showroster record
1	("Free Agent","Alan Labouseur",74)
2	("Free Agent","Female Mii",73)
3	("Free Agent","Male Mii",72)

	showroster record
1	("DK Wilds","Baby DK",17)
2	("DK Wilds","Blue Kritter",55)
3	("DK Wilds","Brown Kritter",57)
4	("DK Wilds","Diddy Kong",10)
5	("DK Wilds","Dixie Kong",23)
6	("DK Wilds","Donkey Kong",9)
7	("DK Wilds","Funky Kong",24)
8	("DK Wilds","King K. Rool",26)
9	("DK Wilds",Kritter,27)
10	("DK Wilds","Red Kritter",56)
11	("DK Wilds","Tiny Kong",25)

	showroster record
1	("Peach Monarchs","Baby Daisy",16)
2	("Peach Monarchs","Baby Peach",15)
3	("Peach Monarchs","Blue Toad",47)
4	("Peach Monarchs",Daisy,4)
5	("Peach Monarchs","Green Toad",49)
6	("Peach Monarchs",Peach,3)
7	("Peach Monarchs","Petey Piranha",37)
8	("Peach Monarchs","Purple Toad",50)
9	("Peach Monarchs",Toad,18)
10	("Peach Monarchs",Toadette,19)
11	("Peach Monarchs",Toadsworth,20)
12	("Peach Monarchs","Yellow Toad",48)

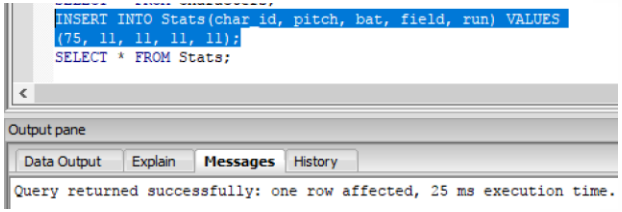


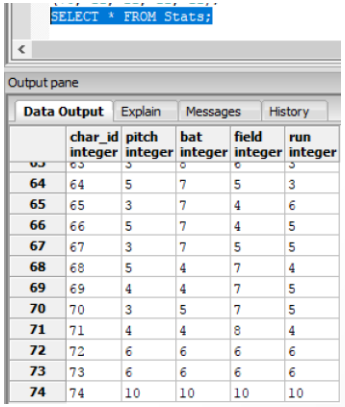
Triggers

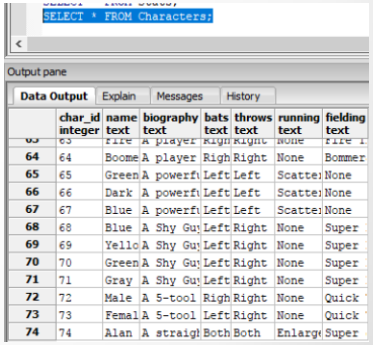


Trigger 1: Create a trigger that checks if a player is added to the Characters table, when added to the stats table, if they have any stat less than zero or greater than 10, remove them from the database.

```
CREATE OR REPLACE FUNCTION newStat()
RETURNS TRIGGER AS
$$
BEGIN
        IF (NEW.bat < 0 OR NEW.bat > 10) OR (NEW.pitch < 0 OR NEW.pitch > 10) OR (NEW.field< 0 OR NEW.field > 10) OR (NEW.run < 0
        OR NEW.run> 10) THEN
        DELETE FROM Stats WHERE pitch = NEW.pitch;
        DELETE FROM Stats WHERE bat = NEW.bat;
        DELETE FROM Stats WHERE field = NEW.field;
        DELETE FROM Stats WHERE run = NEW.run;
        DELETE FROM Stats WHERE char_id = NEW.char_id;
        DELETE FROM Characters WHERE char_id = NEW.char_id;
        END IF;
        RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER newStat
AFTER INSERT ON Stats
FOR EACH ROW
EXECUTE PROCEDURE newStat();

INSERT INTO Characters(char_id, name, biography, bats, throws, running, fielding) VALUES
(75, 'Trevor Pirone', 'Really cute.', 'Right', 'Right', 'Enlarge', 'None');
SELECT * FROM Characters;
INSERT INTO Stats(char_id, pitch, bat, field, run) VALUES
(75, 11, 11, 11, 11);
SELECT * FROM Stats;
SELECT * FROM Characters;
```



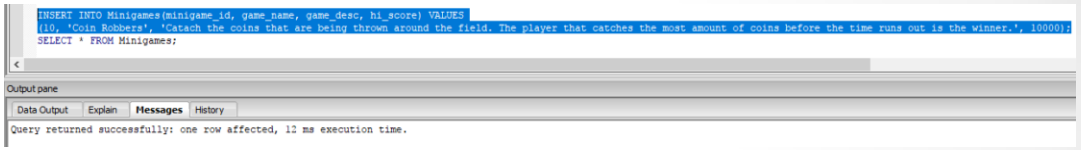




Triggers

Trigger 2: Create a trigger that checks if the high score of a new minigame is greater than 9999 (the max value allowed in the game). If so, delete the minigame from the database.

```
CREATE OR REPLACE FUNCTION newMinigame()
RETURNS TRIGGER AS
$$
BEGIN
        IF (NEW.hi_score > 9999) THEN
        DELETE FROM Minigames WHERE minigame_id = NEW.minigame_id;
        DELETE FROM Minigames WHERE game_name = NEW.game_name;
        DELETE FROM Minigames WHERE game_desc = NEW.game_desc;
        DELETE FROM Minigames WHERE hi_score = NEW.hi_score;
        END IF;
        RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER newMinigame
AFTER INSERT ON Minigames
FOR EACH ROW
EXECUTE PROCEDURE newMinigame();

INSERT INTO Minigames(minigame_id, game_name, game_desc, hi_score) VALUES
(10, 'Coin Robbers', 'Catach the coins that are being thrown around the field. The player that catches the most amount of coins before the time runs out is the winner.', 10000);
SELECT * FROM Minigames;
```



	minigame_id integer	game_name text	game_desc text	hi_score integer
1	1	Bob-omb De	Hit bob-om	6000
2	2	Wall Ball	Thorw the	5000
3	3	Piranha Pa	Prevent th	4000
4	4	Gem Catch	Maneuver t	3500
5	5	Barrel Bash	Fend off k	2500
6	6	Ghost K	Throw base	3000
7	7	Blooper Ba	Collect th	4000
8	8	Grafitti Ru	Paint as m	3500
9	9	Bowser Pink	Keep a spi	5000

Security

Admin Role:
The admin will have access to modify all tables present in the database.

```
CREATE ROLE ADMIN;
GRANT ALL
ON ALL TABLES IN SCHEMA PUBLIC
TO ADMIN;
```

Captain Role:
The captain will have the privilege to select, update, and delete information on teams, stadiums, and minigames.

```
CREATE ROLE CAPTAIN;
GRANT SELECT, UPDATE, DELETE
ON Teams, Stadiums, Minigames
TO CAPTAIN;
```

Guest Role:
The guest can use select statements on all tables, but is revoked from inserting, updating, or deleting while interacting with the database.

```
CREATE ROLE GUEST;
GRANT SELECT
ON ALL TABLES IN SCHEMA PUBLIC
TO GUEST;
REVOKE INSERT, UPDATE, DELETE
ON ALL TABLES IN SCHEMA PUBLIC
FROM GUEST;
```

Implementation Notes

It should be noted that the Stats table only reflects the stats of characters without enabling star players. (When a player is a star all stats not maxed out increase by 1.)

This database does not take into account scout missions that appear in the game's story mode due to sheer size and time.

Items are available in every game and stadium, but when playing an exhibition game, items can be turned off, so the database is assuming that items are always enabled.

Implementation Notes

Similar to the items, star powers (star pitches and star swings) can also be disabled when playing an exhibition game.

In the game, the player has the option to pick from 12 captains. However, captains also play for other captains teams (i.e. Luigi on Mario's team, Daisy on Peach's, etc.) The captains that play for other captains are known as sub-captains which is not covered in the database. So, if a user wanted to see the roster for the Luigi Knights, the team would be empty because of the aforementioned reason.

Known Problems

Some known problems include…

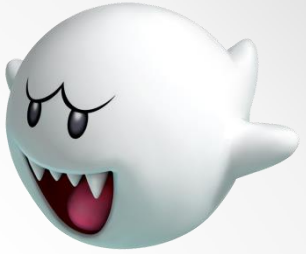
- The teamOwned stored procedure, although it works, when specifying a non-captain character, it returns an empty result instead of having a result specifying only captains own teams.
- Missing scout missions from the story mode due to size and time. Does not break the database, however. (This would have made for some cool triggers though.)
- Not a groundbreaking issue, but items and stadiums do not really relate because all items can be used in every stadium. In fact, I would say items are on their own, but because it had to be implemented it best fit with stadium.
- (Maybe a more ideal fit would have been relating it to an instance of game and having the ability to enable or disable the use of items?)

Future Enhancements

Including, but not limited to…

- Adding scout missions into the database.
- Adding a stats table that takes into account a player when they are a star (and if they even have a star)!
- Rearranging the items table in a manner that makes it more involved in the database.
- Creating a games table which takes into account enabling rules specific to that game (allow star pitches/swings, mercy rule, innings, items, etc.).
- Chemistry table (which characters have good chemistry/bad chemistry).

Conclusion



Regardless of the outcome of the project, I felt that I learned a lot about building and managing a database system. I also found the use of views, stored procedures and triggers very interesting as I had no prior knowledge of them before this class.

For what it was worth, this game happens to be one of my favorite games of all time and implementing a database for it has made me learn even more about the game that I did not know prior to this project (I found which characters had the highest and lowest stat averages thanks to Query 3.).

Although there are elements of the game missing in the database, I can always go back and modify it at my leisure. It's not perfect, but I think it's definitely satisfying enough for Mario to pick out his best lineups to defeat Bowser and rescue the Baseball Kingdom!

THE END

"Buttered broccoli! I can't believe I lost!"
-Bowser Jr.