G6 Pokedex Database

Team 6:

Lucas Attias

Spencer Mantis

Justin Forkert

Syracuse University

IST-659 Data Admin Concept & Db Mgmt

Professor Gregory Michael Zink

June 18, 2024

Table of Contents

Overview Narrative

Introduction

This database aims to support catching and battling Pokémon across all games by connecting players to Pokémon information including types, base stats, and generations. Users can create an account to track the different Pokémon that they have caught. Users are also able to query the database without an account and use the G6 Pokedex as a resource while they play. The G6 Pokedex will be powered by SQL to query the database and will be a hybrid between existing Pokémon trackers and informational sites.

Purpose of the Database

The purpose of this database is to be a reference point for Pokémon gamers. The database will link Pokémon and the user. It will allow users to view attributes of different Pokémon including their types, base stats, generations, and more. Players can develop battle strategies based on their queries and target specific Pokémon that users desire to catch.

Target Audience

Gamers who want to keep track of, organize, and gain insight into the Pokémon they have caught, plan to catch, and are battling against. For the competitive gamer looking to get an edge, but also the casual fan who wants to inquire more about Pokémon.

Database Structure Overview

High Level Business Requirements

- Provide a user-friendly platform to connect users to their Pokémon
- Provide a way to easily search for Pokémon by different attributes
- Provide a way for users to save Pokémon to a watchlist that can be easily accessed
- Provide a place to store Pokémon caught in games and keep track of your progress

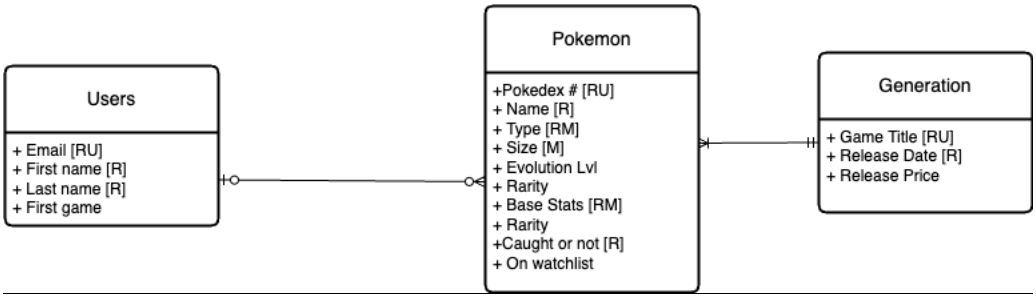
Functional Requirements

- User Registration and Authentication
- Searching for Pokémon
    - For specific Pokémon
    - For Pokémon of specific types
    - Ability to query by generation and/or base stat with an advanced search
- Pokémon inserts
- Saved Pokémon watchlist
- Pokémon catch list

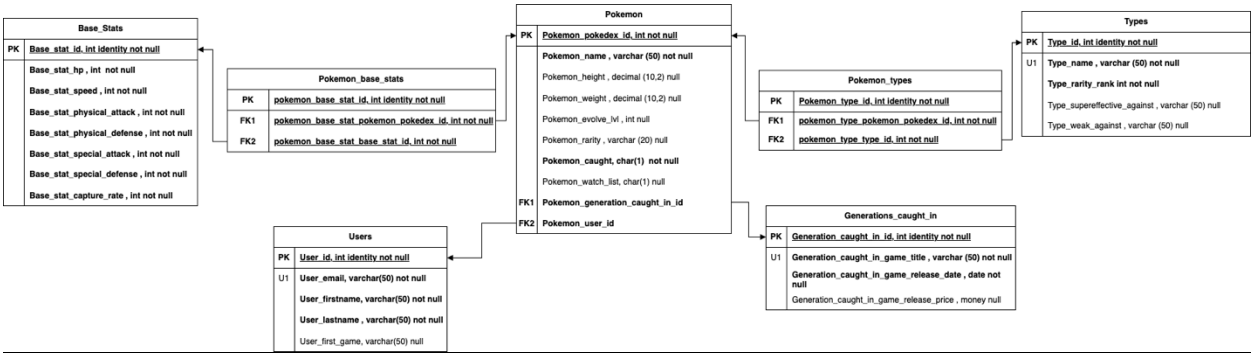
Data Analysis of the Facts

Below is a data analysis of the facts listing the entities, attributes, and relationships in the data model.

Conceptual Data Model



Logical Data Model



Entities and Attributes

Entity	Attribute	Data Type	Constraint
pokemon	pokemon_pokedex_id	int not null	Primary Key
	pokemon_name	varchar(50) not null	
	pokemon_height	decimal null	
	pokemon_weight	decimal null	
	pokemon_evolve	int null	
	pokemon_rarity	varchar(20) null	
	pokemon_caught	char(1) not null	
	pokemon_watch_list	char(1) null	
	pokemon_generation_caught_in_id	int not null	Foreign Key
	pokemon_user_id	int null	Foreign Key
user	user_id		Primary Key
	user_email		Unique Key
	user_firstname		
	user_lastname		
	user_first_game		
generation_caught_in	generation_caught_in_id	int not null	Primary Key
	generation_caught_in_game_title	varchar(50) not null	Unique Key
	generation_caught_in_game_release_date	date not null	
	generation_caught_in_game_release price	money null	
pokemon_base_stats	pokemon_base_stat_id	int not null	Primary Key
	pokemon_base_stat_pokemon_pokedex_id	int null	
	pokemon_base_stats_base_stat_id	int null	
base_stats	base_stat_id	int not null	Primary Key
	base_stat_hp	int not null	
	base_stat_speed	int not null	

	base_stat_physical_attack	int not null	
	base_stat_physical_defense	int not null	
	base_stat_special_attack	int not null	
	base_stat_special_defense	int not null	
	base_stat_capture_rate	int not null	
pokemon_types	pokemon_type_id	int not null	Primary Key
	pokemon_type_pokemon_pokedex_id	int not null	Foreign Key
	pokemon_type_type_id	int not null	Foreign Key
types	type_id	int not null	Primary Key
	type_name	varchar (50) not null	Unique Key
	type_supereffective_against	varchar(50) null	
	type_weak_against	varchar(50) null	

Data Requirements

Entities & Attributes:

Entity	Attribute	Props	Description
Pokemon	Pokedex #	RU	Unique identification # for each Pokemon
	Name	R	Name of Pokemon
	Type	RM	All associated types for a pokemon
	Size	M	Height and weight of the pokemon
	Evolution Level		Level the Pokemon evolves at
	Rarity		Classification of how rare a pokemon is
	Base Stats	RM	Hp, speed, physical/special attack, physical/special defense of Pokemon
	Caught	R	Has the user Caught It?
	Watchlist		Has the user added it to their watchlist?
User	First Name	R	First name of the user
	Last Name	R	Last name of the user
	Email	RU	Email of the user
	First Game		First Pokemon game the user played
Generations	Game Title	RU	Gametitle generartion was introduced in
	Game Release Date	R	Date of generatrion release
	Release Price		Price of game at release

Relationships:

Relationship	Entity	Rules	Min	Max	Entity 2
Pokemon introduced in generation	Pokemon	Is introdcued in	1	1	Generation
	Generation	Introduces	1	M	Pokemon
User and Pokemon	User	Has	0	M	Pokemon
	Pokemon	Has	0	1	User

Data Questions and Answers

1. Add transactions to ensure data integrity

```
166
167  /*Insert into users*/
168  BEGIN TRY
169      BEGIN TRANSACTION;
170  INSERT INTO users (user_firstname, user_lastname, user_email)
171  VALUES
172      ('Ash', 'Ketchum', 'ash@pokemon.com'),
173      ('Misty', 'Waterflower', 'misty@pokemon.com'),
174      ('Brock', 'Rock', 'brock@pokemon.com'),
175      ('Gary', 'Oak', 'gary@pokemone.com'),
176      ('Jessie', 'Team Rocket', 'jessie@pokemon.com');
177          COMMIT;
178  END TRY
179  BEGIN CATCH
180      IF @@TRANCOUNT > 0 THROW 50001, 'Duplicate Email',1
181          ROLLBACK;
182      throw;
183  END CATCH
184  GO
```

Messages

```
8:20:17 PM    Started executing query at Line 168
              (0 rows affected)
              Msg 50001, Level 16, State 1, Line 13
              Duplicate Email
              Total execution time: 00:00:00.008
```

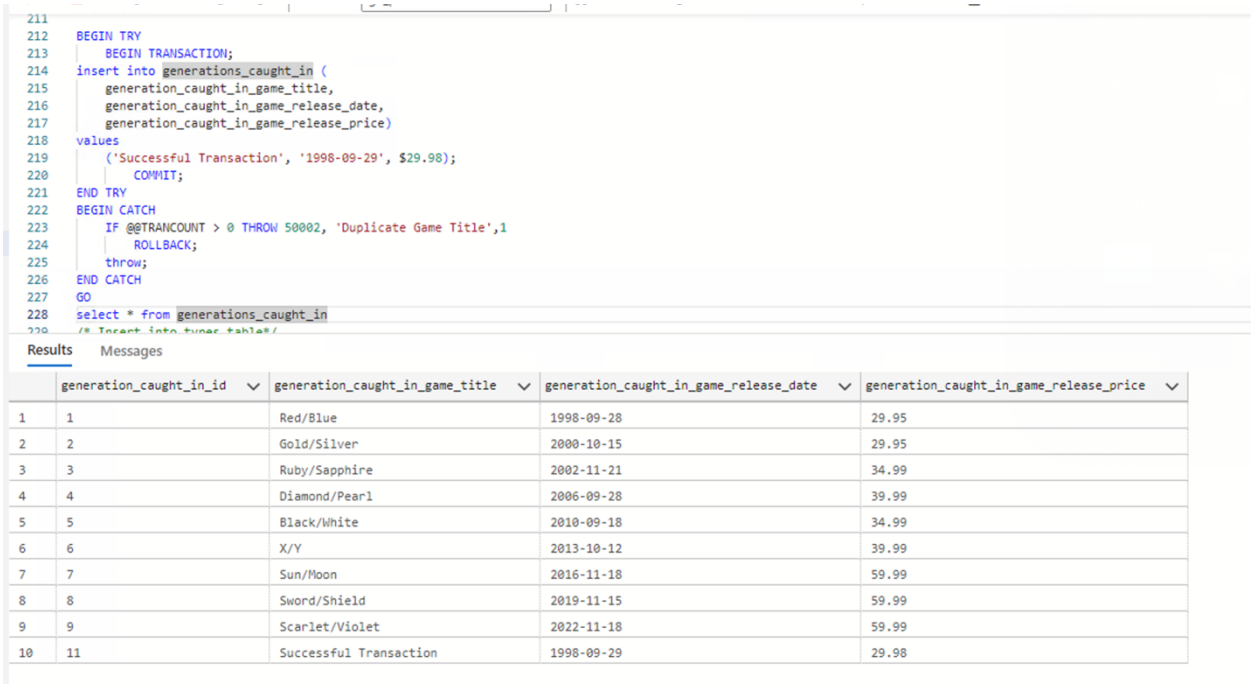
Figure shows the "Duplicate Email" error message displayed when users enter duplicate email values into the users table.

```
185
186  /* Insert into generations_caught_in table*/
187  BEGIN TRY
188      BEGIN TRANSACTION;
189  insert into generations_caught_in (
190      generation_caught_in_game_title,
191      generation_caught_in_game_release_date,
192      generation_caught_in_game_release_price)
193  values
194      ('Red/Blue', '1998-09-28', $29.95),
195      ('Gold/Silver', '2000-10-15', $29.95),
196      ('Ruby/Sapphire', '2002-11-21', $34.99),
197      ('Diamond/Pearl', '2006-09-28', $39.99),
198      ('Black/White', '2010-09-18', $34.99),
199      ('X/Y', '2013-10-12', $39.99),
200      ('Sun/Moon', '2016-11-18', $59.99),
201      ('Sword/Shield', '2019-11-15', $59.99),
202      ('Scarlet/Violet', '2022-11-18', $59.99);
203          COMMIT;
204  END TRY
205  BEGIN CATCH
206      IF @@TRANCOUNT > 0 THROW 50002, 'Duplicate Game Title',1
207          ROLLBACK;
208      throw;
209  END CATCH
210  GO
211
212  /* Insert into types table*/
```

Messages

```
8:25:05 PM    Started executing query at Line 187
              (0 rows affected)
              Msg 50002, Level 16, State 1, Line 20
              Duplicate Game Title
              Total execution time: 00:00:00.010
```

Figure shows the "Duplicate Game Title" error message that displays when a user enters a duplicate game title value into the generations_caught_in table.



```
211
212  BEGIN TRY
213      BEGIN TRANSACTION;
214  insert into generations_caught_in (
215      generation_caught_in_game_title,
216      generation_caught_in_game_release_date,
217      generation_caught_in_game_release_price)
218  values
219      ('Successful Transaction', '1998-09-29', $29.98);
220          COMMIT;
221  END TRY
222  BEGIN CATCH
223      IF @@TRANCOUNT > 0 THROW 50002, 'Duplicate Game Title',1
224          ROLLBACK;
225      throw;
226  END CATCH
227  GO
228  select * from generations_caught_in
```

Results | Messages

	generation_caught_in_id	generation_caught_in_game_title	generation_caught_in_game_release_date	generation_caught_in_game_release_price
1	1	Red/Blue	1998-09-28	29.95
2	2	Gold/Silver	2000-10-15	29.95
3	3	Ruby/Sapphire	2002-11-21	34.99
4	4	Diamond/Pearl	2006-09-28	39.99
5	5	Black/White	2010-09-18	34.99
6	6	X/Y	2013-10-12	39.99
7	7	Sun/Moon	2016-11-18	59.99
8	8	Sword/Shield	2019-11-15	59.99
9	9	Scarlet/Violet	2022-11-18	59.99
10	11	Successful Transaction	1998-09-29	29.98

Figure shows a successful transaction into the generation_caught_in table with no error messages.

2. A complex query of a specific Pokémon with a specific attribute (fastest Pokémon in gen 2)

```
346
347   /*Fastest pokemon in generation 2*/
348   SELECT TOP 1 p.pokemon_name, bs.base_stat_speed
349   FROM pokemon p
350   JOIN pokemon_base_stats pbs ON p.pokemon_pokedex_id = pbs.pokemon_base_stat_pokemon_pokedex_id
351   JOIN base_stats bs ON pbs.pokemon_base_stats_base_stat_id = bs.base_stat_id
352   WHERE p.pokemon_generation_caught_in_id = 2
353   ORDER BY bs.base_stat_speed DESC
354   GO
```

Results Messages

	pokemon_name	base_stat_speed
1	Kingdra	85

Figure shows a complex query displaying the fastest Pokémon in generation 2.

3. Output all Pokémon that are super effective against a specific type

```
624   /*Super effective against a specific type i.e Ground*/
625   SELECT p.pokemon_pokedex_id, p.pokemon_name, t.type_name, t.type_supereffective_against
626   FROM pokemon p
627   JOIN pokemon_types pt ON p.pokemon_pokedex_id = pt.pokemon_type_pokemon_pokedex_id
628   JOIN types t ON t.type_id = pt.pokemon_type_type_id
629   JOIN pokemon_base_stats pbs ON pbs.pokemon_base_stat_pokemon_pokedex_id = p.pokemon_pokedex_id
630   WHERE t.type_supereffective_against LIKE '%Ground%'
631   GO
632
```

Results Messages

	pokemon_pokedex_id	pokemon_name	type_name	type_supereffective_against
1	149	Dragonite	Flying	Grass/Ground/Bug/Ghost
2	334	Altaria	Flying	Grass/Ground/Bug/Ghost
3	373	Salamence	Flying	Grass/Ground/Bug/Ghost
4	384	Rayquaza	Flying	Grass/Ground/Bug/Ghost

Figure shows a query for displaying which pokemon are super effective against a specific type (i.e. Ground).

4. Show all Pokémon of a specific type (i.e. Ground)

```
482
483  /* TVF for users who frequently filter Pokémon by a specific type*/
484  IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID(N'dbo.PokemonByType'))
485      DROP FUNCTION dbo.PokemonByType;
486  GO
487
488  CREATE FUNCTION dbo.PokemonByType(@type_name varchar(50))
489  RETURNS TABLE
490  AS
491  RETURN (
492      SELECT p.pokemon_pokedex_id,
493             p.pokemon_name,
494             t.*
495      FROM pokemon p
496      JOIN pokemon_types pt ON p.pokemon_pokedex_id = pt.pokemon_type_pokemon_pokedex_id
497      JOIN types t ON pt.pokemon_type_type_id = t.type_id
498      WHERE t.type_name = @type_name
499  );
500  GO
```

```
641
642  /*Test Function to show Pokemon of a certain type, show all ground pokemon*/
643  SELECT *
644  FROM dbo.PokemonByType('Ground');
```

Results | Messages

	pokemon_pokedex_id	pokemon_name	type_id	type_name	type_rarity_rank	type_supereffective_against	type_weak_agaianst
1	329	Vibrava	10	Ground	7	Electric/Poison/Rock	Water/Grass/Ice/
2	330	Flygon	10	Ground	7	Electric/Poison/Rock	Water/Grass/Ice/

Figures show table value function for outputting all Pokémon of a specific type. In this example, all ground Pokémon in the database is queried.

5. Casual player view of the most commonly used basic info and stats

```
/*A view that will commonly be used by casual players to see basic pokemon info and stats*/
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.VIEWS WHERE TABLE_NAME = 'v_casual_player')
BEGIN
    DROP VIEW v_casual_player;
END
GO

CREATE VIEW v_casual_player as
SELECT
    p.pokemon_pokedex_id,
    p.pokemon_name,
    t.type_name,
    t.type_supereffective_against,
    t.type_weak_agaianst,
    bs.base_stat_hp,
    bs.base_stat_speed
FROM
    pokemon p
JOIN
    pokemon_types pt ON p.pokemon_pokedex_id = pt.pokemon_type_pokemon_pokedex_id
JOIN
    types t ON t.type_id = pt.pokemon_type_type_id
JOIN
    pokemon_base_stats pbs ON pbs.pokemon_base_stat_pokemon_pokedex_id = p.pokemon_pokedex_id
JOIN
    base_stats bs ON bs.base_stat_id = pbs.pokemon_base_stats_base_stat_id;
GO
```

Figure above shows view being created to allow casual fans to search for common information.

```
/* #4 See Casual Player View. These are the most commonly used basic pokemon info and stats. This shows all pokemon with an hp over 60*/
select * from v_casual_player
where base_stat_hp > 60
GO
```

Results Messages

	pokemon_pokedex_id	pokemon_name	type_name	type_supereffective_against	type_weak_agaianst	base_stat_hp	base_stat_speed
1	148	Dragonair	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	61	70
2	149	Dragonite	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	91	80
3	149	Dragonite	Flying	Grass/Ground/Bug/Ghost	Electric/Ice/Rock	91	80
4	230	Kingdra	Water	Fire/Ice/Steel	Electic/Grass	75	85
5	230	Kingdra	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	75	85
6	330	Flygon	Ground	Electric/Poison/Rock	Water/Grass/Ice/	80	100
7	330	Flygon	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	80	100
8	334	Altaria	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	75	80
9	334	Altaria	Flying	Grass/Ground/Bug/Ghost	Electric/Ice/Rock	75	80
10	372	Shelgon	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	65	50
11	373	Salamence	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	95	100
12	373	Salamence	Flying	Grass/Ground/Bug/Ghost	Electric/Ice/Rock	95	100
13	380	Latias	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	80	110
14	380	Latias	Psychic	Fight	Bug/Ghost/Dark	80	110
15	381	Latios	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	80	110
16	381	Latios	Psychic	Fight	Bug/Ghost/Dark	80	110
17	384	Rayquaza	Dragon	Fire/Water/Electric/Grass	Ice/Dragon/Fairy	105	95
18	384	Rayquaza	Flying	Grass/Ground/Bug/Ghost	Electric/Ice/Rock	105	95

Figure shows select statement from the casual player view that queries the called view to display all Pokémon with a base stat HP above 60.

6. a. View of all Pokémon in gen 1
   b. View of all Pokémon in gen 2
   c. View of all Pokémon in gen 3

```
503   /*View of all pokemon for generation filtering*/
504   IF EXISTS (SELECT * FROM sys.views WHERE name = 'v_pokemon_generation')
505   BEGIN
506       DROP VIEW v_pokemon_generation;
507   END
508   GO
509
510   CREATE VIEW v_pokemon_generation AS
511   SELECT
512       p.pokemon_pokedex_id,
513       p.pokemon_name,
514       gci.generation_caught_in_game_title,
515       gci.generation_caught_in_id
516   FROM
517       pokemon p
518   JOIN
519       generations_caught_in gci
520   ON
521       gci.generation_caught_in_id = p.pokemon_generation_caught_in_id
522   GO
523
```

Figure shows generation view being created to display all the Pokémon of different generations.

```
652   /*View sorted for all pokemon in generation 1*/
653   Select pokemon_pokedex_id, pokemon_name, generation_caught_in_game_title from v_pokemon_generation
654       WHERE generation_caught_in_id = 1
655   GO
656
657   /*View sorted for all pokemon in generation 2*/
658   Select pokemon_pokedex_id, pokemon_name, generation_caught_in_game_title from v_pokemon_generation
659       WHERE generation_caught_in_id = 2
660   GO
661
662   /*View sorted for all pokemon in gernation 3*/
663   Select pokemon_pokedex_id, pokemon_name, generation_caught_in_game_title from v_pokemon_generation
664       WHERE generation_caught_in_id = 3
665   GO
666
```

Figure shows the generations view being executed three times to display all of the Pokémon in each of the first three generations.

Results Messages

	pokemon_pokedex_id	pokemon_name	generation_caught_in_game_title
1	25	Pikachu	Red/Blue
2	147	Dratini	Red/Blue
3	148	Dragonair	Red/Blue
4	149	Dragonite	Red/Blue

	pokemon_pokedex_id	pokemon_name	generation_caught_in_game_title
1	230	Kingdra	Gold/Silver

	pokemon_pokedex_id	pokemon_name	generation_caught_in_game_title
1	329	Vibrava	Ruby/Sapphire
2	330	Flygon	Ruby/Sapphire
3	334	Altaria	Ruby/Sapphire
4	371	Bagon	Ruby/Sapphire
5	372	Shelgon	Ruby/Sapphire
6	373	Salamence	Ruby/Sapphire
7	380	Latias	Ruby/Sapphire
8	381	Latios	Ruby/Sapphire
9	384	Rayquaza	Ruby/Sapphire

Figure shows the output of the three generation view executions to display all Pokémon in each of the first three generations.

7. View for super effective against every type

```
576  /*A view to show all super effective pokemon vs every type*/
577  IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.VIEWS WHERE TABLE_NAME = 'v_super_effective_pokemons')
578  BEGIN
579      DROP VIEW v_super_effective_pokemons;
580  END
581  GO
582
583  CREATE VIEW v_super_effective_pokemons AS
584  SELECT
585      p.pokemon_name,
586      CASE
587          WHEN t.type_supereffective_against LIKE '%Ground%' THEN 'Ground'
588          WHEN t.type_supereffective_against LIKE '%Electric%' THEN 'Electric'
589          WHEN t.type_supereffective_against LIKE '%Bug%' THEN 'Bug'
590          WHEN t.type_supereffective_against LIKE '%Dark%' THEN 'Dark'
591          WHEN t.type_supereffective_against LIKE '%Dragon%' THEN 'Dragon'
592          WHEN t.type_supereffective_against LIKE '%Fight%' THEN 'Fight'
593          WHEN t.type_supereffective_against LIKE '%Fire%' THEN 'Fire'
594          WHEN t.type_supereffective_against LIKE '%Flying%' THEN 'Flying'
595          WHEN t.type_supereffective_against LIKE '%Ghost%' THEN 'Ghost'
596          WHEN t.type_supereffective_against LIKE '%Grass%' THEN 'Grass'
597          WHEN t.type_supereffective_against LIKE '%Ice%' THEN 'Ice'
598          WHEN t.type_supereffective_against LIKE '%Normal%' THEN 'Normal'
599          WHEN t.type_supereffective_against LIKE '%Poison%' THEN 'Poison'
600          WHEN t.type_supereffective_against LIKE '%Psychic%' THEN 'Psychic'
601          WHEN t.type_supereffective_against LIKE '%Rock%' THEN 'Rock'
602          WHEN t.type_supereffective_against LIKE '%Steel%' THEN 'Steel'
603          WHEN t.type_supereffective_against LIKE '%Water%' THEN 'Water'
604          WHEN t.type_supereffective_against LIKE '%Fairy%' THEN 'Fairy'
605          ELSE ''
606      END AS super_effective_type
607  FROM
608      pokemon p
609  JOIN
610      pokemon_types pt ON p.pokemon_pokedex_id = pt.pokemon_type_pokemon_pokedex_id
611  JOIN
612      types t ON t.type_id = pt.pokemon_type_type_id;
613  GO
614
```

Figure shows a view being created to display all Pokémon that are super effective against different types.

```sql
/*View for every pokemon that is super effective against each type*/
SELECT
    super_effective_against_ground,
    super_effective_against_electric,
    super_effective_against_bug,
    super_effective_against_dark,
    super_effective_against_dragon,
    super_effective_against_fight,
    super_effective_against_fire,
    super_effective_against_flying,
    super_effective_against_ghost,
    super_effective_against_grass,
    super_effective_against_ice,
    super_effective_against_normal,
    super_effective_against_poison,
    super_effective_against_psychic,
    super_effective_against_rock,
    super_effective_against_steel,
    super_effective_against_water,
    super_effective_against_fairy
FROM (
    SELECT
        MAX(CASE WHEN super_effective_type = 'Ground' THEN pokemon_name ELSE '' END) AS super_effective_against_ground,
        MAX(CASE WHEN super_effective_type = 'Electric' THEN pokemon_name ELSE '' END) AS super_effective_against_electric,
        MAX(CASE WHEN super_effective_type = 'Bug' THEN pokemon_name ELSE '' END) AS super_effective_against_bug,
        MAX(CASE WHEN super_effective_type = 'Dark' THEN pokemon_name ELSE '' END) AS super_effective_against_dark,
        MAX(CASE WHEN super_effective_type = 'Dragon' THEN pokemon_name ELSE '' END) AS super_effective_against_dragon,
        MAX(CASE WHEN super_effective_type = 'Fight' THEN pokemon_name ELSE '' END) AS super_effective_against_fight,
        MAX(CASE WHEN super_effective_type = 'Fire' THEN pokemon_name ELSE '' END) AS super_effective_against_fire,
        MAX(CASE WHEN super_effective_type = 'Flying' THEN pokemon_name ELSE '' END) AS super_effective_against_flying,
        MAX(CASE WHEN super_effective_type = 'Ghost' THEN pokemon_name ELSE '' END) AS super_effective_against_ghost,
        MAX(CASE WHEN super_effective_type = 'Grass' THEN pokemon_name ELSE '' END) AS super_effective_against_grass,
        MAX(CASE WHEN super_effective_type = 'Ice' THEN pokemon_name ELSE '' END) AS super_effective_against_ice,
        MAX(CASE WHEN super_effective_type = 'Normal' THEN pokemon_name ELSE '' END) AS super_effective_against_normal,
        MAX(CASE WHEN super_effective_type = 'Poison' THEN pokemon_name ELSE '' END) AS super_effective_against_poison,
        MAX(CASE WHEN super_effective_type = 'Psychic' THEN pokemon_name ELSE '' END) AS super_effective_against_psychic,
        MAX(CASE WHEN super_effective_type = 'Rock' THEN pokemon_name ELSE '' END) AS super_effective_against_rock,
        MAX(CASE WHEN super_effective_type = 'Steel' THEN pokemon_name ELSE '' END) AS super_effective_against_steel,
        MAX(CASE WHEN super_effective_type = 'Water' THEN pokemon_name ELSE '' END) AS super_effective_against_water,
        MAX(CASE WHEN super_effective_type = 'Fairy' THEN pokemon_name ELSE '' END) AS super_effective_against_fairy
    FROM v_super_effective_pokemons
    GROUP BY pokemon_name
) AS AllSuperEffective
```

```sql
WHERE
    super_effective_against_ground <> ''
    OR super_effective_against_electric <> ''
    OR super_effective_against_bug <> ''
    OR super_effective_against_dark <> ''
    OR super_effective_against_dragon <> ''
    OR super_effective_against_fight <> ''
    OR super_effective_against_fire <> ''
    OR super_effective_against_flying <> ''
    OR super_effective_against_ghost <> ''
    OR super_effective_against_grass <> ''
    OR super_effective_against_ice <> ''
    OR super_effective_against_normal <> ''
    OR super_effective_against_poison <> ''
    OR super_effective_against_psychic <> ''
    OR super_effective_against_rock <> ''
    OR super_effective_against_steel <> ''
    OR super_effective_against_water <> ''
    OR super_effective_against_fairy <> '';
GO
```

Results | Messages

	super_effective_against_ground	super_effective_against_electric	super_effective_against_bug	super_effective_against_dark	super_effecti
1	Altaria	Altaria			
2		Bagon			
3		Dragonair			
4	Dragonite	Dragonite			
5		Dratini			
6		Flygon			
7		Kingdra			
8		Latias			
9		Latios			
10					
11	Rayquaza	Rayquaza			
12	Salamence	Salamence			
13		Shelgon			
14		Vibrava			

Figures show view to display all Pokémon that are super effective against different types being called.

8. Show a user's catchlist

```
/*CTE to show all pokemon caught by user one*/
WITH All_Pokemon_caught_for_user AS (
    SELECT
        s.user_firstname + ' ' + s.user_lastname AS user_name,
        p.pokemon_pokedex_id,
        p.pokemon_name
    FROM
        users s
    JOIN
        pokemon p
    ON
        p.pokemon_user_id = s.user_id
    WHERE
        s.user_id = 1 and p.pokemon_caught ='Y'
)
SELECT
    user_name,
    pokemon_pokedex_id,
    pokemon_name
FROM
    All_Pokemon_caught_for_user;
GO

/*CTE to show all pokemon on a user's watchlist*/
```

Results Messages

	user_name	pokemon_pokedex_id	pokemon_name
1	Ash Ketchum	25	Pikachu
2	Ash Ketchum	147	Dratini
3	Ash Ketchum	148	Dragonair
4	Ash Ketchum	329	Vibrava
5	Ash Ketchum	334	Altaria
6	Ash Ketchum	373	Salamence

Figure shows a common table expression to display all the Pokémon caught by user 1.

9. Show a user's watchlist

```
700
701  /*CTE to show all pokemon on a user's watchlist*/
702  WITH All_Pokemon_watchlist_for_user AS (
703      SELECT
704          s.user_firstname + ' ' + s.user_lastname AS user_name,
705          p.pokemon_pokedex_id,
706          p.pokemon_name
707      FROM
708          users s
709      JOIN
710          pokemon p
711      ON
712          p.pokemon_user_id = s.user_id
713      WHERE
714          s.user_id = 2 and p.pokemon_watch_list ='Y'
715  )
716  SELECT
717      user_name,
718      pokemon_pokedex_id,
719      pokemon_name
720  FROM
721      All_Pokemon_watchlist_for_user;
722  GO
723
```

Results Messages

	user_name	pokemon_pokedex_id	pokemon_name
1	Misty Waterflower	149	Dragonite
2	Misty Waterflower	330	Flygon
3	Misty Waterflower	371	Bagon
4	Misty Waterflower	380	Latias
5	Misty Waterflower	384	Rayquaza

Figure shows a common table expression to display all the Pokémon saved to user 2's watchlist.

10. Show every Pokémon on both a watchlist and caught list to help understand how users interact with the database.

```
552
553  /*A view to see all pokemon in the database that appear on both the caught list and watchlist of every user*/
554  IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.VIEWS WHERE TABLE_NAME = 'v_all_pokemon_on_watchlist_all_users')
555  BEGIN
556      DROP VIEW v_all_pokemon_on_watchlist_all_users;
557  END
558  GO
559
560  CREATE VIEW v_all_pokemon_on_watchlist_all_users AS
561  SELECT
562      s.user_firstname + ' ' + s.user_lastname AS user_name,
563      p.pokemon_pokedex_id,
564      p.pokemon_name
565  FROM
566      users s
567  JOIN
568      pokemon p
569  ON
570      p.pokemon_user_id = s.user_id
571  WHERE
572      p.pokemon_caught = 'Y'
573      AND p.pokemon_watch_list = 'Y';
574  GO
575
```

```
726  /*A view to see all pokemon in the database that appear on both the caught list and watchlist for each user*/
727  SELECT
728      user_name,
729      pokemon_pokedex_id,
730      pokemon_name
731  FROM
732      v_all_pokemon_on_watchlist_all_users;
733  GO
734
```

Results Messages

	user_name	pokemon_pokedex_id	pokemon_name
1	Misty Waterflower	149	Dragonite
2	Misty Waterflower	371	Bagon
3	Ash Ketchum	373	Salamence
4	Misty Waterflower	384	Rayquaza

Figure showing a view created and called to display all Pokémon in the database on both the watchlist and catchlist of all users.

11. Show the average stat of all Pokémon (i.e. HP)

```
673    /*Average hp of all Pokemon*/
674    SELECT AVG(base_stat_hp) AS average_base_stat_hp
675    FROM base_stats;
676    GO
677
```

Results Messages

	average_base_stat_hp
1	69

Figures above show a query to display the average HP among all the Pokémon's base stats in the database

12. Show how many Pokémon have a speed over a value of 80

```
667    /*How many pokemon have a speed over 80?*/
668    SELECT COUNT(p.pokemon_pokedex_id) AS pokemon_count
669    FROM pokemon p
670    JOIN pokemon_base_stats pbs ON pbs.pokemon_base_stat_pokemon_pokedex_id = p.pokemon_pokedex_id
671    JOIN base_stats bs ON bs.base_stat_id = pbs.pokemon_base_stats_base_stat_id
672    WHERE bs.base_stat_speed > 80;
673    GO
674
```

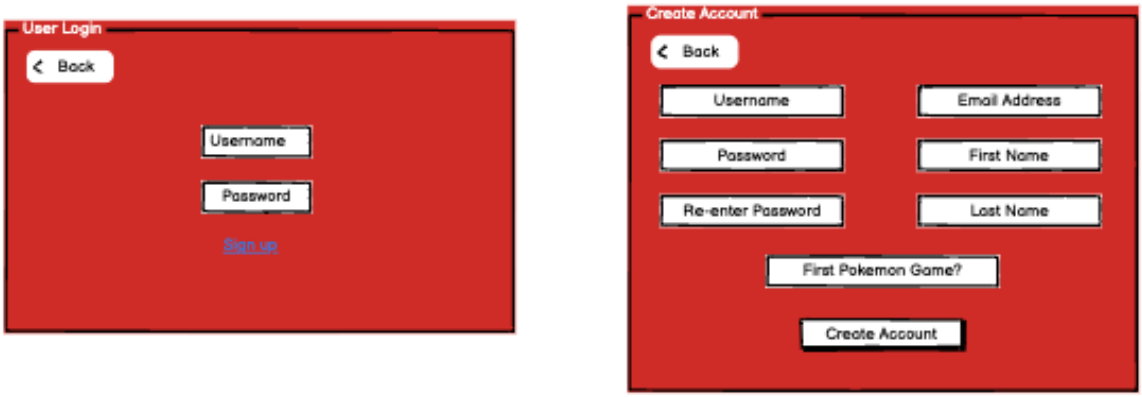
Results Messages

	pokemon_count
1	7

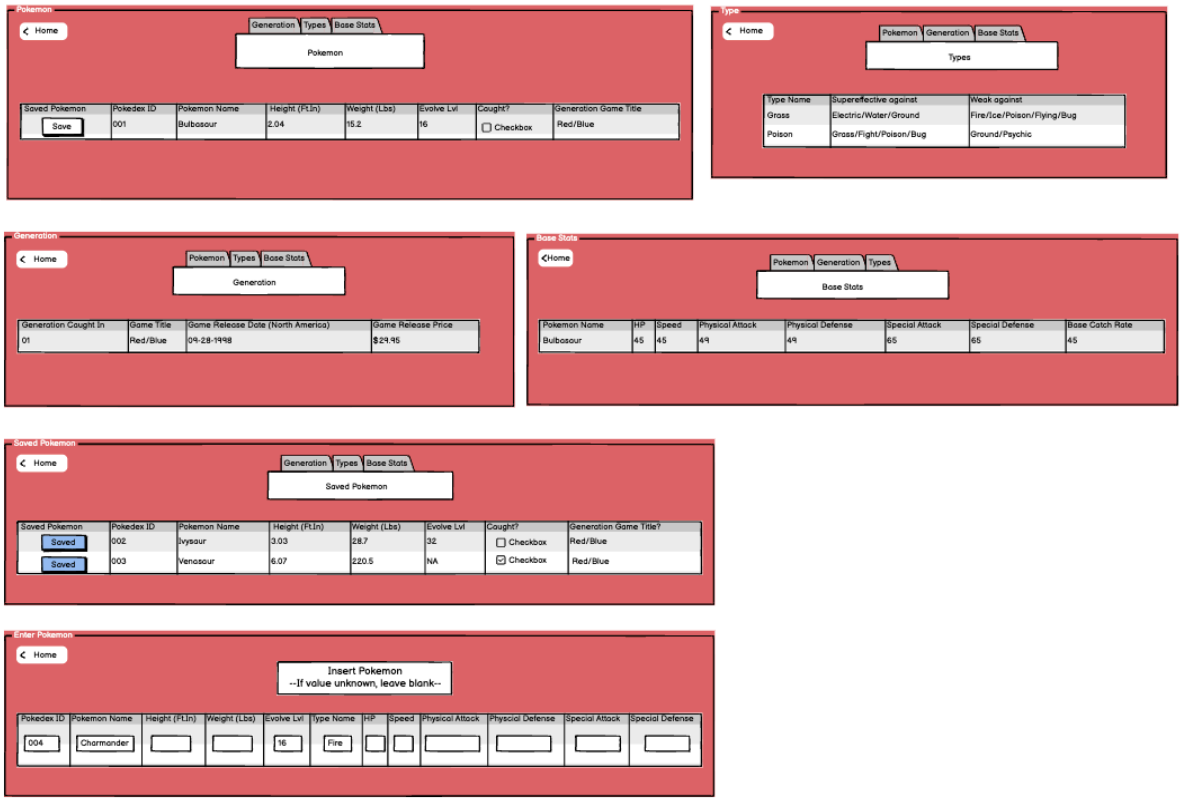
Figure shows a count of how many Pokémon have a base speed stat greater than a value of 80.

Application Screens

Home page and advanced search screens



User account sign in and create account screens



Output of all tables, saved Pokémon watchlist, and insert Pokémon screens

Team 6 Team Log

4/26 – Formed Teams chat

5/4 – Decided on “Pokemon” topic for project and submitted proposal to professor

5/15 – Meeting with team 6 members to discuss different models required for this project

5/21 – Meeting with team 6 members to discuss our created models and map out our next steps

5/29 – Completed Relationships Table, Conceptual Model, Logical Model using Draw.io and edited the models

6/1 – Discussed team member roles moving forward and remaining project requirements

6/5 – Completed presentation stage of project including application screens mockup using Balsamiq design tool

6/10 – Completed the up/down script and created the database G6_Pokedex

6/12 – Inserted all values into the tables “types” and “generation_caught_in”. inserted all values into the tables “pokemon” and “base_stats”. Inserted values into all bridge tables and successfully tested the table joins

6/13 – Completed three queries in the G6_Pokedex.dbo, added values into the “user” table, checked the normalization of the data, and edited application screens mockup in Balsamiq to better align with the database created

6/14 – Completed three queries in the G6_Pokedex.dbo, and added transactions, views, and Indexes to the G6_Pokedex.dbo

6/15 – Completed executive slides deck and overview narrative, completed reflection, completed data questions and answers

6/16 – Wrapped up SQL document, wrapped up Project Report document, completed PowerPoint document

6/17 – Submitted Project

Reflective Conclusion

Team 6 has been working to create a Pokedex for gamers to utilize while playing through the Pokémon video games. With so many different Pokémon in existence having so many different attributes, it is currently very difficult to search for Pokémon by these attributes. It is very surprising that the most popular database currently for Pokémon, serebii.net, doesn't have this feature available to users already. Following the implementation of our database, Pokémon research becomes much more streamlined. Team 6 is very proud of the work put in to create this database and a lot of planning went into this project before the actual design came to fruition.

Once we created the entities and relationships the project mapped itself out very nicely. It was strategic for the team to start at a high level and gradually increase specificity. Small tweaks were made to the original plan after the work had started, but for the most part the design followed the plan. One issue we had as a group is we wanted to add so many features that we just simply did not have enough time to get them all done. For example, if the team had been allotted more time, the indexes would have been completed to provide more efficiency within the database. Because we did not have more time, we needed to selectively choose what features should be required for the database to get the job done.

If more time was given, this database would exceed any resource that is currently available for researching Pokémon. If the database was completely populated, I have no doubt that it would be widely accepted and beloved by the Pokémon community. This community can be extremely critical and analytical, and team 6 created the G6 Pokedex database with the user in mind. The G6 Pokedex gives the user the edge they need to win more battles, catch more Pokémon, and be the very best that no one ever was.