Rony Richard,
Jade Miller,
Tyler Lowry,
Isaac Maughan

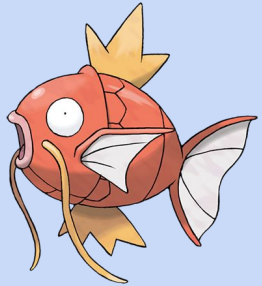
POKÉDEX DATABASE

Rationale

Pokémon have many qualities that are critical to game play. However, the original Pokédex database shows the Pokémon's height, weight, and evolutions first. This improved Pokédex database prioritizes data important to game play.

Design of UI Forms

The database's UI Forms tie directly to the rationale behind this database. Information important to game play is what the user will see first. For example: the Pokémon's type and attacks.



SQL Test Scripts

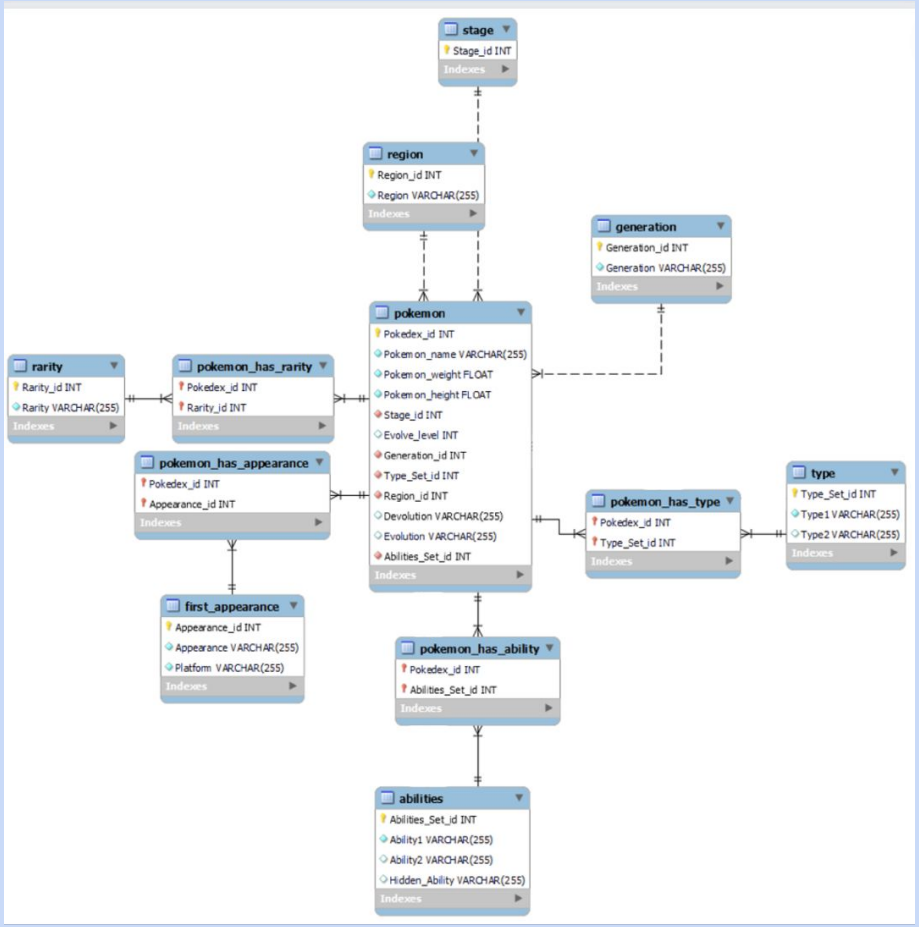
Our test scripts executed statements in a controlled setting and verified expected results notably when creating, reading, updating, and deleting data.

The test scripts themselves will be shown and explained more in depth later in the presentation.

Entity Relationship Design Model

POKÉDEX DATABASE

The entity relationship database is the map or blueprint of how a database is set. Based on ERD, people create the schematic model of the database using ERD which allows data integration and data analysis for the database to function as a whole. It's a blueprint that allows the user to determine how to structure and design the database before you would reverse engineer it into code.



Test Case Index

Pokedex_id Index

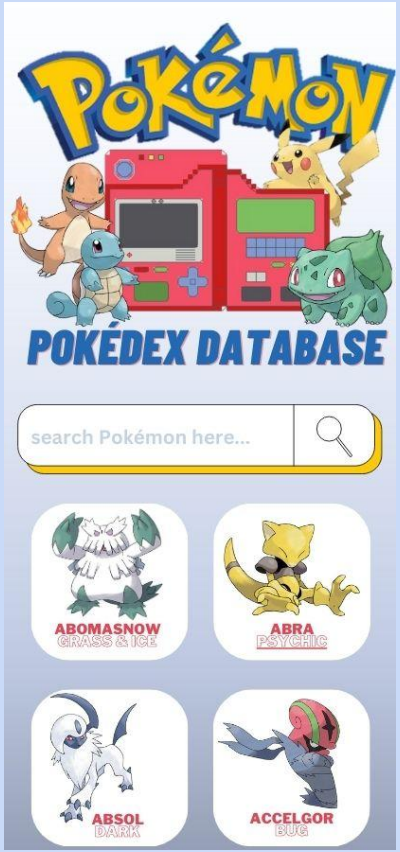
CREATE INDEX idx_pokemon_id
ON pokemon (Pokedex_id)

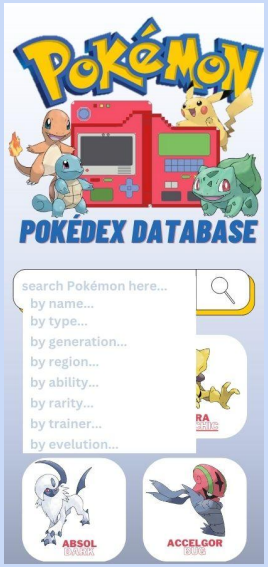
Pokemon_name Index

CREATE INDEX idx_pokemon_name
ON pokemon (Pokedex_name)

	Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
▶	pokemon	0	PRIMARY	1	Pokedex_id	A	38	NULL	NULL		BTREE
	pokemon	1	Stage_id	1	Stage_id	A	3	NULL	NULL		BTREE
	pokemon	1	Generation_id	1	Generation_id	A	8	NULL	NULL		BTREE
	pokemon	1	Type_Set_id	1	Type_Set_id	A	16	NULL	NULL		BTREE
	pokemon	1	Region_id	1	Region_id	A	8	NULL	NULL		BTREE
	pokemon	1	Abilities_Set_id	1	Abilities_Set_id	A	16	NULL	NULL	YES	BTREE
	pokemon	1	idx_pokemon_name	1	Pokemon_name	A	38	NULL	NULL		BTREE
	pokemon	1	idx_pokedex_id	1	Pokedex_id	A	38	NULL	NULL		BTREE

User Interface Mock-ups

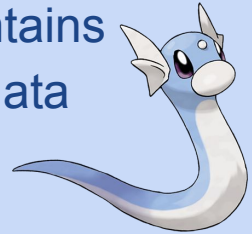




Users can search for a Pokémon in the search bar or look for a Pokémon listed below in alphabetical order

This User Interface makes human-computer interaction simple

Each Pokémon have their own page that contains their data



Test Case SQL Script Files

```sql
-- Test Case 1: Create Pokemon
-- Description: Verify the ability to create a new Pokemon entry in the system.
INSERT INTO pokemon (Pokedex_id, Pokemon_name, Pokemon_weight, Pokemon_height, Stage_id, Evolve_level, Generation_id, Type_Set_id, Region_id, Devolution, Evolution, Abilities_Set_id)
VALUES ('1000', 'Testmon', '10.0', '1.0', '1', '20', '8', '3', '1', 'None', 'EvolveMon', '4');

-- Test Case 2: Read Pokemon
-- Description: Verify the ability to retrieve Pokemon information from the system.
SELECT * FROM pokemon WHERE Pokedex_id = '1000';

-- Test Case 3: Update Pokemon
-- Description: Verify the ability to update Pokemon information in the system.
UPDATE pokemon SET Pokemon_weight = '15.0' WHERE Pokedex_id = '1000';

-- Test Case 4: Delete Pokemon
-- Description: Verify the ability to delete a Pokemon entry from the system.
DELETE FROM pokemon WHERE Pokedex_id = '1000';
```

Test case SQL script files are files that contain SQL statements used to perform specific test scenarios on a database. These files are commonly used in database testing to ensure that the database functions correctly and meets the desired requirements. Test case SQL script files typically include a set of SQL statements that simulate different operations on the database, such as data insertion, modification, retrieval, and deletion.

POKÉMON
POKÉDEX DATABASE

Create & Read

```
INSERT INTO pokemon
(Pokedex_id, Pokemon_name,
Pokemon_weight,
Pokemon_height, Stage_id,
Evolve_level, Generation_id,
Type_Set_id, Region_id,
Devolution, Evolution,
Abilities_Set_id)
VALUES ('1000', 'Testmon', '10.0',
'1.0', '1', '20', '8', '3', '1', 'None',
'EvolveMon', '4');
```

```
SELECT * FROM pokemon
WHERE Pokedex_id = '1000';
```

POKÉMON
POKÉDEX DATABASE

Update

```
UPDATE pokemon SET
Pokemon_weight =
'15.0' WHERE
Pokedex_id = '1000';
```

Delete

```
DELETE FROM
pokemon WHERE
Pokedex_id = '1000';
```

This Update statement looks for Pokémon with a weight of 15 and sets those Pokémon's Pokédex Id to 1000 so that the following Delete statement can delete those Pokémon

POKÉDEX DATABASE

Test Case Queries

```
SELECT
    P.Pokemon_name,
    FA.Appearance,
    FA.Platform
FROM
    Pokemon AS P
    LEFT JOIN Pokemon_has_appearance AS PHA ON P.Pokedex_id = PHA.Pokedex_id
    LEFT JOIN First_appearance AS FA ON PHA.Appearance_id = FA.Appearance_id
```

Results

Pokemon_name	Appearance	Platform
Arceus	Pokemon Diamond/Pearl	Nintendo DS
Blastoise	Pokemon Red/Blue	Game Boy
Blaziken	Pokemon Ruby/Sapphire	Game Boy Advance
Brionne	Pokemon Sword/Shield	Nintendo Switch
Bulbasaur	Pokemon Red/Blue	Game Boy
Chesnaught	Pokemon Sun/Moon	Nintendo 3DS
Chespin	Pokemon Sun/Moon	Nintendo 3DS
Combusken	Pokemon Ruby/Sapphire	Game Boy Advance
Dartrix	Pokemon Sword/Shield	Nintendo Switch
Decidueye	Pokemon Sword/Shield	Nintendo Switch
Dialga	Pokemon Diamond/Pearl	Nintendo DS
Espurr	Pokemon Sun/Moon	Nintendo 3DS
Gardevoir	Pokemon Ruby/Sapphire	Game Boy Advance
Grookey	Pokemon Scarlet/Violet	Nintendo Switch

This test query selects the Pokémon's names, first appearance, and the Platform where they first appear, testing if the joins have been created correctly.

Test Case Queries

Pokedex_id	Pokemon_name	Pokemon_weight	Pokemon_height	Stage_id	Ability1	Ability2	Hidden_Ability	Type1	Type2
280	Ralts	6.6	0.4			e	Telepathy	Psychic	Fairy
281	Kirlia	20.2	0.8			e	Telepathy	Psychic	Fairy
282	Gardevoir	48.4	1.6			e	Telepathy	Psychic	Fairy
677	Espurr	3.5	0.3	1	Keen Eye	Infiltrator	Own Tempo	Psychic	NULL
678	Meowstic	8.5	0.6	2	Keen Eye	Infiltrator	Prankster	Psychic	NULL

Results

```
SELECT
    P.Pokedex_id,
    P.Pokemon_name,
    P.Pokemon_weight,
    P.Pokemon_height,
    P.Stage_id,
    AB.Ability1,
    AB.Ability2,
    AB.Hidden_Ability,
    T.Type1,
    T.Type2
FROM
    Pokemon AS P
    LEFT JOIN Pokemon_has_ability AS PA ON P.Pokedex_id = PA.Pokedex_id
    LEFT JOIN Abilities AS AB ON PA.Abilities_Set_id = AB.Abilities_Set_id
    LEFT JOIN Pokemon_has_type AS PT ON P.Pokedex_id = PT.Pokedex_id
    LEFT JOIN Type AS T ON PT.Type_Set_id = T.Type_Set_id

WHERE
    AB.Ability2 IS NOT NULL;
```

A Pokemon having a second ability is rare, so this query searches for Pokemon that fit that criteria.

Any questions?