University of British Columbia, Vancouver

Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: ___4_____

Date: ____August 4, 2024______

Group Number: ___7_____

Name	Student Number	CS Alias (User ID)	Preferred Email Address
Hayden Chan	**49928377**	**p3x1c**	**haydenphchan@gmail.com**
Kristen Tan	**55327829**	**q7n0y**	**kristendenise9@gmail.com**
Cody Zhao	**36025468**	**r0j8e**	**codyz2770@gmail.com**

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your email address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

a. **A short description of the final project, and what it accomplished.**

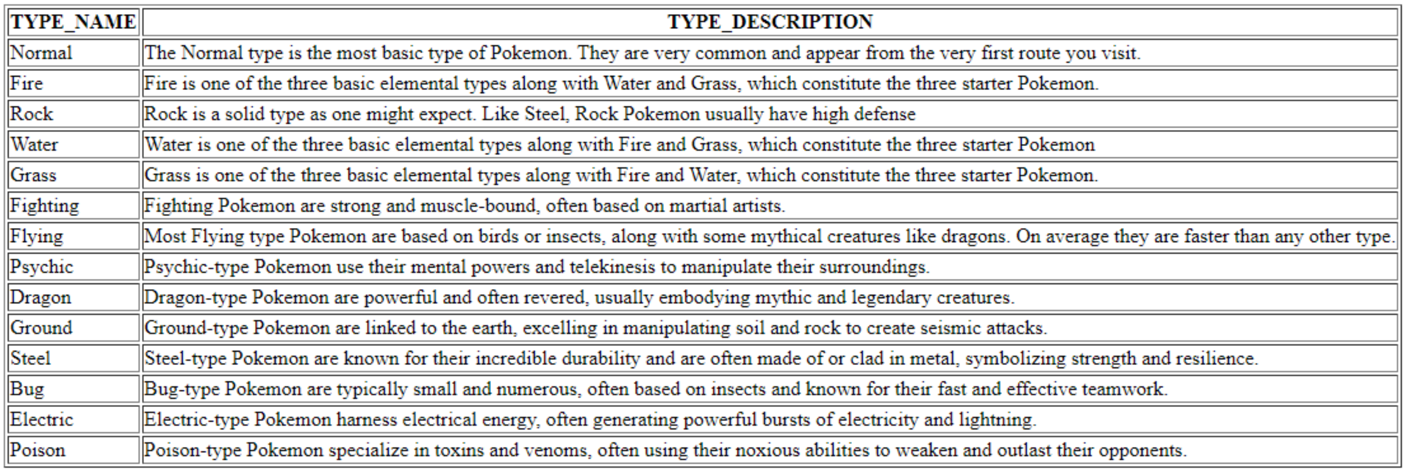
The final project was a comprehensive web-based Pokemon Database management system. It included various features allowing users to interact with the database, such as resetting the database, inserting new Pokemon moves with attributes like name, damage, accuracy, and type, and associating these moves with specific Pokemon. The project also enabled updating existing move information dynamically fetched from the database. It provided a functionality to count the tuples in the "Move" table, and showcased the release year and generation of each Pokemon through table joins. Users could project specific columns from the "Trainer_Info", "Uses", and "Trainer_Origin" tables, and filter the "Trainer_Info" table based on multiple conditions using AND/OR clauses. Additionally, the system could identify trainers in the game Pokemon Red, and display each type's minimum, maximum, average damage, and the number of moves. It included an aggregation feature to view types with more than a certain number of moves, and a nested aggregation feature to show each type's average damage excluding those below specified thresholds.

b. **A description of how your final schema differed from the schema you turned in.**

No Changes were made to the final schema.

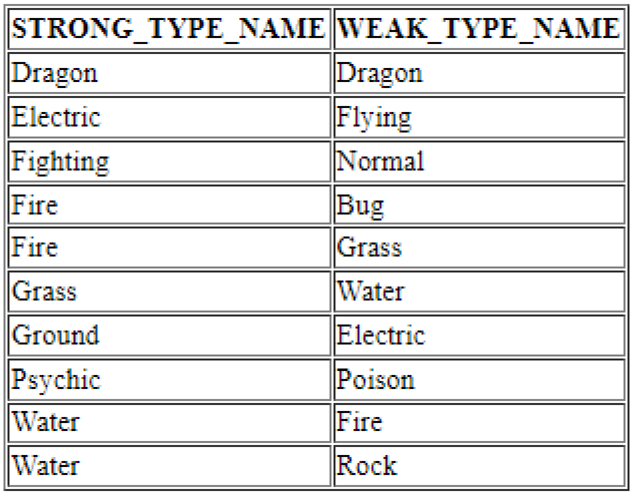
c. **A copy of the schema and screenshots that show what data is present in each relation after the SQL script from item #2 is run.**

Type(type_name, description)



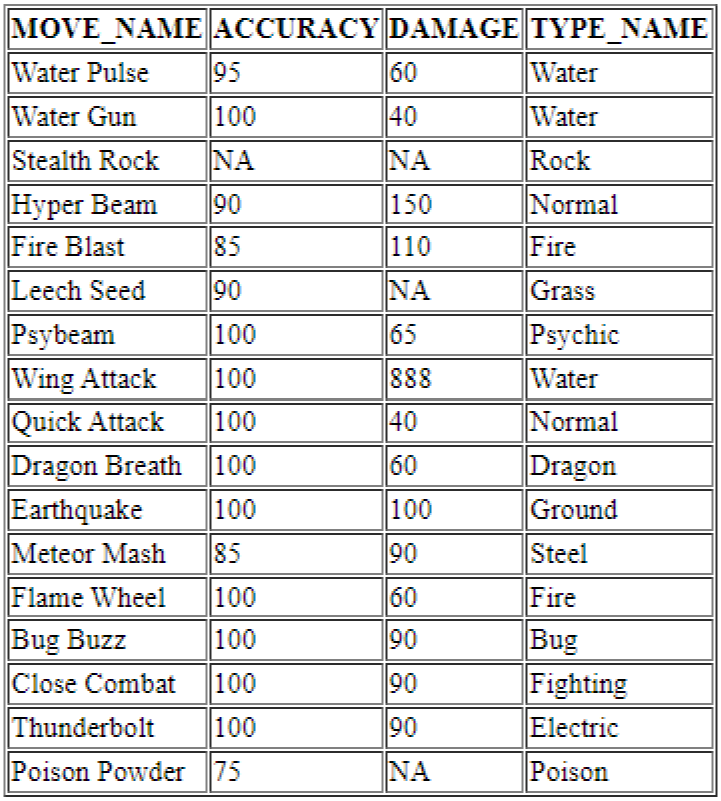
TYPE_NAME	TYPE_DESCRIPTION
Normal	The Normal type is the most basic type of Pokemon. They are very common and appear from the very first route you visit.
Fire	Fire is one of the three basic elemental types along with Water and Grass, which constitute the three starter Pokemon.
Rock	Rock is a solid type as one might expect. Like Steel, Rock Pokemon usually have high defense
Water	Water is one of the three basic elemental types along with Fire and Grass, which constitute the three starter Pokemon
Grass	Grass is one of the three basic elemental types along with Fire and Water, which constitute the three starter Pokemon.
Fighting	Fighting Pokemon are strong and muscle-bound, often based on martial artists.
Flying	Most Flying type Pokemon are based on birds or insects, along with some mythical creatures like dragons. On average they are faster than any other type.
Psychic	Psychic-type Pokemon use their mental powers and telekinesis to manipulate their surroundings.
Dragon	Dragon-type Pokemon are powerful and often revered, usually embodying mythic and legendary creatures.
Ground	Ground-type Pokemon are linked to the earth, excelling in manipulating soil and rock to create seismic attacks.
Steel	Steel-type Pokemon are known for their incredible durability and are often made of or clad in metal, symbolizing strength and resilience.
Bug	Bug-type Pokemon are typically small and numerous, often based on insects and known for their fast and effective teamwork.
Electric	Electric-type Pokemon harness electrical energy, often generating powerful bursts of electricity and lightning.
Poison	Poison-type Pokemon specialize in toxins and venoms, often using their noxious abilities to weaken and outlast their opponents.

TypeStrength(**strong_type_name**, **weak_type_name**)



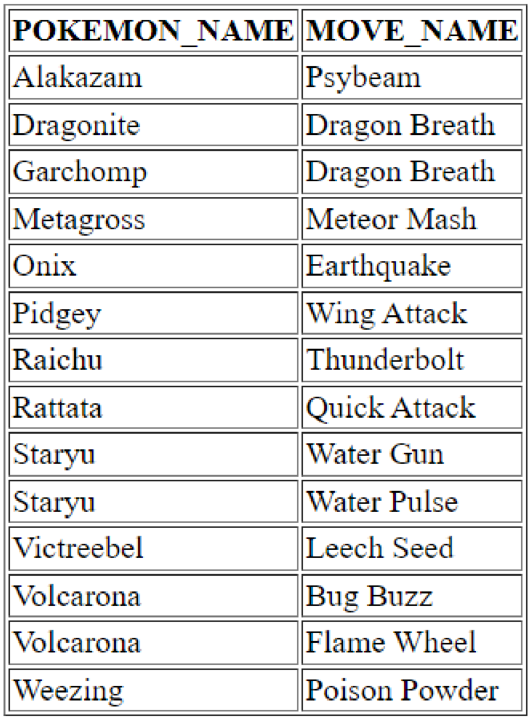
STRONG_TYPE_NAME	WEAK_TYPE_NAME
Dragon	Dragon
Electric	Flying
Fighting	Normal
Fire	Bug
Fire	Grass
Grass	Water
Ground	Electric
Psychic	Poison
Water	Fire
Water	Rock

Move(move_name, accuracy, damage, **type_name**)



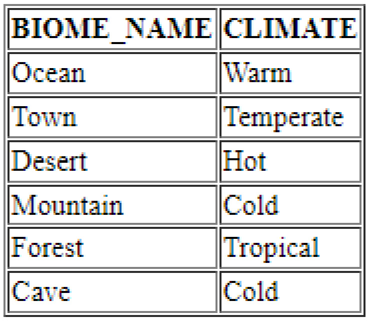
MOVE_NAME	ACCURACY	DAMAGE	TYPE_NAME
Water Pulse	95	60	Water
Water Gun	100	40	Water
Stealth Rock	NA	NA	Rock
Hyper Beam	90	150	Normal
Fire Blast	85	110	Fire
Leech Seed	90	NA	Grass
Psybeam	100	65	Psychic
Wing Attack	100	888	Water
Quick Attack	100	40	Normal
Dragon Breath	100	60	Dragon
Earthquake	100	100	Ground
Meteor Mash	85	90	Steel
Flame Wheel	100	60	Fire
Bug Buzz	100	90	Bug
Close Combat	100	90	Fighting
Thunderbolt	100	90	Electric
Poison Powder	75	NA	Poison

CanLearn (**pokemon_name, move_name**)



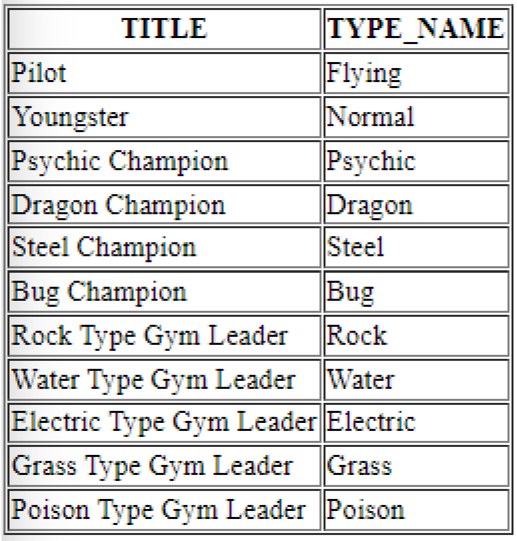
POKEMON_NAME	MOVE_NAME
Alakazam	Psybeam
Dragonite	Dragon Breath
Garchomp	Dragon Breath
Metagross	Meteor Mash
Onix	Earthquake
Pidgey	Wing Attack
Raichu	Thunderbolt
Rattata	Quick Attack
Staryu	Water Gun
Staryu	Water Pulse
Victreebel	Leech Seed
Volcarona	Bug Buzz
Volcarona	Flame Wheel
Weezing	Poison Powder

Habitat(biome_name, climate)



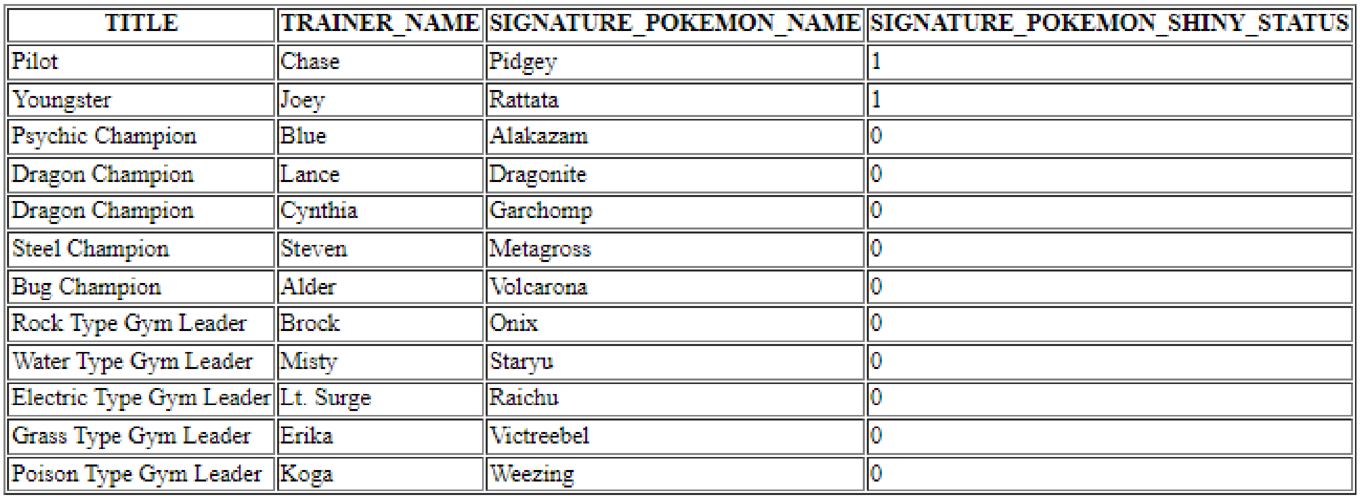
BIOME_NAME	CLIMATE
Ocean	Warm
Town	Temperate
Desert	Hot
Mountain	Cold
Forest	Tropical
Cave	Cold

Title_Type (title, **type_name**)



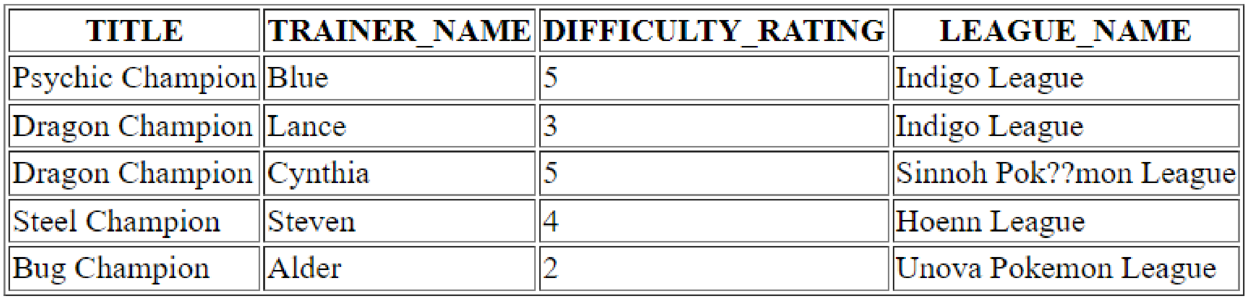
TITLE	TYPE_NAME
Pilot	Flying
Youngster	Normal
Psychic Champion	Psychic
Dragon Champion	Dragon
Steel Champion	Steel
Bug Champion	Bug
Rock Type Gym Leader	Rock
Water Type Gym Leader	Water
Electric Type Gym Leader	Electric
Grass Type Gym Leader	Grass
Poison Type Gym Leader	Poison

Trainer_Info (title, trainer_name, **signature_pokemon_name, signature_pokemon_shiny_status**)



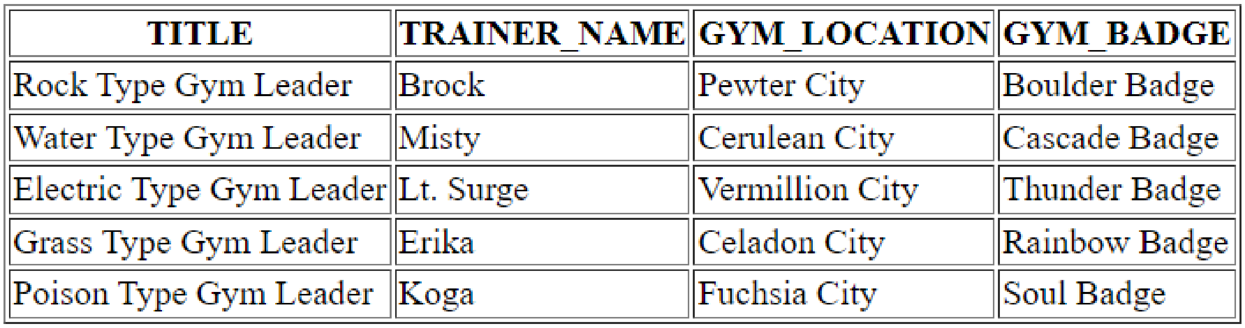
TITLE	TRAINER_NAME	SIGNATURE_POKEMON_NAME	SIGNATURE_POKEMON_SHINY_STATUS
Pilot	Chase	Pidgey	1
Youngster	Joey	Rattata	1
Psychic Champion	Blue	Alakazam	0
Dragon Champion	Lance	Dragonite	0
Dragon Champion	Cynthia	Garchomp	0
Steel Champion	Steven	Metagross	0
Bug Champion	Alder	Volcarona	0
Rock Type Gym Leader	Brock	Onix	0
Water Type Gym Leader	Misty	Staryu	0
Electric Type Gym Leader	Lt. Surge	Raichu	0
Grass Type Gym Leader	Erika	Victreebel	0
Poison Type Gym Leader	Koga	Weezing	0

Champion(**title, trainer_name**, difficulty_rating, league_name)



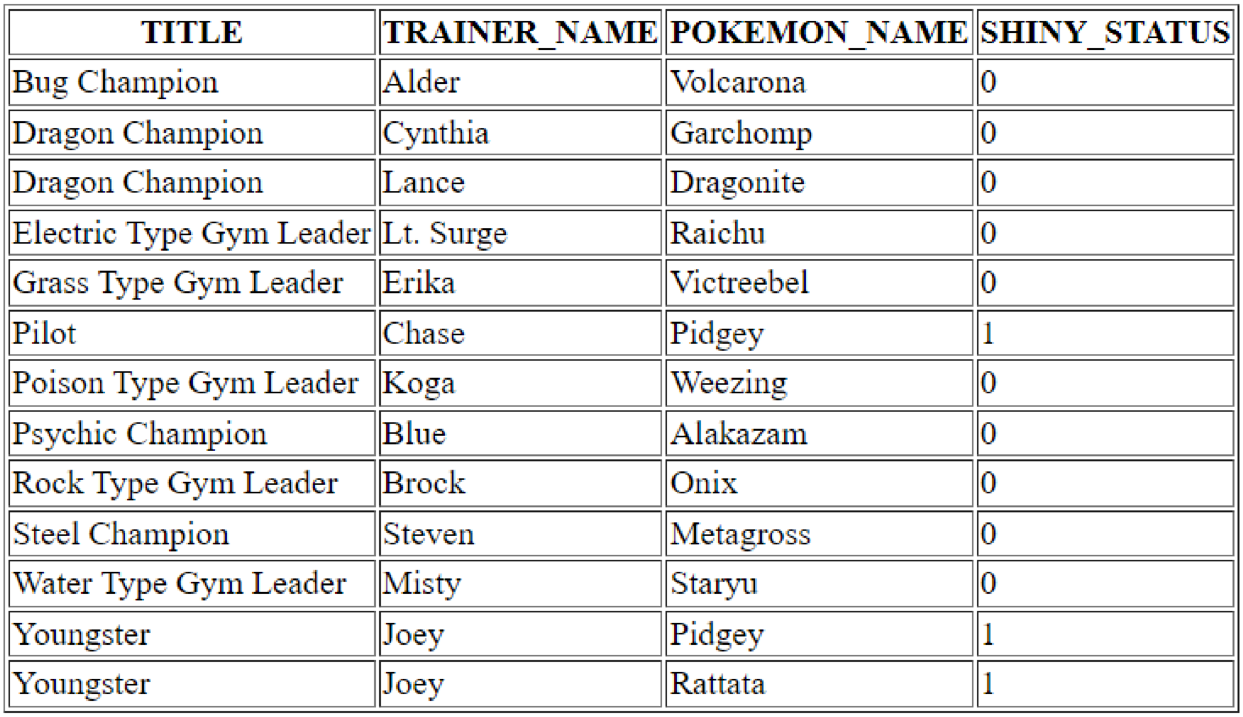
TITLE	TRAINER_NAME	DIFFICULTY_RATING	LEAGUE_NAME
Psychic Champion	Blue	5	Indigo League
Dragon Champion	Lance	3	Indigo League
Dragon Champion	Cynthia	5	Sinnoh Pok??mon League
Steel Champion	Steven	4	Hoenn League
Bug Champion	Alder	2	Unova Pokemon League

Gym Leader(**title, trainer_name**, gym_location, gym_badge)



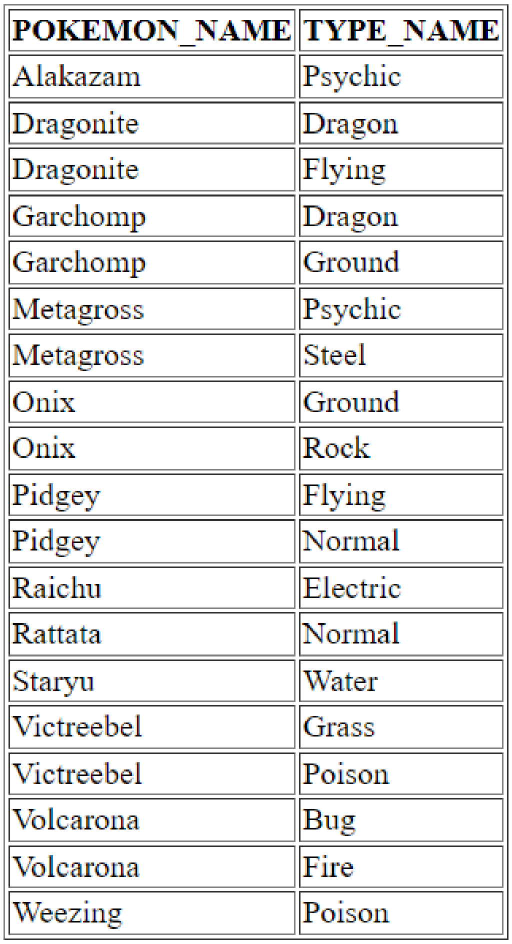
TITLE	TRAINER_NAME	GYM_LOCATION	GYM_BADGE
Rock Type Gym Leader	Brock	Pewter City	Boulder Badge
Water Type Gym Leader	Misty	Cerulean City	Cascade Badge
Electric Type Gym Leader	Lt. Surge	Vermillion City	Thunder Badge
Grass Type Gym Leader	Erika	Celadon City	Rainbow Badge
Poison Type Gym Leader	Koga	Fuchsia City	Soul Badge

Uses (**trainer_name, title, pokemon_name, shiny_status**)



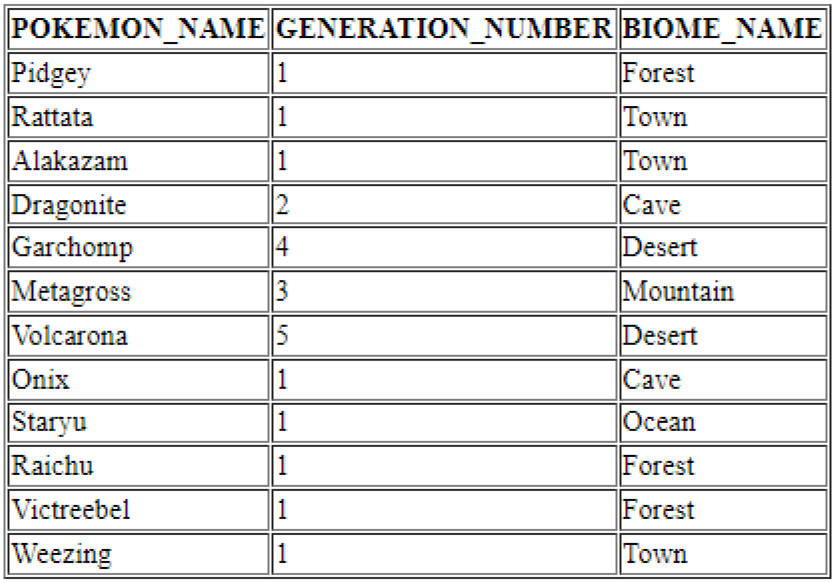
TITLE	TRAINER_NAME	POKEMON_NAME	SHINY_STATUS
Bug Champion	Alder	Volcarona	0
Dragon Champion	Cynthia	Garchomp	0
Dragon Champion	Lance	Dragonite	0
Electric Type Gym Leader	Lt. Surge	Raichu	0
Grass Type Gym Leader	Erika	Victreebel	0
Pilot	Chase	Pidgey	1
Poison Type Gym Leader	Koga	Weezing	0
Psychic Champion	Blue	Alakazam	0
Rock Type Gym Leader	Brock	Onix	0
Steel Champion	Steven	Metagross	0
Water Type Gym Leader	Misty	Staryu	0
Youngster	Joey	Pidgey	1
Youngster	Joey	Rattata	1

Categorised (**pokemon_name, type_name**)



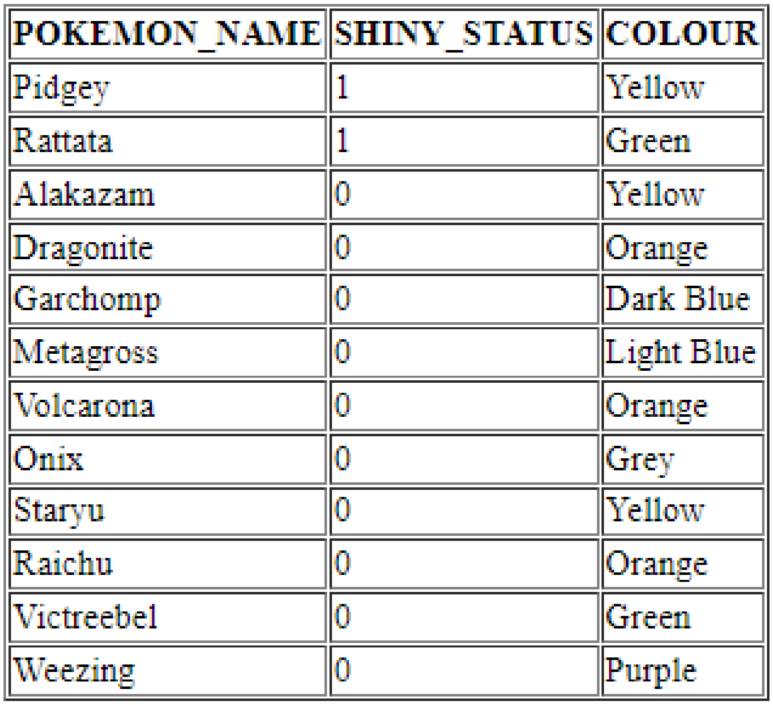
POKEMON_NAME	TYPE_NAME
Alakazam	Psychic
Dragonite	Dragon
Dragonite	Flying
Garchomp	Dragon
Garchomp	Ground
Metagross	Psychic
Metagross	Steel
Onix	Ground
Onix	Rock
Pidgey	Flying
Pidgey	Normal
Raichu	Electric
Rattata	Normal
Staryu	Water
Victreebel	Grass
Victreebel	Poison
Volcarona	Bug
Volcarona	Fire
Weezing	Poison

Pokemon_Basic_Info (pokemon_name, **generation_number, biome_name**)



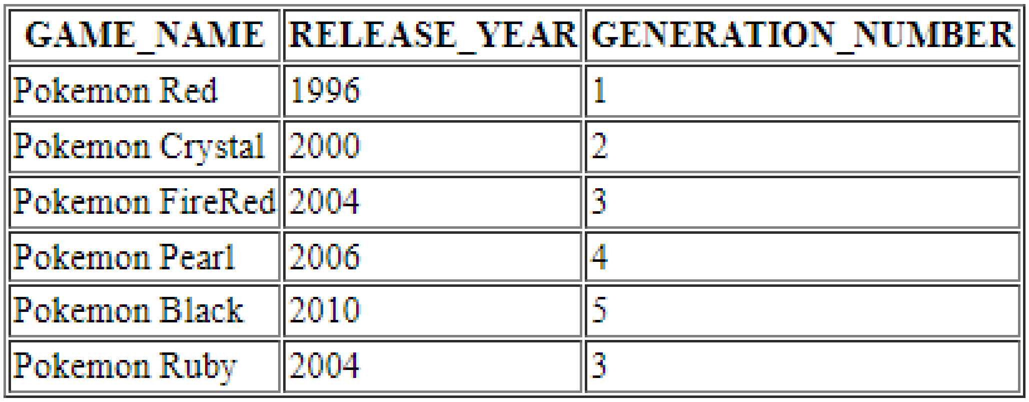
POKEMON_NAME	GENERATION_NUMBER	BIOME_NAME
Pidgey	1	Forest
Rattata	1	Town
Alakazam	1	Town
Dragonite	2	Cave
Garchomp	4	Desert
Metagross	3	Mountain
Volcarona	5	Desert
Onix	1	Cave
Staryu	1	Ocean
Raichu	1	Forest
Victreebel	1	Forest
Weezing	1	Town

Pokemon_Colour (pokemon_name, shiny_status, colour)



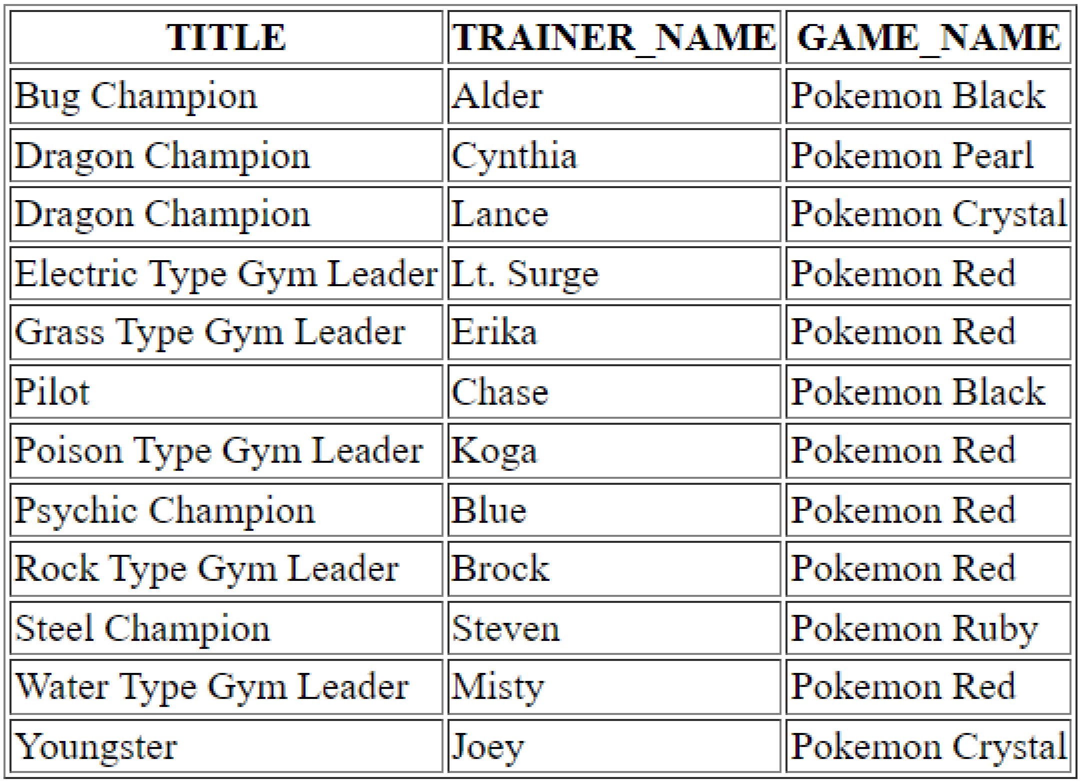
POKEMON_NAME	SHINY_STATUS	COLOUR
Pidgey	1	Yellow
Rattata	1	Green
Alakazam	0	Yellow
Dragonite	0	Orange
Garchomp	0	Dark Blue
Metagross	0	Light Blue
Volcarona	0	Orange
Onix	0	Grey
Staryu	0	Yellow
Raichu	0	Orange
Victreebel	0	Green
Weezing	0	Purple

Game(game_name, release_year, **generation_number**)



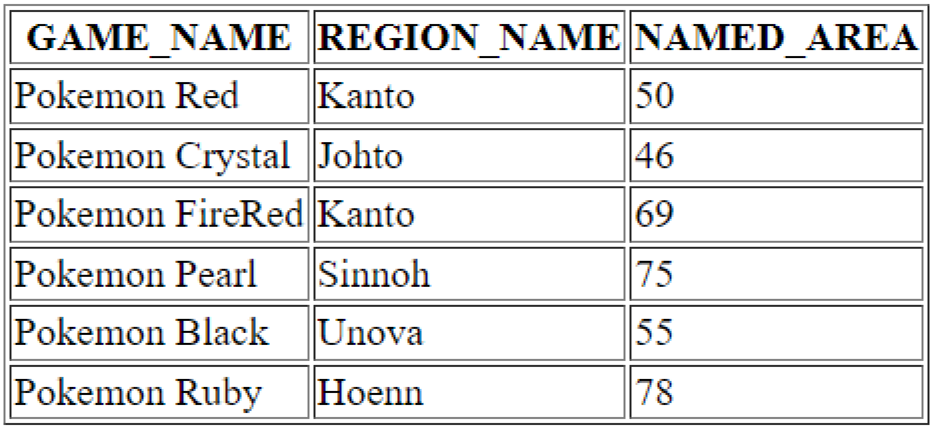
GAME_NAME	RELEASE_YEAR	GENERATION_NUMBER
Pokemon Red	1996	1
Pokemon Crystal	2000	2
Pokemon FireRed	2004	3
Pokemon Pearl	2006	4
Pokemon Black	2010	5
Pokemon Ruby	2004	3

Trainer_Origin(**trainer_name, title, game_name**)



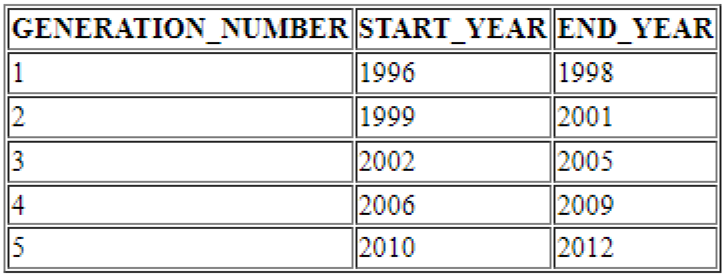
TITLE	TRAINER_NAME	GAME_NAME
Bug Champion	Alder	Pokemon Black
Dragon Champion	Cynthia	Pokemon Pearl
Dragon Champion	Lance	Pokemon Crystal
Electric Type Gym Leader	Lt. Surge	Pokemon Red
Grass Type Gym Leader	Erika	Pokemon Red
Pilot	Chase	Pokemon Black
Poison Type Gym Leader	Koga	Pokemon Red
Psychic Champion	Blue	Pokemon Red
Rock Type Gym Leader	Brock	Pokemon Red
Steel Champion	Steven	Pokemon Ruby
Water Type Gym Leader	Misty	Pokemon Red
Youngster	Joey	Pokemon Crystal

Game_Region(**game_name**, region_name, named_area)



GAME_NAME	REGION_NAME	NAMED_AREA
Pokemon Red	Kanto	50
Pokemon Crystal	Johto	46
Pokemon FireRed	Kanto	69
Pokemon Pearl	Sinnoh	75
Pokemon Black	Unova	55
Pokemon Ruby	Hoenn	78

Generation(generation_number, start_year, end_year)



GENERATION_NUMBER	START_YEAR	END_YEAR
1	1996	1998
2	1999	2001
3	2002	2005
4	2006	2009
5	2010	2012

d. A list of all SQL queries used and where it can be found in the code (i.e., file name and line number(s)). For SQL query requirements, check the rubric listed on Canvas for Milestone 4.

SQL Query	**Location in Code**

INSERT Operation
 	
if (filter_var($_POST['insDamage'], FILTER_VALIDATE_INT) === false) {
echo '<p style="color: red; font-weight: bold;">The damage must be an integer!</p>';
} else if (empty($_POST['insMoveName'])) {
echo '<p style="color: red; font-weight: bold;">Move Name can not be null!</p>';
} else {
//Getting the values from user and insert data into the table
$moveinfo = array(
":bind1" => $_POST['insMoveName'],
":bind2" => $_POST['insAcc'],
":bind3" => $_POST['insDamage'],
":bind4" => $_POST['insMoveType']
);
	Pokemon.php (line 438-465)

$canlearn = array(
":bind5" => $_POST['insMoveName'],
":bind6" => $_POST['insPokemon']
);
 	
$movetable = array(
$moveinfo
);
 	
$canlearntable = array(
$canlearn
);
 	
executeBoundSQL("INSERT INTO Move (move_name, accuracy, damage, type_name) VALUES (:bind1, :bind2, :bind3, :bind4)", $movetable);
 	
executeBoundSQL("INSERT INTO CanLearn (pokemon_name, move_name) VALUES (:bind6, :bind5)", $canlearntable);	

DELETE Operation $deltrainer = $_POST['DeleteTrainer']; $sql = "DELETE FROM Trainer_Info WHERE	Pokemon.php (line 479-482)

trainer_name ='" . $deltrainer . "'";
echo $sql;
 	
executePlainSQL($sql);	

UPDATE Operation
 	
$UpdateMove = $_POST['UpdateMove'];
$UpdateDamage = $_POST['UpdateDamage'];
$UpdateAccuracy = $_POST['UpdateAcc'];
$UpdateType = $_POST['UpdateType'];
 	
if (filter_var($_POST['UpdateDamage'], FILTER_VALIDATE_INT) === false) {
echo '<p style="color: red; font-weight: bold;">The damage must be an integer!</p>';
} else {
// you need the wrap the old name and new name values with single
$sql = "UPDATE Move SET move_name ='" . $UpdateMove . "', damage='" . $UpdateDamage . "', accuracy='" . $UpdateAccuracy . "', type_name='" . $UpdateType . "' WHERE move_name='" . $UpdateMove . "'";
 	
executePlainSQL($sql);
	Pokemon.php (line 399-412)

oci_commit($db_conn);
echo "Update Complete!";
}	

Selection
 	
$columns = isset($_GET['columns']) ? implode(", ", $_GET['columns']) : "*";
 	
$conditions = [];
if (isset($_GET['filter_columns']) && isset($_GET['operators']) && isset($_GET['values'])) {
foreach ($_GET['filter_columns'] as $index => $column) {
$operator = $_GET['operators'][$index];
$value = $_GET['values'][$index];
 	
$conditions[] = "$column $operator '$value'";
}
}
 	
if ($_GET['combine'] == "AND") {
$filterClause = !empty($conditions) ? " WHERE " . implode(" AND ", $conditions) : "";
} else {
$filterClause =
	Pokemon2.php (line 576-608)

!empty($conditions) ? " WHERE " . implode(" OR ", $conditions) : "";
}
$sql = "SELECT $columns FROM Trainer_Info" . $filterClause;
//echo $sql ;
if ($value == NULL) {
// echo "error";
$result = executePlainSQL("SELECT $columns FROM Trainer_Info");
printResult($result);
} else if ($_GET['filter_columns'][0] === "signature_pokemon_shiny_status" && filter_var($_GET['values'][0], FILTER_VALIDATE_INT) === false) {
echo '<p style="color: red; font-weight: bold;">Filter value for shiny status is not an integer</p>';
} else {
$result = executePlainSQL($sql);
printResult($result);
}	

Projection	Pokemon2.php (line 551-570)

$table = $_GET['table'];
 	
$columns = isset($_GET['columns']) ? implode(", ", $_GET['columns']) : "*";
 	
$conditions = [];
if (isset($_GET['filter_columns'])) {
foreach ($_GET['filter_columns'] as $index => $column) {
}
}
 	
$sql = "SELECT $columns FROM $table";
//echo $sql ;
 	
if ($table === "emptyTable") {
echo '<p style="color: red; font-weight: bold;">No Table Chosen!</p>';;
} else {
$result = executePlainSQL($sql);
 	
printResult($result);
}	

Join $query = "SELECT Pokemon_Basic_Info.pokemon_na	Pokemon2.php (line 643-647)

me, Generation.generation_number, Generation.start_year, Generation.end_year FROM Generation RIGHT JOIN Pokemon_Basic_Info ON Generation.generation_number=Pokemon_Basic_Info.generation_number WHERE Generation.generation_number ='" . $gennum . "' "; $result = executePlainSQL($query);	
Aggregation with Group By switch ($GroupByOptions) { case 'Minimum': $result = executePlainSQL("Select Move.type_name as Type, min (damage) as Minimum_Damage From Move, Type Where damage IS NOT NULL Group by Move.type_name Order by min (damage) desc");	Pokemon3.php (line 477-512)

```
printResult($result);
                                    break;

                           case 'Maximum':
                                    $result =
executePlainSQL("Select
Move.type_name as Type, max
(damage) as Maximum_Damage


From Move, Type


Where damage IS NOT NULL


Group by Move.type_name


Order by max (damage) desc");

printResult($result);
                                    break;

                           case 'Average':
                                    $result =
executePlainSQL("Select
Move.type_name as Type, avg
(damage) as Average_Damage


         From Move, Type


         Where damage IS NOT NULL
```

Group by Move.type_name
Order by avg (damage) desc");
printResult($result);
break;
case 'Amount of Moves':
$result = executePlainSQL("Select Move.type_name as Type, count (distinct move_name) as Number_Of_Moves
From Move, Type
Group by Move.type_name
Order by count (distinct move_name) desc");
printResult($result);
break;
}	

Aggregation with Having $query = "Select Move.type_name as Type, count(distinct move_name) as Amount_Of_Moves	Pokemon3.php (line 700-703)

From Move, Type Group by Move.type_name Having count(distinct move_name) > " . $HavingAmount . " order by count(distinct move_name) desc";	

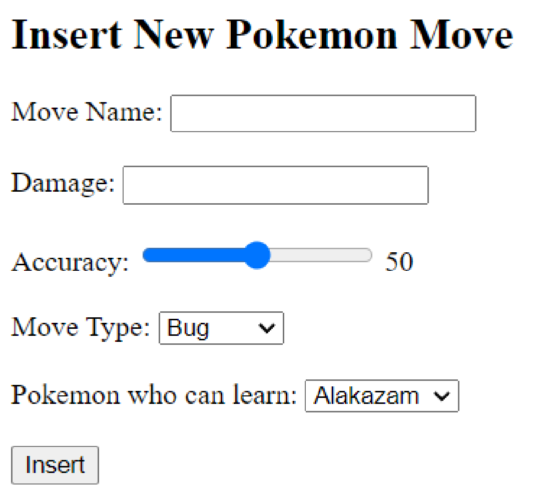
Nested Aggregation with Group By
switch ($NestedOptions) {
case 'Minimum':
$result = executePlainSQL("Select Move.type_name, avg(Move.damage)
From Move, Type
Group by Move.type_name
Having avg(Move.damage) >= (SELECT min(AVG(damage)) AS average_damage
FROM Move, Type
group by Move.type_name)
order by avg(Move.damage) desc");
	Pokemon3.php (line 670-703)

Division
 	
$sql = "SELECT T.trainer_name
FROM
Trainer_Info T
WHERE
NOT EXISTS (
 	
SELECT G.game_name
 	
FROM Game G
 	
WHERE G.game_name = 'Pokemon Red' AND NOT EXISTS (
 	
SELECT C.trainer_name
 	
FROM Trainer_Origin C
 	
WHERE C.trainer_name = T.trainer_name
 	
AND C.game_name = G.game_name
)
)";
	Pokemon3.php (line 617-628)

e. **Screenshots demonstrating the functionality of each query using the GUI. We want to see a before/during/after progression of events. For example, the before screenshot would be what data is in the table before you run the query, the during screenshot(s) is how the query is triggered using the GUI, and the after screenshot is what data is in your table afterwards. Please label each set of screenshots with the name of the query it is meant to address (e.g., “Insert Operation”).**

<table>
<tr><th>SQL Query</th><th>SCREENSHOT OF FUNCTIONALITY</th></tr>
<tr><td>INSERT Operation</td><td>

Before:



Insert New Pokemon Move

Move Name:

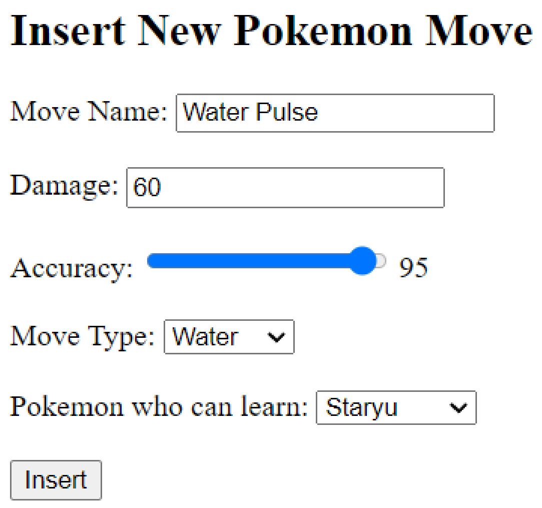
Damage:

Accuracy: 50

Move Type: Bug

Pokemon who can learn: Alakazam

During:



Insert New Pokemon Move

Move Name: Water Pulse

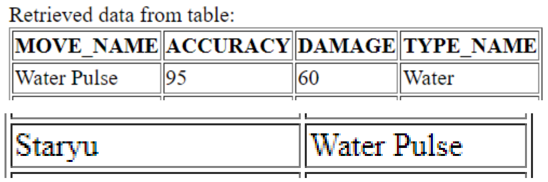
Damage: 60

Accuracy: 95

Move Type: Water

Pokemon who can learn: Staryu

After:



Retrieved data from table:

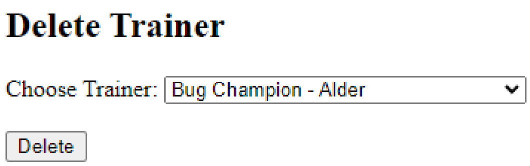
<table>
<tr><th>MOVE_NAME</th><th>ACCURACY</th><th>DAMAGE</th><th>TYPE_NAME</th></tr>
<tr><td>Water Pulse</td><td>95</td><td>60</td><td>Water</td></tr>
</table>

<table>
<tr><td>Staryu</td><td>Water Pulse</td></tr>
</table>

</td></tr>
<tr><td>DELETE Operation</td><td>

Before:

</td></tr>
</table>

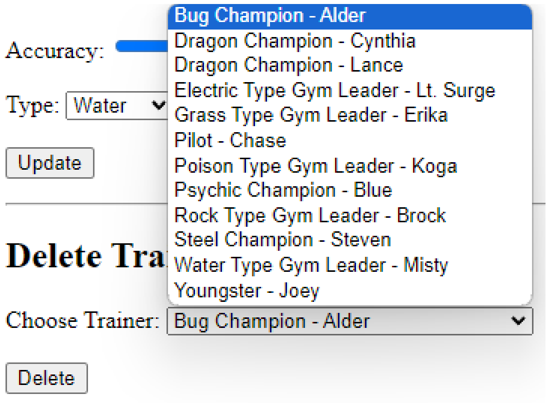


Delete Trainer

Choose Trainer: Bug Champion - Alder

Delete

During:



Accuracy:

Type: Water

Update

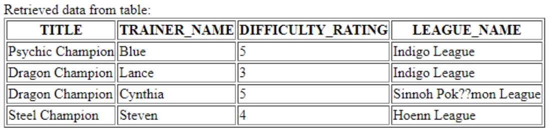
Delete Tra

Bug Champion - Alder
Dragon Champion - Cynthia
Dragon Champion - Lance
Electric Type Gym Leader - Lt. Surge
Grass Type Gym Leader - Erika
Pilot - Chase
Poison Type Gym Leader - Koga
Psychic Champion - Blue
Rock Type Gym Leader - Brock
Steel Champion - Steven
Water Type Gym Leader - Misty
Youngster - Joey

Choose Trainer: Bug Champion - Alder

Delete

After:

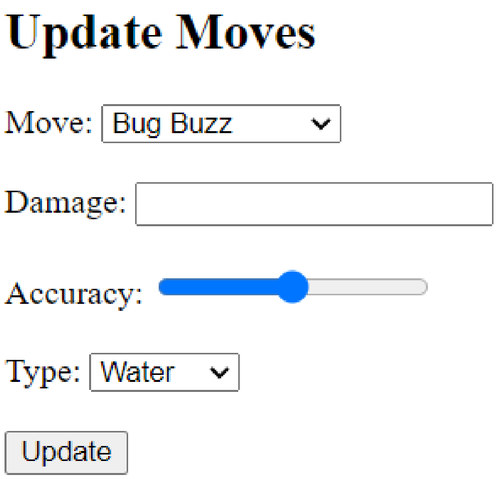


Retrieved data from table:

TITLE	TRAINER_NAME	DIFFICULTY_RATING	LEAGUE_NAME
Psychic Champion	Blue	5	Indigo League
Dragon Champion	Lance	3	Indigo League
Dragon Champion	Cynthia	5	Sinnoh Pok??mon League
Steel Champion	Steven	4	Hoenn League

UPDATE Operation

Before:



Update Moves

Move: Bug Buzz

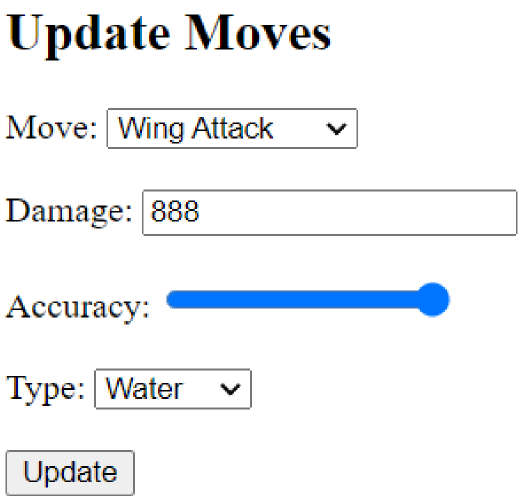
Damage:

Accuracy:

Type: Water

Update

During:



Update Moves

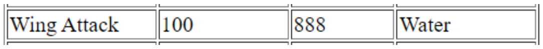
Move: Wing Attack

Damage: 888

Accuracy:

Type: Water

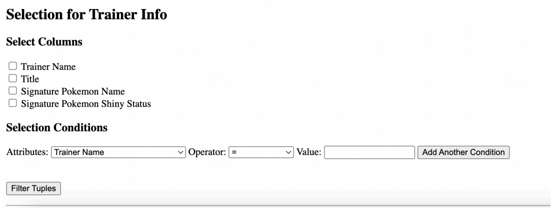
After:



Wing Attack	100	888	Water

Selection

Before:



Selection for Trainer Info

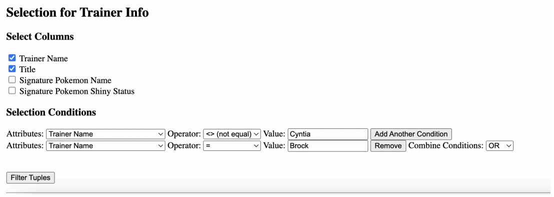
Select Columns

☐ Trainer Name
☐ Title
☐ Signature Pokemon Name
☐ Signature Pokemon Shiny Status

Selection Conditions

Attributes: Trainer Name Operator: = Value:

During:



Selection for Trainer Info

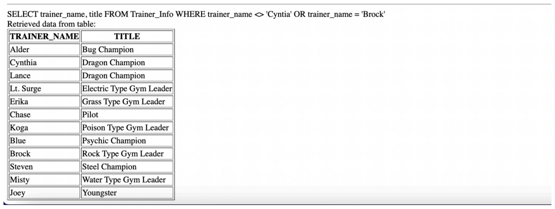
Select Columns

☑ Trainer Name
☑ Title
☐ Signature Pokemon Name
☐ Signature Pokemon Shiny Status

Selection Conditions

Attributes: Trainer Name Operator: <> (not equal) Value: Cyntia
Attributes: Trainer Name Operator: = Value: Brock Combine Conditions: OR

After:

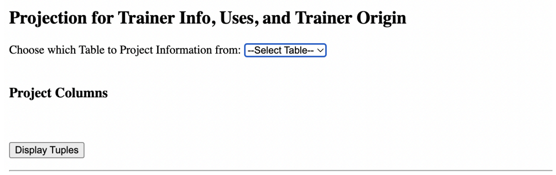


SELECT trainer_name, title FROM Trainer_Info WHERE trainer_name <> 'Cyntia' OR trainer_name = 'Brock'
Retrieved data from table:

TRAINER_NAME	TITLE
Alder	Bug Champion
Cynthia	Dragon Champion
Lance	Dragon Champion
Lt. Surge	Electric Type Gym Leader
Erika	Grass Type Gym Leader
Chase	Pilot
Koga	Poison Type Gym Leader
Blue	Psychic Champion
Brock	Rock Type Gym Leader
Steven	Steel Champion
Misty	Water Type Gym Leader
Joey	Youngster

Projection

Before

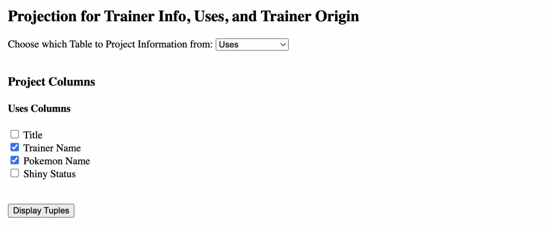


Projection for Trainer Info, Uses, and Trainer Origin

Choose which Table to Project Information from: --Select Table--

Project Columns

During:



Projection for Trainer Info, Uses, and Trainer Origin

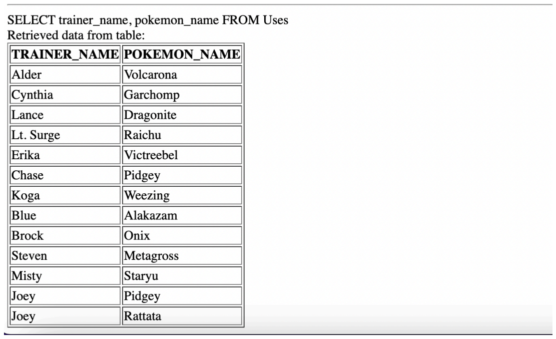
Choose which Table to Project Information from: Uses

Project Columns

Uses Columns

☐ Title
☑ Trainer Name
☑ Pokemon Name
☐ Shiny Status

After:

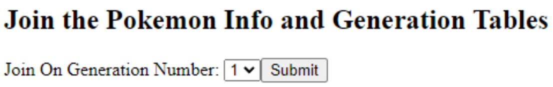


SELECT trainer_name, pokemon_name FROM Uses
Retrieved data from table:

TRAINER_NAME	POKEMON_NAME
Alder	Volcarona
Cynthia	Garchomp
Lance	Dragonite
Lt. Surge	Raichu
Erika	Victreebel
Chase	Pidgey
Koga	Weezing
Blue	Alakazam
Brock	Onix
Steven	Metagross
Misty	Staryu
Joey	Pidgey
Joey	Rattata

Join

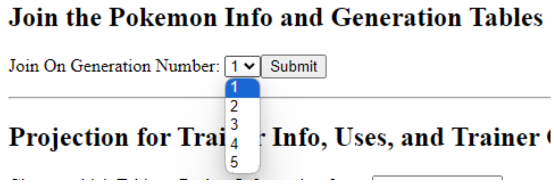
Before:



Join the Pokemon Info and Generation Tables

Join On Generation Number: 1

During:

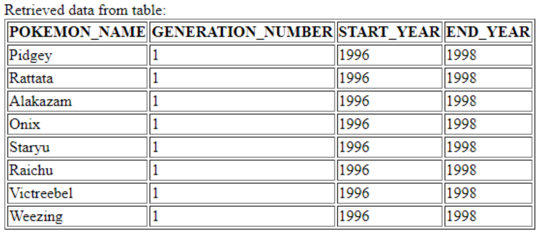


Join the Pokemon Info and Generation Tables

Join On Generation Number: 1

Projection for Trai r Info, Uses, and Trainer

After:

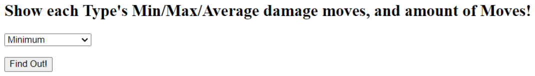


Retrieved data from table:

POKEMON_NAME	GENERATION_NUMBER	START_YEAR	END_YEAR
Pidgey	1	1996	1998
Rattata	1	1996	1998
Alakazam	1	1996	1998
Onix	1	1996	1998
Staryu	1	1996	1998
Raichu	1	1996	1998
Victreebel	1	1996	1998
Weezing	1	1996	1998

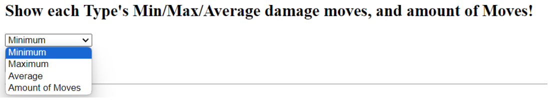
Aggregation with Group By

Before



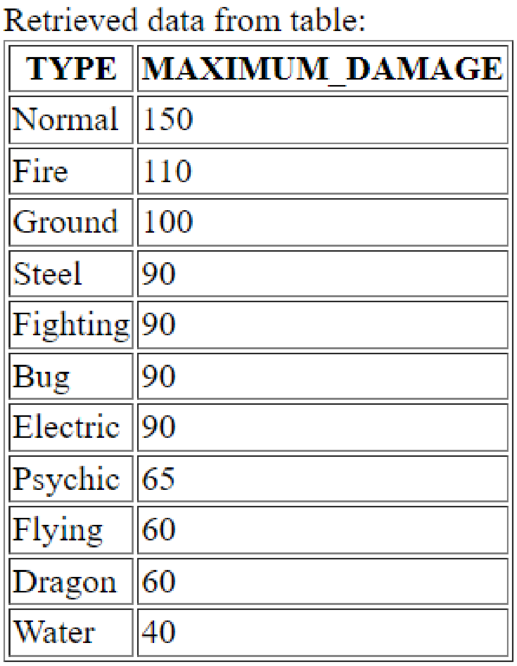
Show each Type's Min/Max/Average damage moves, and amount of Moves!

During



Show each Type's Min/Max/Average damage moves, and amount of Moves!

After

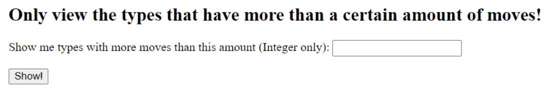


Retrieved data from table:

TYPE	MAXIMUM_DAMAGE
Normal	150
Fire	110
Ground	100
Steel	90
Fighting	90
Bug	90
Electric	90
Psychic	65
Flying	60
Dragon	60
Water	40

Aggregation with Having

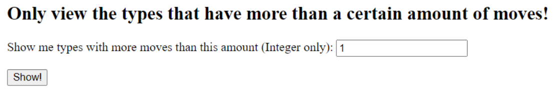
Before



Only view the types that have more than a certain amount of moves!

Show me types with more moves than this amount (Integer only):

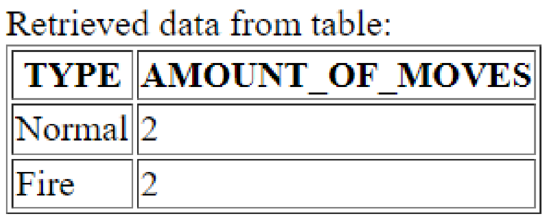
During



Only view the types that have more than a certain amount of moves!

Show me types with more moves than this amount (Integer only): 1

After

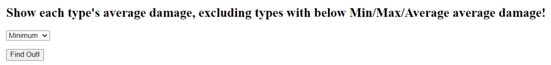


Retrieved data from table:

TYPE	AMOUNT_OF_MOVES
Normal	2
Fire	2

Nested Aggregation with Group By

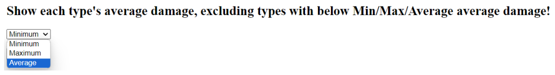
Before



Show each type's average damage, excluding types with below Min/Max/Average average damage!

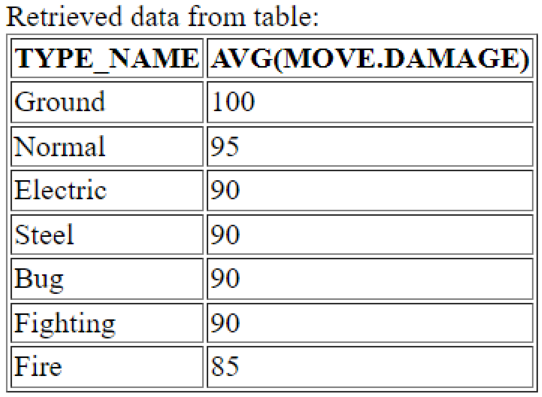
Minimum

During



Show each type's average damage, excluding types with below Min/Max/Average average damage!

After

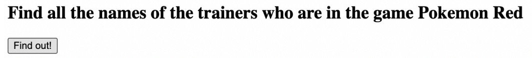


Retrieved data from table:

TYPE_NAME	AVG(MOVE.DAMAGE)
Ground	100
Normal	95
Electric	90
Steel	90
Bug	90
Fighting	90
Fire	85

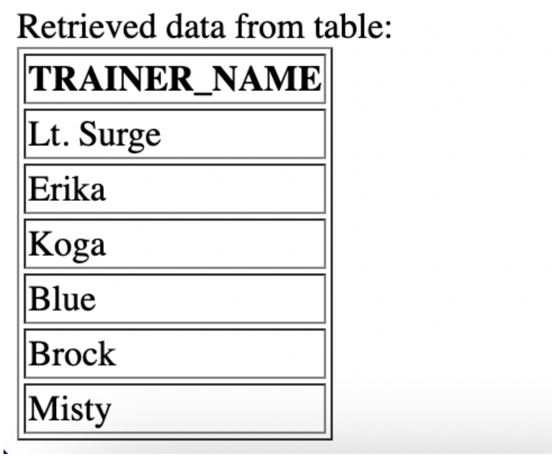
Division

Before:



Find all the names of the trainers who are in the game Pokemon Red

After:

<table>
<tr>
<td></td>
<td>


Retrieved data from table:

TRAINER_NAME
Lt. Surge
Erika
Koga
Blue
Brock
Misty

</td>
</tr>
</table>