Design-Your-Own Database

Nicholas Suchy

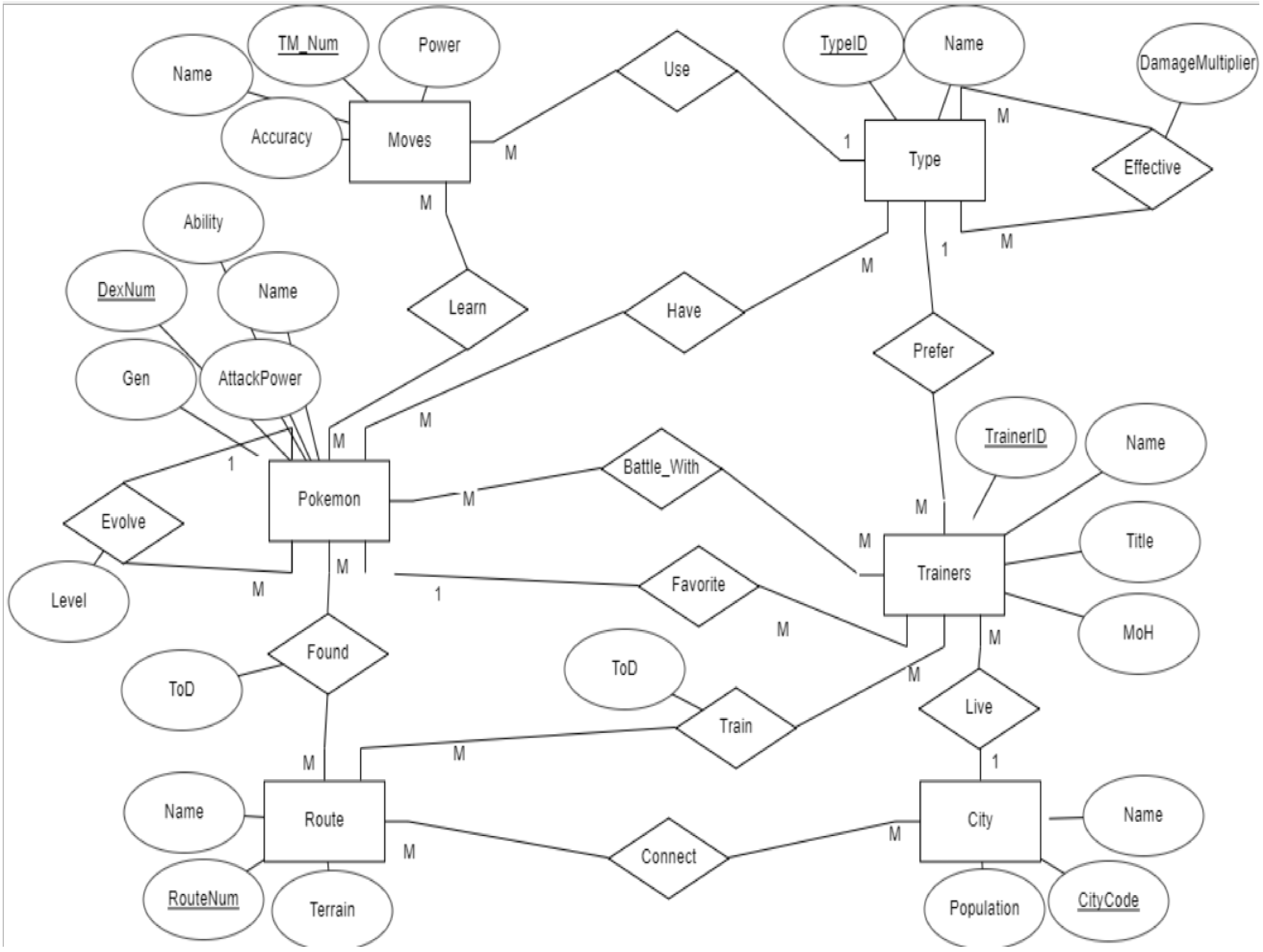
CMPT 308 – Database Systems

Section 111

This project is based off the Pokemon video game series. Specifically, this system will model the Sinnoh region, released in the 4th generation of the game. The system must be designed to satisfy the following requirements:

1. A region contains many species of Pokemon that can be found on a variety of routes. A Pokemon can also have multiple types. Pokemon can battle with many trainers and a Pokemon can be many different trainers' favorite. Pokemon learn many moves in order to succeed in battle. Pokemon are uniquely identified by their Pokedex number.
2. Pokemon can evolve into another Pokemon at a certain level. A Pokemon can evolve into many different Pokemon, but a Pokemon can only evolve from one Pokemon.
3. Routes contain a variety of different species of Pokemon. Pokemon can only be found on routes at certain times of day. In addition, a route connects to many cities. Like Pokemon, a route can have many trainers training on it at once. Routes are specified by their route number.
4. A city can connect to many routes. Cities have many trainers that live within them. Cities can be uniquely identified by their city code.
5. A Trainer lives in a city. However, trainers can be found training on a route at a certain time of day. Trainers have many routes that they may train on. Not every person the region is a trainer. Trainers have one favorite Pokemon, and trainers battle with many Pokemon. In addition, trainers usually have one preferred Pokemon type that they center their team around. Trainers can be uniquely identified by their trainer ID.
6. A type can be preferred by many trainers, shared among many Pokemon, and used by many moves. Types are uniquely denoted by their type ID.
7. Moves can be shared between Pokemon. Moves use a specific typing as well. Moves are uniquely identified by their TM number.
8. When Pokemon attack each other, the elemental type of the move can provide advantages and disadvantages. Some types deal x2 damage to others while some types can deal x.5 or even x0 damage. Finally, some types don't get an advantage or disadvantage over other types.
9. In addition, the system must be able to satisfy the following queries:

a. For each species of Pokemon, list its Pokedex number, name, ability, attack power, number of generation it was released in.
b. For each route in the region, list its route number, name, and terrain.
c. For each city, list its city code, name, and population.
d. For each trainer, list their trainer ID, name, title, and money on hand.
e. For each type, list its type ID, and name.
f. For each move, list the TM number, move name, move power, and the accuracy of the move. Additionally, be sure to list the type ID of the elemental type of the move.
g. For a given route and a given Pokemon, list the Pokemon's Pokedex number, the route's route number, and the time of day said Pokemon can be caught.
h. For a given attacking type and given defending type, list the attacking type ID, the defending type ID, and the damage multiplier of the move.
i. For a given Pokemon that evolves into another Pokemon, list the evolving Pokemon's Pokedex ID, the evolved Pokemon's Pokedex ID, and the level that the evolving Pokemon evolves into the evolved Pokemon.

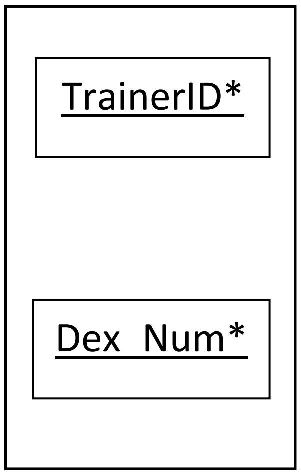


Moves
Name
TM_Num
Power
Accuracy
Use
Type
TypeID
Name
DamageMultiplier
Effective
Learn
Have
Prefer
Ability
DexNum
Name
Gen
AttackPower
Pokemon
Evolve
Level
Battle_With
Trainers
TrainerID
Name
Title
MoH
Favorite
Found
ToD
Train
ToD
Live
Route
Name
RouteNum
Terrain
Connect
City
Name
Population
CityCode
M
1

```
CREATE TABLE "111SUCHYN"."XBATTLEWITH"(

TRAINERID   NUMBER(*,0),

DEX_NUM     NUMBER(*,0),

CONSTRAINT PK_XBATTLEWITH PRIMARY KEY ("TRAINERID", "DEX_NUM"),

CONSTRAINT "FK_XBATTLEWITH_XTRAINER" FOREIGN KEY ("TRAINERID")

REFERENCES "111SUCHYN"."XTRAINERS" ("TRAINERID"),

CONSTRAINT "FK_XBATTLEWITH_XPOKEMON" FOREIGN KEY ("DEX_NUM")

REFERENCES "111SUCHYN"."XPOKEMON" ("DEX_NUM"));
```

This table contains data on what Pokemon trainers use in battle. Each row in the table has a trainer ID and a Pokedex number. A row is read by saying that the trainer that matches the trainer ID of the row battles with the Pokemon that matched the Pokedex number of the row.

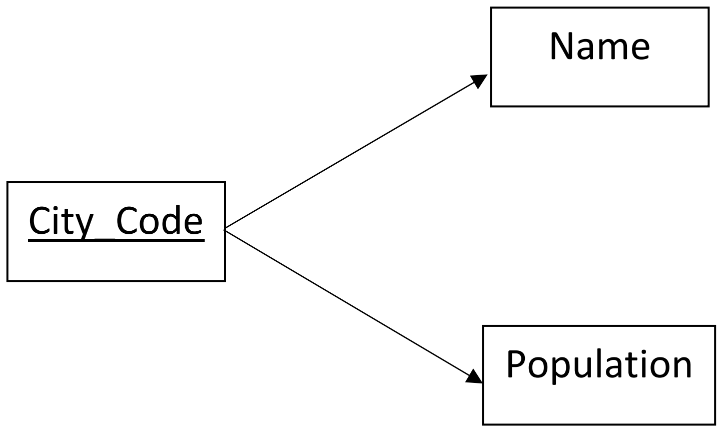
3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies (Asterisks represent foreign keys that reference another table)



```
CREATE TABLE "111SUCHYN"."XCITY"(
"CITY_CODE"  NUMBER(*,0),
"NAME"       VARCHAR2(20 BYTE),
"POPULATION" NUMBER(*,0),
CONSTRAINT "PK_XCITY" PRIMARY KEY ("CITY_CODE"));
```

This table contains data on each city in the region including the city's code, the name of the city and the population of the city.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

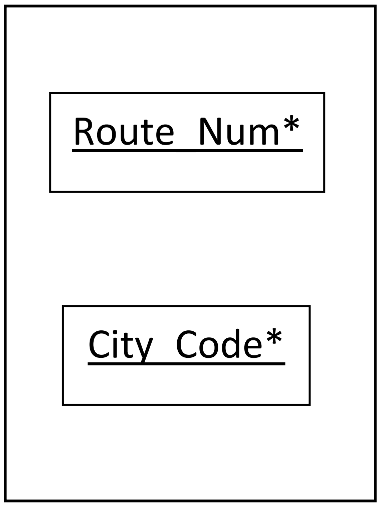


```
CREATE TABLE "111SUCHYN"."XCONNECT" (
"ROUTE_NUM" NUMBER(*,0),
"CITY_CODE" NUMBER(*,0),
CONSTRAINT "PK_XCONNECT" PRIMARY KEY ("ROUTE_NUM", "CITY_CODE"),
CONSTRAINT "FK_XCONNECT_XROUTE" FOREIGN KEY ("ROUTE_NUM")
REFERENCES "111SUCHYN"."XROUTE" ("ROUTE_NUM"),
CONSTRAINT "FK_XCONNECT_XCITY" FOREIGN KEY ("CITY_CODE")
REFERENCES "111SUCHYN"."XCITY" ("CITY_CODE"));
```

This table contains data on what routes connect which cities together. It shows what routes are connected to what cities and what cities touch what routes. If two or more city codes share a common route number within the table, those two cities, referenced by the city code, are connected by that route, referenced by the route number.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

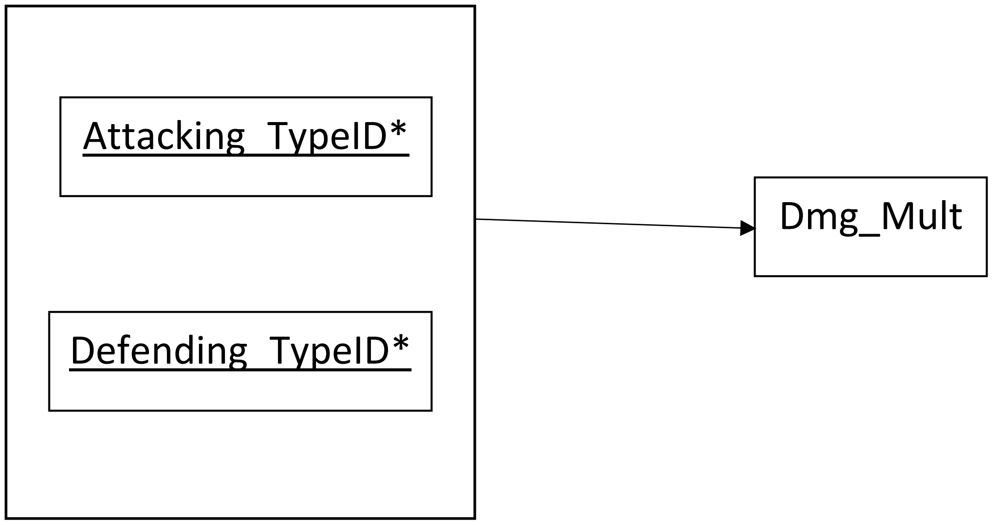


```
CREATE TABLE "111SUCHYN"."XEFFECTIVE" (

"ATTACKING_TYPEID" NUMBER(*,0),

"DEFENDING_TYPEID" NUMBER(*,0),

"DMG_MULT"         NUMBER(4,1),

CONSTRAINT "PK_XEFFECTIVE" PRIMARY KEY ("ATTACKING_TYPEID",
"DEFENDING_TYPEID"),

CONSTRAINT FK_XEFFECTIVE_XTYPE1 FOREIGN KEY ("ATTACKING_TYPEID")

REFERENCES "111SUCHYN"."XTYPE" ("TYPEID"),

CONSTRAINT FK_EEFFECTIVE_XTYPE2 FOREIGN KEY ("DEFENDING_TYPEID")

REFERENCES "111SUCHYN"."XTYPE" ("TYPEID"));
```

This table contains data about type effectiveness. For any two given types, it lists the attacking type, the defending type, and the damage multiplier the attacking type receives on its attack. A row is read by saying the attacking type gets a certain damage multiplier for attacking the specific defending type.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

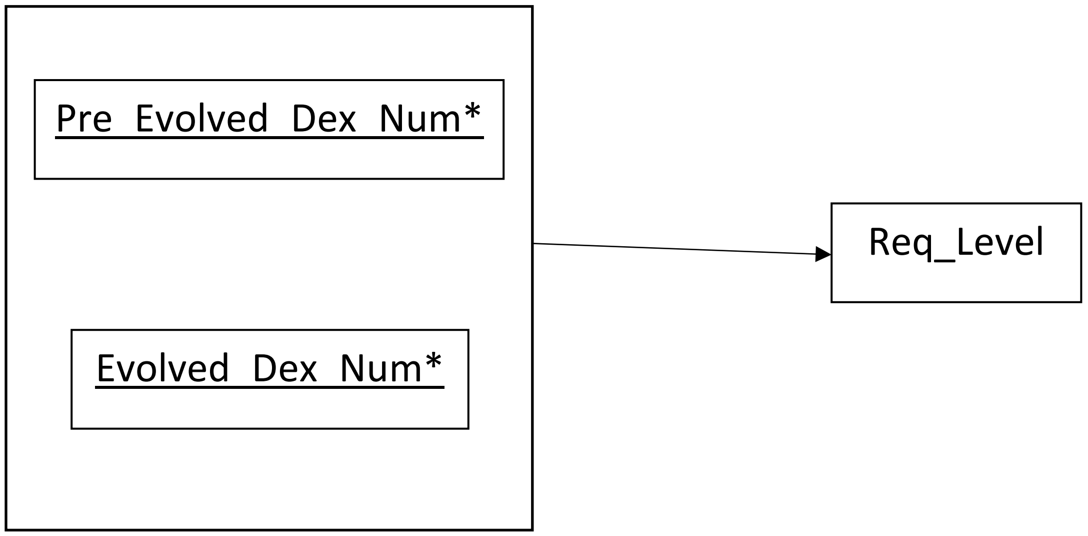


```
CREATE TABLE "111SUCHYN"."XEVOLVE" (

"PRE_EVOLVED_DEX_NUM" NUMBER(*,0),

"EVOLVED_DEX_NUM"     NUMBER(*,0),

"REQ_LEVEL"           NUMBER(*,0),

CONSTRAINT "PK_XEVOLVE" PRIMARY KEY ("PRE_EVOLVED_DEX_NUM",
"EVOLVED_DEX_NUM"),

CONSTRAINT FK_XEVOLVE_XPOKEMON1 FOREIGN KEY(PRE_EVOLVED_DEX_NUM)

REFERENCES "111SUCHYN"."XPOKEMON" ("DEX_NUM"),

CONSTRAINT FK_XEVOLVE_XPOKEMON2 FOREIGN KEY ("EVOLVED_DEX_NUM")

REFERENCES "111SUCHYN"."XPOKEMON" ("DEX_NUM"));
```

This table contains data on how Pokemon evolve. It lists the pre-evolved Pokemon's Pokedex number and it's evolved form's Pokedex number as well as the level requirement that the pre evolved Pokemon evolves at. A row shows that a Pokemon matching the pre evolved Pokedex number evolves into a Pokemon matching the evolved Pokedex number at the required level specified in the row.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

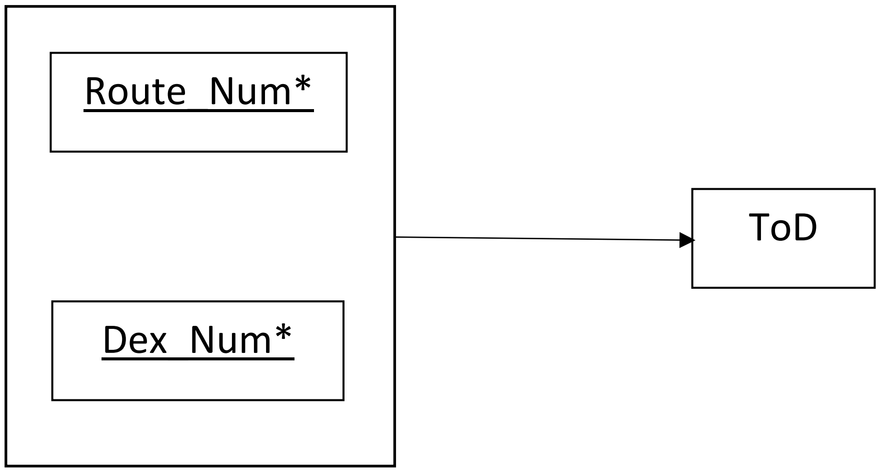


```
CREATE TABLE "111SUCHYN"."XFOUND" (
"ROUTE_NUM" NUMBER(*,0),
"DEX_NUM"   NUMBER(*,0),
"TOD"       VARCHAR2(20 BYTE),
CONSTRAINT "PK_XFOUND" PRIMARY KEY ("ROUTE_NUM", "DEX_NUM"),
CONSTRAINT "FK_XFOUND_XROUTE" FOREIGN KEY ("ROUTE_NUM")
REFERENCES "111SUCHYN"."XROUTE" ("ROUTE_NUM"),
CONSTRAINT "FK_XFOUND_XPOKEMON" FOREIGN KEY ("DEX_NUM")
REFERENCES "111SUCHYN"."XPOKEMON" ("DEX_NUM"));
```

This table contains data on where Pokemon can be found. Pokemon can only be found on routes. Pokemon are also found at certain times of day. A given route number in the table has many Pokedex numbers associated with it showing that Pokemon can be found on that route.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

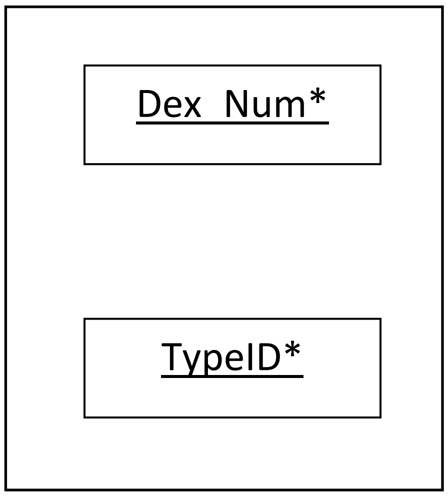


```
CREATE TABLE "111SUCHYN"."XHAVE" (
"DEX_NUM" NUMBER(*,0),
"TYPEID"  NUMBER(*,0),
CONSTRAINT "PK_XHAVE" PRIMARY KEY ("DEX_NUM", "TYPEID"),
CONSTRAINT "FK_XHAVE_XPOKEMON" FOREIGN KEY ("DEX_NUM")
REFERENCES "111SUCHYN"."XPOKEMON" ("DEX_NUM"),
CONSTRAINT "FK_XHAVE_XTYPE" FOREIGN KEY ("TYPEID")
REFERENCES "111SUCHYN"."XTYPE" ("TYPEID"));
```

This table contains data about what types Pokemon have. A given Pokedex number can be associated with many type IDs. Each row represents that a Pokemon with a matching Pokedex number in this table has the type that matches the type ID in the row.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

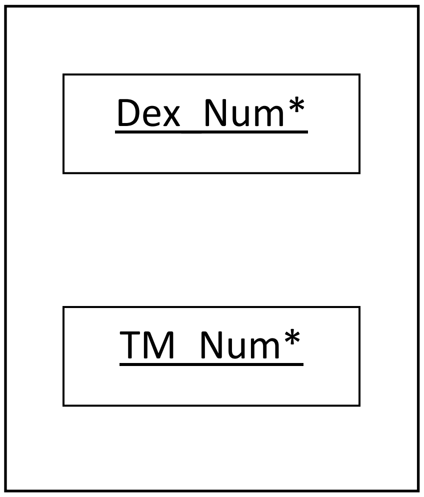


```
CREATE TABLE "111SUCHYN"."XLEARN" (
"DEX_NUM" NUMBER(*,0),
"TM_NUM"  NUMBER(*,0),
CONSTRAINT "PK_XLEARN" PRIMARY KEY ("DEX_NUM", "TM_NUM"),
CONSTRAINT "FK_XLEARN_XPOKEMON" FOREIGN KEY ("DEX_NUM")
REFERENCES "111SUCHYN"."XPOKEMON" ("DEX_NUM"),
CONSTRAINT "FK_XLEARN_XMOVE" FOREIGN KEY ("TM_NUM")
REFERENCES "111SUCHYN"."XMOVE" ("TM_NUM"));
```

This table contains data about what moves a Pokemon can learn. In each row, there is a Pokedex number and a TM number. The Pokemon corresponding to the Pokedex number can learn the move that corresponds to the TM number.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

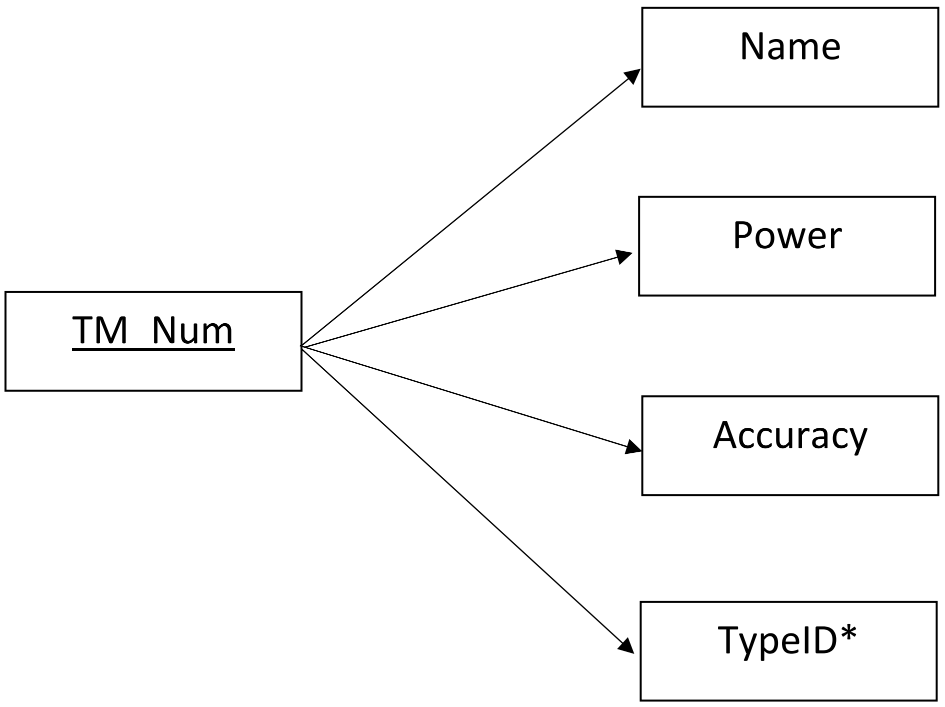


```
CREATE TABLE "111SUCHYN"."XMOVE" (
"TM_NUM"   NUMBER(*,0),
"NAME"     VARCHAR2(20 BYTE),
"POWER"    NUMBER(*,0),
"ACCURACY" NUMBER(*,0),
"TYPEID"   NUMBER(*,0),
CONSTRAINT "PK_XMOVE" PRIMARY KEY ("TM_NUM"),
CONSTRAINT "FK_XMOVE_XTYPE" FOREIGN KEY ("TYPEID")
REFERENCES "111SUCHYN"."XTYPE" ("TYPEID"));
```

This table contains data on moves. Moves are identified by their TM number and have a type ID that references that name of the type the move uses when it attacks. Moves also have a name, power, and accuracy.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

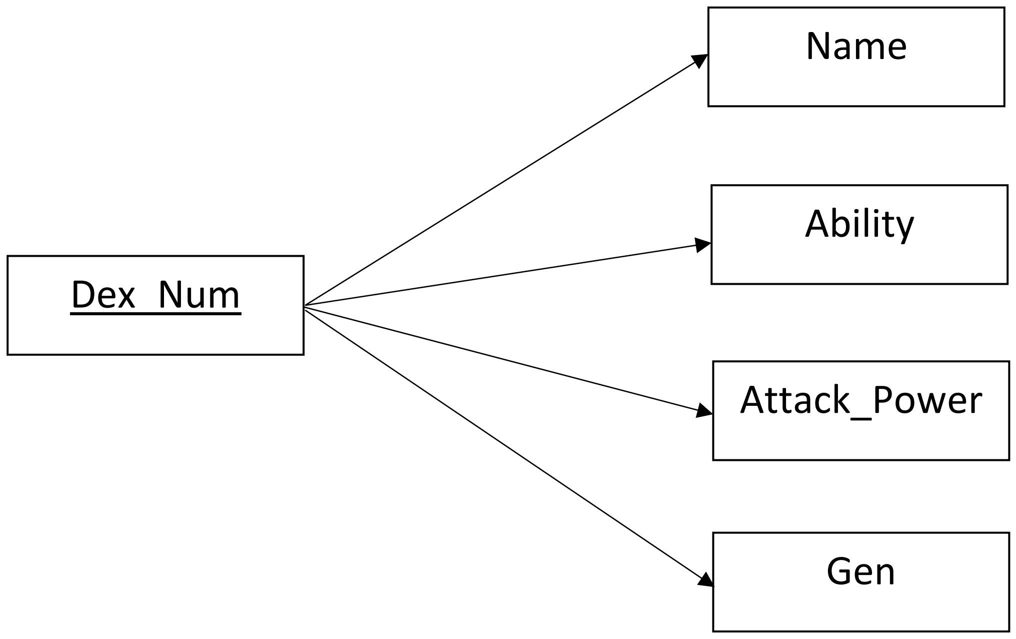


```
CREATE TABLE "111SUCHYN"."XPOKEMON" (
"DEX_NUM"       NUMBER(*,0),
"NAME"          VARCHAR2(20 BYTE),
"ABILITY"       VARCHAR2(20 BYTE),
"ATTACK_POWER"  NUMBER(*,0),
"GEN"           NUMBER(*,0),
CONSTRAINT "PK_XPOKEMON" PRIMARY KEY ("DEX_NUM"));
```

This table contains data on Pokemon. Pokemon can be identified by their Pokedex number and have a name, ability, attack power, and a generation that they were released in.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

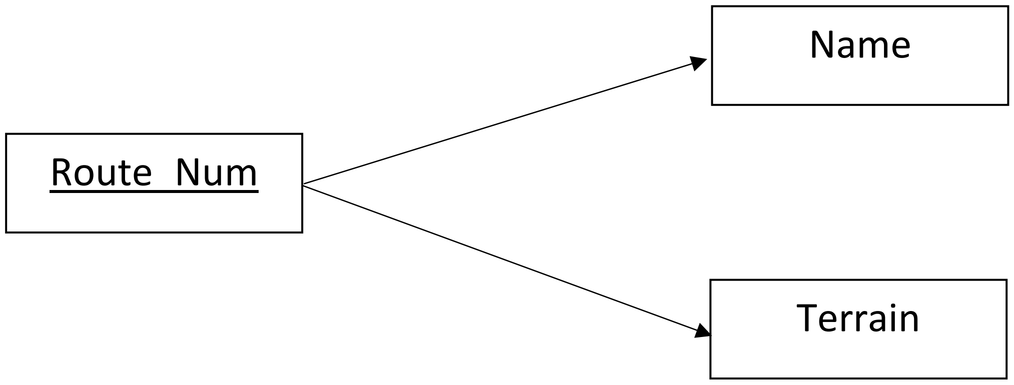


```
CREATE TABLE "111SUCHYN"."XROUTE" (
"ROUTE_NUM" NUMBER(*,0),
"NAME"      VARCHAR2(20 BYTE),
"TERRAIN"   VARCHAR2(20 BYTE),
CONSTRAINT "PK_XROUTE" PRIMARY KEY ("ROUTE_NUM"));
```

This table contains data about routes in the region. Routes are identified by a route number and have a name and terrain.

3rd Normal Form Justification: The table is in 2NF and doesn’t have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)

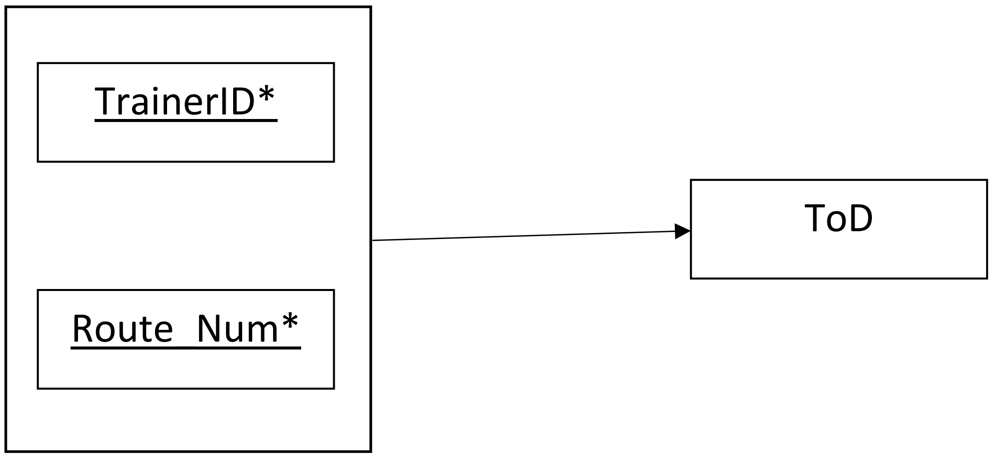


```
CREATE TABLE "111SUCHYN"."XTRAIN" (
"TRAINERID" NUMBER(*,0),
"ROUTE_NUM" NUMBER(*,0),
"TOD"       VARCHAR2(20 BYTE),
CONSTRAINT "PK_XTRAIN" PRIMARY KEY ("TRAINERID", "ROUTE_NUM"),
CONSTRAINT "FK_XTRAIN_XTRAINER" FOREIGN KEY ("TRAINERID")
REFERENCES "111SUCHYN"."XTRAINERS" ("TRAINERID"),
CONSTRAINT "FK_XTRAIN_XROUTE" FOREIGN KEY ("ROUTE_NUM")
REFERENCES "111SUCHYN"."XROUTE" ("ROUTE_NUM"));
```

This table contains data on what routes trainers train on. Each row contains a trainer ID, a route number, and a time of day. A row says that the trainer matching the given trainer ID, trains on the given route matching the route number at the specified time of day.

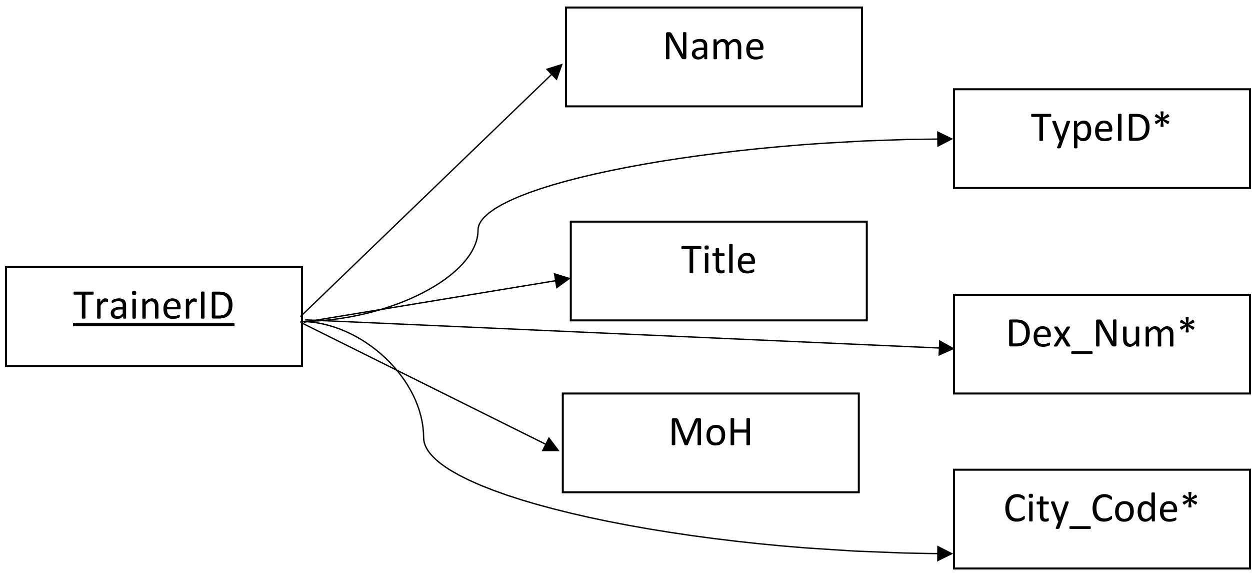
3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)



```
CREATE TABLE "111SUCHYN"."XTRAINERS" (
"TRAINERID" NUMBER(*,0),
"NAME"      VARCHAR2(20 BYTE),
"TITLE"     VARCHAR2(20 BYTE),
"MOH"       NUMBER(*,0),
"TYPEID"    NUMBER(*,0),
"DEX_NUM"   NUMBER(*,0),
"CITY_CODE" NUMBER(*,0),
CONSTRAINT "PK_XTRAINERS" PRIMARY KEY ("TRAINERID"),
CONSTRAINT "FK_XTRAINERS_XTYPE" FOREIGN KEY ("TYPEID")
REFERENCES "111SUCHYN"."XTYPE" ("TYPEID"),
CONSTRAINT "FK_XTRAINERS_XPOKEMON" FOREIGN KEY ("DEX_NUM")
REFERENCES "111SUCHYN"."XPOKEMON" ("DEX_NUM"),
CONSTRAINT "FK_XTRAINERS_XCITY" FOREIGN KEY ("CITY_CODE")
REFERENCES "111SUCHYN"."XCITY" ("CITY_CODE"));
```

This table contains data about trainers. Trainers are identified by their trainer ID and have a name, title, amount of money on hand. Additionally, trainers have a preferred type, that matches the row's type ID, a favorite Pokemon, that matches the row's Pokedex number, and a city that they live in, as designated by the row's city code.

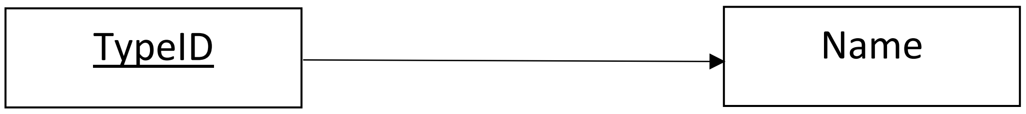


```
CREATE TABLE "111SUCHYN"."XTYPE" (
"TYPEID" NUMBER(*,0),
"NAME"   VARCHAR2(20 BYTE),
CONSTRAINT "PK_XTYPE" PRIMARY KEY ("TYPEID"));
```

This table contains data about types. Types are referenced by others database objects as needed using their type ID and have a name.

3rd Normal Form Justification: The table is in 2NF and doesn't have any transitive dependencies.

(Asterisks represent foreign keys that reference another table)



FinQuery1: Name every trainer that trains in Eterna Forest at night.

```
Create Or Replace View FinQuery1 As
Select Trainers.name
From xTrainers Trainers
Where Not Exists
    (Select *
    From xRoute Route
    Where Route.name = 'Eterna Forest'
    And Not Exists
        (Select *
        From xTrain Train
        Where Trainers.trainerID = Train.trainerID
        And Route.route_num = Train.route_num
        And Train.tod = 'Night'));
```

"NAME"
"Nick"
"Amanda"

Cardinality: 2

FinQuery2: Name moves that are only used by Pokemon that share a similar type with the move

```
Create Or Replace View FinQuery2 As
Select M.name As Move
From xMove M
Where M.TM_num Not In
    (Select L.TM_Num
    From xLearn L
    Where L.dex_num Not In
        (Select P.dex_num
        From xPokemon P, xHave H
        Where P.dex_num = H.dex_num
        And H.typeID = M.typeID));
```

"MOVE"
"Flamethrower"
"Waterfall"
"Hydro Pump"
"Ice Shard"

Cardinality: 4

FinQuery3: Get the title of trainers and their names for trainers that battle with none of the Pokemon that have the fire type.

```
Create Or Replace View FinQuery3 As

Select Trainer.title, Trainer.name As Trainer_Name

From xTrainers Trainer

Where Trainer.trainerID Not In

    (Select Battle.trainerID

    From xBattleWith Battle, xPokemon Pkmn

    Where Battle.dex_num = Pkmn.dex_num

    And Pkmn.dex_num In

        (Select Have.dex_num

        From xHave Have, xType EType

        Where Have.typeID = EType.typeID

        And EType.name = 'Fire'));
```

"TITLE"	"TRAINER_NAME"
"Champion"	"Cynthia"
"School Boy"	"Joey"
"Bug Catcher"	"Sally"
"Hiker"	"Brandon"
"School Girl"	"Mary"
"Swimmer"	"Mark"
"Dragon Tamer"	"Lance"
"Gym Leader"	"Byron"
"Gym Leader"	"Fantina"
"Palm Reader"	"Amanda"

Cardinality: 10

FinQuery4: Certain types cannot be damaged by other types, as designated by a damage multiplier of 0. Name all types along with any types that they are immune to, if any.

```
Create Or Replace View FinQuery4 As

Select Defending_Type.name As Defending_Type,
Ineffective_Attacks.name As Attacking_Type

From xType Defending_Type Left Join

    (Select *

     From xEffective E Join xType Attacking_Type On
     E.attacking_typeID = Attacking_Type.typeID

     Where dmg_mult = 0) Ineffective_Attacks

     On Defending_Type.typeID =
     Ineffective_Attacks.defending_typeID;
```

"DEFENDING_TYPE"	"ATTACKING_TYPE"
"Normal"	"Ghost"
"Fire"	""
"Water"	""
"Grass"	""
"Flying"	"Ground"
"Fighting"	""
"Poison"	""
"Electric"	""
"Ground"	"Electric"
"Rock"	""
"Psychic"	""
"Ice"	""
"Bug"	""
"Ghost"	"Fighting"
"Ghost"	"Normal"
"Steel"	"Poison"
"Dragon"	""
"Dark"	"Psychic"
"Fairy"	"Dragon"

Cardinality: 19

FinQuery5: Name all trainers along with the city that they live in along with any cities in which no trainers live.

```
Create Or Replace View FinQuery5 As

Select T.name As Trainer, C.name As City

From xTrainers T Right Join xCity C On T.city_code =
C.city_code;
```

"TRAINER"	"CITY"
"Mary"	"Twin Leaf Town"
"Joey"	"Sandgem Town"
"Gabbi"	"Jubilife City"
"Nick"	"Jubilife City"
"Brandon"	"Oreburgh City"
"Byron"	"Oreburgh City"
"Patricia"	"Oreburgh City"
"Sally"	"Floaroma Town"
"Amanda"	"Eterna City"
"Fantina"	"Hearthome City"
"Harry"	"Solaceon Town"
"Jane"	"Solaceon Town"
"Lance"	"Celestic Town"
"Cynthia"	"Celestic Town"
"Mark"	"Canalave City"
""	"Pokemon League"

Cardinality: 16

FinQuery6: Name all Pokemon along with the Pokemon they evolve from along with any Pokemon that don't evolve from another Pokemon, if any.

```
Create Or Replace View FinQuery6 As

Select Evolved_Pkmn.name As Evolved_Form, Child_Pkmn.name As
Pre_Evolved_FormFrom xPokemon Child_Pkmn Full Outer Join

    (Select *

    From xEvolve E Right Join xPokemon Evolved_Form

    On E.evolved_dex_num = Evolved_Form.dex_num) Evolved_Pkmn

    On Child_Pkmn.dex_num = Evolved_Pkmn.pre_evolved_dex_num;
```

"EVOLVED_FORM"	"PRE_EVOLVED_FORM"
"Espeon"	"Eevee"
"Umbreon"	"Eevee"
"Leafeon"	"Eevee"
"Glaceon"	"Eevee"
"Sylveon"	"Eevee"
"Togepi"	""
"Togetic"	"Togepi"
"Togekiss"	"Togetic"
"Houndour"	""
"Houndoom"	"Houndour"
"Rhyhorn"	""
"Rhydon"	"Rhyhorn"
"Rhyperior"	"Rhydon"
"Absol"	""
"Giratina"	""
""	"Spiritomb"
""	"Flareon"
""	"Glaceon"
""	"Azumarill"
""	"Palkia"

Cardinality: 148 (showing records 85 – 104)

FinQuery7: When a Pokemon uses a move of a type similar to itself the move gets flat bonus damage know as STAB (Same Type Attack Bonus). Additionally, if the damage multiplier is over 1 when one type attacks another, the damage is super effective. Name fully evolved Pokemon (Pokemon that can't evolve), their attack power, and the moves they learn that receive a STAB bonus and deal supper effective damage to Pokemon that are of Gabbi's favorite type.

```
Create Or Replace View FinQuery7 As

Select Pkmn.name As Pokemon, Pkmn.attack_power, Moves.name As
Move

From xPokemon Pkmn, xEvolve Evolved, xLearn Learn, xMove Moves,
xHave Have

Where Pkmn.dex_num = Evolved.evolved_dex_num

And Pkmn.dex_num Not In

    (Select Not_Fully_Evolved.pre_evolved_dex_num

    From xEvolve Not_Fully_Evolved)

And Pkmn.dex_num = Have.dex_num

And Pkmn.dex_num = Learn.dex_num

And Learn.TM_num = Moves.TM_num

And Moves.typeID = Have.typeID

And Moves.typeID In

    (Select Attacking_Type.typeID

    From xType Attacking_Type, xEffective Effective, xTrainers
Trainer

    Where Attacking_Type.typeID = Effective.attacking_typeID

    And Trainer.typeID = Effective.defending_typeID

    And Trainer.name = 'Gabbi'

    And Effective.dmg_mult > 1);
```

"POKEMON"	"ATTACK_POWER"	"MOVE"
"Torterra"	109	"Wood Hammer"
"Leafeon"	110	"Wood Hammer"
"Luxray"	120	"Thunderbolt"
"Raichu"	90	"Thunderbolt"
"Roserade"	125	"Wood Hammer"
"Jolteon"	110	"Thunderbolt"

Cardinality: 6

FinQuery8: For each type, get the name, the average attack power of Pokemon that have that type, and the number of Pokemon that have that type. Order the resulting table from highest average attack power to lowest.

```
Create Or Replace View FinQuery8 As

Select EType.name As Type, Round(Avg(Pkmn.attack_power)) As
Average_Attack_Power, Count(Have.dex_num) As Number_Pokemon

From xType EType, xPokemon Pkmn, xHave Have

Where EType.typeID = Have.typeID

And Have.dex_num = Pkmn.dex_num

Group By EType.name

Order By Average_Attack_Power Desc;
```

"TYPE"	"AVERAGE_ATTACK_POWER"	"NUMBER_POKEMON"
"Dragon"	118	6
"Ice"	115	3
"Psychic"	101	9
"Fighting"	100	6
"Ghost"	100	9
"Dark"	99	7
"Rock"	98	11
"Ground"	95	14
"Poison"	93	9
"Fire"	93	8
"Grass"	92	7
"Steel"	84	8
"Electric"	81	7
"Flying"	81	13
"Normal"	77	10
"Water"	73	15
"Fairy"	64	7
"Bug"	55	2

Cardinality: 18

FinQuery9: Name cities that are connected to a route with a rocky terrain and have trainers that live there which battle with rock, ground, or steel type Pokemon.

```
Create Or Replace View FinQuery9 As

Select Distinct City.name As City

From xCity City, xConnect Connects, xRoute Route, xTrainers
Trainer, xBattleWith Battle, xHave Have, xType EType

Where City.city_code = Connects.city_code

And Connects.route_num = Route.route_num

And Route.terrain = 'Rocky'

And Trainer.city_code = City.city_code

And Trainer.trainerID = Battle.trainerID

And Battle.dex_num = Have.dex_num

And Have.typeID = EType.typeID

And (EType.name = 'Rock'

Or EType.name = 'Ground'

Or EType.name = 'Steel');
```

"CITY"
"Celestic Town"
"Oreburgh City"

Cardinality: 2

FinQuery10: Name Pokemon and the moves they learn such that the Pokemon's attack power is less than the move's power.

```
Create Or Replace View FinQuery10 As

Select Pkmn.name As Pokemon, Moves.name As Move

From xPokemon Pkmn, xMove Moves, xLearn Learn

Where Pkmn.dex_num = Learn.dex_num

And Learn.TM_num = Moves.TM_num

And Pkmn.attack_power < Moves.power;
```

"POKEMON"	"MOVE"
"Grotle"	"Wood Hammer"
"Torterra"	"Wood Hammer"
"Monferno"	"Flamethrower"
"Infernape"	"Fire Blast"
"Piplup"	"Waterfall"
"Shinx"	"Thunderbolt"
"Luxio"	"Thunderbolt"
"Magikarp"	"Tackle"
"Roselia"	"Wood Hammer"
"Zubat"	"Acrobatics"
"Graveler"	"Earthquake"
"Graveler"	"Stone Edge"
"Onix"	"Earthquake"
"Onix"	"Rock Slide"
"Steelix"	"Earthquake"
"Steelix"	"Iron Tail"
"Shieldon"	"Rock Slide"
"Bastiodon"	"Rock Slide"
"Combee"	"Acrobatics"
"Vespiquen"	"Wood Hammer"

Cardinality: 57