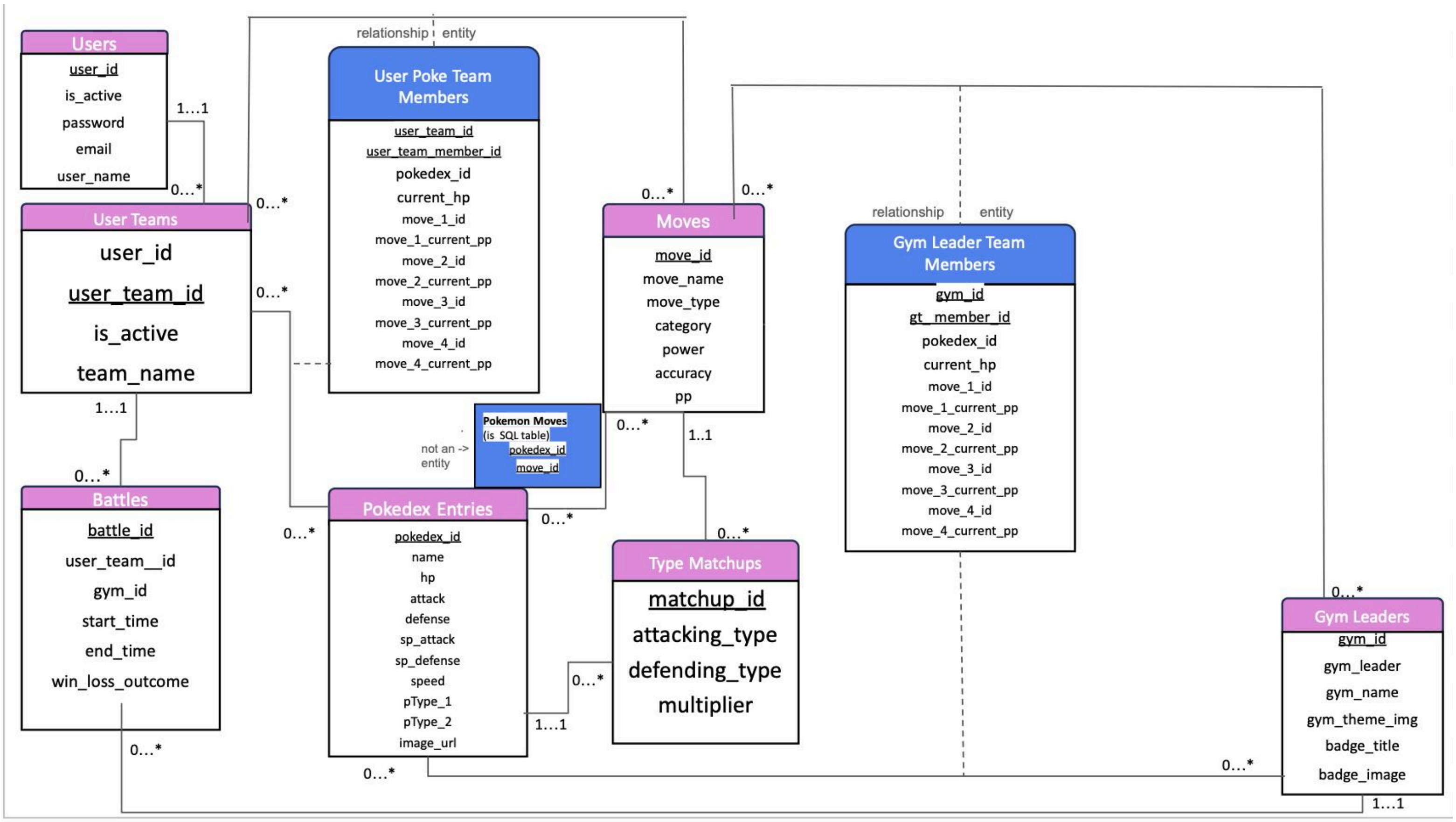
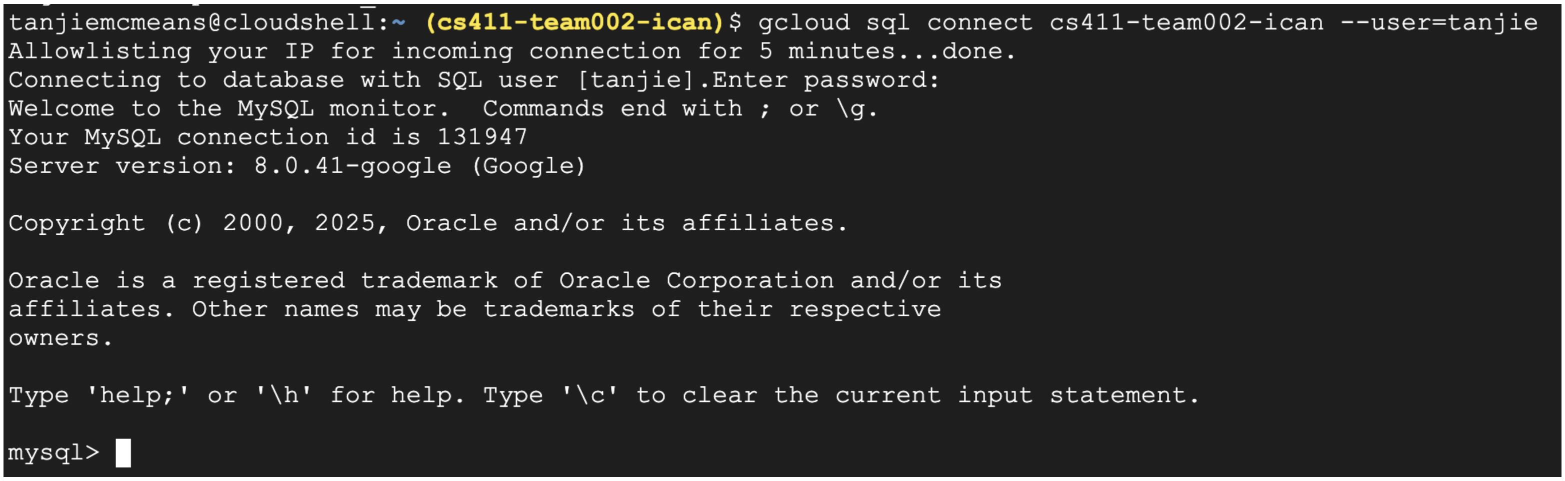
Stage 3: Database Design (PART ONE)

I. ER/UML Diagram (Revised)

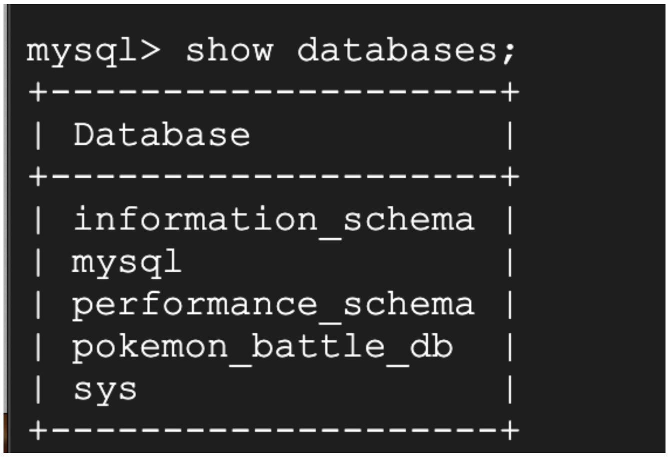


II. Google Cloud Platform

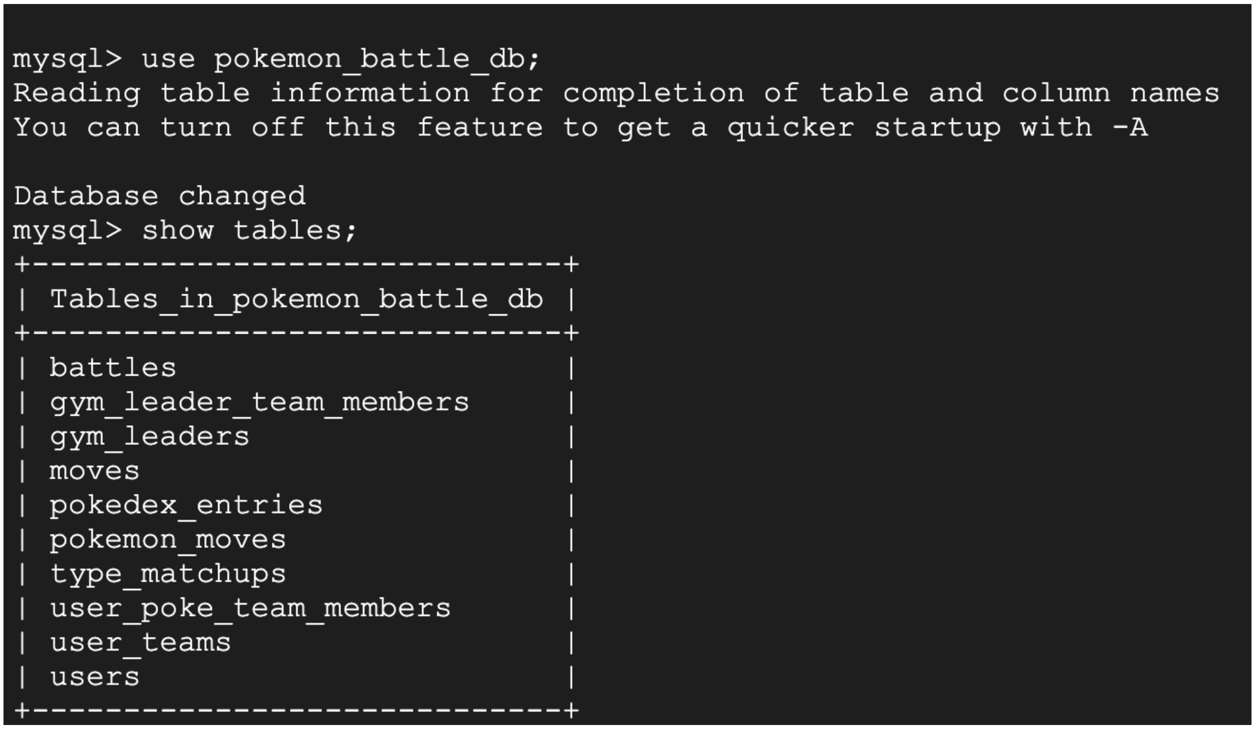
Below is a screenshot of the GCP connection to our database



```
tanjiemcmeans@cloudshell:~ (cs411-team002-ican)$ gcloud sql connect cs411-team002-ican --user=tanjie
Allowlisting your IP for incoming connection for 5 minutes...done.
Connecting to database with SQL user [tanjie].Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 131947
Server version: 8.0.41-google (Google)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| pokemon_battle_db  |
| sys                |
+--------------------+
```



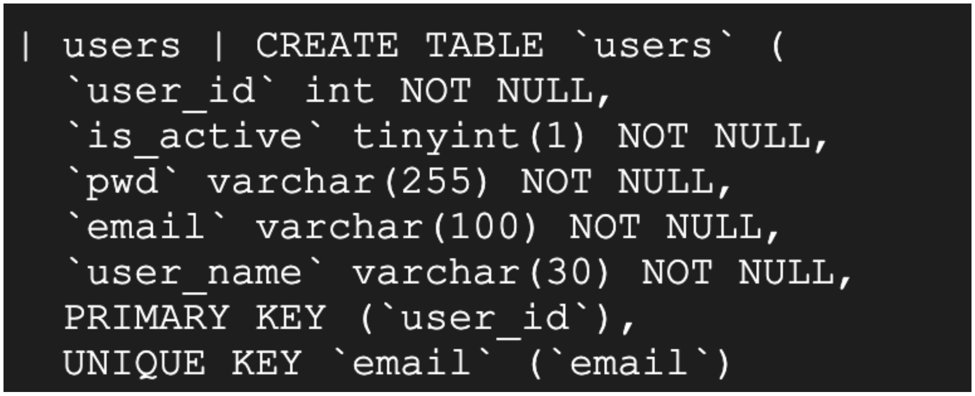
```
mysql> use pokemon_battle_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----------------------------+
| Tables_in_pokemon_battle_db |
+-----------------------------+
| battles                     |
| gym_leader_team_members     |
| gym_leaders                 |
| moves                       |
| pokedex_entries             |
| pokemon_moves               |
| type_matchups               |
| user_poke_team_members      |
| user_teams                  |
| users                       |
+-----------------------------+
```

III. Data Definition Language

Below are the Data Definition Language (DDL) commands used to create the 9 primary tables in our database. The following section details the SQL CREATE TABLE commands that were implemented in our database

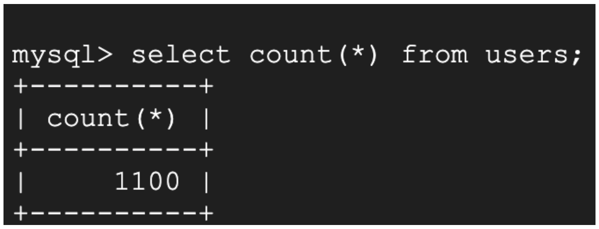
schema. We implemented all 9 main entity tables for our project. The DDL for each table is provided below.

users



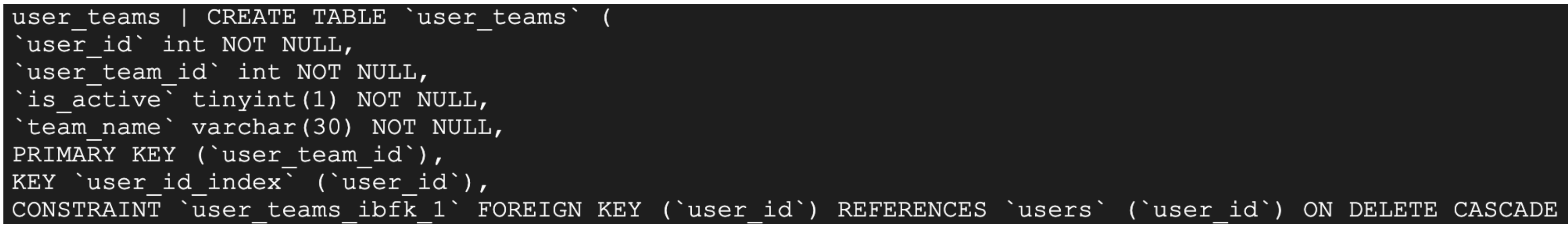
```
| users | CREATE TABLE `users` (
  `user_id` int NOT NULL,
  `is_active` tinyint(1) NOT NULL,
  `pwd` varchar(255) NOT NULL,
  `email` varchar(100) NOT NULL,
  `user_name` varchar(30) NOT NULL,
  PRIMARY KEY (`user_id`),
  UNIQUE KEY `email` (`email`)
```

Data Insertion: To populate this table, we used an LLM to auto-generate a dataset of fake user profiles. The data was then inserted into our database. This table contains 1,100 rows.



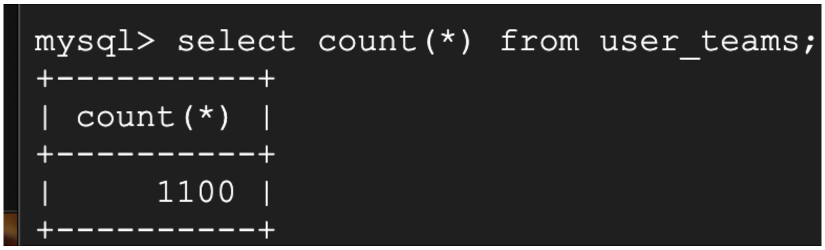
```
mysql> select count(*) from users;
+----------+
| count(*) |
+----------+
|     1100 |
+----------+
```

user_teams



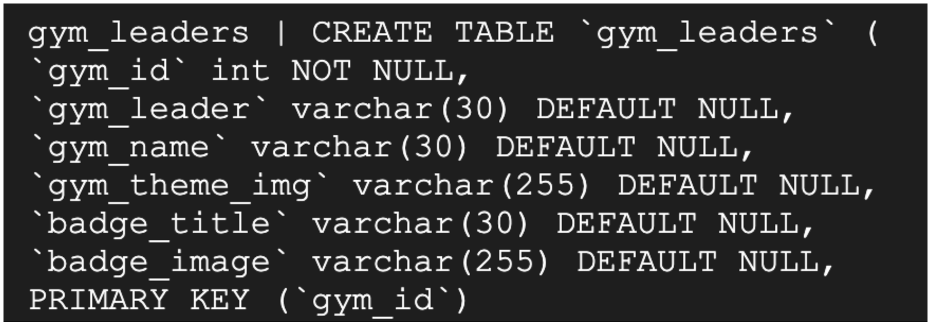
```
user_teams | CREATE TABLE `user_teams` (
`user_id` int NOT NULL,
`user_team_id` int NOT NULL,
`is_active` tinyint(1) NOT NULL,
`team_name` varchar(30) NOT NULL,
PRIMARY KEY (`user_team_id`),
KEY `user_id_index` (`user_id`),
CONSTRAINT `user_teams_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`) ON DELETE CASCADE
```

Data Insertion: To populate this table, we used an LLM to auto-generate a dataset of fake user teams matched to a fake user. The data was then inserted into our database. This table contains 1,100 rows.



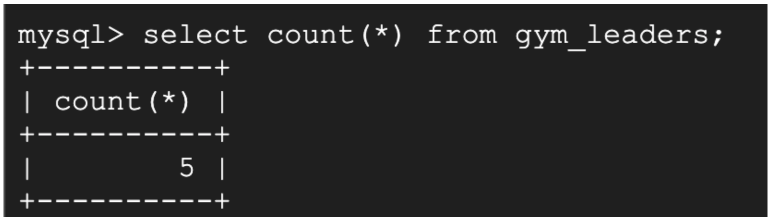
```
mysql> select count(*) from user_teams;
+----------+
| count(*) |
+----------+
|     1100 |
+----------+
```

gym_leaders



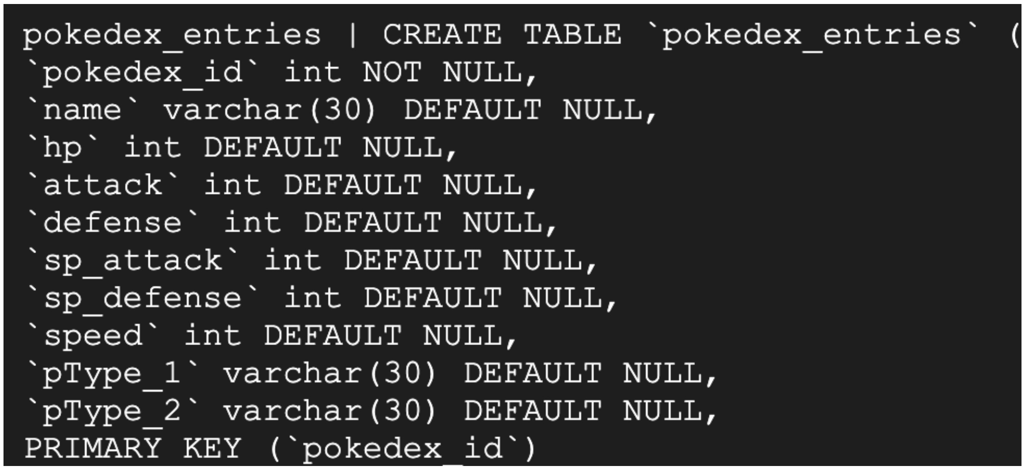
```
gym_leaders | CREATE TABLE `gym_leaders` (
`gym_id` int NOT NULL,
`gym_leader` varchar(30) DEFAULT NULL,
`gym_name` varchar(30) DEFAULT NULL,
`gym_theme_img` varchar(255) DEFAULT NULL,
`badge_title` varchar(30) DEFAULT NULL,
`badge_image` varchar(255) DEFAULT NULL,
PRIMARY KEY (`gym_id`)
```

Data Insertion: To populate this table, we manually created a dataset of gym leaders using Google sheets. The data was then inserted into our database. This table contains 5 rows representing a total of 5 gym leaders.



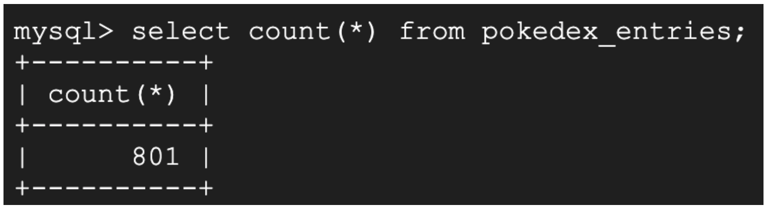
```
mysql> select count(*) from gym_leaders;
+----------+
| count(*) |
+----------+
|        5 |
+----------+
```

pokedex_entries



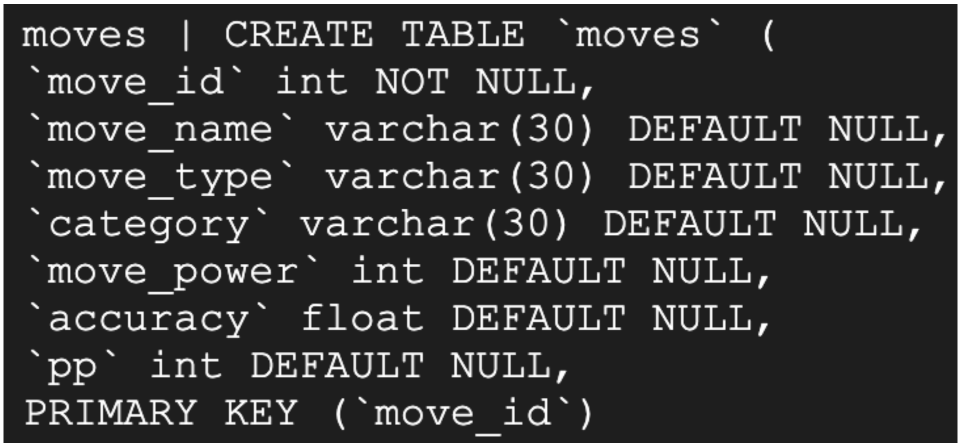
```
pokedex_entries | CREATE TABLE `pokedex_entries` (
`pokedex_id` int NOT NULL,
`name` varchar(30) DEFAULT NULL,
`hp` int DEFAULT NULL,
`attack` int DEFAULT NULL,
`defense` int DEFAULT NULL,
`sp_attack` int DEFAULT NULL,
`sp_defense` int DEFAULT NULL,
`speed` int DEFAULT NULL,
`pType_1` varchar(30) DEFAULT NULL,
`pType_2` varchar(30) DEFAULT NULL,
PRIMARY KEY (`pokedex_id`)
```

Data Insertion: To populate this table, we used raw data from Kaggle (“The Complete Pokemon Dataset”, https://www.kaggle.com/datasets/rounakbanik/pokemon/data, pokemon.csv) and removed attributes that we were not planning to use. The data was then inserted into our database. This table contains 801 rows of real Pokemon entries from the game.



```
mysql> select count(*) from pokedex_entries;
+----------+
| count(*) |
+----------+
|      801 |
+----------+
```

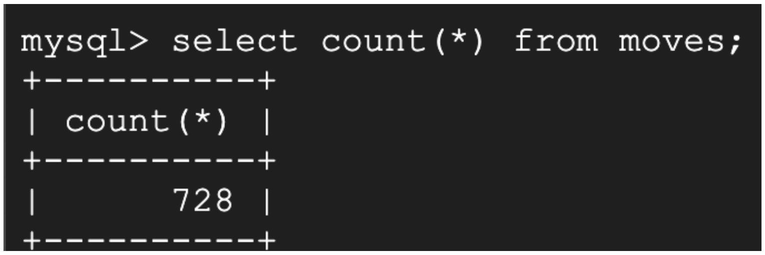
moves



```
moves | CREATE TABLE `moves` (
`move_id` int NOT NULL,
`move_name` varchar(30) DEFAULT NULL,
`move_type` varchar(30) DEFAULT NULL,
`category` varchar(30) DEFAULT NULL,
`move_power` int DEFAULT NULL,
`accuracy` float DEFAULT NULL,
`pp` int DEFAULT NULL,
PRIMARY KEY (`move_id`)
```

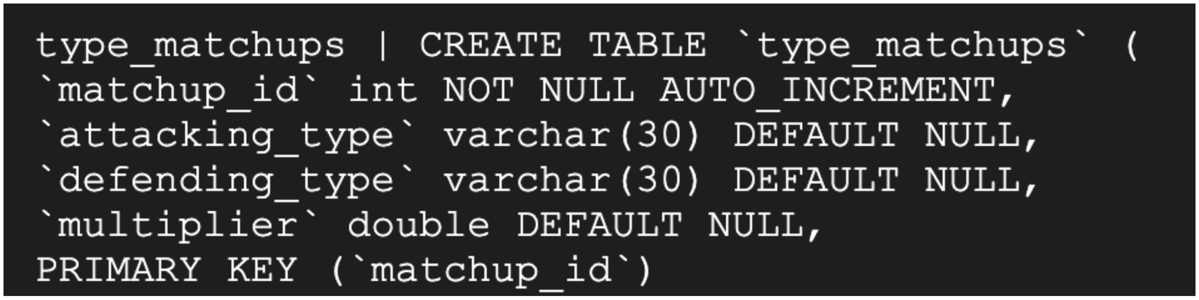
Data Insertion: To populate this table, we used raw data from Kaggle (“Complete Competitive Pokemon Dataset” https://www.kaggle.com/datasets/n2cholas/competitive-pokemon-dataset?select=pokemon-data.csv, move-data.csv) and removed attributes that we were not planning to use. The data was then inserted into our

database. This table contains 728 rows of real moves from the Pokemon game.



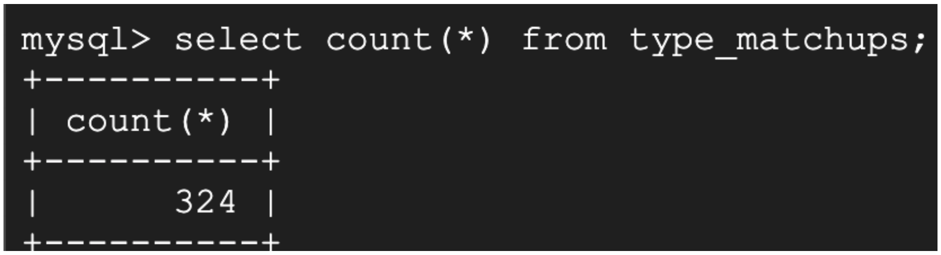
```
mysql> select count(*) from moves;
+----------+
| count(*) |
+----------+
|      728 |
+----------+
```

type_matchups



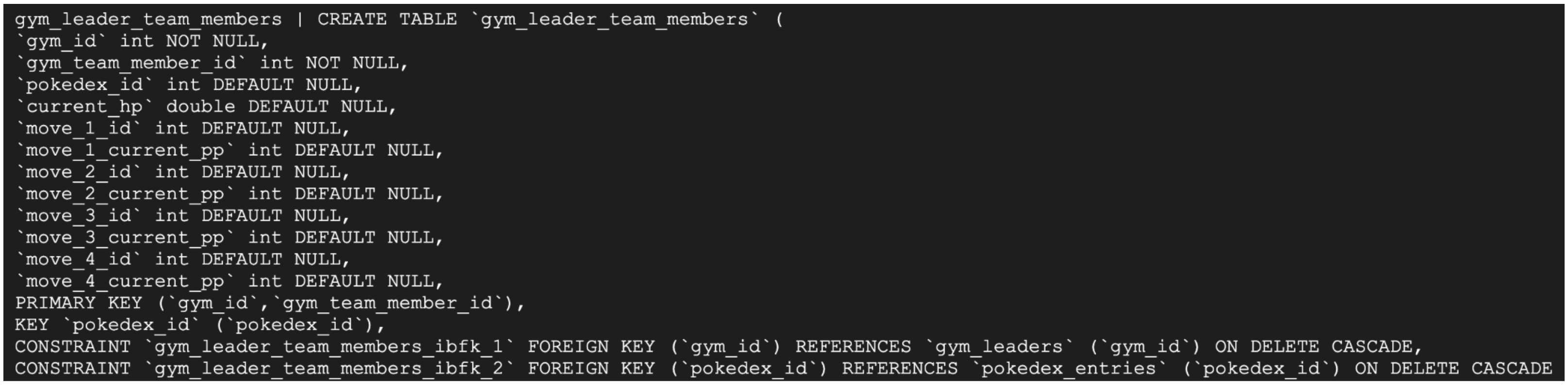
```
type_matchups | CREATE TABLE `type_matchups` (
`matchup_id` int NOT NULL AUTO_INCREMENT,
`attacking_type` varchar(30) DEFAULT NULL,
`defending_type` varchar(30) DEFAULT NULL,
`multiplier` double DEFAULT NULL,
PRIMARY KEY (`matchup_id`)
```

Data Insertion: To populate this table, we used raw data from Kaggle ("Pokemon Type Matchup Data", https://www.kaggle.com/datasets/lunamcbride24/pokemon-type-matchup-data/data, PokeTypeMatchupData.csv) and removed attributes that we were not planning to use. The data was then inserted into our database. This table contains 324 rows of real type matchup multipliers from the Pokemon game.



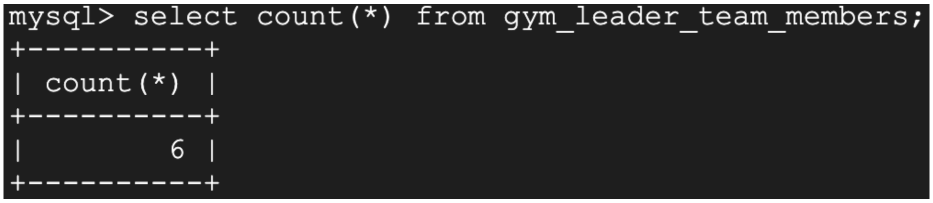
```
mysql> select count(*) from type_matchups;
+----------+
| count(*) |
+----------+
|      324 |
+----------+
```

gym_leader_team_members



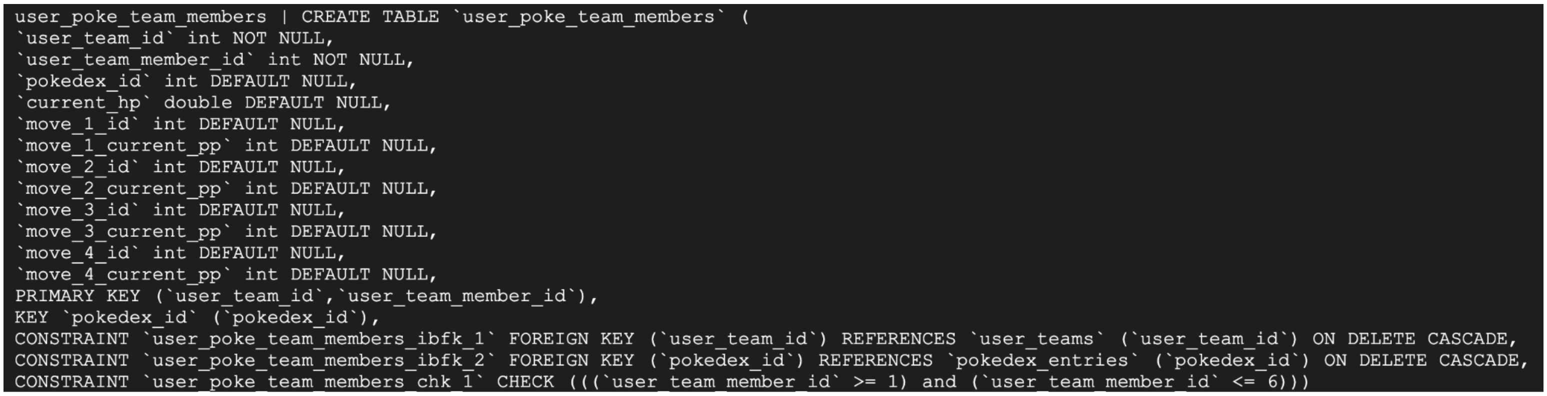
```
gym_leader_team_members | CREATE TABLE `gym_leader_team_members` (
`gym_id` int NOT NULL,
`gym_team_member_id` int NOT NULL,
`pokedex_id` int DEFAULT NULL,
`current_hp` double DEFAULT NULL,
`move_1_id` int DEFAULT NULL,
`move_1_current_pp` int DEFAULT NULL,
`move_2_id` int DEFAULT NULL,
`move_2_current_pp` int DEFAULT NULL,
`move_3_id` int DEFAULT NULL,
`move_3_current_pp` int DEFAULT NULL,
`move_4_id` int DEFAULT NULL,
`move_4_current_pp` int DEFAULT NULL,
PRIMARY KEY (`gym_id`,`gym_team_member_id`),
KEY `pokedex_id` (`pokedex_id`),
CONSTRAINT `gym_leader_team_members_ibfk_1` FOREIGN KEY (`gym_id`) REFERENCES `gym_leaders` (`gym_id`) ON DELETE CASCADE,
CONSTRAINT `gym_leader_team_members_ibfk_2` FOREIGN KEY (`pokedex_id`) REFERENCES `pokedex_entries` (`pokedex_id`) ON DELETE CASCADE
```

Data Insertion: To populate this table, we manually created a dataset of gym team pokemon members using Google sheets. The data was then inserted into our database. This table contains 6 rows representing one gym's pokemon members as a sample set. Other gym's team members are currently not included yet.



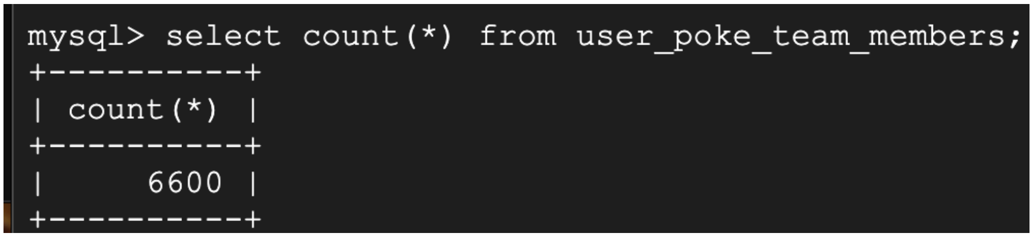
```
mysql> select count(*) from gym_leader_team_members;
+----------+
| count(*) |
+----------+
|        6 |
+----------+
```

user_poke_team_members



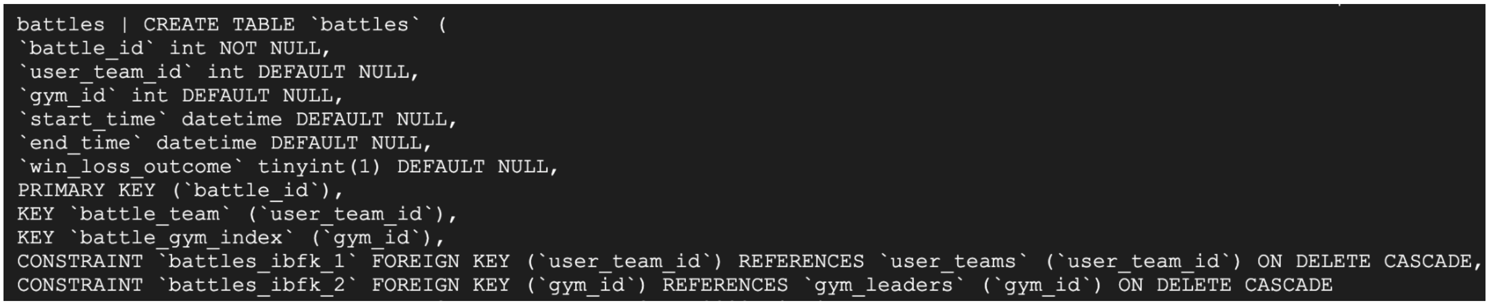
```
user_poke_team_members | CREATE TABLE `user_poke_team_members` (
`user_team_id` int NOT NULL,
`user_team_member_id` int NOT NULL,
`pokedex_id` int DEFAULT NULL,
`current_hp` double DEFAULT NULL,
`move_1_id` int DEFAULT NULL,
`move_1_current_pp` int DEFAULT NULL,
`move_2_id` int DEFAULT NULL,
`move_2_current_pp` int DEFAULT NULL,
`move_3_id` int DEFAULT NULL,
`move_3_current_pp` int DEFAULT NULL,
`move_4_id` int DEFAULT NULL,
`move_4_current_pp` int DEFAULT NULL,
PRIMARY KEY (`user_team_id`,`user_team_member_id`),
KEY `pokedex_id` (`pokedex_id`),
CONSTRAINT `user_poke_team_members_ibfk_1` FOREIGN KEY (`user_team_id`) REFERENCES `user_teams` (`user_team_id`) ON DELETE CASCADE,
CONSTRAINT `user_poke_team_members_ibfk_2` FOREIGN KEY (`pokedex_id`) REFERENCES `pokedex_entries` (`pokedex_id`) ON DELETE CASCADE,
CONSTRAINT `user_poke_team_members_chk_1` CHECK (((`user_team_member_id` >= 1) and (`user_team_member_id` <= 6)))
```

Data Insertion: To populate this table, we used an LLM to auto-generate a dataset of fake user teams member Pokemon matched to a fake user. The data was then inserted into our database. This table contains 6,600 rows, to represent 6 Pokemon per user for 1,100 fake users.



```
mysql> select count(*) from user_poke_team_members;
+----------+
| count(*) |
+----------+
|     6600 |
+----------+
```

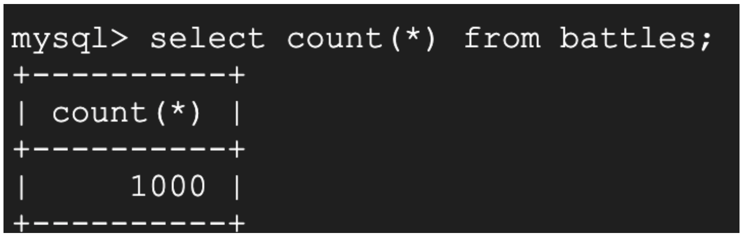
battles



```
battles | CREATE TABLE `battles` (
`battle_id` int NOT NULL,
`user_team_id` int DEFAULT NULL,
`gym_id` int DEFAULT NULL,
`start_time` datetime DEFAULT NULL,
`end_time` datetime DEFAULT NULL,
`win_loss_outcome` tinyint(1) DEFAULT NULL,
PRIMARY KEY (`battle_id`),
KEY `battle_team` (`user_team_id`),
KEY `battle_gym_index` (`gym_id`),
CONSTRAINT `battles_ibfk_1` FOREIGN KEY (`user_team_id`) REFERENCES `user_teams` (`user_team_id`) ON DELETE CASCADE,
CONSTRAINT `battles_ibfk_2` FOREIGN KEY (`gym_id`) REFERENCES `gym_leaders` (`gym_id`) ON DELETE CASCADE
```

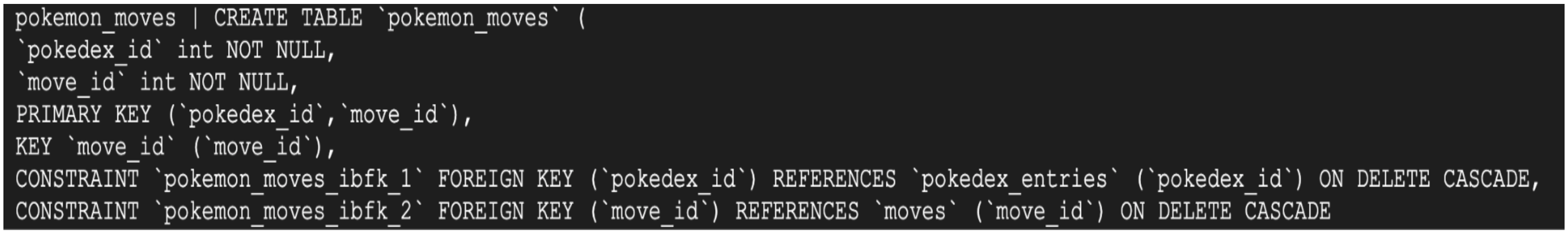
Data Insertion: To populate this table, we used an LLM to auto-generate a dataset of battle statistics matched to a fake user's

team. The data was then inserted into our database. This table contains 1,000 rows.



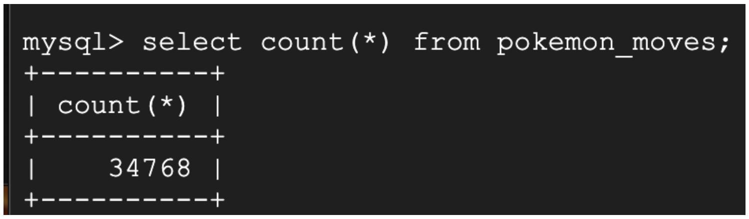
```
mysql> select count(*) from battles;
+----------+
| count(*) |
+----------+
|     1000 |
+----------+
```

pokemon_moves (a SQL relationship table)



```
pokemon_moves | CREATE TABLE `pokemon_moves` (
`pokedex_id` int NOT NULL,
`move_id` int NOT NULL,
PRIMARY KEY (`pokedex_id`,`move_id`),
KEY `move_id` (`move_id`),
CONSTRAINT `pokemon_moves_ibfk_1` FOREIGN KEY (`pokedex_id`) REFERENCES `pokedex_entries` (`pokedex_id`) ON DELETE CASCADE,
CONSTRAINT `pokemon_moves_ibfk_2` FOREIGN KEY (`move_id`) REFERENCES `moves` (`move_id`) ON DELETE CASCADE
```

Data Insertion: This table was created to resolve a many-to-many relationship between the pokemon entries table and the moves table. To populate this table, we gave an LLM raw data from Kaggle ("Complete Competitive Pokemon Dataset" https://www.kaggle.com/datasets/n2cholas/competitive-pokemon-dataset?select=pokemon-data.csv, pokemon-data.csv) and asked it to match Pokemon moves that a Pokemon is capable of learning (which was not yet in our database) to Pokedex Ids. The data was then inserted into our database. This table contains 34,768 rows of moves that can be learned by 801 real Pokemon from the game.



```
mysql> select count(*) from pokemon_moves;
+----------+
| count(*) |
+----------+
|    34768 |
+----------+
```