Table 1: User Login Data

(From Chatgpt)
This table stores user authentication information. For this kind of data, each record should be unique to each user.

Fields:

- `user_id` (Primary Key): A unique identifier for each user, typically an auto-incremented integer.
- `email`: The user's email address. Must be unique.
- `password`: The user's password.

Sample Schema:

sql

Copy code

```sql
CREATE TABLE user_login (
 user_id INT AUTO_INCREMENT PRIMARY KEY,
 email VARCHAR(255) UNIQUE NOT NULL,
 password VARCHAR(255) NOT NULL
);
```

Table 2: Niwa Fire Information

This table stores information about fire risks based on terrain and is predicted for every day of the week. Given that data includes daily predictions for different sites, you should consider both the station name and the date as key parts of your primary key.

Fields:

- `date`: The date of the prediction.
- `station_name`: The name of the station.
- `forest_risk`, `scrub_risk`, `grass_risk`: Risk levels for different types of terrain.
- `ffmc`, `dmc`, `dc`, `isi`, `bui`, `fwi`: Various fire weather indices.
- `temp`, `rh`, `dir`, `wsp`, `rn24`, `gc_percent`: Weather conditions affecting fire risk.

- Composite Primary Key: `date`, `station_name`

Sample Schema:

sql

Copy code

```sql
CREATE TABLE fire_info (
 date DATE,
 station_name VARCHAR(255),
 forest_risk INT,
 scrub_risk INT,
 grass_risk INT,
 ffmc FLOAT,
 dmc FLOAT,
 dc FLOAT,
 isi FLOAT,
 bui FLOAT,
 fwi FLOAT,
 temp FLOAT,
 rh INT,
 dir INT,
 wsp FLOAT,
 rn24 FLOAT,
 gc_percent INT,
 PRIMARY KEY (date, station_name)
);
```

Considerations:

1. Normalization: Ensure that data is normalized to avoid redundancy and maintain data integrity.
2. Foreign Keys: If there are relationships between the tables that need to be enforced, consider adding foreign keys. For instance, if fire information relates to a specific user, a `user_id` foreign key could be added to the `fire_info` table.
3. Indexes: Besides primary keys, consider creating indexes on frequently queried fields to improve performance.

By setting up your tables in this manner, you ensure that your database is well structured for efficient querying and data integrity.