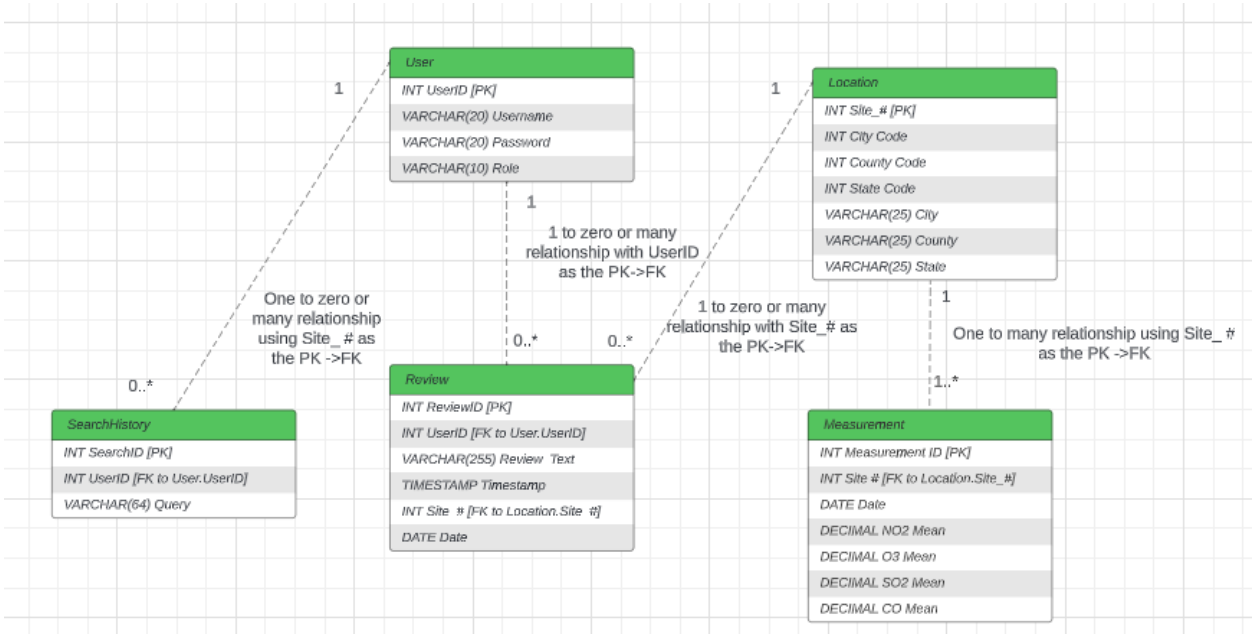
Database Conceptualization for Pollution Visualization Application

ER/UML Diagram



**Written in UML notation that can be referenced here:* https://www.gleek.io/blog/er-symbols-notations .

Entity Relationship and Key Descriptions, Assumptions

In order to achieve the stated functions of our application, we will need to consider Users of the application, Measurements of pollution, the Locations those measurements are taken at, the Reviews of pollution left by Users, and the Reviews left by Users. We make the following assumptions regarding these entities:

1. Each User can write as many reviews or search queries as they please, or none.
2. Each SearchHistory entry and Review will have only one associated User.
3. Each Measurement will have only one associated location.
4. A given Location can have 1 or more associated Measurements, but must have at least one, otherwise it would not be included in the dataset at all.
5. Each Location can have any amount of Review entries.
6. Each Review will have only one associated Location.

User - The User entity will have UserID as a primary key. The Role field will be a choice between admin, editor, and viewer. The User entity has a one to zero or many relationship with SearchHistory, which has UserID as a Foreign Key referencing User.UserID. Likewise, the User entity has a one to zero or many relationship with the Review entity, which also has UserID as a Foreign Key referencing User.UserId.

Location - The Location entity has Site_# as a primary key. Each location has a one to one or many relationship with Measurement, which has Site_# as a Foreign Key referencing Location.Site_#. The Location entity also has a one to zero or many relationship with the Review entity, which has Site_# as a Foreign Key referencing Location.Site_#.

SearchHistory - The SearchHistory entity has SearchID as a Primary Key and a UserID as a Foreign key referencing User.UserID.

Review - The Review entity has ReviewID as a Primary Key, UserId as a Foreign Key referencing User.UserID, and Site_# as a Foreign Key referencing Location.Site_#.

Measurement - The Measurement entity has MeasurementID as a Primary Key and Site_# as a Foreign key referencing Location.Site_#.

Relational Schema

Following is the translated relational schema based on our UML diagram:

Table-User(UserID: INT [PK], Username: VARCHAR(20), Password: VARCHAR(20), Role: VARCHAR(10))

Table-Location(Site_#: INT [PK], City_Code: INT, Country_Code: INT, State_Code: INT, City: VARCHAR(25), County: VARCHAR(25), State: VARCHAR(25))

Table-SearchHistory(SearchID: INT [PK], UserID: INT [FK to User.UserID], Query: VARCHAR(64))

Table-Review(ReviewID: INT [PK], UserID: INT [FK to User.UserID], Review_Text: VARCHAR(255), Timestamp: Timestamp, Site_#: INT [FK to Location.Stie_#], Date: DATE)

Table-Measurement(MeasurementID: INT [PK], Site_#: INT [FK to Location.Site_#], Date: DATE, NO2_Mean: DECIMAL, O3_Mean: DECIMAL, SO2_Mean: DECIMAL, CO_Mean: DECIMAL)

BCNF Normalization

The schema above has the following functional dependencies for each relation:

User:

UserID -> Username, Password, Role

Measurement:

MeasurementID -> Site_#, Date, {NO2, O3, SO2, CO} Mean

Location:

Site_# -> State, City, County

Review:

ReviewID -> UserID, Review_Text, Timestamp, Site_#, Date

SearchHistory:

Search ID -> User ID, Query

Due to the way our entities were initially defined, we ultimately ended up with relations that had fairly straightforward functional dependencies. Each of the above functional dependencies can be proven to have a superkey for their respective relations on the left-hand side by computing the attribute closures of each:

{UserID}$^+$ = Name, Password, Role
User attributes are: Name, Password, Role
So UserID is a superkey of User

{MeasurementID}$^+$ = Site#, Date, {NO2, O3, SO2, CO} mean
Measurement attributes are: Site_#, Date, NO2_Mean, O3_Mean, SO2_Mean, CO_Mean
So MeasurementID is a superkey of Measurement

{Site_#}$^+$ = State, City, County
Location attributes are: State, City, Country
So Site_# is a superkey of Location

{ReviewID}$^+$ = UserID, Review_Text, Timestamp, Site_#, Date
Review attributes are: User_ID, Review_Text, Timestamp, Site_#, Date
So ReviewID is a superkey of Review

SearchID -> UserID, Query, Timestamp
SearchHistory attributes are: UserID, Query, Timestamp
So SearchID is a superkey of SearchHistory

A set of relations can be said to fit Boyce-Codd Normal Form if, for each relation, the existing non-trivial functional dependencies all have a superkey on their left-hand side. Since the above relational schema fits this definition, it is already Boyce-Codd normalized.