Project Title: Model-Managers

Project Members:
- Kurt Greissman
- David “Skip” McGee

a) Overview

The Model-Managers company enables customers to determine which wind forecast models are most accurate for given locations over specified time intervals. There are a variety of wind forecast models (HRRR, ECMWF, GFS, MBLUE, etc.) which often have different wind forecasts for a given location. Users of these forecast models usually cannot determine what model is the most accurate for their specific location. In other words, they would need to compare the model forecasts to actual data from weather/wind collection devices at specific locations to determine which model is typically more accurate for those locations.

The Model-Managers company solves this problem by storing hourly wind forecasts for a standardized forecast period of 7 days in a database, and then allowing users to interact with a web interface to compare the forecast data against the actual measured data at a location for different time intervals. The 3 criteria used to compare forecast predictions to measured wind are average wind speed, wind direction, and wind gust speed. The Model-Managers company uses the free OpenMeteo API (https://open-meteo.com) to obtain wind forecast data, and a Holfuy (https://api.holfuy.com/) device API to obtain the measured wind forecast data.

The minimal viable product of the Model-Managers company uses 2 weather forecast models (HRRR and ECMWF) and 1 Holfuy device, located at La Bajada Ridge, New Mexico. Model-Managers uses a GitHub repository at https://github.com/skipmcgee/wind-forecast to manage their codebase.

b) Entities:

1. Models:
   These are the individual weather models that the Model-Managers product supports.
   a. modelID: int, auto_increment, unique, not NULL, PK
   b. modelName: varchar(100), not NULL
   Relationships: there is a one to many relationship between models and forecasts. For every model there are many forecasts.

- Locations:
  The locations where customers desire to compare forecasts to measured values.
  a. locationID: int, auto_increment, unique, not NULL, PK
  b. locationName: varchar(300), unique, not NULL
  c. locationLatitude: float, not NULL
  d. locationLongitude: float, not NULL

e. locationAltitude: int, not NULL

Relationships: A one to one relationship exists between locations and sensors. There is only one sensor at a specific location. A one to many relationship exists between locations and forecasts. There are typically greater than one forecast at a single location.

- Sensors:

  The sensors that can measure the weather data at a specific location.

  a. sensorID: int, auto_increment, unique, not NULL, PK
  b. sensorName: varchar(300), not NULL
  c. sensorLatitude: float, not NULL, FK -> locationLatitude
  d. sensorLongitude: float, not NULL, FK, -> locationLongitude
  e. sensorAPIKey: varchar(300), not NULL
  f. sensorNumber: int, not NULL
  g. sensorAltitude: int, not NULL, FK, -> locationAltitude

  Relationships: there is a one to one relationship between sensors and locations.

1. Forecasts:

   Hourly forecast information that is common across all weather models.

   a. forecastID: int, auto_increment, unique, not NULL,
   b. forecastDateTime: date, not NULL
   c. forecastTemperature_2m: float, not NULL
   d. forecastPrecipitation: float, not NULL
   e. forecastWeatherCode: varchar(100), not NULL
   f. forecastPressureMSL: int, not NULL
   g. forecastWindSpeed_10m: float, not NULL
   h. forecastWindDirection_10m: float, not NULL
   i. forecastCape: float, not NULL
   j. forecastLatitude: float, not NULL, FK -> locationLatitude
   k. forecastLongitude: float, not NULL, FK, -> locationLongitude
   l. forecastModel: int, not NULL, FK -> modelID

   Relationships: there is a many to one relationship between forecasts and locations. There are many forecasts at a single location.

2. Readings:

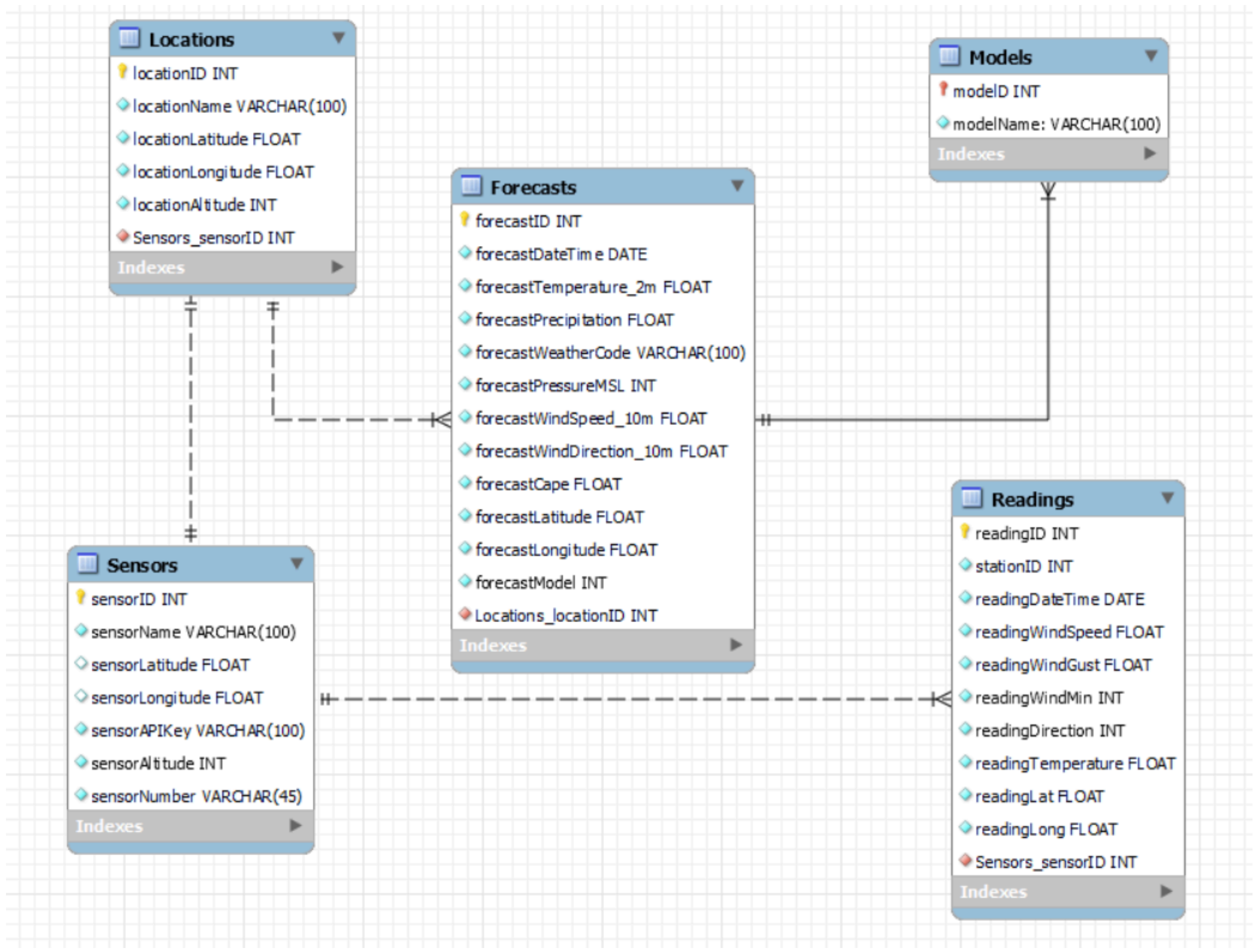
   The actual measured sensor readings at a specific location.

   a. readingID: int, auto_increment, unique, not NULL, PK
   b. stationId: int, not NULL, FK -> sensorID
   c. readingDateTime: date, not NULL
   d. readingWindSpeed: float, not NULL
   e. readingWindGust: float, not NULL
   f. readingWindMin: int, not NULL
   g. readingDirection: int, not NULL
   h. readingTemperature: float, not NULL
   i. readingLat: float, not NULL, FK -> sensorLatitude

j. readingLong: float, not NULL, FK -> sensorLongitude

Relationships: there is a many to one relationship between readings and sensors. There are many readings at a single sensor.

c) Entity Relationship Diagram:



Locations
locationID INT
locationName VARCHAR(100)
locationLatitude FLOAT
locationLongitude FLOAT
locationAltitude INT
Sensors_sensorID INT
Indexes
Models
modelD INT
modelName: VARCHAR(100)
Indexes
Forecasts
forecastID INT
forecastDateTime DATE
forecastTemperature_2m FLOAT
forecastPrecipitation FLOAT
forecastWeatherCode VARCHAR(100)
forecastPressureMSL INT
forecastWindSpeed_10m FLOAT
forecastWindDirection_10m FLOAT
forecastCape FLOAT
forecastLatitude FLOAT
forecastLongitude FLOAT
forecastModel INT
Locations_locationID INT
Indexes
Readings
readingID INT
stationID INT
readingDateTime DATE
readingWindSpeed FLOAT
readingWindGust FLOAT
readingWindMin INT
readingDirection INT
readingTemperature FLOAT
readingLat FLOAT
readingLong FLOAT
Sensors_sensorID INT
Indexes
Sensors
sensorID INT
sensorName VARCHAR(100)
sensorLatitude FLOAT
sensorLongitude FLOAT
sensorAPIKey VARCHAR(100)
sensorAltitude INT
sensorNumber VARCHAR(45)
Indexes