Module 7 Note

Bobby is an up-and-coming HR analyst whose task is to perform employee research.

**Looking Ahead**

In this module, you'll learn about data modeling, engineering, and analysis. Applying your knowledge of DataFrames and tabular data, you'll create entity relationship diagrams (ERDs), import data into a database, troubleshoot common errors, and create queries that use data to answer questions. Databases are used everywhere—small and large businesses, and even individuals working on personal projects—and SQL is one of the most widely used query languages. Its ability to organize and query data, especially on a large scale, makes SQL knowledge a highly sought after skill in the workforce.

**What You Will Learn**

By the end of this module, you will be able to:

* Design an ERD that will apply to the data.
* Create and use a SQL database.
* Import and export large CSV datasets into pgAdmin.
* Practice using different joins to create new tables in pgAdmin.
* Write basic- to intermediate-level SQL statements.

Bobby is an up-and-coming HR analyst whose task is to perform employee research. Specifically, he needs to find answers to the following questions: Who will be retiring in the next few years? And how many positions will Pewlett Hackard need to fill? This analysis will help future-proof Pewlett Hackard by generating a list of all employees eligible for the retirement package. The employee data Bobby needs is only available in the form of six CSV files because Pewlett Hackard has been mainly using Excel and VBA to work with their data.

But now, they have decided to update their methods to use SQL, a definite upgrade considering the amount of data. Your task is to help Bobby build an employee database with SQL by applying your data modeling, engineering, and analysis skills.

To help Bobby prepare for his analysis, he'll need to download his tools: PostgreSQL and pgAdmin. He'll use Postgres to create a database, and pgAdmin to work with the data he'll be importing. These tools are packaged together in a single download, so let's get started with the setup!

## Database Keys

Database keys identify records from tables and establish relationships between tables. There are numerous types of keys. For our purposes, we will focus on primary keys and foreign keys.

#### Primary Keys

The departments.csv file has a dept\_no column with unique identifiers for each row (one department number per department). For example, d001 will always reference the Marketing department, across other worksheets. This unique identifier is known as a **primary key**.

Primary keys are an important part of database design. When a database is being created, each table added must include a primary key in the architecture. Primary keys serve as a link between these tables.

**Entity Relationship Diagrams (ERDs)**

An **entity relationship diagram (ERD)** is a type of flowchart that highlights different tables and their relationships to each other. The ERD does not include any actual data, but it does capture the following pertinent information from each CSV file:

* Primary keys
* Foreign keys
* Data types for each column

The ERD also shows the flow of information from one table to another, as captured in the image below:

In addition to creating new databases, ERDs are used to document existing databases. The visual representation of the tables gives a deeper understanding of the data and the database as a whole.

When creating a diagram, we need to fully understand all of the data being inserted. Datab

Another step is to create a map of the database. This map will show us each table in the database and the flow of data from one table to another. That's right, we're going to make a fancy flowchart. This provides us with an easy reference to the data without actually accessing it. This is called "modeling the data," and we can get started by creating a diagram with an online tool, instead of trying to make one from scratch. Our flow chart will help us navigate through the relationships more easily than if we had all six CSV files open side-by-side.  
  
Using an online tool called Quick Database Diagrams ("Quick DBD" for short), we'll help Bobby start by familiarizing ourselves with the webpage, then create a conceptual ERD.

here are several ways to refer to the map we're about to create. It's also called a flowchart, an entity relationship diagram, and a schema. We'll be using all of these terms in this module, though "ERD" is the most specific.  
  
There are three forms of ERDs: conceptual, logical, and physical. We'll start by helping Bobby with the most basic of the three, the conceptual diagram. As we add more information to our tables, such as data types and keys, we'll advance through the more complex diagrams.

## Conceptual Diagrams

A **conceptual diagram** is an ERD in its simplest form. To create one, we only need two things: a table name and column headers.

weather\_api\_key="dd4489015a0d53691a4504c482fe26b5"

# Geoapify API Key

geoapify\_key = "114208dbe31c49a6aee5748cfdc3a2e5"

### Basic Project Plan

Here's an outline of your project plan:

* **Task:** Collect and analyze weather data across cities worldwide.
* **Purpose:** PlanMyTrip will use the data to recommend ideal hotels based on clients' weather preferences.
* **Method:** Create a Pandas DataFrame with 500 or more of the world's unique cities and their weather data in real time. This process will entail collecting, analyzing, and visualizing the data.

Your analysis of the data will be split into three main parts, or stages.

1. **Collect the Data**
   * Use the NumPy module to generate more than 1,500 random latitudes and longitudes.
   * Use the citipy module to list the nearest city to the latitudes and longitudes.
   * Use the OpenWeatherMap API to request the current weather data from each unique city in your list.
   * Parse the JSON data from the API request.
   * Collect the following data from the JSON file and add it to a DataFrame:
     + City, country, and date
     + Latitude and longitude
     + Maximum temperature
     + Humidity
     + Cloudiness
     + Wind speed
2. **Exploratory Analysis with Visualization**
   * Create scatter plots of the weather data for the following comparisons:
     + Latitude versus temperature
     + Latitude versus humidity
     + Latitude versus cloudiness
     + Latitude versus wind speed
   * Determine the correlations for the following weather data:
     + Latitude and temperature
     + Latitude and humidity
     + Latitude and cloudiness
     + Latitude and wind speed
   * Create a series of maps using GeoViews and the Geoapify API that showcases the following:
     + Latitude and temperature
     + Latitude and humidity
     + Latitude and cloudiness
     + Latitude and wind speed
3. **Visualize Travel Data**

Create a map can display information on specific cities based on a customer's travel preferences. Complete these steps:

* + Filter the Pandas DataFrame based on user inputs for a minimum and maximum temperature.
  + Create a map for the new DataFrame.
  + Find a hotel from the cities' coordinates using the Geoapify API.
  + Store the name of the first hotel in the DataFrame.
  + Add additional information to each point in the map that display information about the city, current maximum temperature, and a hotel in the city.

ownload his tools: PostgreSQL and pgAdmin.

**PostgreSQL**

PostgreSQL, typically referred to as just "Postgres," is a **relational database system**. This type of database consists of tables and their predefined relationships.

Think of it like this: Each CSV file's data will be loaded into a table. If there are six CSV files, then there will also be six tables in Postgres. "Relationships" are how each table relates to another. We'll create tables and define relationships as we progress through the module, so don't worry if this seems confusing right now—we'll get lots of practice.

Another aspect of Postgres is that it will create a local server on your computer, which is where the databases we create will be stored. Then the databases will store the tables and the data. It's a rather intricate filing system.

**NOTE**

For more about PostgreSQL, see the [PostgreSQL documentationLinks to an external site.](https://www.postgresql.org/docs/manuals/) and the [PostgreSQL tutorialLinks to an external site.](https://www.tutorialspoint.com/postgresql/).

**pgAdmin**

pgAdmin is the window into our database: it's where queries are written and executed and where results are viewed. While Postgres holds the files, pgAdmin provides the access. All SQL actions take place within these two programs, so let's install them.