**Relational Databases:**

- Store data in rows and columns (like an Excel sheet) and can quickly and efficiently retrieve data from those tables (especially when there are multiple tables and the relationships between those tables are involved in the query)  
- *Database* – Contains many tables  
- *Relation/Table* – Contains tuples and attributes  
- *Tuple (Row)* – A set of fields that generally represent an object (like a person or artist name)  
- *Attribute (Column/Field)* – One of many possible elements of data corresponding to the object represented by the row (phone number, email address, album name, song name)  
- ***Database Schema*** – The outline for the database that determines rules for each filed (number of characters allowed, integer vs float)

**Structured Query Language (SQL):**

- The language used to issue commands to the database (Create table, retrieve table data, insert data, delete data)  
- Best practice is not to talk directly to database but to talk to an intermediary database application instead  
- Data needs to be cleaned first before it can be manipulated using SQL  
- Four basic functions (CRUD):   
1. **C**reate   
2. **R**ead   
3. **U**pdate   
4. **D**elete

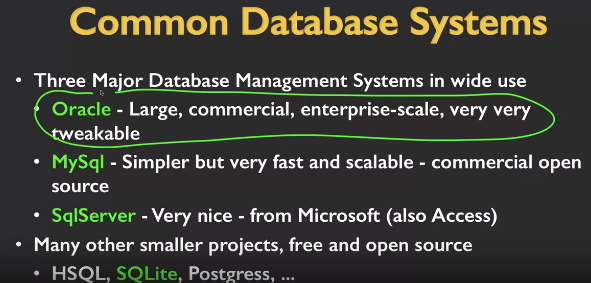
**Database Roles:**

- **Application Developer** – Builds the logic for the application, the UI, and monitors it for problems  
- **Database Administrator** – Monitors and adjusts the database as the program runs in production   
A diagram of software development

Description automatically generatedA diagram of a data analysis structure

Description automatically generated

Database Schema:

- The structure/format of the database. Provides rules and guidance on what can and cannot be contained in the database and how it is supposed to be stored  


- **Oracle** is for enterprise databases  
- **MySQL** is typically used for websites  
-**SQLite** is an embedded database that is built into software and not a standalone database that software must access  
- Already built into Python, just need to import it

**Single Table CRUD:**

- Need to be explicit about how large your data is going to be to help database be fast (variable characters up to 128 of them)  
A screenshot of a computer screen

Description automatically generated

**SQL Insert** – Inserts a new row into a table   
INSERT INTO Users (name, email) VALUES ('Kristin', 'kf@umich.edu')

**SQL Delete** – Deletes a row in a table  
DELETE FROM Users WHERE email='fred@umich.edu'

**SQL Update** – Allows for the updating of existing data in a row  
UPDATE Users SET name = 'Charles' WHERE email = 'csev@umich.edu'

**Retrieving Records** – Retrieves a group of records, either all of them or a subset if you use the WHERE clause  
SELECT \* FROM Users  
SELECT \*FROM Users WHERE email = 'csev@umich.edu'

**Ordering Records** – Used to get results sorted in ascending or descending order  
SELECT \* FROM Users ORDER BY email  
SELECT \* FROM Users ORDER BY name

**Databases:**

- Relational databases are stored as a dictionary with a *key:value* mapping  
- Data is indexed to allow for the quick retrieval of it even amongst large amounts of data  
- Simple operations on a database will be done through the database manager but ore complex/tedious ones will be done using a Python script  
- When we create a database we must label in advance the column names and the type of data that will be stored in it  
A screen shot of a computer program

Description automatically generated  
*connect* – Makes a connection to the database (stored in file ‘music.sqlite’)  
*cursor –* Functions similar to *open()* and allows us to perform operations on the data stored in the database  
A diagram of a diagram of a group

Description automatically generated

A computer code with text

Description automatically generated  
A white background with blue text

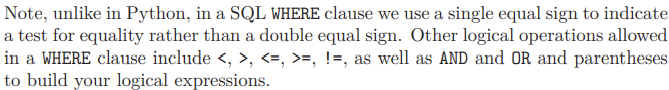
Description automatically generated

A screenshot of a game

Description automatically generated

- *Cursor* does not read all of the data from a database when we execute SELECT so you need to create a for loop for it

**SQL Commands:**

  
  
  
  
  
  
A close up of black text

Description automatically generated  


- *Data Modeling* – The act of deciding how to break up your application data into multiple tables and establishing relationships between those tables  
- *Data Model* – The design document that shows the tables and their relationships  
­- *Database Normalization* – Never put the same string data in a column more than once, better to create a numeric key for the data and reference it from another table  
- Use a primary key to link one table to another, key is a number assigned into table using *id* column   
A close-up of a computer code

Description automatically generated  
- In the Artist Table, 42 is a *primary key* but in subsequent tables that reference it is known as a *foreign key* because it references data in a different table  
A text on a white background

Description automatically generated  
- When we want to retrieve data we use the JOIN keyword to properly extract it during a SELECT statement  
A close up of a text

Description automatically generated  
