Q1 In computer science, a database connection is a feature that enables client software to communicate with database server software, whether or not they are running on the same machine. Sending commands and receiving responses, which are typically in the form of result sets, both require connections. A fundamental idea in data-centric programming is connections. Connection pooling was developed to enhance performance because some DBMS engines take a long time to connect. Without a "open and available" connection to a database, no command may be executed against it. By providing a connection string—a means of addressing a specific database or server and instance as well as user authentication credentials—to an underlying driver or provider, connections are created. A connection can be opened, closed, and its properties (such as the command time-out length or transaction, if one exists) can be set once it has been established. The data access interface and data provider being utilised determine the key/value pairs that make up the Connection String. On each connection, many databases (including PostgreSQL) only permit one action to be carried out at a time. When a database receives a request for data (a SQL Select statement) and returns a result set, the connection is open but not yet ready for subsequent actions since the client is still processing the result set. Other databases do not place this restriction, such as SQL Server 2005 (and later). Databases that allow several operations per connection, however, typically have far higher overhead than those that only allow one operation job at a time.

Q2 Connection pooling is a method created to address this issue. The apps that require database access can share a pool of database connections that have been created. An often-used wrapper for the real database connection is the connection object acquired from the connection pool. The wrapper conceals the specifics of the pool from the application while being aware of its relationship with the pool. For instance, the wrapper object may include a "close" method that may be used in the same way as the database connection's "close" function. In contrast to the method on the database connection, the method on the wrapper may instead return the database connection to the pool rather than actually closing it. When the application calls the methods on the wrapper object, it is not necessary for it to be aware of connection pooling. Instead of keeping a connection open for the duration of the application, this strategy promotes the practise of opening a connection in an application only when necessary and closing it as soon as the work is finished. In this way, many requests can be fulfilled by a small number of connections. This practise is known as multiplexing. On the other hand, a persistent connection is frequently utilised in client/server architectures so that the server status may be controlled. This "state" includes connection-specific functional settings, temporary items, server-side cursors, and more. When the connection pool overflows, the programme fails. When an application requests a connection, this can happen if every connection in the pool is already in use. For instance, if too many users try to access the website at once or if one or more actions are prohibited or just ineffective, the application may consume a connection for too long.