1. SQLite can appropriately be used in these situations:

* Embedded devices and the internet of things  
  Application file format
  + Because an SQLite database requires no administration, it works well in devices that must operate without expert human support. SQLite is a good fit for use in cellphones, set-top boxes, televisions, game consoles, cameras, watches, kitchen appliances, thermostats, automobiles, machine tools, airplanes, remote sensors, drones, medical devices, and robots: the "internet of things".
* Websites
  + SQLite works great as the database engine for most low to medium traffic websites (which is to say, most websites). The amount of web traffic that SQLite can handle depends on how heavily the website uses its database. Generally speaking, any site that gets fewer than 100K hits/day should work fine with SQLite. The 100K hits/day figure is a conservative estimate, not a hard upper bound. SQLite has been demonstrated to work with 10 times that amount of traffic.
* Data analysis
  + People who understand SQL can employ the sqlite3 command-line shell (or various third-party SQLite access programs) to analyze large datasets. Raw data can be imported from CSV files, then that data can be sliced and diced to generate a myriad of summary reports. More complex analysis can be done using simple scripts written in Tcl or Python (both of which come with SQLite built-in) or in R or other languages using readily available adaptors. Possible uses include website log analysis, sports statistics analysis, compilation of programming metrics, and analysis of experimental results. Many bioinformatics researchers use SQLite in this way.
* Experimental SQL language extensions
  + The simple, modular design of SQLite makes it a good platform for prototyping new, experimental database language features or ideas.

2. Other characteristics that differentiate SQLite from other SQL databases are:

* It's serverless
  + Most SQL database engines are implemented as a separate server process. Programs that want to access the database communicate with the server using some kind of interprocess communication (typically TCP/IP) to send requests to the server and to receive back results. SQLite does not work this way. With SQLite, the process that wants to access the database reads and writes directly from the database files on disk. There is no intermediary server process.
* It's compact
  + When optimized for size, the whole SQLite library with everything enabled is [less than 500KiB in size](https://www.sqlite.org/footprint.html) (as measured on an ix86 using the "size" utility from the GNU compiler suite.) Unneeded features can be disabled at compile-time to further reduce the size of the library to under 300KiB if desired.
  + Most other SQL database engines are much larger than this. IBM boasts that its recently released CloudScape database engine is "only" a 2MiB jar file - an order of magnitude larger than SQLite even after it is compressed! Firebird boasts that its client-side library is only 350KiB. That's as big as SQLite and does not even contain the database engine. The Berkeley DB library from Oracle is 450KiB and it omits SQL support, providing the programmer with only simple key/value pairs.

3.

1. The copyright status of SQLite is that it's been dedicated to public domain.
2. Yes, you can use SQLite for a project, so long as you have legal proof if some organizations want it.
3. As long as you get a Warranty of Title of SQLite, you can use it in a product, that way it is shown who the true authors of SQLite are.

4. In order:

* Import the module
* Make a connection to the database
* Create a cursor Object
* Use the execute method of the cursor object to run SQL commands