**Understanding Database Options**

**Project Statement**

Your boss calls you into a meeting. She tells you about a new project that is about to begin. It needs a database back-end, and she would like you to do some research to figure out which database is the best solution. She says that the first phase of the project is just to test feasibility, so they need a very simple, fast solution **(performance)** that doesn’t need a lot of setup **(ease of setup)** or special hardware/software to get going **(low software/hardware requirements)**. It needs to be cross platform (**cross-platform compatibility)** but doesn’t require any special network access (networkable and easy firewall). She says for the feasibility phase, only one user will be accessing the database from their computer **(concurrency)**. She says it needs to be easy to backup to a usb stick so it can be copied from one computer to another **(ease of use / maintenance)**. Cost is of utmost importance during this first phase of this project **(low cost)**. It’s ok if the database backend is replaced with something else in later phases, once all the requirements have been worked out **(adaptability)**. Oh, and one last thing, it needs to be compatible with Python since that’s what will be used throughout this project **(Python compatibility)**.

Given the above information, make a list of features (criteria) that are important to your boss for this phase of the project. Rank how important each criteria is, where 1 is not very important, and 5 is very important. Show your criteria and the rank you assigned each one.

Evaluate the following databases against your criteria:

* *MSSQL*
* *Oracle*
* *SQLite*
* *MySQL (or MariaDB)*
* *PostgreSQL*
* *Microsoft Access*
* *LibreOffice Base*

Rate each database against the criteria, where 0 means it doesn’t meet the criteria at all, and 5 meets the criteria extremely well. Show your rating for each database, against each criteria.

Multiply your numbers and add them up, just like the class example. Find which database should work the best for this phase of the project and state your findings.

**Project Analysis**

**\*Refer to table below to see further details regarding calculations and in bold above (project statement) see criteria defined.**

****

**Findings**

Based on results obtained MySQL will work the best for this phase of the project, considering it did better than the other options in all the criteria evaluated as a whole. Besides, as it was stated in the that this is for a feasibility phase (let’s say for example it could be intended to be use for a proof of concept of a solution) having a specific scope, MySQL is a good option to begin with, considering advantages like it is:

* Open-source (companies/individuals can use the community version - no payments or license fees required).
* Ease of use / maintenance (mild learning curve).
* Python compatibility, straightforward setup, and interoperability.
* Performance, overall speed, and scalability in the long run could be a plus for expected future growth. Additionally, as it’s mentioned that “It’s ok if the database backend is replaced with something else in later phases”, the company can continue with the community version if their business requirements increased and in case this is not enough, they can even migrate to the commercial one at some point without any complications.
* Ease for networking or firewall settings, as straightforward as adding/modifying lines in MySQL config file is possible to allow remote connections, among other things. There is plenty documentation for it, so with basic networking knowledge is possible to work this out.