Cloud Databases

1. Basic principles of NoSQL databases:

Relational databases typically suffer from performance problems in data-intensive applications such as indexing large volumes of documents or high-load webpages. Relational databases are efficient only if they are optimized for frequent small transactions or large transactions with little to no writes. Relational databases handle simultaneous high data requests and frequent data changes poorly. NoSQL databases on the other hand can handle many read/write operations.  
Relational databases are using different tables to store data and joins to retrieve data. NoSQL databases aren’t using tables and try to avoid joins. Motivations for this include the simplicity of design or the simpler “horizontal” scaling to clusters of machines. The data structures (key-value pair, graph, …) used in NoSQL differ from the ones used by relational databases, making some operations faster. The suitability of a NoSQL database depends on the specific problem to be solved. Some NoSQL databases compromise consistency in favor of availability and speed. Database changes are propagated too all nodes eventually (milliseconds), sometimes resulting to inaccurate or wrong, so-called stale reads. Barriers for the further spread of NoSQL stores are the use of low-level query languages instead of SQL, the inability to perform joins across tables, and enormous previous investments in existing relational databases.

2. Benefits of NoSQL databases over an SQL database:

NoSQL databases are more powerful and more scalable in comparison to relational ones.  
NoSQL databases are horizontally scalable which means that they can handle increased traffic simply by adding more servers to the database. NoSQL databases have the ability to become larger and much more powerful, making them the preferred choice for large or constantly evolving data sets.  
Other benefits are:  
+ store unstructured, semi-structured, or structured data  
+ enable easy updates to schemas and fields (NoSQL databases often allow developers to directly change the structure of the data)  
+ developer-friendly (move to NoSQL databases has been driven primarily by developers who find it easier to create different types of applications than with relational databases)  
+ take full advantage of the cloud to deliver zero downtime (many NoSQL databases can be upgraded and allow the structure of the database to change with zero downtime)

3. Three examples of data where NoSQL is the most suitable storage alternative:

- NoSQL databases are the best for modern applications where data models evolve, and scalability is essential  
- NoSQL databases are often used in agile projects because schema changes – which are common – don’t require a lot of overhead  
- data with rapid data growth and scalability  
- large amounts of unstructured or semi-structured data requires NoSQL as it doesn’t fit into a relational model  
- big data that is/should be used for data mining, as NoSQL is largely horizontally scalable