2.

* Fundamentally, a Sink allows data to be sent out of Kafka to an external data source, such as a database, while a Source allows this data to be brought into Kafka.
* When constructing an integration between two systems via the ETL paradigm, Kafka mainly is a tool which facilitates the ‘Transfer’ part of the process, but largely leaves it up to the applications/sources/sinks on either side to handle Extracting and Loading the data respectively. By providing a mechanism (connector) for this, the load is taken off of the applications and put onto Kafka, which now facilitates the connection completely.
* Each connection is broken up into what are known as Tasks which act as a set of connector replicas that run alongside each other. That way if there is an error in one task, the other task is still able to continue processing data.

3.

* A KV database is a non-relational database in which the actual data is stored in a more object-like manner, indexed in the database via a key.
* One main advantage is that you can store basically anything in any format in a KV database. Moreover, you can store multiple different types of object records side by side in the same table, which reduces the degree of coupling between the application and the database.
* One of the main disadvantages of this type of approach is that the schema of each object is not fixed, which leaves it up to the transacting application to extract the required data not knowing necessarily what the schema of the record it is querying in advance. One way to manage this would be keeping the different schemas on the application side and having a mandatory field in the KV database record object specifying the schema used.
* Some examples include AWS DynamoDB, CouchDB, Redis and InfinityDB