Q1. What is the concept of a metaclass?

In object-oriented programming, a metaclass is a class whose instances are classes. Just as an ordinary class defines the behavior of certain objects, a metaclass defines the behavior of certain classes and their instances. Not all object-oriented programming languages support metaclasses.

Q2. What is the best way to declare a class's metaclass?

There are several ways to do this, but one way is to set \_\_metaclass\_\_ at the module level. This way, all classes of this module will be created using this metaclass, and we just have to tell the metaclass to turn all attributes to uppercase.

Q3. How do class decorators overlap with metaclasses for handling classes?

Ans = Anything you can do with a class decorator, you can of course do with a custom metaclass (just apply the functionality of the "decorator function", i.e., the one that takes a class object and modifies it, in the course of the metaclass's \_\_new\_\_ or \_\_init\_\_ that make the class object!-)

Q4. How do class decorators overlap with metaclasses for handling instances?

because the decorator returns the original class, instances are made from it, not from a wrapper object. In fact, instance creation is not intercepted at all.