In docs-soliditylang-org-en-v0.8.21.pdf, p65

A function type A is implicitly convertible to a function type B if and only if their parameter types are identical, their return types are identical, their internal/external property is identical and 1. the state mutability of A is more restrictive than the state mutability of B.

In particular:

• pure functions can be converted to view and non-payable functions

• view functions can be converted to non-payable functions

1. payable functions can be converted to non-payable functions

No other conversions between function types are possible.

The rule about payable and non-payable might be a little confusing, but in essence, if a function is payable, this means that it also accepts a payment of zero Ether, so it also is non-payable. On the other hand, a non-payable function will reject Ether sent to it, so non-payable functions cannot be converted to payable functions. To clarify, 3.rejecting ether is more restrictive than not rejecting ether. This means you can override a payable function with a non-payable but not the other way around.

Above statement 1,2,3 contradict each other.