Das wichtigste Prinzip ist folgendes: in Unicode werden Sonderzeichen (also Umlaute, ß, etc.) als Mehrbytezeichen abgespeichert. Es gibt verschiedene Transformationsformate die entscheiden in wievielen Bytes genau. Du nimmst immer die Anzahl der Zeichen mit der Anzahl der Bytes mal um das Ergebnis zu erhalten. Ich habe dir unten aufgeschrieben wieviel Bytes jede Codierung hat und wie man das berechnet. Einen Unterschied dazu gibt es nur bei UTF-8, denn hier werden normale Zeichen in einem Byte und Umlaute in 2 Bytes gespeichert.

In Isolatin wird jedes Zeichen mit 1 Byte (8 Bit) gespeichert, egal ob Umlaute, Sonderzeichen oder nicht.
“für” hat hier 3 Byte und “Straße” 6 Byte.

In UTF-8 wird ein Zeichen grundsätzlich in einem Byte (8 Bit) gespeichert. Deutsche Umlaute und auch “ß“ benötigen 2 Byte. Es gibt noch andere Zeichen die dann in 3 oder 4 Byte gespeichert werden, wie z.B. “€”
Hier hat “für” 1+2+1 = 4 Byte (2 wegen ü) und “Straße” (1+1+1+1+2+1) = 7 Byte (2 wegen ß).

UTF-16 speichert jedes Zeichen in 2 oder 4 Byte (16 Bit). Hier sind für uns nur die 2 Byte interessant denn UTF-16 speichert hier auch die Umlaute und die meisten Sonderzeichen. Damit hat “für” 3*2 = 6 Bytes und “Straße” hat 6*2 = 12 Bytes in UTF-16.

UTF-32 speichert jedes Zeichen in 4 Byte (32 Bit).
Hier hat “für” 3*4 = 12 Byte und “Straße” hat 6 * 4 = 24 Byte in UTF-32.