**KV Cache & MLA**

假设输入 在 iter=1，则其长度为 ，在 iter=2，长度为

我们可以认为：

我们考虑计算矩阵 , 在前，因此是行成列，因此

因此我们看上去只需要计算新的数据和权重相乘就可以获得 ，而不需要重复计算 。

我们考虑 矩阵 ，由于 causal attention即：

观察 iter=2

我们再去考虑

考虑最后都是 ，因此, 因此 和 iter=1 一致。而 则会引入新数据。

因此最直觉的做法是直接缓存 。但是这不是最优点。

考虑一个输入 输出，而我们需要 ，因此我们不需要获得 。我们可以认为 是产生 的一个过程矩阵。

因此我们关注最后需要产生的一个数据

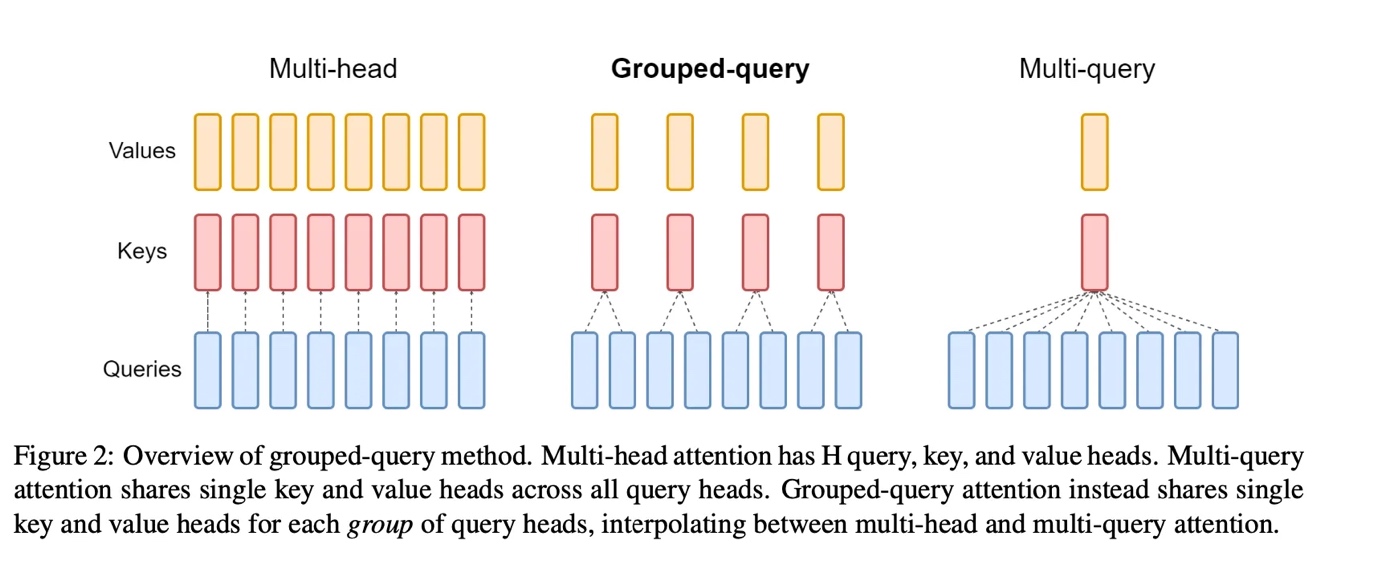
会发现这里需要 ，而 和 已被计算，所以直接缓存即可。因此这里我们只会去缓存 而不去缓存 （因为无需缓存）。

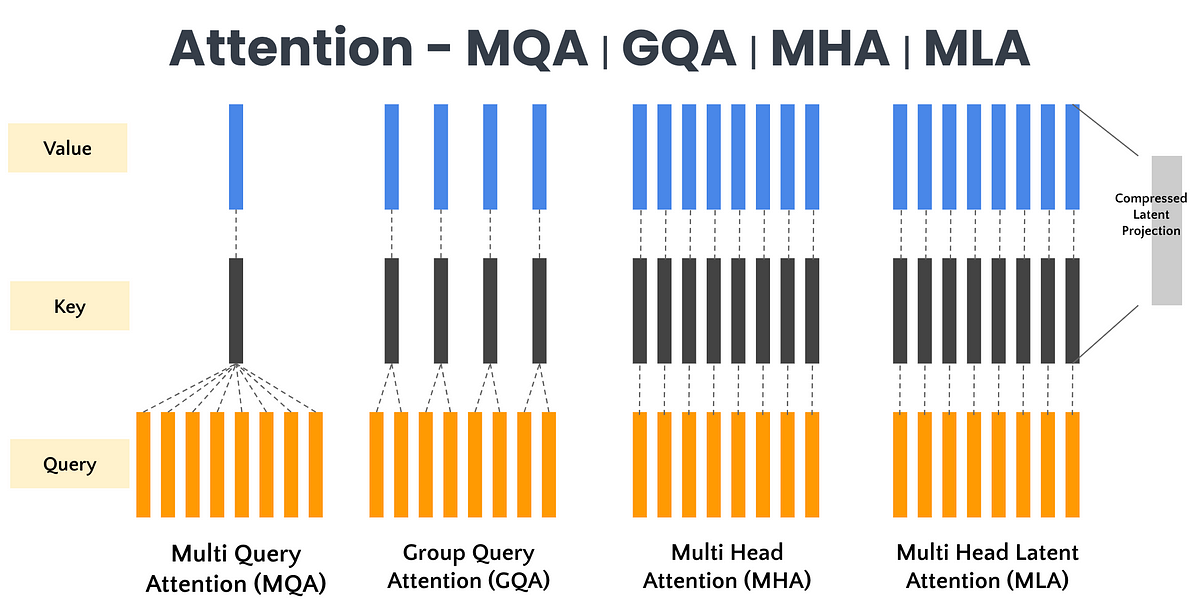
这里会发现 会随着长度增加，而越来越大。

**Grouped Query Attention (GQA), Multi Query Attention (MQA)**

计算 时，随着 sequence length 增加，在增大。因此通过 HBM 和SRAM的通信成为短板。（类似于DRAM 和 L1）。

因此我们需要降低 的大小。最简单的做法是降低数量，因此提出了 MQA 和 GQA以分别表示如下：





而MLA则使用 Latent Space表示。假设隐层为