Sea R la tabla con menor cantidad de bloques: *listadoempresas*, S la tabla *vendedores* y M=12 los bloques de memoria disponibles para la operación.

Uso la estrategia de **Loops anidados por bloque:**

$$\begin{aligned} Cost(R \bowtie S) &= B(R) + \lceil B(R) / (M-2) \rceil * B(S) \\ &= 1000 + \lceil 1000 / (12-2) \rceil * 2000 \\ &= 1000 + 100 * 2000 = \mathbf{201.000} \end{aligned}$$

No se puede aplicar **loop con único índice** porque no hay índices aprovechables

Pruebo con **Hash Grace**:
Con M=12 se pueden hacer 11 particiones. Pero como $B(R)/(M-1)=1000/11 \approx 90$ entonces las particiones de R no entrarán en memoria y no se puede aplicar Hash Grace eficientemente, se necesitaria más memoria.

Pruebo con **Sort Merge**:
Dijeron que la condición $M >= \lceil B(R)/V(A,R) \rceil + \lceil B(S)/V(A,S) \rceil + 1$, en la práctica no se la tiene en cuenta.
Lo considero porque la condición del join es de igualdad y las tablas van a quedar ordenadas con sort externo.

$$\begin{aligned} Cost(OrdM(R)) &= 2 * B(R) * \lceil \log_{M-1}(B(R)) \rceil \\ &= 2 * 1000 * \lceil \log_{11}(1000) \rceil \\ &= 2 * 1000 * 3 = \mathbf{6000} \\ Cost(OrdM(S)) &= 2 * B(S) * \lceil \log_{M-1}(B(S)) \rceil \\ &= 2 * 2000 * \lceil \log_{11}(2000) \rceil \\ &= 2 * 2000 * 4 = \mathbf{16.000} \end{aligned}$$

$$\begin{aligned} Cost(R \bowtie S) &= Cost(OrdM(R)) + Cost(OrdM(S)) + B(R) + B(S) \\ &= 6000 + 16.000 + 1000 + 2000 = \mathbf{25.000} \end{aligned}$$

Cantidad de filas que serán devueltas:

$$\begin{aligned} n(R \bowtie S) &= n(R) * n(S) / máx(V(A,R) , V(A,S)) \\ &= 10.000 * 50.000 / máx(100, 50) \\ &= 50.0000.000 / 100 = \mathbf{5.000.000} \end{aligned}$$

Conclusión: Sort Merge termina siendo la estrategia más eficiente con un costo de 25.000 bloques, devolviendo 5.000.000 de filas.