Efficient and Portable Einstein Summation in SQL

Mark Blacher[1], Julien Klaus[1], Christoph Staudt[1], Sören Laue[2], Viktor Leis[3], Joachim Giesen[1]

[1]Friedrich Schiller University Jena, [2]University of Hamburg, [3]Technical University of Munich

Why Einstein Summation in SQL?

While ML workloads include training as well as inferencing, supporting the latter efficiently is an immediate need.

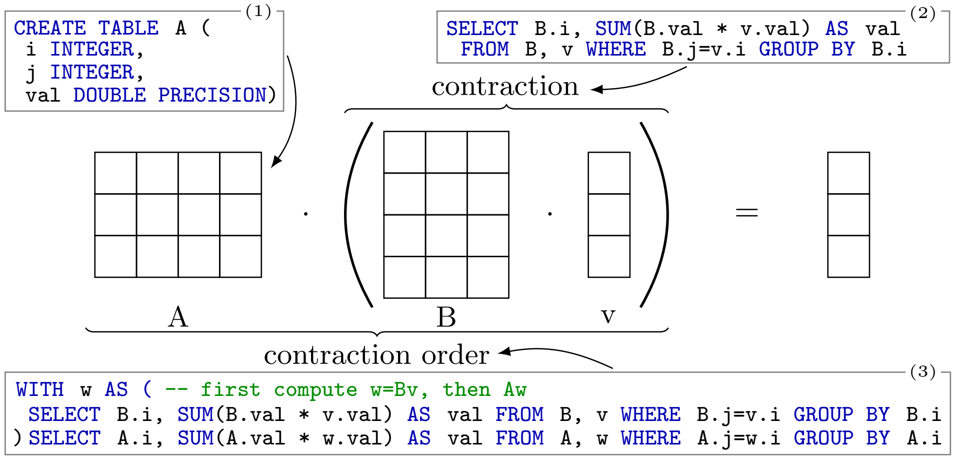
The seattle report on databaseresearch, Commun. ACM, August 2022

Einstein Summation in SQL Essentials

(1) **Tensors** are realized as **relations** (**COO** sparse **format**)

(2) **Tensor contractions** are implemented as **inner joins** paired with **GROUP BY** clauses and the **SUM** function

(3) **Common table expressions** are used to **optimize** the tensor **contraction order**



(1)

```
CREATE TABLE A (
 i INTEGER,
 j INTEGER,
 val DOUBLE PRECISION)
```

(2)

```
SELECT B.i, SUM(B.val * v.val) AS val
  FROM B, v WHERE B.j=v.i GROUP BY B.i
```

(3)

```
WITH w AS ( -- first compute w=Bv, then Aw
 SELECT B.i, SUM(B.val * v.val) AS val FROM B, v WHERE B.j=v.i GROUP BY B.i
) SELECT A.i, SUM(A.val * w.val) AS val FROM A, w WHERE A.j=w.i GROUP BY A.i
```

The Role of Query Engines in Einstein Summation

Planning and **execution times** (#SAT problem with 952 clauses)

DBMS	Planning time	Execution time
opt_einsum (NumPy backend)	0.00 s	3.69 s
SQLite	0.03 s	0.37 s
HyPer (interpreted)	0.87 s	0.08 s
PostgreSQL	1.51 s	20.19 s
DuckDB	*N/A*	*N/A*
DuckDB (no optimizations)	0.20 s	0.97 s

🤔 Disable query optimizations for Einstein summation queries?

Try it Out Yourself

🌍 sql-einsum.ti2.uni-jena.de

Acknowledgments

This work was supported by the Carl Zeiss Foundation within the project “Interactive Inference”.
🌍 *inference.uni-jena.de*