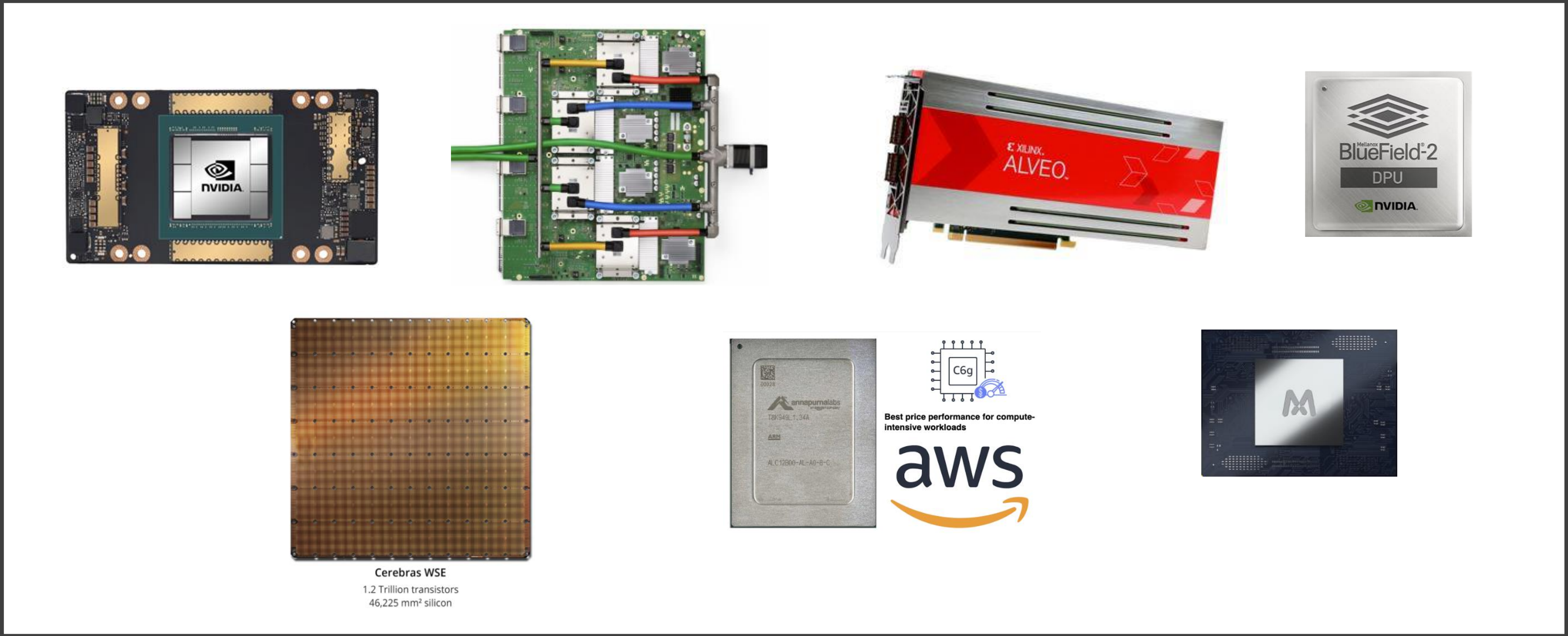
Rebalancing Memory and Compute with CXL Computational Memory

HARRY KIM / CPO, MetisX

Domain Specific Architecture

The next decade will see a Cambrian explosion of novel computer architectures, meaning exciting times for computer architects in academia and in industry.

- John L. Hennessy and David A. Patterson

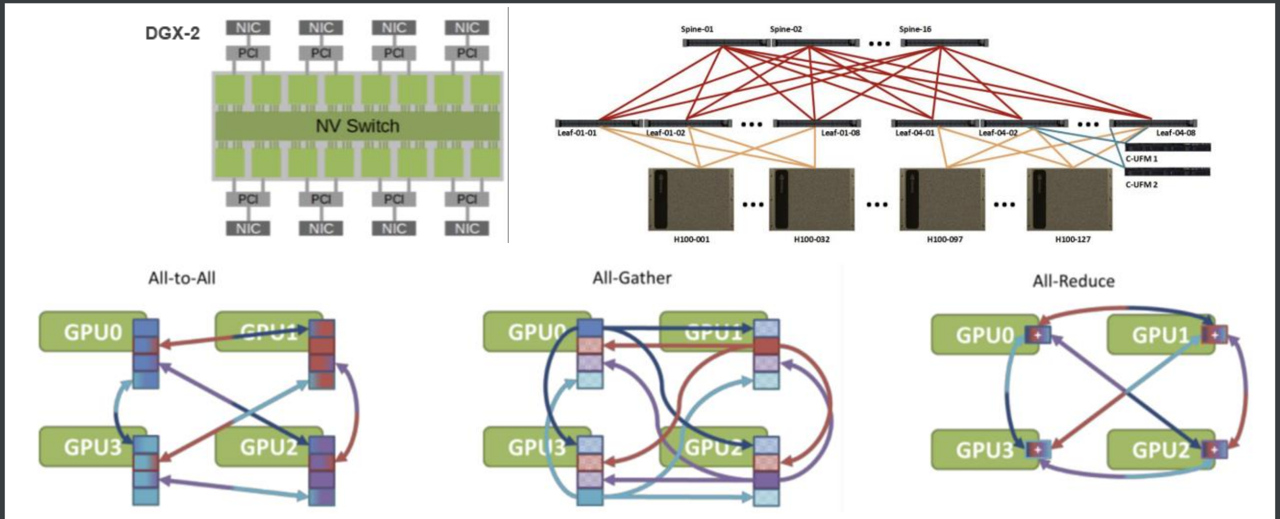


Scaling Constraints

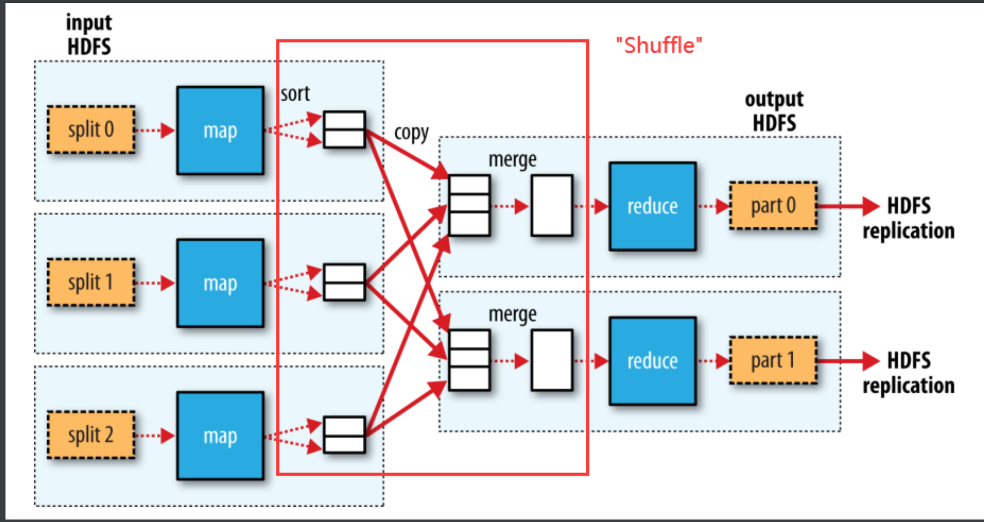
Low GPU/CPU utilization due to data movements between Cards, Servers, Racks and even Datacenters

Scaling hardware to meet growing compute and/or memory demands, through complex network and storage topology.

Adding more nodes makes it increasingly difficult to achieve linear performance gains, because when data is distributed across several nodes, it eventually needs to be gathered again.



Collective Communications in GPU pod



Shuffle in Spark Cluster

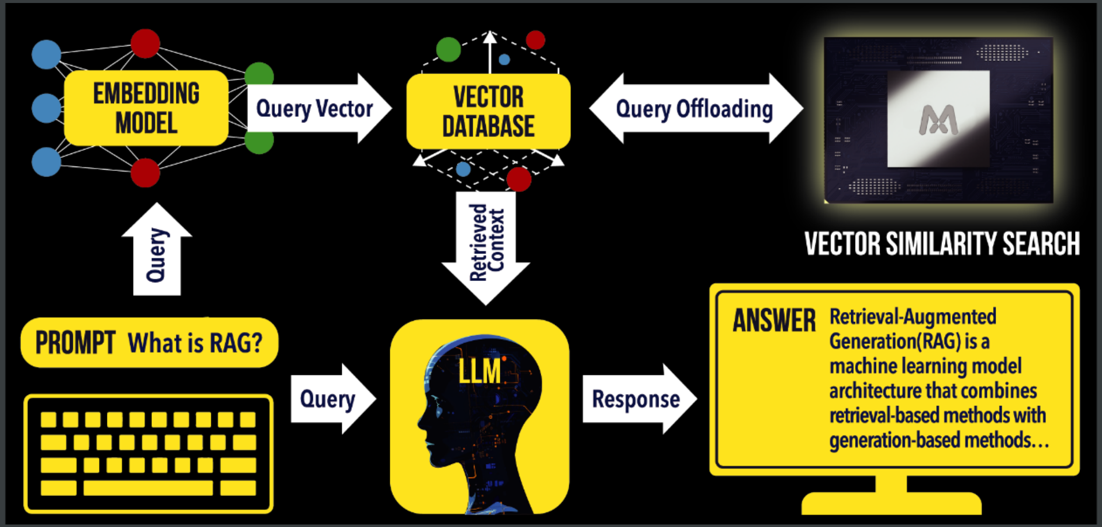
Vector Databases for RAG

Vector Databases
- Explicit External Memory for Retrieval
- New Data, Your Private Data
- Find and Augment Relevant Documents on LLM Generation

Memory Requirements
- Vectors: FP32 x 4K Dimensions x 10M Chunks = 150GB
  (c.f. Wikipedia: 6M Pages, 8M Chunks)
- Meta data, Key columns, Documents

Compute Requirements
- Cosine Distance: A few TFLOPS(not 100 TFLOPS)
- Cannot Compromise Accuracy in a Business Context
  Medical, Legal, Financial, Military, etc.
- Filtering with null, meta data, complex conditions
- DATABASE add/delete/update, Data governance

→ Bigger Memory
→ Transparent, Cache Coherent Memory
→ FP Vector + General DB Query Processing



Retrieval Augmented Generation Flow

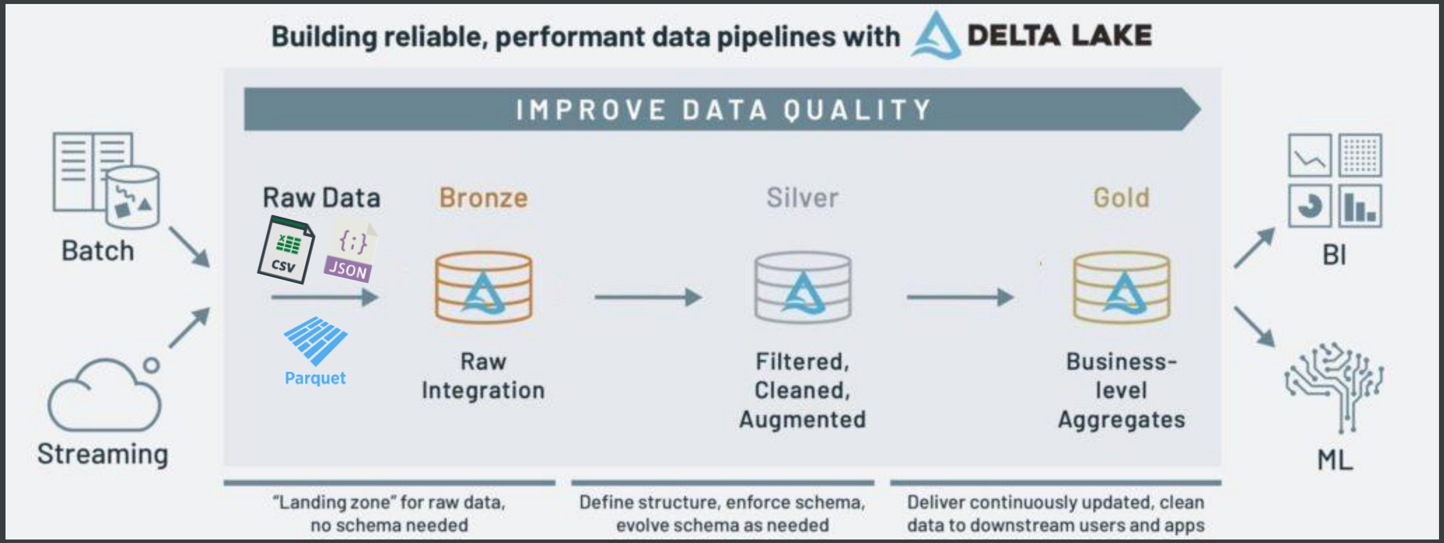
Rank	Model	Model Size (Million Parameters)	Memory Usage (GB, fp32)	Embedding Dimensions
1	SFR-Embedding-2_R	7111	26.49	4096
2	gte-Qwen2-7B-instruct	7613	28.36	3584
3	neural-embedding-v1			
4	NV-Embed-v1	7851	29.25	4096
5	voyage-large-2-instruct			1024
6	Linq-Embed-Mistral	7111	26.49	4096
7	SFR-Embedding-Mistral	7111	26.49	4096
8	gte-Qwen1.5-7B-instruct	7099	26.45	4096
9	gte-Qwen2-1.5B-instruct	1776	6.62	4096
10	voyage-lite-02-instruct	1220	4.54	1024

Huggingface MTEB Leaderboard

Data Processing Pipelines

Refining Unstructured Raw Data Transforms it into a Cleaned, Reliable and Structured Source

Feeding “high quality data” to ML/AI models both for training and inference (garbage in, garbage out)
Cluster with 1000s of nodes to process TBs of data – LOG files, Comments, Likes, ...
Data movement among nodes : (de-)compression, (de-)serialization, OOM or disk spill, snapshot(failover)
→ Bigger/Transparent/Coherent Memory
→ Query Processing + Integer Operations(Strings, Compression, Encode/Decode)



Business Intelligence

Data Mining
Visualization
Analysis Reporting
Decision Support

ML Training/Inference

Sentiment Analysis
Recommendation
Prediction
Generation

Source : https://www.databricks.com/kr/glossary/medallion-architecture

Under the Hood

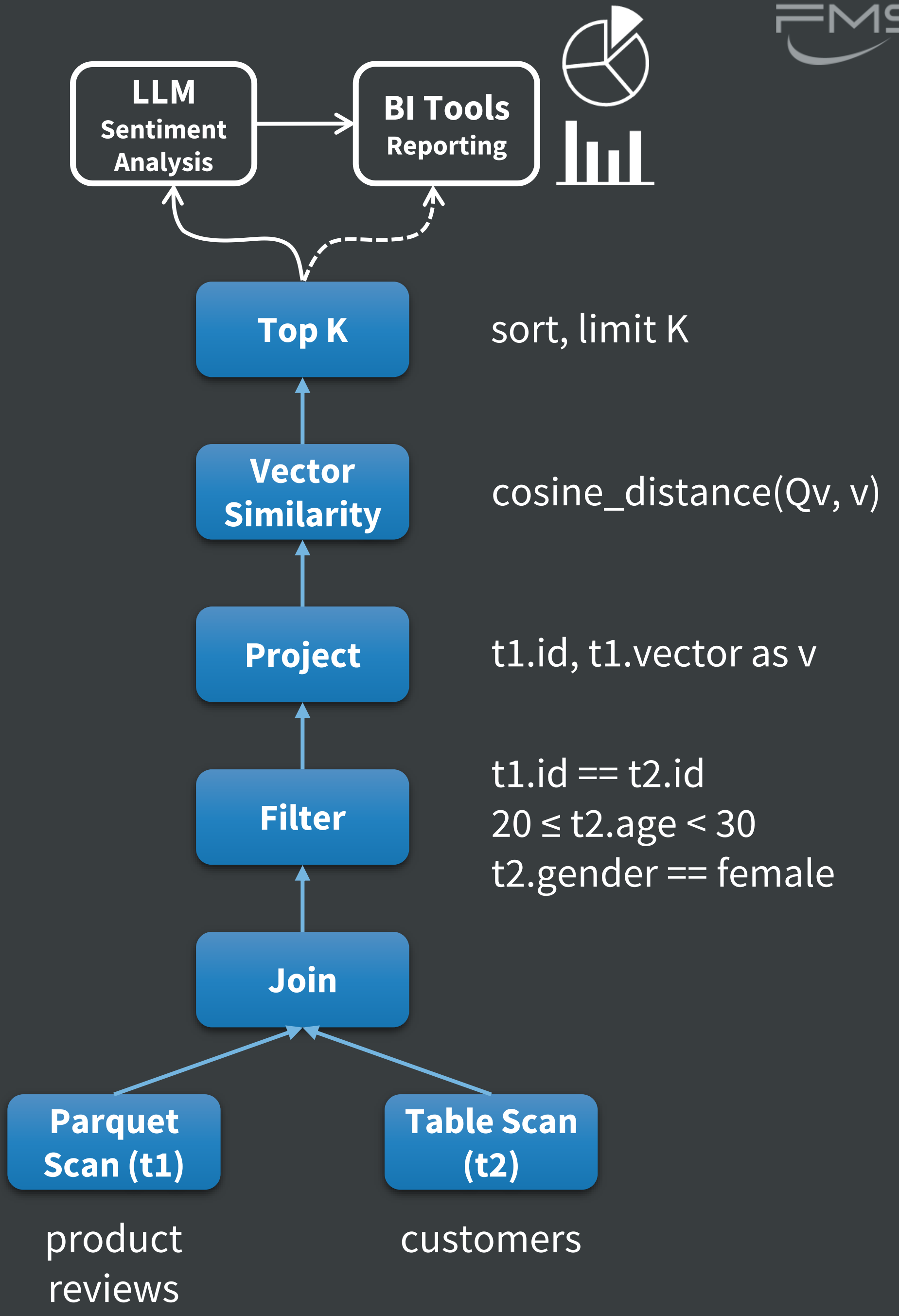
Q: How do women in their 20s react to the product's design?

Applying simple, repetitive operators on massive data

Raw data are usually compressed and encoded

Many "if" statements for filtering *In the US? In Europe?*

→ Distributed Execution Engines such as Apache Spark
MapReduce, Data Parallelism, In-Memory



More Data

When you have More Data,
You need More Memory as well as More Compute

But, Adding More Nodes doesn't Work, it's not Efficient.

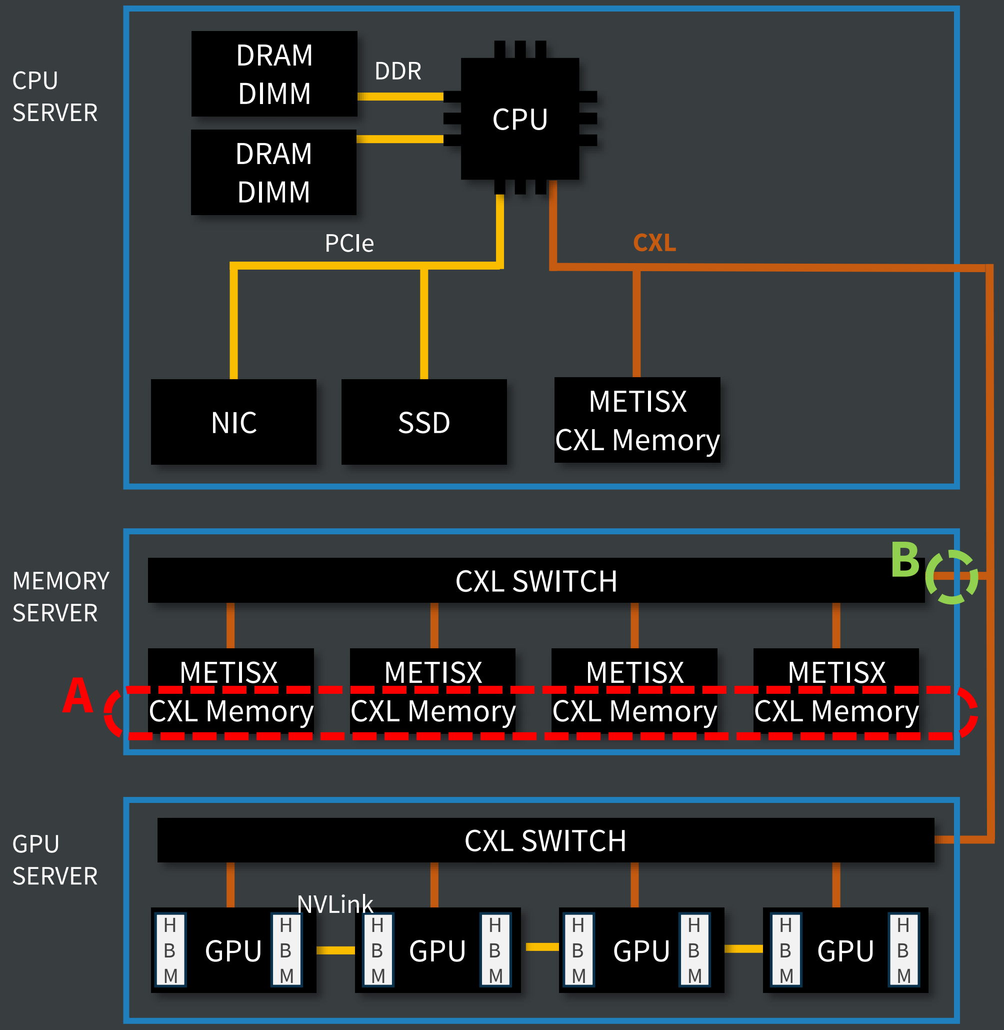
Vector Databases	Scale-out Data Analytics
Vector databases prefer in-memory data structures for fast response times and are generally not scalable.	OOM(Out of Memory) and disk spill are major issues that trouble data analysts.
X86 CPUs find it challenging to provide sufficient computational power for high-dimensional vector operations.	CPUs also used in supercomputers for scientific computations are overpowered for handling simple integers or strings.
Storing vector data on GPUs is impractical due to limited and expensive VRAM.	Similarly, Tensor Cores on GPUs during SQL processing are just unused silicon.

We need a domain-specific alternative beyond simply adding more nodes.

CXL Memory Expansion & Disaggregation

1. **Memory Expansion**
   - Direct Attached in the chassis
   - ~10 TB of DRAM
2. **Memory Disaggregation**
   - Expand with Switch(es)
   - Rack-scale Pooling / Sharing
   - Peer-to-Peer Access(UIO, .mem)
3. **Compute**
   - More data with longer latency -> Performance?
   - Load-store from far memory is time and power consuming
   - Eventually going to be overwhelmed by growing data
4. **CXL Memory Utilization**
   - A: Raw memory bandwidth
   - B: CXL bandwidth
   - A >> B

→ Near Memory Processing



CXL Memory + Data Processing

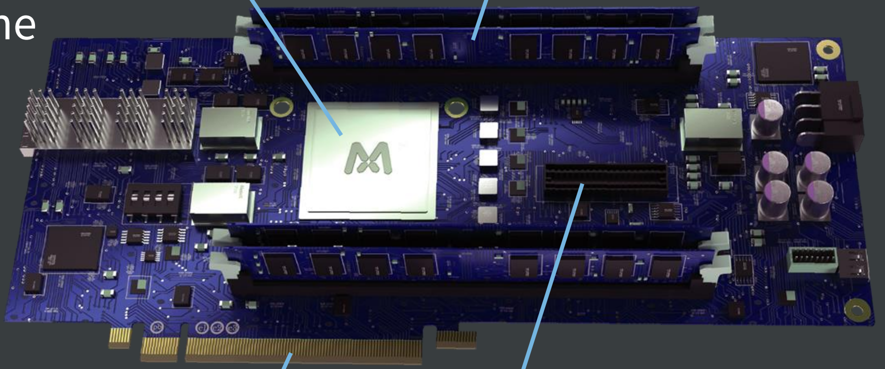
CXL Computational Memory for Large-scale Data

Available 2Q 2025

Novel CXL Hardware

1000s of Custom RISC-V Cores + TFLOPS Vector Engine

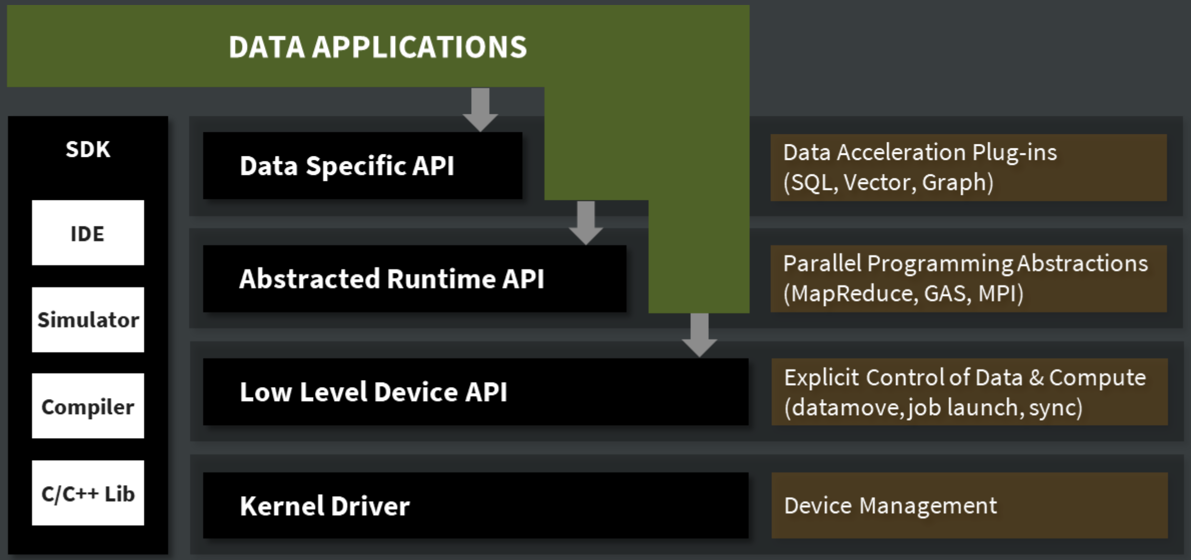
DDR5 x 4Ch ~1TB



CXL 3.0 HDM-DB with Back-invalidation Cache Coherence

SSD-backed CXL Expansion

Rich Software Framework



THANK YOU

harry.kim@metisx.com

http://metisx.com

https://www.linkedin.com/company/metisx/

Visit MetisX Booth # 734