**Solid State Drive (SSD)**

Solid State Drive (SSD): The data is stored on interconnected flash memory chips. The chips can be installed on a PCI Express (PCIe) card. (I/O Bandwidth: 3254.94 MB/s)

Hard Disk Drive (HDD): A metal platter with a magnetic coating that stores your data, a read/write head on an arm accesses the data while the platters are spinning. (I/O Bandwidth: 907.75 MB/s)

**Benefits of the SSD:**
- Super low latency: Zero seeking time
- Very fast read and write speeds

**Data Partitioning**

Chunks are both the I/O and processing units.
- **Structure:** A chunk is composed of columns connected by address pointers.
- **Metadata:** Each chunk contains the minimum and maximum values for each column.

**Storage System Architecture**

**Pull Based Model**
- The computation drives all data movement
- No natural way to coordinate data requests
- Lose CPU cycles due to memory access latency

**Push Based Model**
- Data flow drives the computation
- Redundant memory access since many data are dropped
- Endless memory access until all requests are satisfied

**Asynchronous Multi-Threading**

- The schema and chunk list are generated from metadata and passed as input to scheduler.
- Multiple worker threads are assigned with messages and sent to execution units.

**Caching**

- Cache layer stores chunk-column for the upcoming queries to reduce disk I/O.
- Cache is organized like a 2D array: [ChunkID][ColID].
- The cache capacity is maintained by an eviction mechanism.

**Push Based Prefetching Model**

- Batch queries
- Merge requests
- Read without interval

**Results**

**Loading Size Adjustment for SSD**

**I/O Requests**

**Execution Time with HD**

**Last Reading Request with HD**

**Execution Time with SSD**

**Last Reading Request with SSD**

**Contributions**

- Design a cache layer to take advantage of the high bandwidth of SSD and minimize the amount of I/O access.
- Introduce a push-based prefetching model that can dynamically read chunks in an optimal order to maximize the I/O throughput.
- The push-based prefetching mechanism allows multiple queries merge the requests for the same chunk, in order to share the I/O access.
- The push-based prefetching mechanism reduces the total read time of batch queries processing.