What’s in Apache Spark 2.0.0?

- Dataset evolution: SparkSession (entry to Dataset/DataFrame - not SQLContext(sc))
- Off-heap caching: Overcome GC limits, compressed object pointers (compressed oops - up to 32GB Java heap - 32 bits/4 bytes, 64 bits/8 bytes if over)

Project Tungsten - Closer to bare metal

- Memory management and binary processing
  - Java serialized object: JVM GC, 2 bytes UTF-16 encoding, header, hash code
  - C-style memory access - sun.misc.Unsafe
    - allocateMemory, copyMemory, freeMemory, getAddress, getInt, putInt
  - Spark manages memory
- Code generation: Don't create object, compare binary

Catalyst Optimizer

- represent query as tree/manipulate
- rule-based and cost-based optimization
- analysis, logical optimization, physical planning, and code generation to compile parts of queries to Java bytecode
- Literal(value: Int), Attribute(name: String)
- Add(left: TreeNode, right: TreeNode)
- Rule: for example tree.transform (add(lit1,lit2) = lit(1+2)
- Analysis: look up column names/types/tables from Catalog
- Logical Optimization: rule-based with constant folding, predicate pushdown, projection pruning, null propagation, Boolean expression simplification, and other rules
- Physical plans - cost model, code gen- expressions (+ predicate pushdown)

Project Tungsten 2.0 - reduce CPU bottlenecks

- Virtual function calls
- Use CPU registers instead of cache/memory

Simple aggregate query with filter

- count how many items have the sk 1000

Pre-2.0 Apache Spark: Volcano Iterator Model

- Graefe, 1994 paper "Volcano" - iterator, virtual function call
- "elegantly compose arbitrary combinations of operators"