

Contents

Acknowledgements	v
Abstract (English/Français)	vii
Table of Contents	xv
List of figures	xix
List of tables	xxi
1 Introduction	1
1.1 The Information Age	1
1.2 Data Management	1
1.3 Data Analytics	3
1.3.1 Data Analytics Challenges	4
1.4 Implications on the DBMS architecture	4
1.5 Solid-State Storage and Work Sharing for Efficient Scaleup Data Analytics	5
1.5.1 Evolution of the storage hierarchy	6
1.5.2 Summary and Contributions	7
1.5.3 Published papers	8
1.6 Outline (How to read this thesis)	9
2 Related Work and Background	11
2.1 Hardware Trends in Storage	11
2.1.1 Fifty years of hard disks	11
2.1.2 From hard disks to solid-state storage	12
2.1.3 Flash-based solid-state storage	13
2.1.4 Phase Change Memory	14
2.1.5 Flash vs. PCM	15
2.1.6 More solid-state storage technologies	17
2.1.7 Sequential Reads and Random Writes in Storage Systems	18
2.2 Data Analysis Systems	18
2.2.1 High-Throughput Updates and Data Analysis	19
2.2.2 Adaptive organization of data and updates	20

- 2.2.3 Orthogonal Uses of SSDs for Data Warehouses 21
- 2.3 Optimizing Data Analysis Queries Using Work-Sharing 21
 - 2.3.1 Sharing in the I/O layer and in the execution engine 22
 - 2.3.2 Simultaneous Pipelining 23
 - 2.3.3 The QPipe execution engine 23
 - 2.3.4 Global Query Plans with shared operators 24
 - 2.3.5 The CJOIN operator 25
 - 2.3.6 Quantifying work sharing opportunities 27
- 2.4 Bloom filters 28
 - 2.4.1 Bloom filters' applications in data management 28
 - 2.4.2 Bloom filters for evolving workloads and storage 28
- 2.5 Indexing for Data Analysis 29
 - 2.5.1 SSD-aware indexing 29
 - 2.5.2 Specialized optimizations for tree indexing 29
 - 2.5.3 Indexing for data warehousing 30
- 2.6 Knowledge-based Data Analysis 30
 - 2.6.1 RDF datasets and benchmarks 30
 - 2.6.2 Storing RDF data 31
- 2.7 Summary 32
- 3 Online Updates for Data Analytics using Solid-State Storage 33**
 - 3.1 Introduction 33
 - 3.1.1 Efficient Online Updates for DW: Limitations of Prior Approaches 34
 - 3.1.2 Our Solution: Cache Updates in SSDs for online updates in DWs 35
 - 3.1.3 Contributions 38
 - 3.2 Efficient Online Updates and Range Scans in Data Warehouses 38
 - 3.2.1 Basic Concepts and Focus of the Study 39
 - 3.2.2 Conventional Approach: In-Place Updates 40
 - 3.2.3 Prior Proposals: Indexed Updates (IU) 41
 - 3.3 MaSM Design 42
 - 3.3.1 Basic Ideas 43
 - 3.3.2 MaSM-2M 44
 - 3.3.3 MaSM-M 46
 - 3.3.4 MaSM- α M 49
 - 3.3.5 Further Optimizations 49
 - 3.3.6 Transaction Support 50
 - 3.3.7 Achieving The Five Design Goals 52
 - 3.4 Analysis and Modeling of MaSM 53
 - 3.4.1 Theorems About MaSM Behavior 54
 - 3.4.2 SSD Wear vs. Memory Footprint 55
 - 3.4.3 Memory Footprint vs. Performance 56
 - 3.4.4 Modeling the I/O Time of Range Queries 58

3.4.5	LSM analysis and comparison with MaSM	62
3.5	Experimental Evaluation	64
3.5.1	Experimental Setup	64
3.5.2	Experiments with Synthetic Data	65
3.5.3	TPC-H Replay Experiments	71
3.6	Discussion	71
3.6.1	Shared-Nothing Architectures	71
3.6.2	MaSM for deeper memory hierarchies and new storage technologies	72
3.6.3	Can SSDs Enhance Performance and Be Cost Efficient?	73
3.6.4	General support of MaSM	73
3.6.5	Applying MaSM to Cloud Data Management	75
3.7	Conclusion	75
4	Enhancing Data Analytics with Work Sharing	77
4.1	Introduction	77
4.1.1	Methodologies for sharing data and work	77
4.1.2	Integrating Simultaneous Pipelining and Global Query Plans	78
4.1.3	Optimizing Simultaneous Pipelining	79
4.1.4	Simultaneous Pipelining vs. Global Query Plans	79
4.1.5	Contributions	80
4.1.6	Outline	81
4.2	Integrating SP and GQP	81
4.2.1	Benefits of applying SP to GQP	81
4.2.2	CJOIN as a QPipe stage	82
4.2.3	SP for the CJOIN stage	83
4.3	Shared Pages Lists for SP	84
4.3.1	Design of a SPL	87
4.3.2	Linear Window of Opportunity	87
4.4	Experimental Methodology	88
4.5	Experimental Analysis	89
4.5.1	Impact of concurrency	90
4.5.2	Impact of data size	92
4.5.3	Impact of Similarity	95
4.5.4	SSB query mix evaluation	96
4.6	Discussion	97
4.7	Conclusions	98
5	BF-Tree: Approximate Tree Indexing	101
5.1	Introduction	101
5.1.1	Implicit Clustering	101
5.1.2	The capacity-performance trade-off	102
5.1.3	Indexing for modern storage	103
5.1.4	Approximate Tree Indexing	103

5.2	Making approximate indexing competitive	105
5.3	Bloom Filter Tree (BF-Tree)	106
5.3.1	BF-Tree architecture	106
5.3.2	Creating and Updating a BF-Tree	107
5.4	Read-optimized BF-Trees	110
5.5	Modeling BF-Trees and B ⁺ -Trees	111
5.5.1	Parameters and model	111
5.5.2	Discussion on BF-Tree size and compression	115
5.6	Experimental evaluation	116
5.6.1	Experimental Methodology	116
5.6.2	BF-Tree for primary key	117
5.6.3	BF-Tree for non-unique attributes	121
5.6.4	BF-Tree for TPCCH	123
5.6.5	Summary	125
5.7	BF-Tree as a general index	125
5.7.1	BF-Tree vs. interpolation search	125
5.7.2	Range scans	126
5.7.3	Querying in the presence of inserts and deletes	126
5.8	Optimizations	129
5.9	Conclusions	129
6	Graph Data Management using Solid-State Storage	131
6.1	Introduction	131
6.1.1	Contributions	132
6.1.2	Outline	133
6.2	Path Processing	133
6.3	RDF processing on PCM	137
6.3.1	The RDF-tuple	139
6.3.2	The Internals of Pythia	140
6.3.3	Experimental Workload	141
6.3.4	Experimental Evaluation	142
6.3.5	Comparison with RDF-3X	143
6.4	PCM Opportunities	144
6.4.1	Algorithm redesign	145
6.5	SPARQL code for test queries	145
6.6	Conclusions	146
7	The Big Picture	149
7.1	What we did	149
7.2	Storage is constantly evolving: Opportunities and challenges	150
7.2.1	Persistent main memory: Database Management Systems vs. Operating Systems	151
7.2.2	Software stack is too slow	151

7.3	Ever increasing concurrency in analytics	151
7.3.1	Extending support of work sharing	152
7.3.2	Column-stores and high concurrency	152
	Bibliography	153
	Curriculum Vitae	167