Inhaltsverzeichnis

Vorwort ... 9

Danksagungen ... 11

Über dieses Buch ... 15

1 **Ein neues Paradigma für Big Data** ... 17
1.1 Aufbau des Buches ... 18
1.2 Skalierung mit einer herkömmlichen Datenbank ... 19
1.2.1 Skalierung mit einer Warteschlange ... 19
1.2.2 Skalierung durch Sharding ... 20
1.2.3 Erste Probleme mit der Fehlertoleranz ... 21
1.2.4 Probleme mit fehlerhaften Daten ... 21
1.2.5 Was ist schiefgegangen? ... 21
1.2.6 Inwiefern sind Big-Data-Verfahren hilfreich? ... 22
1.3 NoSQL ist kein Allheilmittel ... 22
1.4 Grundlagen ... 23
1.5 Erwünschte Eigenschaften eines Big-Data-Systems ... 24
1.5.1 Belastbarkeit und Fehlertoleranz ... 24
1.5.2 Lesen und Aktualisieren mit geringen Latenzzeiten ... 25
1.5.3 Skalierbarkeit ... 25
1.5.4 Allgemeingültigkeit ... 25
1.5.5 Erweiterbarkeit ... 25
1.5.6 Ad-hoc-Abfragen ... 25
1.5.7 Minimaler Wartungsaufwand ... 26
1.5.8 Fehlerbehebung ... 26
1.6 Schwierigkeiten vollständig inkrementeller Architekturen ... 26
1.6.1 Komplexität im Betrieb ... 27
1.6.2 Extreme Komplexität, um letztendliche Konsistenz zu erzielen ... 28
1.6.3 Keine Fehlertoleranz gegenüber menschlichem Versagen ... 30
1.6.4 Vollständig inkrementelle Lösung kontra Lambda-Architektur ... 31
1.7 Lambda-Architektur ... 32
1.7.1 Batch-Layer ... 33
1.7.2 Serving-Layer ... 35
1.7.3 Batch- und Serving-Layer erfüllen fast alle Anforderungen ... 35
1.7.4 Speed-Layer ... 36
1.8 Die neuesten Trends ... 39
1.8.1 Prozessoren werden kaum noch schneller ... 39
1.8.2 Elastic Clouds ... 39
1.8.3 Ein lebhaftes Open-Source-Ökosystem für Big Data ... 40
1.9 Beispielanwendung: SuperWebAnalytics.com ... 41
1.10 Zusammenfassung ... 41

Teil I Batch-Layer ... 43

2 **Das Datenmodell für Big Data** ... 45
2.1 Die Eigenschaften von Daten ... 46
2.1.1 Daten sind ursprünglich ... 49
2.1.2 Daten sind unveränderlich ... 52
2.1.3 Daten sind beständig korrekt ... 54
2.2 Das faktenbasierte Modell zur Repräsentierung von Daten ... 55
2.2.1 Faktenbeispiele und ihre Eigenschaften ... 56
2.2.2 Vorteile des faktenbasierten Modells ... 58
2.3 Graphenschemata ... 62
2.3.1 Elemente eines Graphenschemas ... 62
2.3.2 Die Notwendigkeit, dem Schema zu gehorchen ... 63
2.4 Ein vollständiges Datenmodell für SuperWebAnalytics.com ... 64
2.5 Zusammenfassung ... 65

3 **Das Datenmodell für Big Data: Praxis** ... 67
3.1 Wozu ein Serialisierungs-Framework? ... 67
3.2 Apache Thrift ... 68
3.2.1 Knoten ... 69
3.2.2 Kanten ... 69
3.2.3 Eigenschaften ... 70
3.2.4 Alles in Datenobjekten zusammenfassen ... 71
3.2.5 Weiterentwicklung des Schemas ... 71
3.3 Für Serialisierungs-Frameworks geltende Beschränkungen ... 72
3.4 Zusammenfassung ... 73

4 **Datenspeicherung im Batch-Layer** ... 75
4.1 Speicheranforderungen des Stammdatensatzes ... 76
4.2 Auswahl einer Speicherlösung für den Batch-Layer ... 77
4.2.1 Schlüssel-Werte-Datenbank zum Speichern des Stammdatensatzes verwenden ... 77
4.2.2 Verteilte Dateisysteme ... 78
4.3 Funktionsweise verteilter Dateisysteme ... 79
4.4 Speichern des Stammdatensatzes mit einem verteilten Dateisystem ... 81
4.5 Vertikale Partitionierung ... 83
4.6 Verteilte Dateisysteme sind maschinennah ... 84
4.7 Speichern des SuperWebAnalytics.com-Stammdatensatzes in einem verteiltem Dateisystem ... 85
4.8 Zusammenfassung ... 86

5 **Datenspeicherung im Batch-Layer: Praxis** ... 87
5.1 Verwendung des Hadoop Distributed File Systems ... 87
5.1.1 Das Problem mit kleinen Dateien ... 89
5.1.2 Eine allgemeinere Abstrahierung ... 89
5.2 Datenspeicherung im Batch-Layer mit Pail ... 91
5.2.1 Grundlegende Pail-Operationen ... 91
5.2.2 Objekte serialisieren und in Pails speichern ... 93

5.2.3 Pail-Operationen ... 95
5.2.4 Vertikale Partitionierung mit Pail ... 96
5.2.5 Pail-Dateiformat und Komprimierung ... 97
5.2.6 Vorteile von Pail zusammengefasst ... 98
5.3 Speichern des Stammdatensatzes für SuperWebAnalytics.com ... 99
5.3.1 Ein strukturiertes Pail für Thrift-Objekte ... 101
5.3.2 Ein einfaches Pail für SuperWebAnalytics.com ... 102
5.3.3 Ein geteiltes Pail zur vertikalen Partitionierung des Datensatzes ... 103
5.4 Zusammenfassung ... 107

6 Batch-Layer ... 109
6.1 Beispiele ... 110
6.1.1 Anzahl der Pageviews innerhalb eines bestimmten Zeitraums ... 110
6.1.2 Vorhersage des Geschlechts ... 110
6.1.3 Einflussreiche Tweets ... 111
6.2 Berechnungen im Batch-Layer ... 112
6.3 Neuberechnungsalgorithmen kontra inkrementelle Algorithmen ... 114
6.3.1 Performance ... 115
6.3.2 Fehlertoleranz gegenüber menschlichem Versagen ... 116
6.3.3 Allgemeine Anwendbarkeit des Algorithmus ... 117
6.3.4 Auswahl eines Algorithmustyps ... 118
6.4 Skalierbarkeit im Batch-Layer ... 118
6.5 MapReduce: Ein Paradigma für Big-Data-Berechnungen ... 120
6.5.1 Skalierbarkeit ... 121
6.5.2 Fehlertoleranz ... 123
6.5.3 Allgemeine Anwendbarkeit von MapReduce ... 123
6.6 Maschinennähe ... 126
6.6.1 Berechnungen in mehreren Schritten sind nicht intuitiv ... 126
6.6.2 Die manuelle Implementierung von Joins ist sehr kompliziert ... 126
6.6.3 Enge Kopplung der logischen und physischen Ausführung ... 128
6.7 Pipe-Diagramme: Eine allgemeinere Auffassung von Stapelverarbeitungsberechnungen ... 129
6.7.1 Konzepte der Pipe-Diagramme ... 130
6.7.2 Ausführen von Pipe-Diagrammen via MapReduce ... 134
6.7.3 Combiner-Aggregator ... 135
6.7.4 Beispiele für Pipe-Diagramme ... 136
6.8 Zusammenfassung ... 138

7 Batch-Layer: Praxis ... 139
7.1 Ein Beispiel zur Veranschaulichung ... 140
7.2 Typische Schwierigkeiten Daten verarbeitender Tools ... 142
7.2.1 Proprietäre Sprachen ... 142
7.2.2 Mangelhaft einbindungsfähige Abstraktionen ... 143
7.3 Einführung in JCascalog ... 144
7.3.1 Das JCascalog-Datenmodell ... 144
7.3.2 Aufbau einer JCascalog-Abfrage ... 146
7.3.3 Abfragen mehrerer Datensätze ... 147
7.3.4 Gruppierung und Aggregatoren ... 150

7.3.5 Schrittweise Abarbeitung einer Abfrage ... 151
7.3.6 Benutzerdefinierte Prädikate ... 154
7.4 Einbindung ... 159
7.4.1 Subqueries kombinieren ... 160
7.4.2 Dynamisch erzeugte Subqueries ... 161
7.4.3 Prädikatmakros ... 164
7.4.4 Dynamisch erzeugte Prädikatmakros ... 167
7.5 Zusammenfassung ... 170

8 Beispiel eines Batch-Layers: Architektur und Algorithmen ... 171
8.1 Design des Batch-Layers für SuperWebAnalytics.com ... 172
8.1.1 Unterstützte Abfragen ... 172
8.1.2 Batch-Views ... 173
8.2 Überblick über den Workflow ... 176
8.3 Aufnahme neuer Daten ... 177
8.4 URL-Normalisierung ... 178
8.5 User-ID-Normalisierung ... 179
8.6 Deduplizierung der Pageviews ... 184
8.7 Berechnung der Batch-Views ... 184
8.7.1 Zeitlicher Verlauf der Pageviews ... 185
8.7.2 Zeitlicher Verlauf der eindeutig unterschiedlichen Besucher ... 186
8.7.3 Analyse der Bounce-Rate ... 187
8.8 Zusammenfassung ... 188

9 Beispiel eines Batch-Layers: Implementierung ... 189
9.1 Ausgangspunkt ... 189
9.2 Vorbereitung des Workflows ... 190
9.3 Aufnahme neuer Daten ... 191
9.4 URL-Normalisierung ... 195
9.5 User-ID-Normalisierung ... 197
9.6 Deduplizierung der Pageviews ... 204
9.7 Berechnung der Batch-Views ... 204
9.7.1 Zeitlicher Verlauf der Pageviews ... 204
9.7.2 Zeitlicher Verlauf der eindeutig unterschiedlichen Besucher ... 207
9.7.3 Berechnung der Bounce-Rate ... 209
9.8 Zusammenfassung ... 212

Teil II Serving-Layer ... 213

10 Serving-Layer ... 215
10.1 Performancekennzahlen des Serving-Layers ... 216
10.2 Lösung des Problems »Normalisierung kontra Denormalisierung« durch den Serving-Layer ... 219
10.3 Anforderungen an eine Datenbank für den Serving-Layer ... 221
10.4 Gestaltung eines Serving-Layers für SuperWebAnalytics.com ... 222
10.4.1 Zeitlicher Verlauf der Pageviews ... 223
10.4.2 Zeitlicher Verlauf eindeutig unterschiedlicher Besucher ... 223

10.4.3 Berechnung der Bounce-Rate ... 224
10.5 Vergleich mit einer vollständig inkrementellen Lösung ... 225
10.5.1 Vollständig inkrementelle Lösung ... 225
10.5.2 Vergleich mit einer auf der Lambda-Architektur beruhenden Lösung ... 231
10.6 Zusammenfassung ... 232

11 Serving-Layer: Praxis ... 233
11.1 ElephantDB: Grundlagen ... 233
11.1.1 Views in ElephantDB erzeugen ... 234
11.1.2 Views in ElephantDB deployen ... 234
11.1.3 ElephantDB verwenden ... 235
11.2 Einrichtung des Serving-Layers für SuperWebAnalytics.com ... 237
11.2.1 Zeitlicher Verlauf der Pageviews ... 237
11.2.2 Zeitlicher Verlauf eindeutig unterschiedlicher Besucher ... 240
11.2.3 Berechnung der Bounce-Rate ... 241
11.3 Zusammenfassung ... 242

Teil III Speed-Layer ... 243

12 Echtzeit-Views ... 245
12.1 Berechnung von Echtzeit-Views ... 246
12.2 Speichern der Echtzeit-Views ... 248
12.2.1 Letztendliche Genauigkeit ... 249
12.2.2 Im Speed-Layer gespeicherter Zustand ... 249
12.3 Schwierigkeiten bei inkrementeller Berechnung ... 250
12.3.1 Gültigkeit des CAP-Theorems ... 251
12.3.2 Das komplexe Zusammenwirken von CAP-Theorem und inkrementellen Algorithmen ... 253
12.4 Asynchrone kontra synchrone Aktualisierungen ... 254
12.5 Echtzeit-Views verwerfen ... 256
12.6 Zusammenfassung ... 258

13 Echtzeit-Views: Praxis ... 259
13.1 Cassandras Datenmodell ... 259
13.2 Cassandra verwenden ... 261
13.2.1 Cassandra für Fortgeschrittene ... 263
13.3 Zusammenfassung ... 264

14 Warteschlangen und Streamverarbeitung ... 265
14.1 Warteschlangen ... 265
14.1.1 Warteschlangen mit nur einem Abnehmer ... 266
14.1.2 Warteschlangen mit mehreren Abnehmern ... 268
14.2 Streamverarbeitung ... 269
14.2.1 Warteschlangen und Worker ... 270
14.2.2 Fallstricke beim Warteschlangen-Worker-Ansatz ... 271

14.3 Streamverarbeitung one-at-a-time auf höherer Ebene ... 272
14.3.1 Storm-Modell ... 272
14.3.2 Gewährleistung der Nachrichtenverarbeitung ... 277
14.4 SuperWebAnalytics.com: Speed-Layer ... 279
14.4.1 Aufbau der Topologie ... 281
14.5 Zusammenfassung ... 282

15 Warteschlangen und Streamverarbeitung: Praxis ... 283
15.1 Definition einer Topologie mit Apache Storm ... 283
15.2 Apache Storm-Cluster und Bereitstellung ... 286
15.3 Gewährleistung der Nachrichtenverarbeitung ... 288
15.4 Implementierung des Speed-Layers ... 291
15.5 Zusammenfassung ... 296

16 Streamverarbeitung kleiner Stapel ... 297
16.1 Genau einmalige Verarbeitung ... 297
16.1.1 Verarbeitung in streng festgelegter Reihenfolge ... 298
16.1.2 Streamverarbeitung kleiner Stapel ... 299
16.1.3 Topologien zur Verarbeitung kleiner Stapel ... 300
16.2 Grundlegende Konzepte der Streamverarbeitung kleiner Stapel ... 302
16.3 Erweiterte Pipe-Diagramme zur Beschreibung der Streamverarbeitung kleiner Stapel ... 304
16.4 Fertigstellung des Speed-Layers für SuperWebAnalytics.com ... 305
16.4.1 Zeitlicher Verlauf der Pageviews ... 305
16.4.2 Berechnung der Bounce-Rate ... 306
16.5 Eine weitere Methode zur Berechnung der Bounce-Rate ... 311
16.6 Zusammenfassung ... 312

17 Streamverarbeitung kleiner Stapel: Praxis ... 313
17.1 Trident verwenden ... 313
17.2 Fertigstellung des Speed-Layers für SuperWebAnalytics.com ... 317
17.2.1 Zeitlicher Verlauf der Pageviews ... 317
17.2.2 Berechnung der Bounce-Rate ... 320
17.3 Fehlertolerante Verarbeitung kleiner Stapel im Arbeitsspeicher ... 326
17.4 Zusammenfassung ... 328

18 Die Lambda-Architektur im Detail ... 329
18.1 Definition von Datenhaltungssystemen ... 329
18.2 Batch- und Serving-Layer ... 331
18.2.1 Inkrementelle Stapelverarbeitung ... 331
18.2.2 Ressourcennutzung des Batch-Layers messen und optimieren ... 338
18.3 Speed-Layer ... 343
18.4 Query-Layer ... 343
18.5 Zusammenfassung ... 345

Stichwortverzeichnis ... 347