Von Apache Lucene zu Multi-Model Data Stores (Document, Key-Value, Graph) mit multidimensionaler bitemporaler 6NF Ablage + Dynamische Links

DDVUG

8. Tagung (Herbsttagung 2018)

42 data morrow

DATA MODELING ZONE EUROPE 2018

data modeling zone

EUROPE2018

NEW! 3rd day extra

24th & 25th September 2018

Dusseldorf, Germany

Agenda

- Fuzzy / Roter Faden?

DISCLAIMER:
Denn das ist alles nur geklaut
Das ist alles gar nicht meine
Das ist alles nur geklaut
Doch das weiß ich nur ganz alleine
Das ist alles nur geklaut
Und gestohlen
Nur gezogen
Und geraubt
Entschuldigung, das hab' ich mir erlaubt
Aus „Alles nur geklaut“: Die Prinzen

KI ist eindeutig das neue Schlagwort - die Data Fashion Show geht weiter.

- Letztes Jahr verschwand das Schlagwort Big Data endgültig, aber alle sprachen von Data Science. In diesem Jahr sprachen alle über KI und maschinelles Lernen. Ich schätze, es ist ein Teil der Zugehörigkeit zum Datenberuf, sich jedes Jahr an das neueste Schlagwort anzupassen! Meine Keynote wurde vom Redaktionsleiter der Konferenz von "data science for grown ups" in "succeeding in AI" umbenannt. In meinem eigenen Unternehmen war es schwierig, den Leuten zu erklären, dass unsere Informatiker auch an KI arbeiten können (im Grunde genommen habe ich in den meisten Präsentationen KI geschrieben). Ich habe mich von nun an entschieden, Datenwissenschaftler die Daten- und KI-Wissenschaftler zu nennen, um zu signalisieren, dass es sich um die gleichen Personen handelt. Aus kritischer Sicht könnte man argumentieren, dass die echte KI noch etwas Zeit in Anspruch nimmt. Die Art von KI, über die die meisten Leute sprechen, ist nicht ein einzelner Algorithmus, sondern ein System einer Vielzahl von miteinander verbundenen Algorithmen, die in eine gut verwaltete Algorithmusarchitektur eingebettet sind, die Maschinen sinnvoll machen kann, denken, lernen und handeln (siehe auch meinen letzten Artikel über Smart Machines). Soweit ich weiß, hat das bisher noch niemand erreicht. Mein Kollege Dat Tran, Head of Data Science bei Idealo, hat es heute in einem Linkedin-Posting gut zusammengefasst: "Wir sind so weit entfernt von dem, was die Leute echte KI nennen. Alles, was wir tun, ist eine mathematische Optimierung einiger Daten. Und wenn die Daten schlecht sind, ist auch die KI schlecht". Eine Sache, auf die ich in meiner heutigen Keynote hingewiesen habe, war, dass wir es versäumt haben, eine Wissenstaxonomie für unsere Maschinen zu erstellen, sie alle müssen sich auf Rohdaten verlassen, wenn wir maschinelle Lernalgorithmen ausführen. Aber wie können Maschinen etwas über die Gesellschaft lernen, wenn sie keine Bücher lesen?
- Übersetzt mit www.DeepL.com/Translator

Dr. Alexander Borek Global Head of Data & Analytics at Volkswagen Financial Services AG
Succeeding in Data & AI: Insights from the Data Leaders Summit Europe 2018
Veröffentlicht am 19. Oktober 2018 auf LinkedIn

Der Anfang:
Herodot 1,26-33 Kroisos und Solon

„Wenn du den Halys überschreitest, wirst du ein großes Reich zerstören.“

(1,26) Τελευτήσαντος δὲ Ἀλυάττεω ἐξεδέξατο τὴν βασιληίην Κροῖσος ὁ Ἀλυάττεω, ἐτέων ἐὼν ἡλικίην πέντε καὶ τριήκοντα, ὃς δὴ Ἑλλήνων πρώτοισι ἐπεθήκατο Ἐφεσίοισι. Ἔνθα δὴ οἱ Ἐφέσιοι πολιορκεόμενοι ὑπ' αὐτοῦ ἀνέθεσαν τὴν πόλιν τῇ Ἀρτέμιδι, ἐξάψαντες ἐκ τοῦ νηοῦ σχοινίον ἐς τὸ τεῖχος· ἔστι δὲ μεταξὺ τῆς τε παλαιῆς πόλιος, ἣ τότε ἐπολιορκέετο, καὶ τοῦ νηοῦ ἑπτὰ στάδιοι. Πρώτοισι μὲν δὴ τούτοισι ἐπεχείρησε ὁ Κροῖσος, μετὰ δὲ ἐν μέρει ἑκάστοισι Ἰώνων τε καὶ Αἰολέων, ἄλλοισι ἄλλας αἰτίας ἐπιφέρων, τῶν μὲν ἐδύνατο μέζονας παρευρίσκειν, μέζονα ἐπαιτιώμενος, τοῖσι δὲ αὐτῶν καὶ φλαῦρα ἐπιφέρων.	Nach dem Tod des Alyattes übernahm dessen Sohn Kroisos die Herrschaft, der bereits in einem Alter von 35 Jahren stand und unter allen Hellenen zuerst die Ephesier angriff. Als nun die Ephesier von ihm belagert wurden, weihten sie ihre Stadt der Artemis, nachdem sie von dem Tempel bis zur Stadtmauer ein Seil gebunden hatten; es liegen aber zwischen der alten Stadt, die damals belagert wurde, und dem Tempel sieben Stadien. Diese also griff Kroisos zuerst an, nachher aber griff er die übrigen Ionier und Aioler der Reihe nach an, indem er bei jeder Stadt einen anderen Grund vorgab, einen erheblicheren, wenn er einen solchen ausfindig machen konnte; bei einigen von ihnen nahm er auch einen ganz geringen Vorwand.

Übersetzung bearbeitet nach J.Chr.F.Bähr heruntergeladen von http://www.gottwein.de/Grie/herod/hdt01026.php am 24.10.2018

Analyse der deutschen Übersetzung

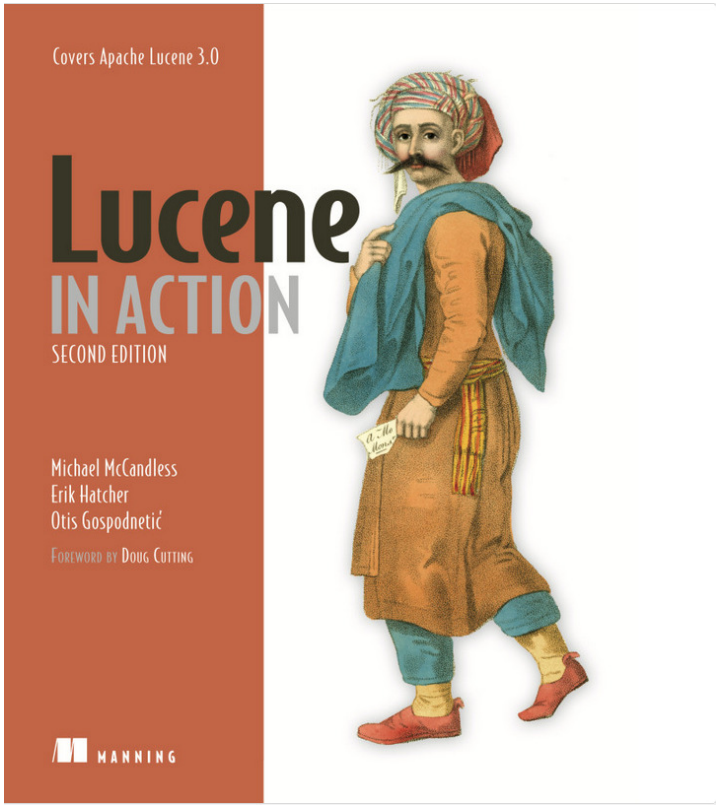
- Wörter 118
- Eindeutige Wörter 86
- „die“ ist das am häufigsten vorkommende Wort. 5 x
- Wörter mit Länge größer als 4 = 58
- Wichtige Wörter = ca. 8 – 12
- U.a. Aioler, Alyattes, Artemis, belagert, Ephesier, Hellenen, Herrschaft, Ionier, Kroisos, Stadt, Stadtmauer, Tempel …

Lucene

- Vor MongoDB, vor Cassandra, vor "NoSQL" gab es Lucene.
- Wusstest du, dass Doug Cutting 1999 die ersten Versionen von Lucene geschrieben hat?
- Um die Dinge in Zusammenhang zu bringen, war dies etwa zu der Zeit, als Google mehr ein Forschungsprojekt als eine tatsächlich vertrauenswürdige Anwendung war. Googles Proof-of-Concept-Suchmaschine war immer noch ein weitläufiger Satz von Desktop-Computern in Stanfords Forschungslabors.
- Übersetzt mit www.DeepL.com/Translator

Lucene: The Good Parts auf Parse.ly by Andrew Montalenti March 12, 2015
https://blog.parse.ly/post/1691/lucene/ heruntergeladen am 24.10.2018

Lucene in Action



- ... Lucene nähert sich den Problemen der Datenexploration aus der Sicht der "Information Retrieval", nicht aus der Sicht der "Database Management Theory". ...
- Lucene's Schöpfer erwägte: Wie unterstützen wir Abfragen, die normale Benutzer tatsächlich eingeben werden? Wie können wir schnell alle Daten, die wir haben, auf einen Schlag durchsuchen? Wie ordnen wir die Ergebnisse an, wenn es mehr als eine wahrscheinliche Übereinstimmung gibt? Wie fassen wir die gesamte Ergebnismenge zusammen, auch wenn wir nur genügend Platz haben, um einen Teil der Ergebnismenge anzuzeigen?
- Damals existierten Solr und Elasticsearch noch nicht.
- Übersetzt mit www.DeepL.com/Translator

Lucene: The Good Parts auf Parse.ly by Andrew Montalenti March 12, 2015
https://blog.parse.ly/post/1691/lucene/ heruntergeladen am 24.10.2018

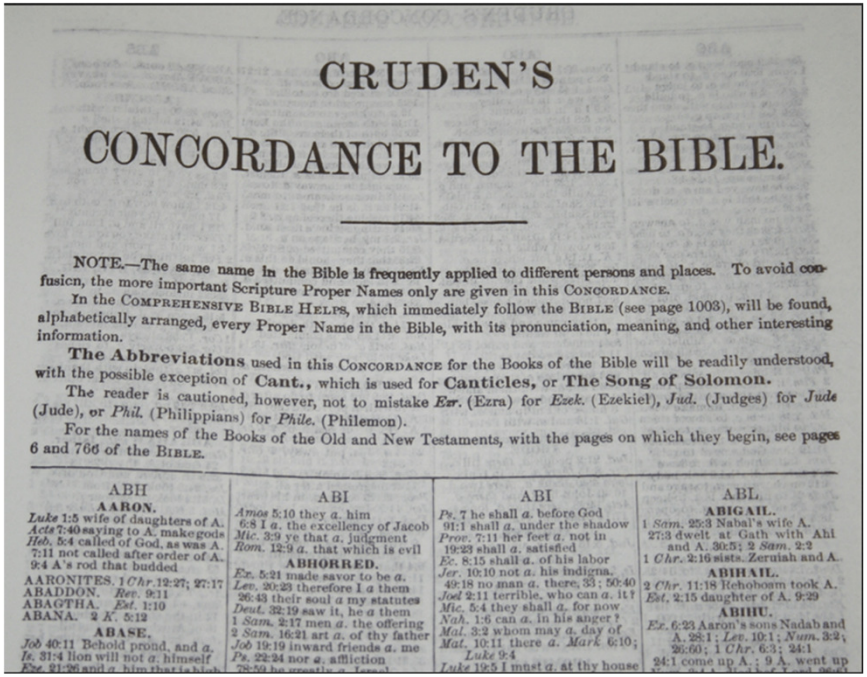
Lucene in Action: https://www.manning.com/books/lucene-in-action-second-edition

Vokabular und Corpus

- Vokabular: Der gesamte Satz eindeutiger Begriffe in einem Corpus.
- Corpus: der gesamte Satz von Dokumenten in einem Index

Übrigens, dies ist eine ziemlich alte Technik der Datengewinnung. Das erste vollständige Vokabular eines komplexen Textes wurde im Jahr 1262 von 500 sehr geduldigen Mönchen erstellt. Das fragliche Dokument war natürlich die Bibel, und der Wortschatz wurde als Konkordanz bezeichnet. Wie lautet das Sprichwort? "Es gibt nichts Neues unter der Sonne."

Lucene: The Good Parts auf Parse.ly by Andrew Montalenti March 12, 2015 https://blog.parse.ly/post/1691/lucene/ heruntergeladen am 24.10.2018



CRUDEN'S

CONCORDANCE TO THE BIBLE.

NOTE.—The same name in the Bible is frequently applied to different persons and places. To avoid confusion, the more important Scripture Proper Names only are given in this CONCORDANCE.

In the COMPREHENSIVE BIBLE HELPS, which immediately follow the BIBLE (see page 1003), will be found, alphabetically arranged, every Proper Name in the Bible, with its pronunciation, meaning, and other interesting information.

The Abbreviations used in this CONCORDANCE for the Books of the Bible will be readily understood, with the possible exception of **Cant.**, which is used for **Canticles**, or **The Song of Solomon.**

The reader is cautioned, however, not to mistake *Ezr.* (Ezra) for *Ezek.* (Ezekiel), *Jud.* (Judges) for *Jude* (Jude), or *Phil.* (Philippians) for *Phile.* (Philemon).

For the names of the Books of the Old and New Testaments, with the pages on which they begin, see pages 6 and 766 of the BIBLE.

ABH

AARON.
Luke 1:5 wife of daughters of A.
Acts 7:40 saying to A. make gods
Heb. 5:4 called of God, as was A.
7:11 not called after order of A.
9:4 A's rod that budded
AARONITES. 1 *Chr.* 12:27; 27:17
ABADDON. *Rev.* 9:11
ABAGTHA. *Est.* 1:10
ABANA. 2 *K.* 5:12

ABASE.
Job 40:11 Behold proud, and *a.*
Is. 31:4 lion will not *a.* himself

ABI

Amos 5:10 they *a.* him
6:8 I *a.* the excellency of Jacob
Mic. 3:9 ye that *a.* judgment
Rom. 12:9 *a.* that which is evil

ABHORRED.
Ex. 5:21 made savor to be *a.*
Lev. 20:23 therefore I *a.* them
26:43 their soul *a.* my statutes
Deut. 32:19 saw it, he *a.* them
1 *Sam.* 2:17 men *a.* the offering
2 *Sam.* 16:21 art *a.* of thy father
Job 19:19 inward friends *a.* me
Ps. 22:24 nor *a.* affliction

ABI

Ps. 7 he shall *a.* before God
91:1 shall *a.* under the shadow
Prov. 7:11 her feet *a.* not in
19:23 shall *a.* satisfied
Ec. 8:15 shall *a.* of his labor
Jer. 10:10 not *a.* his indigna.
49:18 no man *a.* there, 33; 50:40
Joel 2:11 terrible, who can *a.* it?
Mic. 5:4 they shall *a.* for now
Nah. 1:6 can *a.* in his anger?
Mal. 3:2 whom may *a.* day of
Mat. 10:11 there *a.* *Mark* 6:10;
Luke 9:4
Luke 19:5 I must *a.* at thy house

ABL

ABIGAIL.
1 *Sam.* 25:3 Nabal's wife A.
27:3 dwelt at Gath with Ahi
and A. 30:5; 2 *Sam.* 2:2
1 *Chr.* 2:16 sists. Zeruiah and A.

ABIHAIL.
2 *Chr.* 11:18 Rehoboam took A.
Est. 2:15 daughter of A. 9:29

ABIHU.
Ex. 6:23 Aaron's sons Nadab and
A. 28:1; *Lev.* 10:1; *Num.* 3:2;
26:60; 1 *Chr.* 6:3; 24:1
24:1 come up A.; 9 A. went up

Ein Beispiel für eine biblische Konkordanz, die etwas moderner ist als die, an der die Mönche gearbeitet haben.

Begriffserklärungen

- Hier sind einige Begriffe, die Sie in den Lucene und Information Retrieval Communities finden, die in den SQL und Datenbank Communities nicht annähernd so verbreitet sind. Lucene definiert sogar den Begriff " **term** " neu.

- **document**: ein Datensatz; die Sucheinheit; die zurückgegebene Sache als Suchergebnis ("keine Zeile")
- **field**: ein typisierter Slot in einem Dokument zum Speichern und Indizieren von Werten ("keine Spalte").
- **index** : eine Sammlung von Dokumenten, typischerweise mit dem gleichen Schema ("keine Tabelle").
- **corpus**: der gesamte Satz von Dokumenten in einem Index
- **inverted index**: interne Datenstruktur, die Begriffe nach ID auf Dokumente abbildet.
- **term**: aus dem Quelldokument extrahierter Wert, der für den Aufbau des invertierten Index verwendet wird.
- **vocabulary**: der gesamte Satz eindeutiger Begriffe in einem Korpus.
- **uninverted index**: alias "Felddaten", Anordnung aller Feldwerte pro Feld, in Dokumentenreihenfolge
- **doc values**: alternative Art der Speicherung des uninvertierten Index auf der Festplatte (Lucene-spezifisch)

Lucene: The Good Parts auf Parse.ly by Andrew Montalenti March 12, 2015
https://blog.parse.ly/post/1691/lucene/ heruntergeladen am 24.10.2018

Umkehrung unseres Corpus

- Beginnen wir mit einem einfachen Corpus von: Zwei Dokumente, doc1 und doc2, enthalten beide das Feld "tag", Typ "string", mit dem Text "big data". Es gibt auch doc3, gleiche Struktur, aber sein Tag enthält den Text „small data".
- Wie können wir mit diesem kleinen Corpus Dinge finden?
- Anstatt so zu speichern:

```
doc1={"tag": "big data"}
doc2={"tag": "big data"}
doc3={"tag": "small data"}
```

Umkehrung unseres Corpus

- Wir können den "invertierten Index" speichern. Ja, was ist das denn?

```
big=[doc1,doc2]
data=[doc1,doc2,doc3]
small=[doc3]
```

- Ah, also ist es kein Index von Dokumenten zu Begriffen, es ist ein Index von Begriffen zu Dokumenten. Schlau. Wenn wir die Daten so organisieren, können wir Dokumente schneller nach Wert finden. Wenn ich nach „big" suche, bekomme ich doc1 und doc2 zurück. Wenn ich nach „small" suche, bekomme ich doc3 zurück. Wenn ich nach „data" suche, bekomme ich alle Dokumente zurück. Dies ist im Grunde die Kerndatenstruktur in Lucene und in der Suche im Allgemeinen. Ein Hurra für den invertierten Index!

Nicht in meinem Vokabular

- In den obigen Dokumenten habe ich 3 "Begriffe", und die Annahme ist, dass ich sie durch grundlegende Whitespace-Tokenisierung generiert habe. So hatte mein ursprünglicher Korpus die Feldwerte [„big data", „small data"], aber meine generierten Begriffe sind [„big", „small", „data"].

- Das deutet bereits auf etwas Interessantes an Begriffen hin. Wenn Informationen in Ihren Feldwerten wiederholt werden, werden sie durch Herausziehen der Begriffe komprimiert.

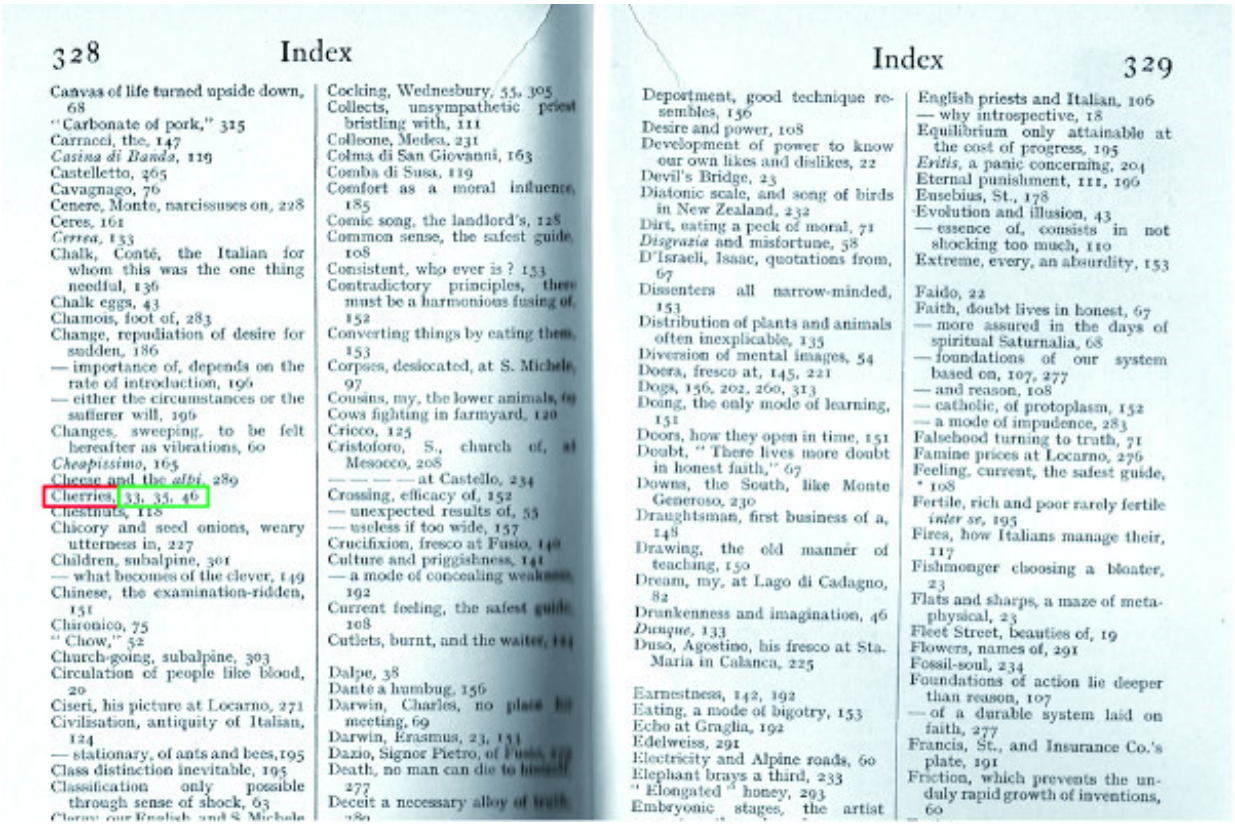
Nicht in meinem Vokabular

- Begriffe sind interessant, wenn Sie Daten haben, die sich häufig unter Ihren Dokumenten wiederholen. In diesem kleinen Beispiel wird der Begriff „data" in beiden Dokumenten wiederholt, erfordert aber nur einen Eintrag im invertierten Index. Stellen Sie sich die gleiche Art von Corpus vor wie oben, aber wenn Sie 1.000 Gesamtdokumente haben, dann ist die Hälfte mit „big data" und die andere mit „small data" versehen. In diesem Fall haben Sie vielleicht:

```
data=[1,2,3,...,1000]
big=[1,3,5,7,9,...,999]
small=[2,4,6,8,...,1000]
```

Aus „BIG DATA“ wird „SMALL DATA“

Buchindex
Bin ich Krösus, oder was?



328 Index

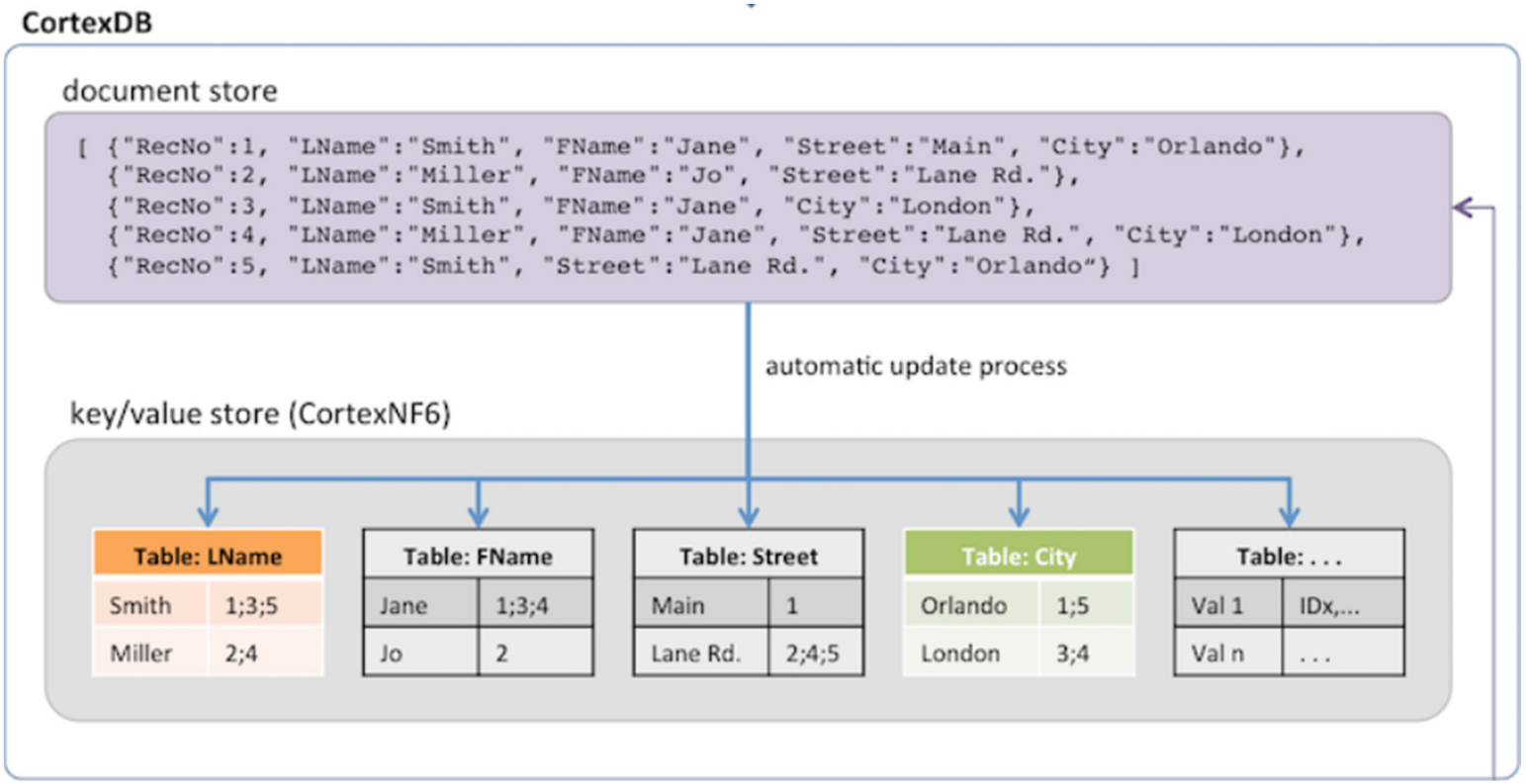
CortexDB

- Die CortexDB ist eine multi-model Datenbank, die für die Ablage von Datensätzen das Konzept eines document store nutzt, alle Felder und Feldinhalte redundanzfrei in einen mehrdimensionalen key/value-store überträgt (6. Normalform / 6NF), sowie Transaktionszeitpunkt und Gültigkeitszeitpunkt von Informationen bitemporal speichern kann. Darüber hinaus ist die ein-eindeutige Verbindung zwischen Datensätzen als Verweisstruktur möglich ("GraphDB").

?

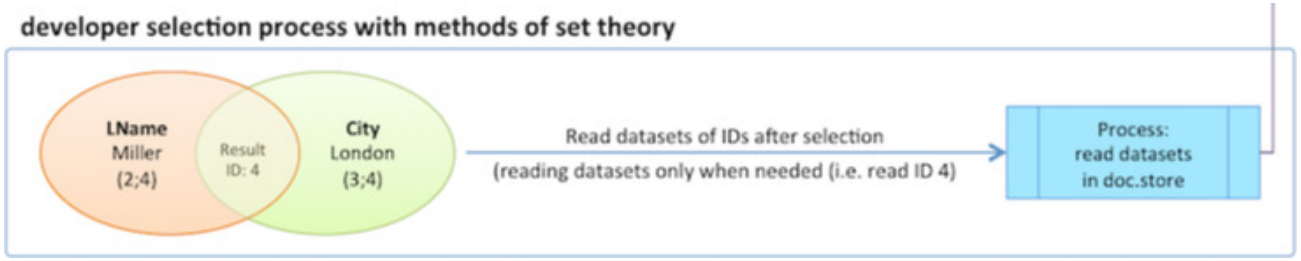
Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

CortexDB



Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

CortexDB und Abfrage



Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

CortexDB im Speziellen

- Bei der CortexDB handelt es sich um eine schemalose Datenbank (vergleichbar mit *Document Stores*), die mit einer inhaltsbasierenden mehrstufigen Indexstruktur arbeitet. Das bedeutet, dass jeder Inhalt *weiß*, in welchen Datensätzen und Feldern er vorkommt und jedes Feld *weiß*, welche verschiedenen Inhalte existieren.

Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

CortexDB im Speziellen

- Diese Indexstruktur bildet daher über alle Felder und deren Inhalte die gesamte Datenbank ab, so dass alle Abfragen in der Indexstruktur behandelt und die schemalos gespeicherten Daten nur zur Ausgabe verwendet werden. Die geringe Größe ermöglicht sehr schnelle Abfragen in beliebiger Abfragekombinationen. Administratoren und Entwickler können daher sofort mit diesem Index arbeiten und brauchen daher (auch bei Änderungen am Datenmodell) keinerlei Anpassungen oder Optimierungen vornehmen.

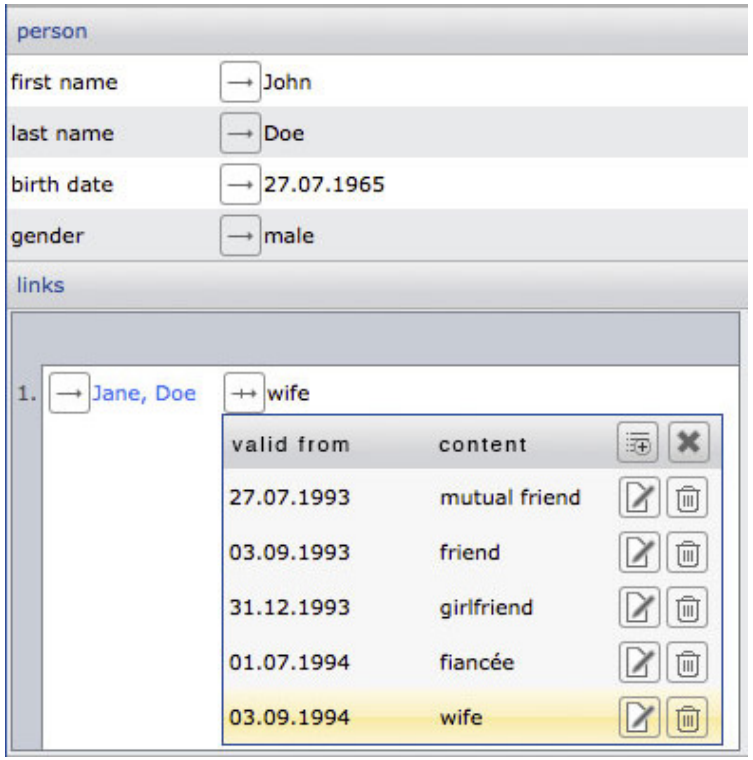
Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

CortexDB und Multi-Model

- Die CortexDB bietet die Möglichkeit, unterschiedliche Datenbank-Funktionen im Rahmen der schemalosen Struktur zu verwenden.
- So ist es beispielsweise möglich, dass über Verweisfelder Strukturen aufgebaut werden, wie sie bei Graph-Datenbanken zu finden sind. Weiterhin können einzelne Felder (optional) beliebig häufig in einem Datensatz verwendet werden (z.B. für Bankverbindungen oder eMail-Adressen); genauso können vergangene, aktuelle und künftige Werte je Feld eines Datensatzes gespeichert werden (bitemporale Datenbank). Zudem wird intern die 6. Normalform ähnlich in Anlehnung an Key/Value-Stores gespeichert (mehrdimensionaler Key/Value-Store).

Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

CortexDB und Multi-Model cortex



Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

CortexDB und Bitemporal

cortex

data 42 morrow

- Zu einem Datensatz speichert die CortexDB die Transaktionszeit von Änderungen. Ergänzend dazu kann jeder Inhalt aller Felder einzeln innerhalb eines Datensatzes mit einem Gültigkeitszeitpunkt versehen werden (Attribut-Zeitstempelung). Daher ist zu jedem Datensatz nicht nur der Zustand eines Datensatzes bei der letzten Transaktion ersichtlich, sondern auch der komplette "Lebenszyklus" (Historisierung). In diesem Fall spricht man von bitemporaler Datenbank.

- Dabei gilt, dass die Transaktionszeit automatisch (implizit) vom Server gepflegt wird und die Gültigkeit eines Wertes vom Anwender (explizit) bzw. von Automatismen wie dem Datenimport oder Schnittstellen. Wurde keine Gültigkeit gepflegt, gilt "unbekannt" für diesen Wert so lange, bis dieser mit einem neuen Wert überschrieben (eine Transaktion überschreibt diesen Wert) oder ein weiterer Wert mit einer Gültigkeit hinzugefügt wird.

Cortex AG https://docs.cortex-ag.com/de/ heruntergeladen am 24.10.2018

Identity Mapping / Record Linkage / Dynamic Links in Data Vault

- Mögliche dynamische Verbindungen werden von Algorithmen bewertet und von Maschine Learning Algorithmen bestimmt.

Das Problem

FIRMA DATENQUELLE 1	FIRMA DATENQUELLE 2
Ähnlichkeit Algorithmus GmbH	Ähnlichkeit Algorythmus GmbH
Completely Different Limited	Völlig Unterschiedlich Limited
Der ganz andere Verein eV	Der völlig andere Verein e.V.

Kein gemeinsamer Identifikator

- Es gibt mehrere aktualisierbare Stammdatenquellen (z.B. Firmendaten).
- Es gibt keinen gemeinsamen Schlüssel für einfache Joins.
- Die Datenzusammenführung muss auf der Grundlage der Zeichenkettenähnlichkeit erfolgen.
- - Name, Adresse, E-Mail, Ort, etc.
- Welcher Ähnlichkeitsalgorithmus ist zu verwenden?
- Identitätsabbildung und Deduplizierung stellen tatsächlich das gleiche Problem dar.

Dejan Sarka PASS SQL Saturday Sofia 2016

Trade-Offs

- Wie viele Daten möchten Sie automatisch abgleichen?
- Je mehr Daten Sie automatisch abgleichen, desto mehr Fehler können Sie erwarten.
- Wie viele Fehler können Sie sich mit dem automatischen Algorithmus leisten?
- Unmöglich, automatisch 100% korrekte Übereinstimmungen zu erhalten.
- Für eine Genauigkeit von 100% müssen die Data Stewards den endgültigen Abgleich manuell durchführen.

Performance Probleme

- Für ein ungefähre Zusammenführen kann jede Zeile aus einer Tabelle mit jeder Zeile aus einer anderen Tabelle verbunden werden.
- Cross Join?
- Selbst bei kleinen Datensätzen können Cross Joins zu Performance-Problemen führen.
- Daten müssen in kleineren Teilen abgebildet werden.
- Partitionierung
- Sortieren der Nachbarschaft
- Datenmenge verringern

Dejan Sarka PASS SQL Saturday Sofia 2016

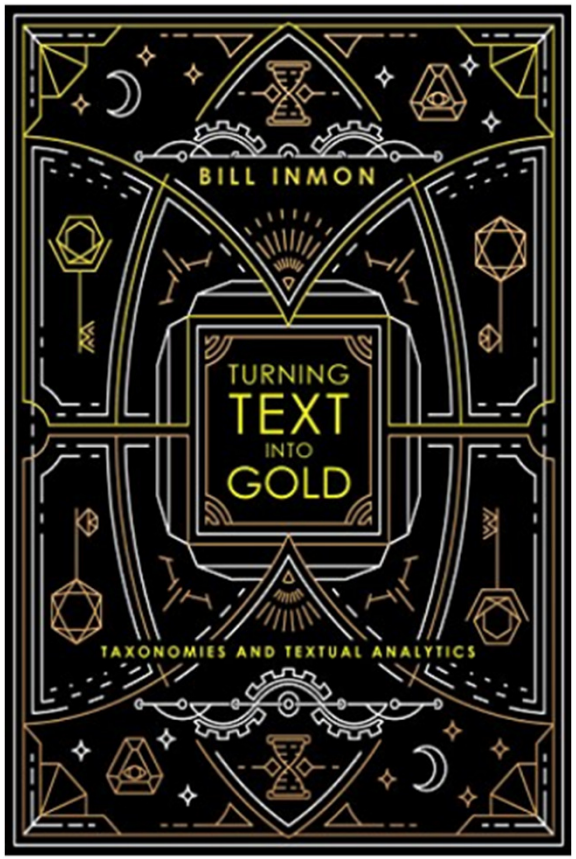
Iterativer Ansatz

- Matching in Iterationen durchführen
- Beginnen Sie mit exakten Übereinstimmungen (Inner Join)
- Dann fügen Sie ähnliche Zeichenketten zusammen.
- Dann fügen Sie weniger ähnliche Zeichenketten zusammen.
- Dann manuell zusammenführen
- Die Lösung hängt vom Werkzeug ab.
- Bereinigen Sie die Daten so weit wie möglich, bevor Sie sie abgleichen. ???

MDS Ähnlichkeitsalgorithmen

- Master Data Services implementiert einige bekannte Ähnlichkeitsalgorithmen.
- Levenshtein-Distanz (auch bekannt als Edit-Distanz)
- Jaccard-Index
- Jaro-Winkler Abstand
- Simil-Algorithmus (auch bekannt als Ratcliff/Obershelp)
- NGrams
- Implementiert in mdq.NGrams und mdq.Similarity CLR Funktionen

Empfehlung



- Es kann ein komplexer Prozess sein, um Text in ein strukturiertes Format zu konvertieren. Die Hauptschwierigkeit besteht darin, mit Unklarheiten im Zusammenhang mit dem Text umzugehen, was erfordert, dass wir verschiedene Techniken wie Homographieauflösung, Näherungsanalyse, Negativitätsableitung usw. anwenden.

Uli Bethke https://sonra.io/2018/10/11/the-future-of-etl-and-the-limitations-of-data-virtualisation-and-noetl/

Vielen Dank für ihre Aufmerksamkeit