Lernzettel - Datenbanken I und II

Kurs TIF20A

12. November 2022

Inhaltsverzeichnis

Teil I

Von Datenbanken zu Data Warehouse zu Data Lakes

1 Die „Philosophie“ von Data Lakes

1. Mit welchem Wachstum an Daten "kämpft" das durchschnittliche Unternehmen jährlich?

   Das durchschnittliche Unternehemen hat aktuell die größte Hürde in den progressiven Anstieg des Datenvolums selbst.

2. Wie viele Datenquellen gilt es dabei (im Durchschnitt) abzudecken?

   Im Durchschnitt hat ein Unternehmen rund 33 verschiedene Datenquellen zu managen, um eine Diversität der Analysen gewährleisten zu können.

3. Ist der Hauptgrund für das Investment in einen Data-Lake die damit verbundenen niedrigeren Transaktionskosten (und wenn nicht, was ist/sind die anderen Gründe?)

   Das niedrigen Invenstitionskosten stellen zwar ebenfalls ein Grund für ein Umstieg auf ein Data-Lake dar, allerdings sind die progressive Steigerung der Innerbetrieblichen Effizienz, sowie die dauerhafte Bereistellung von Daten aus diversen Abteilungen, Mainframes und Legacy Systemen vorreiter bei der Begründung.

4. Ist das Ziel eines Data-Lake die bestehenden Legacy-Systeme zu ersetzen? Erkläre Deine Antwort.

   Nein, das Ziel eines Data-Lakes ist es nicht, bestehende Legacy-System egänzlich zu erstezen.

   Das eigentliche Ziel eines Data-Lakes besteht darin, den Zustrom neuer Datentypen zu nutzen und gleichzeitig die vorhandenen Legacy-Datensysteme zu verwenden.

   Aufgrund langähriger Investitionen in ältere Datenverwaltungs-Technologien (wie zum Beispiel Data Warehouse oder Mainframe-Konzepte), kann ein Data Lake dem Unternehemen nicht nur Unternehmen dabei helfen, das Potenzial neuerer und und vielfältigeren Datentypen zu nutzen, sondern auch dazu beizutragen, diese effizienter zumachen, indem Kapazitäten auf die neuere, flexiblere Infrastruktur verlagert werden.

5. Kommt Deiner Meinung nach die erhöhte User-Satisfaction daher, dass die Unternehmen einen Data-Lake eingeführt haben, oder ist der wahre Grund die Tatsache, dass es sich (wahrscheinlich) um progressive Unternehmen handelt, die Ihren Mitarbeitern die richtigen Werkzeuge an die Hand geben wollen (und die Mitarbeiter sowieso glücklich sind, in so einem tollen Unternehmen zu arbeiten?)

   Die Einführung eines Data-Lakes bezweckt noch keinen erkennbaren Mehrwert in Bezug auf User-Satisfaction.

   Zu einer allgemeinen User- bzw- Mitarbeiter Zufriedenheit zählen noch weitere unternehmensspezifische Maßstäbe dazu. Diese können beispielsweise die Durchführung von Arbeitsprozessen, ein getaktetes Zeitmanagement, ein angenehmes Arbeitsklima oder auch managebare Workloads beinhalten.

   Zudem gehören entsprechend anderweitige Metriken zum Data-Lake selbst dazu, wie beispielsweise die Datenqualität, Schnelligkeit der Informationsbereitstellung und auch die Aussagekraft bei analytischen Auswertungen.

6. Welche 3 Metriken korrelieren am meisten mit einer guten Data Lake-Performance, bzw Ergebnissen im Unternehmen?

   - Effizient der Datenkapazität:
     Analysen können aus einer vielzahl von diversen Quellen und Datentypen erstellt werden, sodass der Zeitaufwand zur Datensuchung erheblich reduziert wird
   - Zugänglichkeit zu den Daten:
     Mit den richtigen Daten, die aus einer Vielzahl von Quellen erfasst werden, sind führende Unternehmen in der Lage, diese Informationen an Datenexperten und Entscheidungsträger weiterzugeben ohne bürokratischen Mehraufwand seitens der IT-Abteilung
   - Aktualität der Daten:
     Alle Nutzer haben ein effektives Zeitfenster, in dem die richtige Information ihre Entscheidungen beeinflussen kann. Die Fähigkeit, Informationen rechtzeitig zu erhalten, unabhängig von der Länge ihres "Entscheidungsfensters", ist entscheidend.

7. Was für Daten könnten für einen Data-Lake interessant sein, obwohl sie überhaupt nicht aus firmeneigenen Quellen stammen?

   - Wetterdaten
   - Marktanalysen mit vergleichbaren Produkten und dessen Entwicklungen von Konkurrenzunternehmen
   - Unternehmensweite Sicherheitsrisikenanalysen
   - Analysen zur allgemeinen Nutzerbedürfnissen, um die Firmeneigenen Produkte entsprechend anzupassen

8. Wenn es um Data-Lakes geht, wie siehst Du die Gewichtung zwischen "Philosophie" und "Technologie"?. Illustriere Deine Antwort mit Beispielen.

   Aus eigener Sicht handelt es sich bei einem Data-Lake zunächst um eine sehr ausgeprägte Philosophische Gewichtung, da hier das vorrangige Motto „Daten sammeln. Egal welche Daten. Egal von welchen Typ. Hauptsache sammeln.“ ist. Zu einer richtigen „Technologie“ wird ein Data-Lake subjektiv, sobald analytische Analysen daraus hervorgehoben werden. Ehe diese nicht stattfinden, existieren die Daten zwar, allerdings bekommen sie erst durch die gewollte Anwendung einen existenziellen Wert.

9. Was erwiderst Du, wenn Dir ein Anbieter seine Data-Lake-Solution mit den Hauptargumenten "Flexibel und Skalierbar" schmackhaft machen möchte? Gibt es eine bessere "Zusammenfassung" was Data-Lakes denn für einen Nutzen bringen?

   Bei einem Data-Lake sind die Argumente der „Flexibilität“ und „Skalierbarkeit“ berechtigt bei der Entscheidungsfindung zur Einführung einer Data-Lake-Solution, allerdings nicht alleinig.

   Beispielsweise nimmt die progressive Arbeitshaltung und Einstellung der Mitarbeiter gegenüber Datenverwaltung und -zugriff eine ebenso wichtige Rolle mit ein. Da Mitarbeiter mit fortschreitender Technologie auch entsprechende Ansprüche an die Daten haben, wird mittels einer Data-Lake-Solution eine Lösung eingeführt, bei dem den Mitarbeiter per se die für ihn relevante Daten vielfältiger nutzen kann.

   Hinzufügend ist ein Entscheidungskriterium der Mehrzweck für die Geschäftsleitung. Mit einem Data-Lake ist diese in der Lage jedem einzelnen Mitarbeiter verwertbare Datenergebnisse aus vorherigen Rohdaten zu liefern, sodass daraus ein erhöhter Datenfluss entsteht.

2 Die Aufgaben eines Data Lakes, die ein Data Warehouse nicht erfüllt

1. Was kann ein Data Warehouse NICHT (und wie erfüllt ein Data Lake diese Anforderung(en))?

   Ein Data Warehouse ist nicht in der Lage Daten in semistrukturierten Formaten zu speichern, da diese vor dem hochladen in ein Repository erst noch aufbereitet werden müssen für eine entsprechende weiterverarbeitung. Zudem ist die Speicherung mit hohen Kosten und zeitlichen Aufwand zu bewerten, um die Menge an rohen und unstrukturierten Dateen auch einer Vielzahl an Quellen in ein Data Warehouse zu überführen.

   Ein Data-Lake hingegen ist ein skalierbarer, kostengünstiger Speicher, der Rohdaten aus verschiedenen Quellen aufnehmen kann, um sie zu analysieren und zu verfeinern. Anschließend lassen sich Teilmengen der aufbereiteten Daten in andere Systeme verschieben, unter anderem auch in ein Data Warehouse, um sie für High-Performance- Analysen und das Reporting zu verwenden

2. Was sind die 4 Hauptanforderungen an einen Data Lake?

   - Big Data ohne hohe Kosten:

     Die unterschiedlichen Datentypen müssen in der Regel in unterschiedlichen Datenplattformen gespeichert werden, da die aktuellen Plattformen nicht in der Lage sind, eine solche Datenvielfalt zentral zu verarbeiten. So entstehen isolierte Dateninseln, sowie immer weitere komplexere Strukturen, in der Informationen „verloren“ gehen können.

   - Governance, Sicherheit und Compliance:

     Bei der Verwendung von Datenmanagement- und Data-Governance-Tools müssen strenge Zugriffrichtlinien erfolgen, Verschlüsselungen von inaktiven und aktiven Daten, sowie eine einheitliche und ausführliche Protokollierung von Datenzugriffen bzw. -änderungen.

   - Einheitliche und zuverlässige Sicht auf Daten:

     Verschiedene Bereiche benötigen eine andere Aufbereitungsweise der Daten (bspw.: Data Scientists - Rohdaten für Untersuchungen und experimentellen Zwecken, Anderweitige Organisationen - Zuverlässige Datenansicht für Analysen und Reportings)

   - Eine Lösung für alle Datennutzer:

     Immer mehr „Knowledge-Workers“ in einem Unternehmen benötigen Zugriff auf die Daten, wodurch hohe Zugriffslatenzen verursacht werden. Eine Datenmanagementlösung könnte den Anwendern einen Selfservice-Zugriff ermöglichen, allerdings sollte dieses Konzept unkompliziert, gut verwaltbar und mit einem sicheren Zugriff versehen sein.

3. Wie haben sich die Rollen von IT und Endanwendern im Kontext von Data Lakes verändert?

   In Bezug auf die Einführung von Data-Lakes stieg nicht nur die Bedeutung der IT auf die Endanwender, sondern auch dessen Aufgabenfeld. Darunter belaufen sich beispielsweise die Evaluierung, Einführung und Verwaltung von Datenmanagementtools, sowie Data-Governance Anwendungen zur Verwaltung von Nutzerrechten. Ebenso eine weiter ansteigende Rolle ist die Datensicherheit per se, sodass die Daten nicht nur gegen unbefugte Dritte zugänglich sind, sondern auch entsprechende DGVSO-Richtlinien innerbetrieblich eingehalten werden.

4. Erkläre an Hand von Beispielen die 4 Schritte zum Aufbau eines Data Lakes

   - Zusammenführung von unterschiedlichsten Datenquellen:
     - Festlegung eines Speicherortes der gesamtheitlichen Daten, wie beispielsweise der Cloud

  - Cloud-Data-Warehouse Technologien sind bereits in der Lage ein Data-Lake innerhalb eines Data Warehouses zu speichern. So entsteht ein Data-Lake, welcher diverse Daten aus unterschiedlichen Quellen aufnimmt und diese nicht einzeln isoliert zu betrachten sind
- Zugriffssicherheit durch Governance gewährleisten
  - Die Cloud-Technologie sollte dabei in der Lage sein Metdadaten mit Kontext zu versehen und als solches auch bereitzustellen. Sprich: Woher kommen die Daten?, Wer benutzt die Daten wann?, etc.
  - Eindeutige Zuweisbarkeit der Daten
  - Hierariche Zugriffsberechtigung
  - Datenverschlüsselung und Schlüsselmanagement für alle Daten
- Aufbereitung der Daten
  - Konsistenz der Daten selbst untereinander
  - Zentrale Plattform für alle Anwendungsfälle, um eine Steigerung der Produktivität im Team zu erlangen
- Ermöglichung des Selfservices
  - Zugänglichkeit der Daten
  - Scale-Out Speicher
  - Skalierung von bereits vorhandenen Rechenclustern
  - Bereitstellung von zusätzlichen Rechencluster

3 Wann brauchen wir denn nun ein Data Lake und wann ein Data Warehouse?

1. Erkläre was denn nun wirklich die Hauptunterschiede zwischen einem Data Warehouse und einem Data Lake sind.
   - Data Warehouse (Verarbeitete Datensätze):
     - Speicherung und Analyse von relationalen / strukturierten Datensätzen
     - Optimierung der Datenstruktur und des Schemas im voraus für zeitlich schnellere SQL-Abfragen
     - Zweck der Daten: Aktuell im Gebrauch
   - Data Lake (Rohdaten):
     - Zentrales Repository
     - Speicherung von unstrukturierten und strukturierten Daten
     - Speicherung des Ist-Zustandes ohne weitere Aufbereitung
     - Zweck der Daten: Noch nicht festgelegt
2. Gib Beispiele für Anwendungen in denen ein, oder die andere (oder beide) die richtige Data XXX Wahl sind.
   - Data Warehouse (Verarbeitete Datensätze):
     - Business Intelligence (BI) für Geschäftsanalytiken und dessen Visualisierungen
     - Batch Berichte

- Data Lake (Rohdaten):
  - Machine Learning
  - Vorhersagung möglicher zukünftiger Ereignisse durch prädiktive Analysen
  - Datenermittlung und -profilierung

Teil II

Key - Value: Redis

4 Prüfungsrelevante Fragen zu Dokumentenbasierten Datenbanken

1. Was ist denn nun ein Key-Value-Store genau?

   Eine Key-Value-Datenbank (Schlüssel-Werte-Datenbank) basiert auf einer Tabelle mit lediglich nur zwei Spalten (eine für den Key und eine für den entsprechenden Value). Der Schlüssel (Primary-Key) sollte dabei eindeutig sein und einen Schema zur genaueren Identifikation folgen.

   Key-Value-Stores stellen In-Memory-Datenbanken (im Arbeitsspeicher) und On-Disk-Lösungen (im Festplattenspeicher) dar.

2. Nenne 3 verschiedene Key-Value Datenbanksysteme

   - Amazon DynamoDB
   - Redis
   - Riak

3. Was ist der Unterschied zu einer Document-Database?

   Ein Key-Value-Store ist speziell für eine Anwendung angedacht und kann zu einer Document-Database umgewandelt werden. Der Unterschied zwischen diesen beiden Databases liegt darin, dass Querys innerhalb einer Dokumentenbasierten-Datenbank ebenfalls in der Value-Spalte möglich ist. Eine Suchabfrage innerhalb der Schlüssel-Wert-Abfrage ist hierbei lediglich auf den Key beschränkt, wodurch der entsprechende Value ausgegeben wird.

4. Bezüglich des "C" in ACID - was ist der Hauptunterschied zu relationalen Datenbanken?

   C = Consistency: Relationale Datenbanken benötigen immer einen eindeutigen Wert, welcher in einen definierten Zustand vorliegen muss, damit die definierten Bedinungen entsprechend erfüllt sind.

   In einer Key-Value-Datenbank müssen die Values keinen Schema befolgen, wodurch mehrere Informationen in der gleichen Spalte speicherbar sind.

5. Welche Normalformen gibt es bei Key-Value-Datenbanken?

   Innerhalb einer Key-Value-Datenbank kann es zu keiner Normalform kommen, da eine entsprechende Normalisierung nur bei relationen Datenbanken durchgeführt werden kann, damit eine eventuell vorliegende Dateninkonsistenz vermieden wird.

6. Welche Datentypen sind akzeptabel als Key? Was ist der Hauptindex für Key-Value Datenbanken, welchen Datentyp hat er, und wie viele andere Indices kann eine KV-Datenbank haben?

   - Ein einzelner Key kann jeglichen Datentyp annehmen, allerdings ist dies abhängig von der Einschränkung, welche von der verwendeten Datenbank-Software auferlegt wird
   - Als Hauptindex in einer Key-Value-Datenbank dient der Primary-Key (Schlüssel), welcher wie bereits aufgeführt jeglichen Datentyp annehmen kann
   - Weitere Indizes sind im Rahmen einer Key-Value-Datenbank nicht notwendig, da hinter einem Key bereits alle zugehörigen Informationen als String, Integer, Dokument, BLOB-Datei, oder anderweitigen Datentypen hinterlegt werden

7. Angenommen wir haben eine Studentendatenbank (Key-Value), wo der Key die Matrikelnummer ist, und der Value die wichtigen Daten wie Name und Lieblingsbier. Mit welcher Suchmöglichkeit kann man alle Studis finden, die gerne Guinness trinken?

   Die einzige Suchmöglichkeit besteht in der Abfolge entsprechenden Key-Value-Pairs. Eine spezifische Abfrage innerhalb der Value-Spalte ist in einer Schlüssel-Werte-Datenbank nicht möglich, sodass alle Keys einzeln abgefragt und die dazugehörigen Values ausgegeben werden müssen. Daraus ableitend lässt sich eine aufwändige manuelle Ergebnissführung aufschlüsseln.

8. Warum bietet sich bei einem Online-Warenkorb (mit großen Umsatzzahlen natürlich, wie z.B. Amazon) eher eine Key-Value Datenbank an um den Inhalt des Warenkorbs zu speichern, anstatt einer relationalen Datenbank? "Weil es schneller ist" reicht nicht als Antwort. Sei ganz spezifisch!

   - Würde für ein Online-Warenkorb eine relationale Datenbank verwendet werden, so würde dies einen deutlich merkbaren Mehraufwand bei der Erstellung und Aktualisierung des Datenbankschemas (zunehmend bei höherer Normalform), sowie eine langsame Datenabfrage seitens der Kunden mit sich ziehen. Der Warenkorb selbst würde dabei lediglich eine Hilfstabelle mit entsprechenden Fremdschlüssel abbilden, welche eine Referenz auf jeden enthaltenen Eintrag mit einen eindeutigen Primärschlüssel darstellen würde. Die Zugriffszeit bei einen einzigen User wäre dabei nicht Auffällig, wohingegen bei beispielsweise Millionen zeitgleichen Zugriffen, dies einen erheblichen Unterschied wiedergeben würde
   - Dahingegen werden mit der Nutzung einer Key-Value-Datenbank einfache Strukturen realisiert, welche mit nur einen Block von eindeutigen Primary-Keys repräsentiert werden können, welche für weiterführende Informationen ohne Abschweifungen auf relationalen Tabellen zur Verfügung stehen. Bei Abfragen sind beispielsweise auch keine *Join-Befehle* von Nöten

   Heutzutage viele Daten die keiner einheitlichen Struktur folgen (Bsp. Tweets)

   Keine Placeholder, wie „null“ erforderlich (BSp. für Artikel Bilder)

9. Ist Windows Explorer eine Key-Value-Datenbank?

   Nein, der Windows Explorer stellt weiterhin ein voreingestelltes Dateimanagementsystem von Microsoft Windows dar, welches eine solche Datenbank nutzen könnte, aber keine Key-Value-Datenbank darstellt.

10. Einer der Artikel behauptet: "key-value stores are not considered suitable for applications requiring frequent updates". Ist das nicht genau das Gegenteil von der Aussage in Aufgabe 8 (Warenkörbe, Session-Daten, usw..). Diskutiere, was hier abgeht/wer Recht hat/was das eigentliche Problem ist!

    Key-Value-Stores finden keinen sinnvollen Einsatz, wenn komplexe und vorallem regelmäßige Querys erfolgen, da hinter jedem Key ein unterschiedlich großer BLOB-Value mit allen gesammelten Daten

zu diesen Key entsprechend hinterlegt ist. Bei einem Update von einen Datensatz innerhalb eines Values, muss der gesamte Value-Block aktualisiert werden.

Das eigentliche Problem besteht darin, genauer zu definieren, was man genau unter „requiring frequent updates“ zu verstehen ist. Handelt es sich beispielsweise um eine Webanwendung zur Online-Ticketbuchuung ist ein sehr häufiges und wiederkehrendes Updaten der Page bei der Eröffnung des Ticketverkaufes möglich, allerdings wird hierbei nicht der Dateninhalt der hinterlegten Web-Datenbank verändert, sondern lediglich bereits vorhandene Daten zu einem bestimmten Zeitpunkt freigeschaltet. Hierfür wäre eine Key-Value-Datenbank weiterhin empfehlenswert. Handelt es sich allerdings um überschaubare Datensätz, wie beispielsweise innerhalb einer Kundendatenbank, empfiehlt sich eine relatione Datenbank, da neben der Vermeidung von Dateninkonsistenz, eine Aktualisierung auf einzelne Datensätze erfolgt.

11. Nenne 3 verschiedene Anwendungen wo Key-Value-Datenbanken Vorteile gegenüber Relationalen Datenbanken haben, und beschreibe spezifisch für jeden Fall, was genau der Vorteil ist (und: was potentielle Nachteile sind)

    - Session-Daten
      - Vorteil: Schnelle Lese- und Schreibgeschwindigkeit, wenn beispielsweise der Key das entsprechende Session-Datum und der Value die aufgerufenen Webseiten darstellt
      - Nachteil: Eingeschränkte Abfragemöglichkeit, wenn nach einer aufgerufenen Webseite gesucht wird, das Session-Datum allerdings nicht vorhanden ist
    - Online Shopping-Profil mit persönlichen Präferenzen
      - Vorteil: Flexible Skalierbarkeit bei Hinzufügen neuer Profilinformationen (Key = Profil-Identifikationsnummer und Value = Zugehörige Profilinformationen), sowie schnelle Schreibgeschwindigkeit neuer Profilinformationen in den Key-Abhängigen Value-Block
      - Nachteil: Benötigter Mehraufwand bei der Aktualisierung, da nicht nur ein Value-Wert aktualisiert wird, sondern der gesamte Value-Block
    - Multimedia Storage
      - Vorteil: Speicherung und Hinterlegung als BLOB-Datei innerhalb des Value-Blocks möglich (Innerhalb einer relationalen Datenbank muss eine Auslagerung dieser vorgenommen werden mit einem Pointer auf die Speicherstelle in der Tabelle selbst)

12. Be- oder widerlege die folgende Aussage (idealerweise an Hand eines Beispiels) "Dokumentendatenbanken sind das Gleiche wie Kev-Value Datenbanken"

    Eine Dokumentenbasierte-Datenbank ist eine weiterführung der Key-Value-Datenbank, wobei ein Dokument im Zusammenhang für einzelne, aber in sich unterschiedliche strukturierte Einheiten steht. Hierbei ist eine (komplexere) Abfrage im Value-Block möglich.

    Beispiel: Datenbank mit studentischen Informationen. welche als JSON-Format hintelregt sind-Key = Martikelnummer und Value = Name und Lieblingsmodul. Eine Abfrage kann nun ausgeben, welche Studenten das Lieblingsmodul Datenbanken haben.

13. Erkläre an Hand eines Beispiels was die Auswirkungen sind, wenn man im laufenden Betrieb die Feldstruktur einer KV-Datenbank verändert, und vergleiche es mit dem, was analog bei einer relationalen Datenbank passieren würde.

    - Da der Key-Value-Store kein einheitliches Schema verlangt oder vorgibt, lassen sich Änderungen an der Datenbank im laufenden Betrieb vornehmen. So kann man ein neues Feld einführen, während gleichzeitig Aktionen in anderen Einträgen erfolgen.

- Im Rahmen einer relationen Datenbank muss diese vorerst abgetrennt werden vom laufenden Betrieb, um die vorgenommene Änderung wirksam zu machen

Teil III

Theoriefragen aus den Jupyter Notebooks

5 PyMongo

1. Warum ist die *aggregation* einer Pipeline langsamer als der *find*-Befehl?
   - *find* = Iterator (Lazy-Evalutation), d.h. es wird nur der aktuelle Eintrag betrachtet
   - *aggregation* = Jeder Eintrag wird in eine Liste hinzugefügt, wobei die Liste jedesmal um einen Eintrag erweitert wird und somit immer umfangreicher und größer wird

6 Schema Design

1. Beispiele für *embedding*, *referencing* und *hybrid*
   (a) **Embedding**
      - Buchung und Speicherung von Projektzeiten (jedes Projekt stellt eigenes Dokument dar)
      - User in einem Helpdesk-System (jeder User stellt eigenes Dokument dar)
      - GPS-bezogene Wetterdaten mit Zeitstempel (jeder GPS-Standort stellt eigenes Dokument dar)

   (b) **Referencing**
      - Kundendaten mit Referenz auf Persönliche Daten, Freigabeberechtigungen, Bestellungen / Rechnungen, Aufträge, etc.
      - Mitarbeiterdaten
      - SAP-basierte Systeme

   (c) **Hybrid**
      - Netzwerkanalyse in Grafana (Daten aus definierten Zeitraum werden in eigenes Dokument gelegt und ältere Daten referenziert)
      - Inventarverwaltung (Auflistung des gesamten Lagerbestandes mit genaueren Information zu jedem Artikel in ausgelagerten Dokumenten) - Messdaten einer Produktionsmaschine (vgl. Hybrid-Prinzip Netzwerkanalyse)

2. Arten von Referenzen
   (a) Manual references
      - Speichert Primary-Key (typischerweise *_id*-Feld) in ein anderes Dokument, um dieses als Referenz zu verwenden
      - Anwendung führt eine zweite Abfrage aus, um das referenzierte Dokument zurückzugeben - Referenzen sind simple und ausreichend für die meisten Use-Cases

- Verwendung innerhalb einer Collection / Datenbank

(b) DBRefs

- Referenzen von einen Dokument zu einem anderen unter Verwendung von Feldern des ersten Dokumenten
- Bspw. *_id*, Collection Name, Datenbank Name, etc.
- Verwendung über mehrere Collections / Datenbanken hinaus

Teil IV

Sonstiges

7 Umrechnung

8 Bit	1 Byte
1024 Byte bzw. 1000 Byte	1 KiB bzw. 1 KB
1024 KiB bzw. 1000 KB	1 Mib bzw. 1 MB
1024 MiB bzw. 1 MB	1 GiB bzw. 1 GB

8 Rechenbeispiel

16 MiB = 16 * 1024 KiB = 16 * 1024 * 1024 Bytes = 16 * 1024 * 1024 * 8 Bit = 134217728 Bit

Annahme (#Design): 32 Bit Floats

134217728 / 32 = 4194304 Datenpunkte

Annahme (#Design): Datenpunkte alle 10 Sek.

4194304 / 0,1 = 41943040 Sekunden

41943040 Sek / 60 = 699050 Minuten 699050 Minuten / 60 = 11650,8 Stunden

11650,8 Stunden / 24 Stunden ~ 485 Tage

9 Datentypen

Typname	Größe	Wertebereich
Boolean	1 bit	true / false
char	16 bit	$\sim 0...65.535$
byte	8 bit	$-128...127$
short	16 bit	$\sim -32.768 - 32.767$
int	32 bit	$\sim -2.147.483.648... + 2.147.483.647$
long	64 bit	$-2^{63}...2^{63}$
float	32 bit	$\pm 1,4^{-45}... \pm 3,4^{38}$
double	64 bit	$\pm 4,9^{-324}... \pm 1,7^{308}$