Karlsruher Institut für Technologie

Datenbanksysteme

Kapitel 11: Datenbank-Funktionalität in der Cloud – ausgewählte Aspekte (Teil 2)

Lehrstuhl für Systeme der Informationsverwaltung, Fakultät für Informatik

Photo by Pattys-photos

Photo by Futurilla

Dynamo Systemarchitektur

- Übersicht (gemäß der hinten genannten Quelle):

Problem	Technik	Vorteil
Partitionierung	Consistent Hashing	Skalierbarkeit, inkrementell
Hohe Verfügbarkeit für das Schreiben	Vector Clocks mit Abgleich beim Lesen	
Umgang mit vorübergehenden Ausfällen	Sloppy Quorum mit hinted handoff	Hohe Verfügbarkeit und Dauerhaftigkeit
Recovery	Anti-Entropy	Synchronisation läuft im Hintergrund ab.
Erkennen von Ausfällen	Gossip-basierte Protokolle	Deckt Anforderung ‚Symmetrie' ab.

  - Nur Auswahl der Probleme/Features.
  - Ausführlichere Behandlung der Themen in weiterführender LV.

Grundlagen
Dynamo
- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.
Begrenzung
Erg.größe
Schluss

Partitionierung

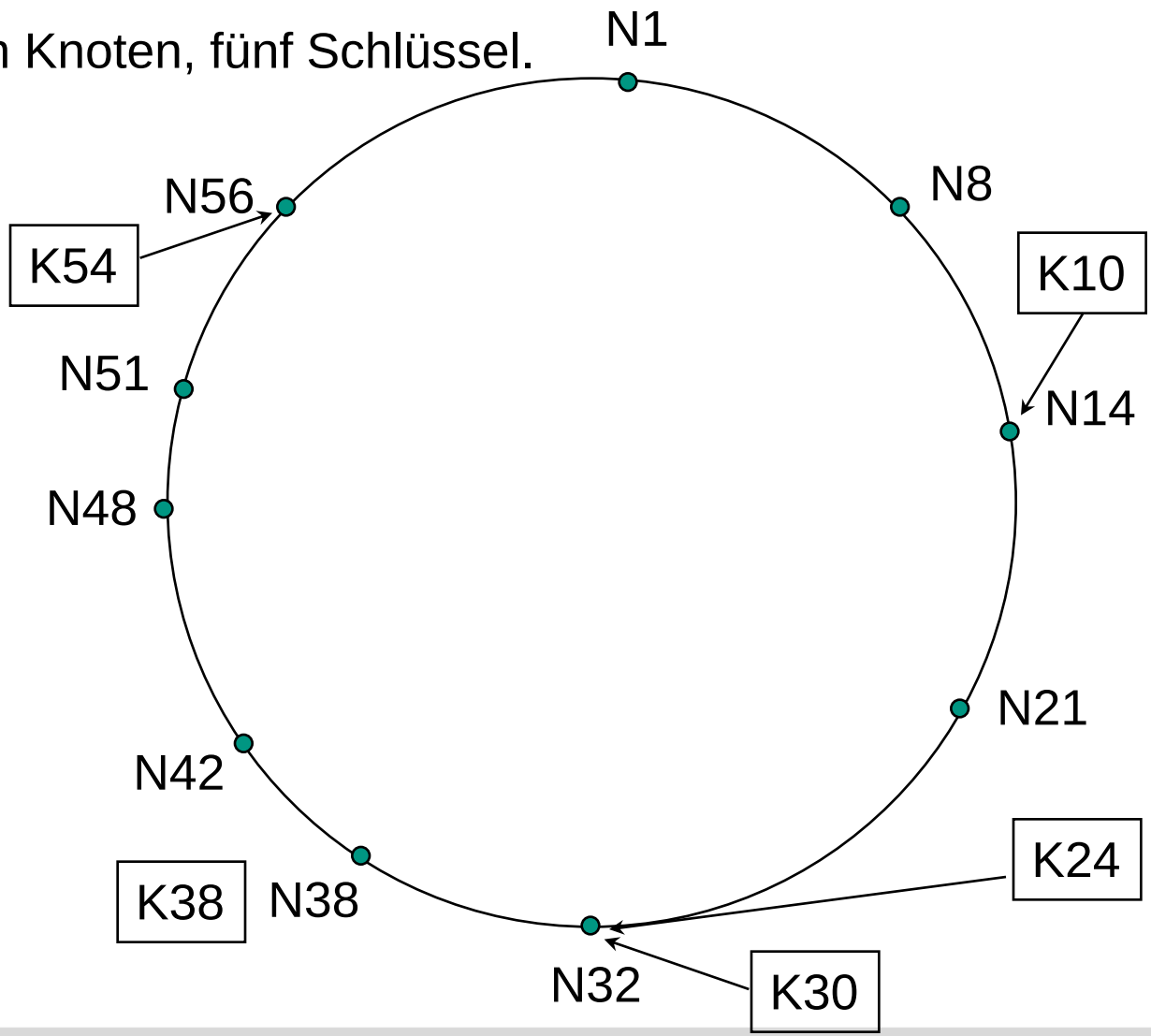
- Ermittlung des Knotens, der Objekt speichert, im Prinzip wie bei Chord.
- Hinzukommen/Ausscheiden eines Knotens betrifft nur seine Nachbarn im Ring.
- Lastverteilung, indem Objekt auf mehrere Positionen im Ring abgebildet wird. (Z. B. grundsätzlich nicht nur eine Hashfunktion, sondern mehrere unterschiedliche.)
- Auch jeder Knoten entspricht mehreren Positionen im Ring. (“Virtuelle Knoten”)
  - Illustration auf folgender Folie.
  - Bessere Lastverteilung, wenn Knoten ausfällt/hinzukommt.

Grundlagen
Dynamo
- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.
Begrenzung
Erg.größe
Schluss

Identifier Circle

Zehn Knoten, fünf Schlüssel.

m=6



Grundlagen

Dynamo

- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.

Begrenzung Erg.größe

Schluss

Vector Clocks (1)

- Um Eventual Consistency zu erreichen,
  sowohl Vektor-Uhren als auch Quorum-basierter Ansatz.

- Zusammenhang zwischen Quorum-basierten
  und Vector-Clock-basierten Ansätzen:
  - Quorum-basierte Techniken vermeiden Inkonsistenzen.
  - Vector-Clock-basierte Techniken erkennen Inkonsistenzen
    und helfen, sie aufzulösen.

Grundlagen

Dynamo

- Motivation

- Entwurfs-
  entscheid.

- Strukt.
  P2P-
  Systeme

- System-
  arch.

Begrenzung
Erg.größe

Schluss

Vector Clocks (2)

- Vector Clocks – Technik zur Erfassung der Zusammenhänge zwischen Versionen.
- Liste von (Knoten, Zähler)-Paaren. Eine Liste für jede Version.
- Unterschiedliche Knoten können Schreiboperationen absetzen. Hier Differenzierung.

Grundlagen

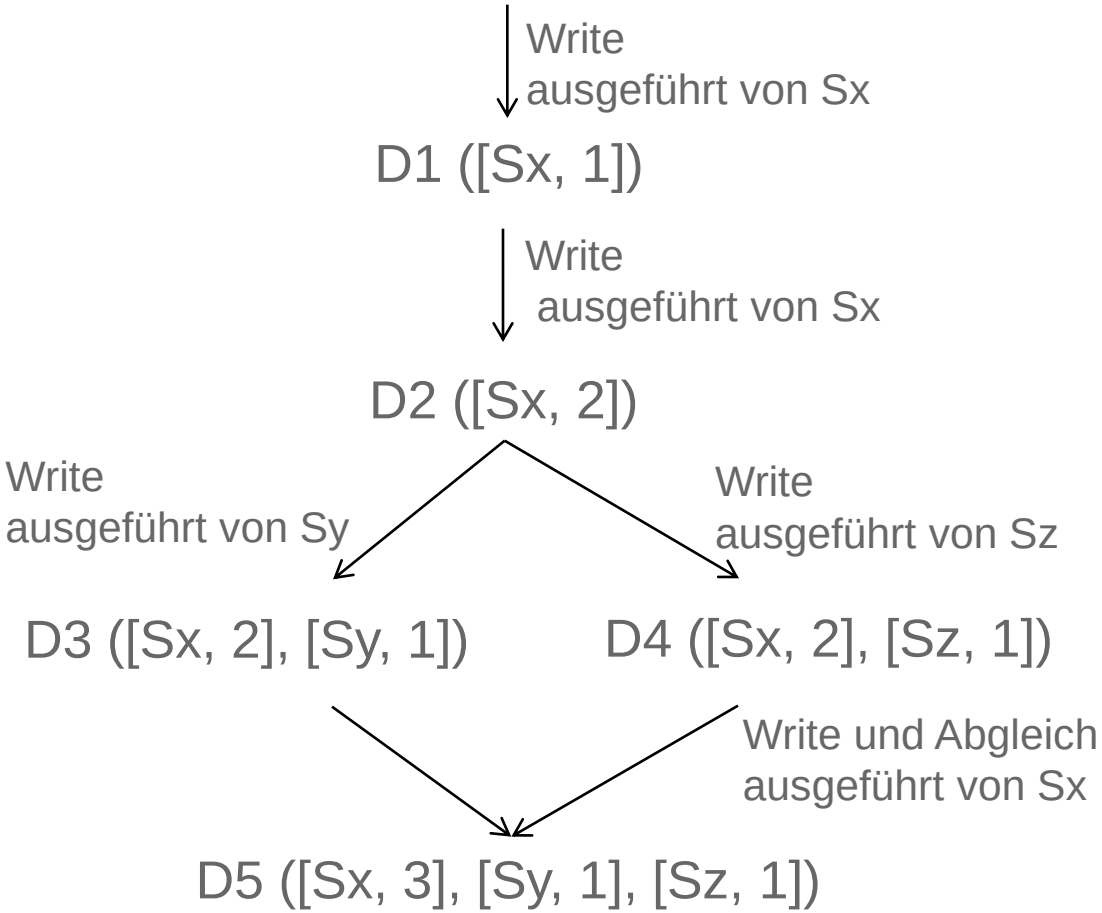
Dynamo

- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.

Begrenzung Erg.größe

Schluss

Vector Clocks – Beispiel



Grundlagen

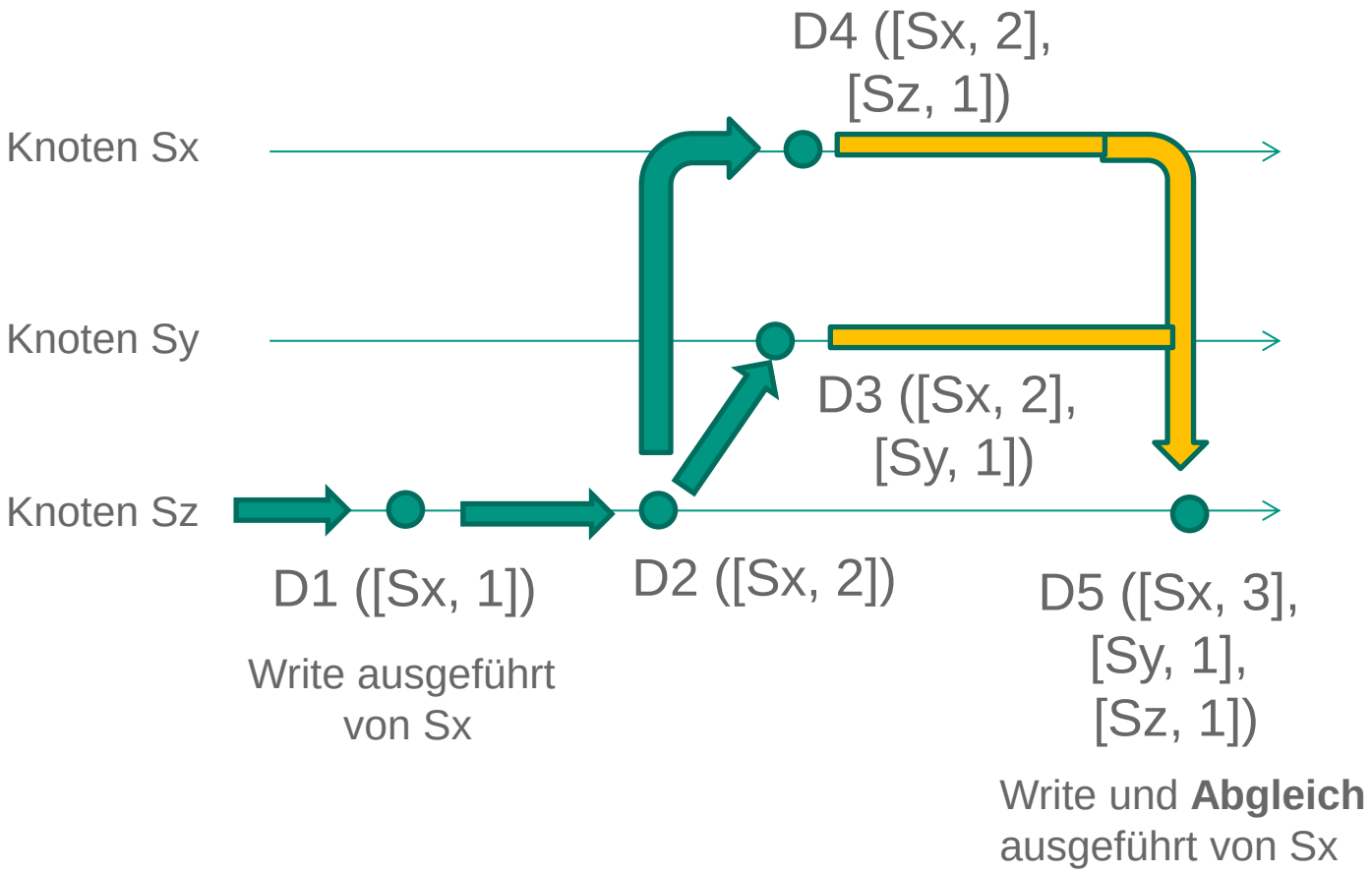
Dynamo

- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.

Begrenzung Erg.größe

Schluss

Vector Clocks – Beispiel



Grundlagen

Dynamo

- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.

Begrenzung Erg.größe

Schluss

Vector Clocks – Anmerkungen zum Beispiel

- Operation D5, die D3 und D4 liest, kann neue, einheitliche Version erstellen.
- Garbage Collection von D1 und D2 nach der Erstellung von D3 und D4 möglich. (Erkennbar anhand der Vector Clocks.)

Grundlagen

Dynamo

- Motivation

- Entwurfs-entscheid.

- Strukt. P2P-Systeme

- System-arch.

Begrenzung Erg.größe

Schluss

Vector Clocks (3) - Partielle Ordnung

- Welche Version hängt von welcher kausal ab?
- Version A ist Vorgänger von Version B, wenn
  - der Zähler für jeden Prozess im Zeitstempel C(A) kleiner oder gleich dem Zähler im Zeitstempel C(B) ist
  - und für mindestens einen dieser Zähler strikt kleiner ist.
- Formal

$$C(A) = (A_1, A_2, A_3 \ldots A_n)$$
$$A \rightarrow B \Leftrightarrow \forall_{1 \leq i \leq n} : A_i \leq B_i \wedge \exists i' : A_{i'} < B_{i'}$$

Grundlagen
Dynamo
- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.
Begrenzung Erg.größe
Schluss

Vector Clocks (4)

- Update (put) muss festlegen,
  welche Versionen er aktualisieren will.
- Dies geht über den sogenannten *Kontext*;
  dies ist Parameter der put-Operation.
  Kontext enthält Vector Clocks.
- Illustration:
  - Erstellen von D5 in Beispiel von eben:
  - System muss wissen, dass D3 und D4 überschrieben werden.
  - Deshalb Vector Clocks von D3 und D4
    als Parameter der Schreiboperation.
- ‘get’ gibt i. Allg. mehrere Versionen zurück.

Grundlagen
Dynamo
- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.
Begrenzung Erg.größe
Schluss

Ausführung von get- und put-Operationen (1)

- Quorum-basierte Techniken
  - garantieren i. Allg. Konsistenz im verteilten Fall.
  - Sie skalieren jedoch nicht.
- Deshalb sind auch Vector-Clock basierte Techniken erforderlich.

Grundlagen

Dynamo

- Motivation

- Entwurfs-entscheid.

- Strukt. P2P-Systeme

- System-arch.

Begrenzung Erg.größe

Schluss

Ausführung von get- und put-Operationen (2)

- Hier nur fehlerfreier Fall.
- Konsistenz mit Hilfe von Quorum-basiertem Protokoll:
  - Lesen:
    Man liest Mindestanzahl von Versionen (R)
    und nimmt die aktuellste.
  - Schreiben:
    Man aktualisiert Mindestanzahl von Kopien (W).
- Üblich ist ist R+W>N.
- Hier kleinere Werte für R und W, wegen besserer Latenz.
- Sogenanntes “sloppy quorum”.

Grundlagen
Dynamo
- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.
Begrenzung Erg.größe
Schluss

Zusammenfassung bis hierhin

- Erläuterung, was Dynamo ist, und wie es aufgebaut ist.
- Keine Datenbanktechnologie im klassischen Sinn.
  Nicht vergleichbar. An manchen Stellen besser,
  an anderen schlechter.
  - Schlechter: ‚Niedrige' Schnittstelle für Anwendungsentwicklung.
  - Besser: Nichtfunktionale Eigenschaften,
    insbesondere Skalierbarkeit, Zuverlässigkeit, Performance.
- Rest des Kapitels:
  - Datenbanktechnologie 'on top' von Dynamo.
  - Halbwegs aktueller Ansatz aus der Forschung.
  - SQL-Anfragen: Ja, aber nicht in voller Schönheit.
    Beschränkungen halten sich aber in Grenzen.

Grundlagen
Dynamo
- Motivation
- Entwurfs-entscheid.
- Strukt. P2P-Systeme
- System-arch.
Begrenzung
Erg.größe
Schluss

Unvorhersehbare Zahl von Anfragen (1)

- Beispiel:
  - Neuer Dienst/neues Informationsangebot im Web, Datenbank-gestützt.
  - Bestimmte Ereignisse lassen Zugriffszahlen explodieren.
  - Aber: I. Allg. dramatische Auswirkungen auf Antwortzeiten/Verfügbarkeit des Dienstes, zumindest kurzfristig.
  - Erfolg wird zum Fluch: Benutzer sind erfahrungsgemäß sehr empfindlich und wandern ab.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Unvorhersehbare Zahl von Anfragen (2)

- Physische Datenunabhängigkeit
  hier möglicherweise nachteilig:
  - Anwender formuliert Anfragen,
    die zu teuren Operationen führen,
    ohne das zu merken.
  - Probleme werden oft erst unter hoher Last sichtbar
    (wenn es zu spät ist).

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Scale Independence

- *Anfrage ist scale-independent* :=
  ihr Laufzeitverhalten ist unabhängig
  von der Größe der Datenbank.
- Im Folgenden wird ein halbwegs aktueller Ansatz
  (einer Gruppe aus Berkeley) vorgestellt.
- Scale Independence wird im Folgenden erreicht durch
  a) Erweiterungen und Beschränkungen der Anfragesprache
     (PIQL, „Performance-Insightful Query Language“)
  b) Überprüfungen beim Übersetzen der Anfrage.
  c) Erweitertes Abschätzen der Antwortzeit.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Klassifikation von Anfragen gemäß ihres Aufwands (1)

- Klasse I („konstant“)
  - Beispiel: Zugriff auf ein Produkt oder auf einen Benutzer, ID-basiert.
  - Für Compiler i. d. R. erkennbar als Zugriff auf einen Schlüssel.
  - Was noch für Beispiele?

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Klassifikation von Anfragen gemäß ihres Aufwands (1)

- Klasse I („konstant“)
  - Beispiel: Zugriff auf ein Produkt oder auf einen Benutzer, ID-basiert.
  - Für Compiler i. d. R. erkennbar als Zugriff auf einen Schlüssel.
  - Manche Anfragen mit LIMIT, ohne Join.
  - Anfrage, die
    - Tupel anhand des Schlüssels identifiziert,
    - dann Fremdschlüssel-Joins vornimmt.
  - Pagination.

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Paginierung Illustration

Adressbuch

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Aachen, Anton
Aaron, Tanya
Berger, Johannes
Berthold, Paul
Christian, Dieter
Christian, Mareike
Dietrich, Richard
Ernst, Thomas

Seite 1 | Seite 2 | Seite 3 | … | Seite 100

PIQL Spracherweiterungen & Beschränkung der Ergebnisgröße (2)

- Wenn Join Fremdschlüssel enthält,
  ist Zwischenergebnis so groß wie eine Eingabe-Relation.
- Veranschaulichung:

Ausleih

Invnr	Name
4711	Meyer
1201	Schulz
0007	Müller
4712	Meyer

Bücher

Invnr	Titel	ISBN	Autor
0007	Dr. No	3-125	James Bond
1201	Objektbanken	3-111	Heuer
4711	Datenbanken	3-765	Vossen
4712	Datenbanken	3-891	Ullman
4717	Pascal	3-999	Wirth

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Klassifikation von Anfragen gemäß ihres Aufwands (2)

- Klasse II („beschränkt“)
  - Beispiel:
    - Es existiert explizite Begrenzung der Anzahl Freunde bei Facebook (5000).
    - Als Kardinalitätsangabe in erweitertem Datenbank-Schema darstellbar.
    - Beispiel für Anfrage also: „Selektiere die Freunde von …“.

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Klassifikation von Anfragen gemäß ihres Aufwands (3)

- Klasse III („linear bzw. sublinear“)
  Beispiel: Ausgabe aller Kunden oder Produkte.
- Klasse IV („super-linear“)
  Beispiel: Clustering-Algorithmus, der Self-Join der zugrundeliegenden Relation ausführt.
- Ziel:
  - Nur Klassen I und II.
    M. E. legitim für Web-basierten Zugriff.
  - Bezüglich Klasse II Generierung von Hinweisen, wie Kardinalitätsangaben aussehen sollten, um Service-Level Objectives einzuhalten.

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

(Key, Value)-Speicherung

MatrNr	Name	GebJahr
123	Böhm	1977
456	Ballack	1977
789	Lahm	1983

- Darstellung mit (Schlüssel, Wert)-Paaren, z. B.:
  - MatrNr → Tupel als Ganzes
  - Name → Menge der entsprechenden Matrikelnummern, als BLOB
  - (456, “456 Ballack 1977“)
  - (Böhm, “123”)
  - (1977, “123 456”)

„Index-Einträge“

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Zugrundeliegende Architektur – Beispiel

- Input:

```
SELECT Name
FROM Player
WHERE GebJahr == 1977
```

- Output:
  - Erster Zugriff auf 'Index' mit Schlüssel 1977.
  - Dann Zugriffe mit Schlüsseln 123, 456.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

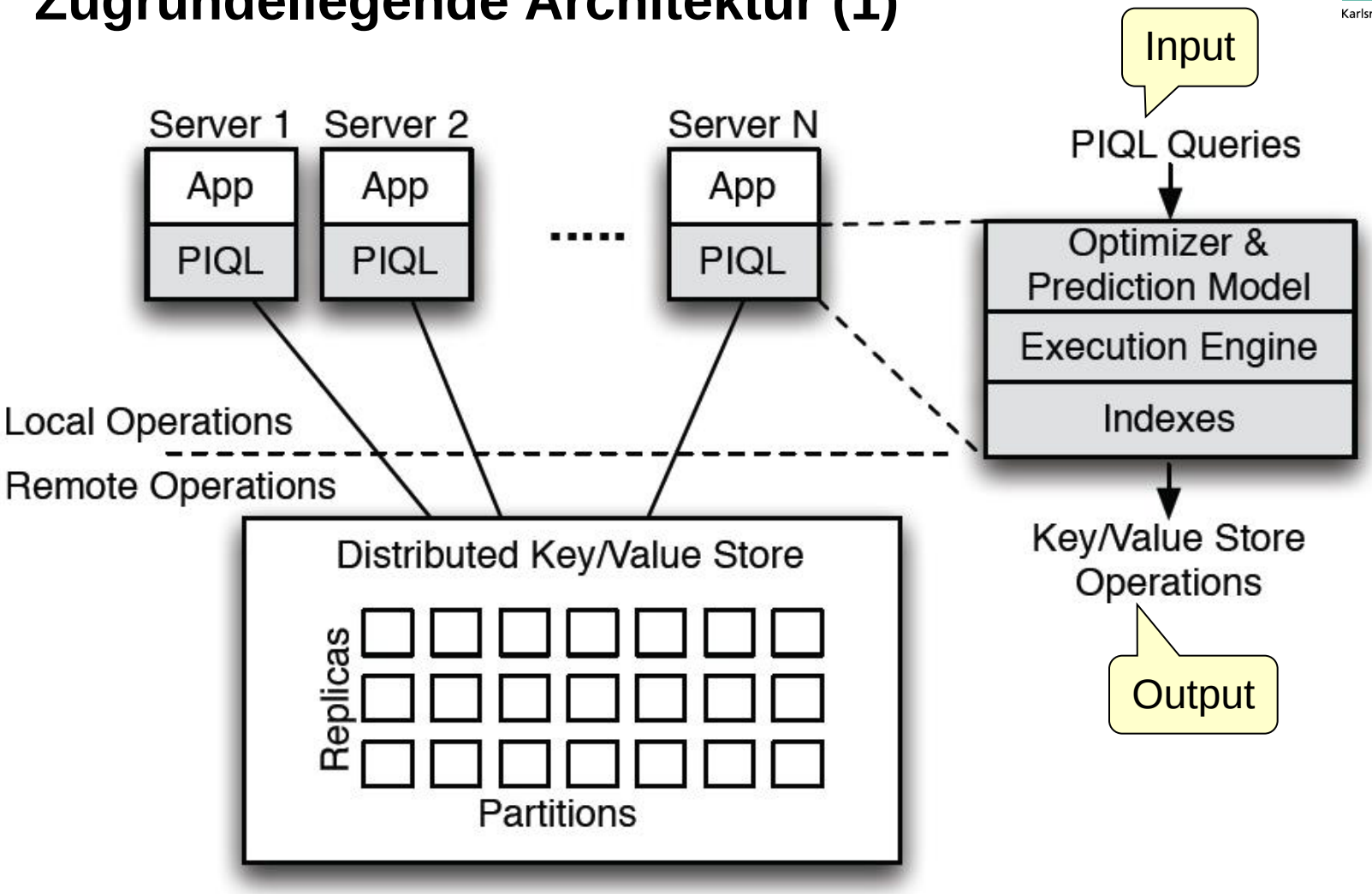
- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Zugrundeliegende Architektur (1)



Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Clients hier nicht explizit dargestellt.

Zugrundeliegende Architektur (2)

- Datenbank Library ist zustandslos,
  d. h. speichert nicht Zustände zwischen Zugriffen.
  Übliche Praxis, um Komplexität zu reduzieren.
  D. h. keine Transaktionen!

- Kostenmaß wird sein:
  Anzahl der Key/Value Store Operationen.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Wie lässt sich garantieren, dass Anfrageergebnis bestimmte Größe hat?

- Übersicht:
  - top-Operator (heißt im Artikel LIMIT-Klausel) in Anfrage.
  - Pagination.
  - Berücksichtigung von Fremdschlüsselbeziehungen.
  - Erweiterungen des Datenbank-Schemas mit Kardinalitätsangaben.

PIQL Spracherweiterungen & Beschränkung der Ergebnisgröße (1)

– Folie von eben –

- Wenn Join Fremdschlüssel enthält,
  ist Zwischenergebnis so groß wie eine Eingabe-Relation.
- Veranschaulichung:

Ausleih

Invnr	Name
4711	Meyer
1201	Schulz
0007	Müller
4712	Meyer

Bücher

Invnr	Titel	ISBN	Autor
0007	Dr. No	3-125	James Bond
1201	Objektbanken	3-111	Heuer
4711	Datenbanken	3-765	Vossen
4712	Datenbanken	3-891	Ullman
4717	Pascal	3-999	Wirth

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

PIQL Spracherweiterungen & Beschränkung der Ergebnisgröße (2)

- Umgekehrte Richtung – Beispiel 1:
  - Angenommen, ein Buch wird maximal zehn Mal ausgeliehen.
  - Dann ist es desolat und wird ausgemustert.
  - Gegeben ein bestimmtes Buch, ist in diesem Fall die Größe des Join-Ergebnisses beschränkt.
  - Zu modellieren als Eigenschaft der Relation ‚Ausleih'. Sie dürfen kein elftes Ausleih-Tupel zu einem Buch generieren.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

PIQL Spracherweiterungen & Beschränkung der Ergebnisgröße (3)

- Umgekehrte Richtung – Beispiel 2:

```
CREATE TABLE Users (
    userId INT,
    firstName VARCHAR(255)
    …
)
CREATE TABLE Tweets(
    owner INT,
    timestamp …
    … )
CREATE TABLE Subscriptions (
    owner INT,
    target INT,
    …
    CARDINALITY LIMIT 100 (ownerUserId)
)
```

Fiktives Beispiel aus dem Twitter-Kontext.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

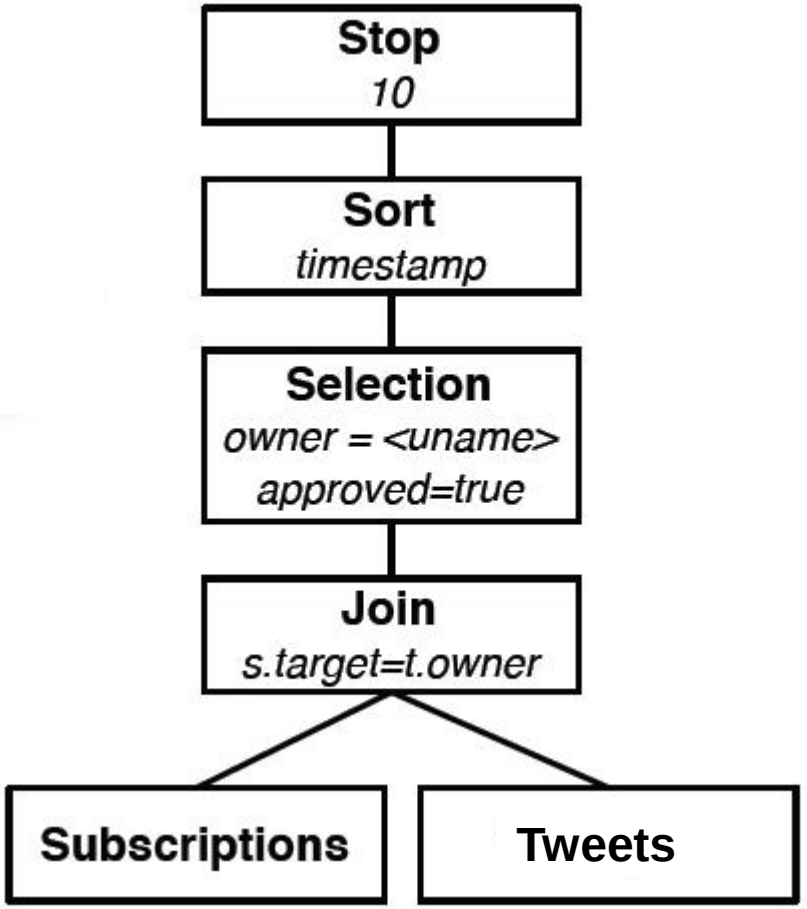
- Umsetzung

- SLO Compliance

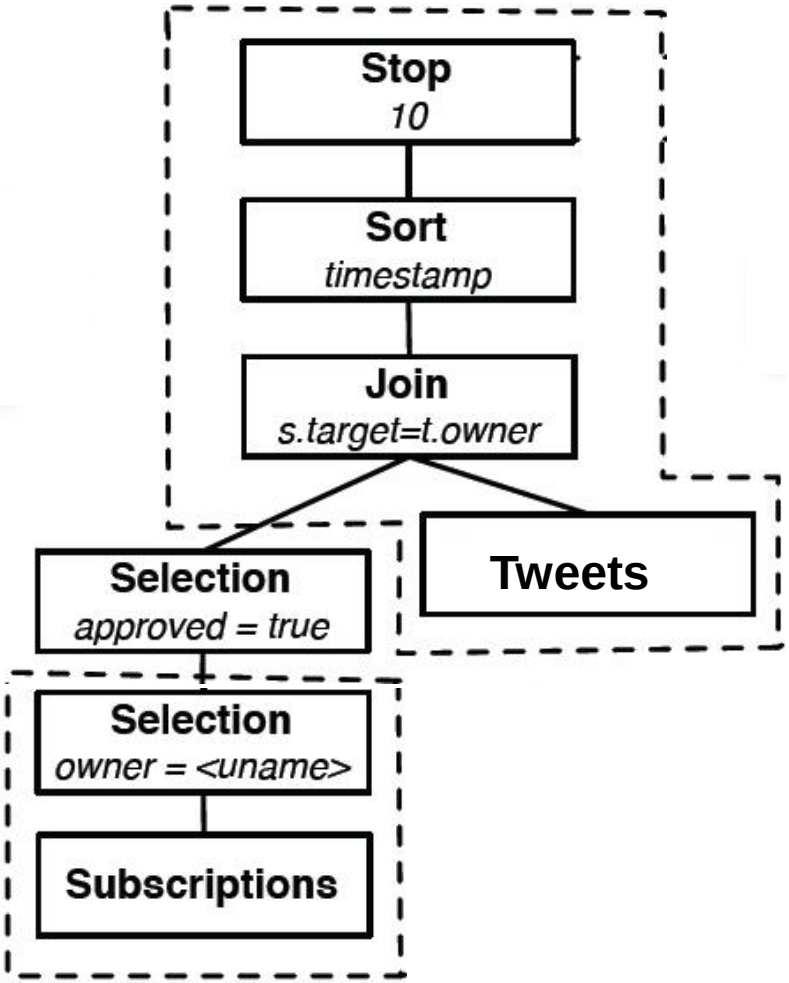
Schluss

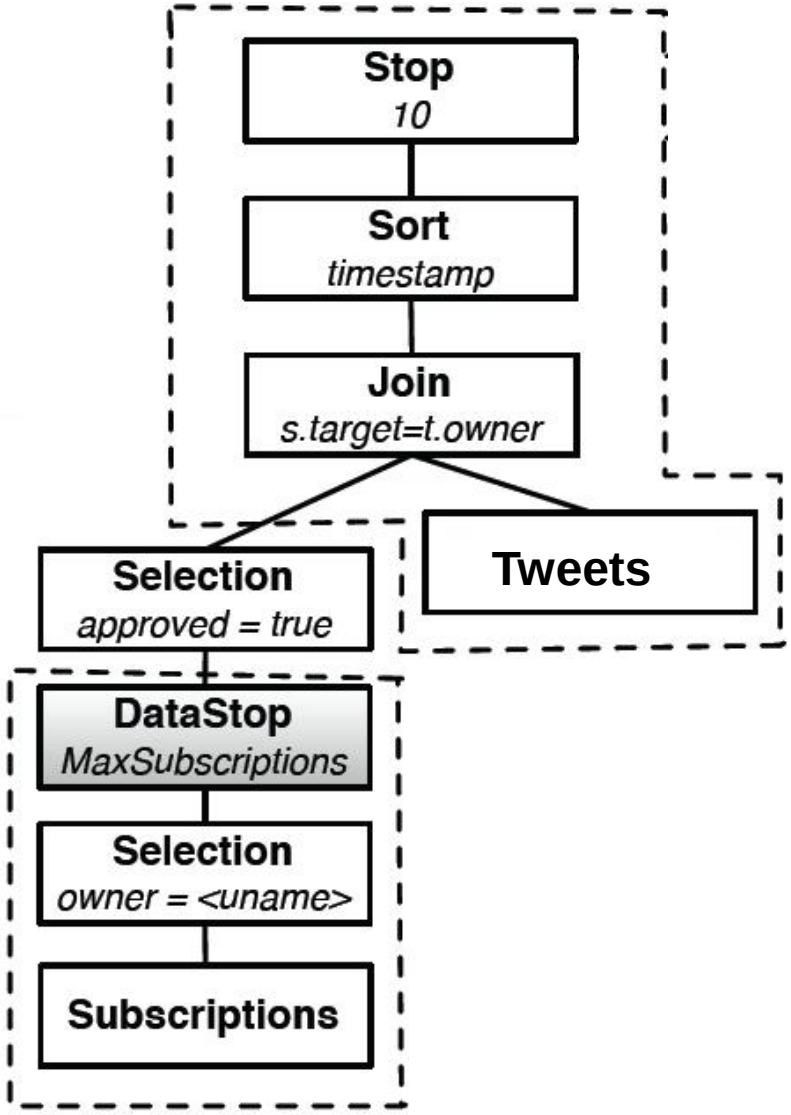
Anfrageübersetzung/-Optimierung (1)

- Anschaulich: „Selektiere die 10 aktuellsten tweets, die ich abonniert habe.“
-
```
SELECT tweets.*
FROM subscriptions s
    JOIN tweets t
WHERE t. owner = s.target
    AND s.owner = <uname>
    AND s.approved = true
ORDER BY t.timestamp DESC
LIMIT 10
```
- Rechts Darstellung als Ausdruck in erweiterter relationaler Algebra.



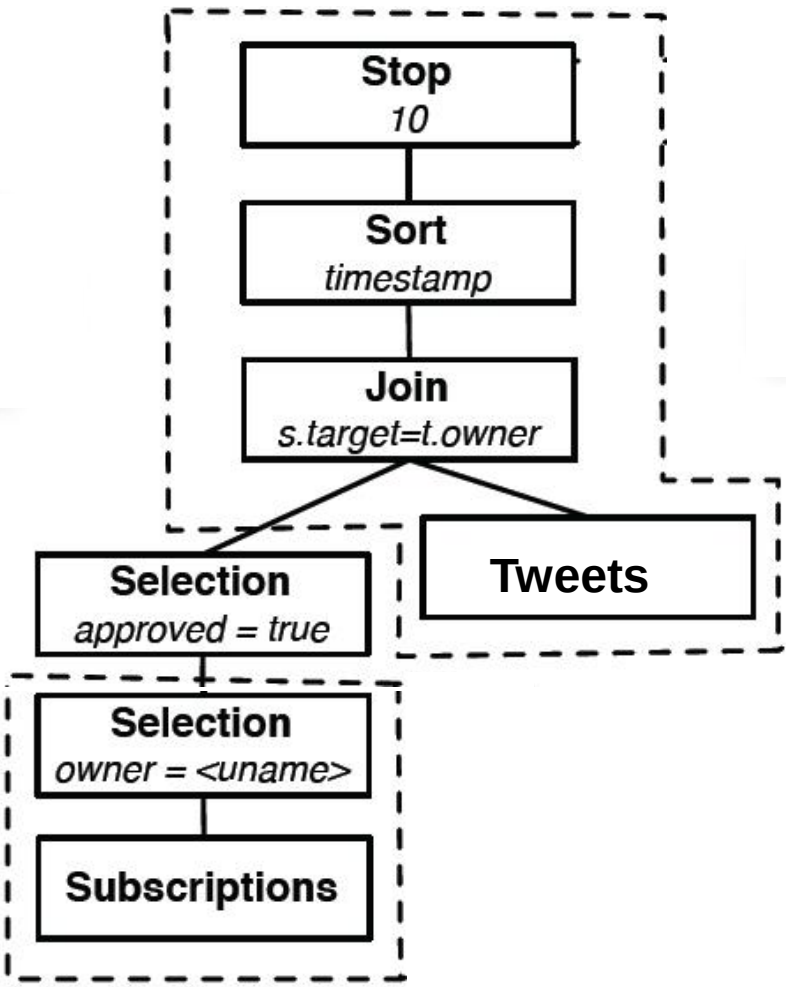
Anfrageübersetzung/-Optimierung (2)

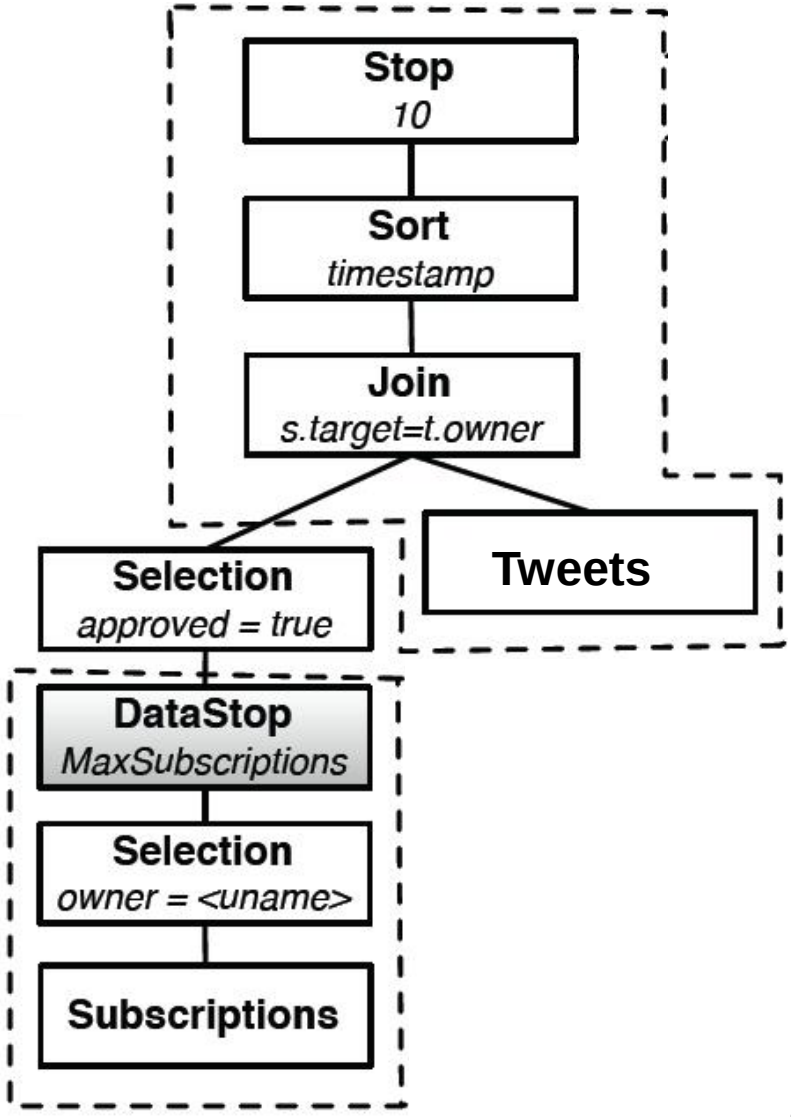




- Grundlagen
- Dynamo
- Begrenzung Erg.größe
- -Intro
- - Architektur
- - Prinzip
- - Umsetzung
- - SLO Compliance
- Schluss

Anfrageübersetzung/-Optimierung (2)





Hier auch andere generische Optimierungen (z. B. Join-Anordnung)

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Anfrageübersetzung/-Optimierung (3)

- Erläuterungen **dataStop**-Operator:
  - Annotation, dass Ausdruck
    nur bestimmte Anzahl von Tupeln generiert.
    (Operator tut/leistet nichts.)
  - Generierung gemäß Kardinalitätsangabe
    im Schema oder für Fremdschlüssel.
  - Platzierung des **dataStop**-Operators hier im Beispiel:
    Käme er nach dem zweiten Select,
    wäre das nicht maximal präzise.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Optimierung auf der physischen Ebene (1)

- Physische Operatoren können i. Allg.
  mehr als einen logischen Operator enthalten.
  (S. b. Beispiele zu Beginn dieses Kapitels.)

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Optimierung auf der physischen Ebene (2)

- Zwei Arten von physischen Operatoren:
  - Ein *remote operator* besteht aus Zugriffen aufs key-value Store und elementaren Verarbeitungsschritten.
    Kostenmodell berücksichtigt nur remote Operatoren.
  - Client-seitige Operatoren für Query-Logik.

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Optimierung auf der physischen Ebene (3)

- Besonderheiten **remote**-Operator:
  - Jeder remote Operator muss explizite Beschränkung der Größe des Zwischenergebnisses enthalten (i. d. R. durch **stop**-Operator in Operator-Darstellung).
  - Hilft, Ausführungsdauer der Anfrage zu beschränken. (Dazu später mehr.)
  - Fehlermeldung, wenn sich derartige Darstellung nicht generieren lässt.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

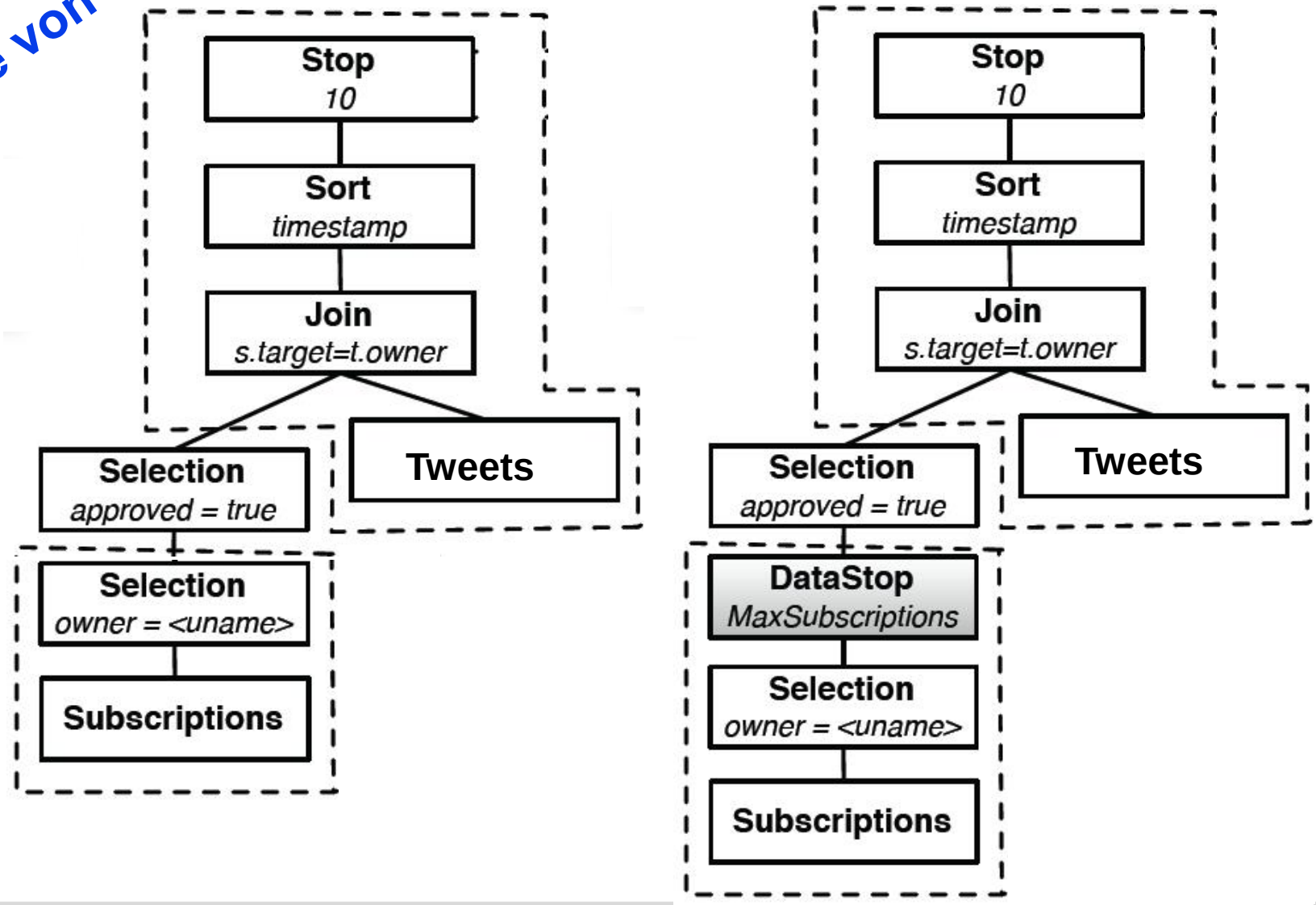
- Umsetzung

- SLO Compliance

Schluss

Optimierung auf der physischen Ebene (4)

– Folie von eben –



Was sind hier die remote Operatoren?

- Übliche Muster logischer Operatoren,
  die immer wieder auftreten.
- Z. B. Zugriff auf eine Relation
  und dann Auswertung diverser Bedingungen.
- Z. B. Join, Auswertung diverser Bedingungen, ggf. sortieren.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

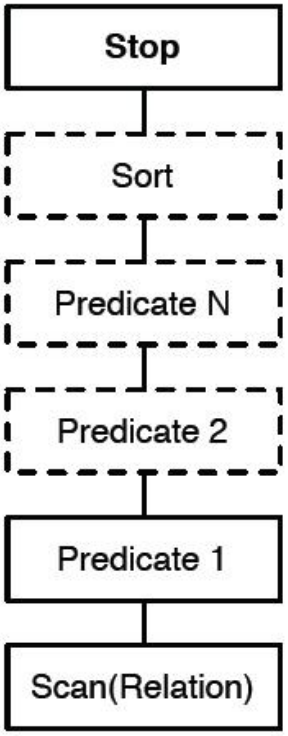
- Umsetzung

- SLO Compliance

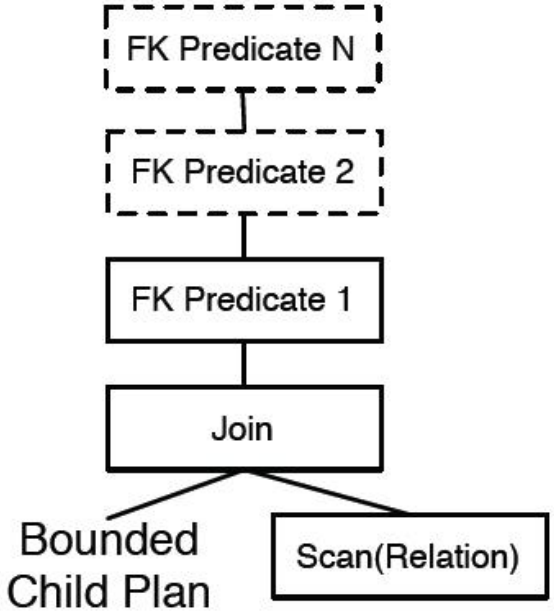
Schluss

Optimierung auf der physischen Ebene (5)

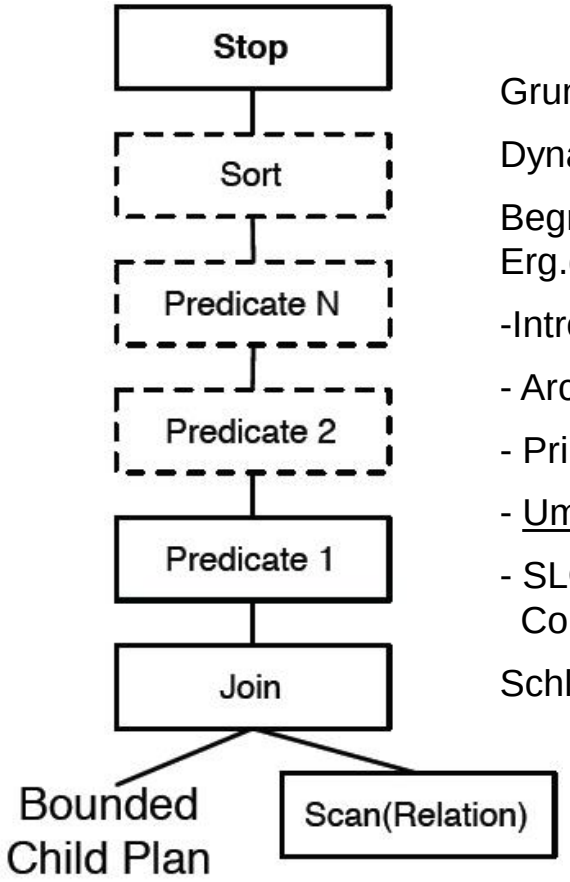
- Welche **remote**-Operatoren gibt es?



(a) IndexScan



(b) IndexFKJoin



(c)SortedIndexJoin

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Optimierung auf der physischen Ebene (6)

- Verfügbare **remote**-Operatoren:
  - **IndexScan**
    - Prädikat muss zusammenhängendem Ausschnitt des indexierten Wertebereichs entsprechen.
    - ‚Sort' muss Sortierreihenfolge des Index sein.
  - **IndexForeignKeyJoin**
    Beschränkung ergibt sich aus Fremdschlüsseleigenschaft.
    - Deshalb im Gegensatz zu den anderen **remote**-Operatoren kein (logischer) **Stop**-Operator.
    - Linker Teilausdruck enthält Fremdschlüssel.
    - SortedIndexJoin hat diese Eigenschaft i. Allg. nicht.

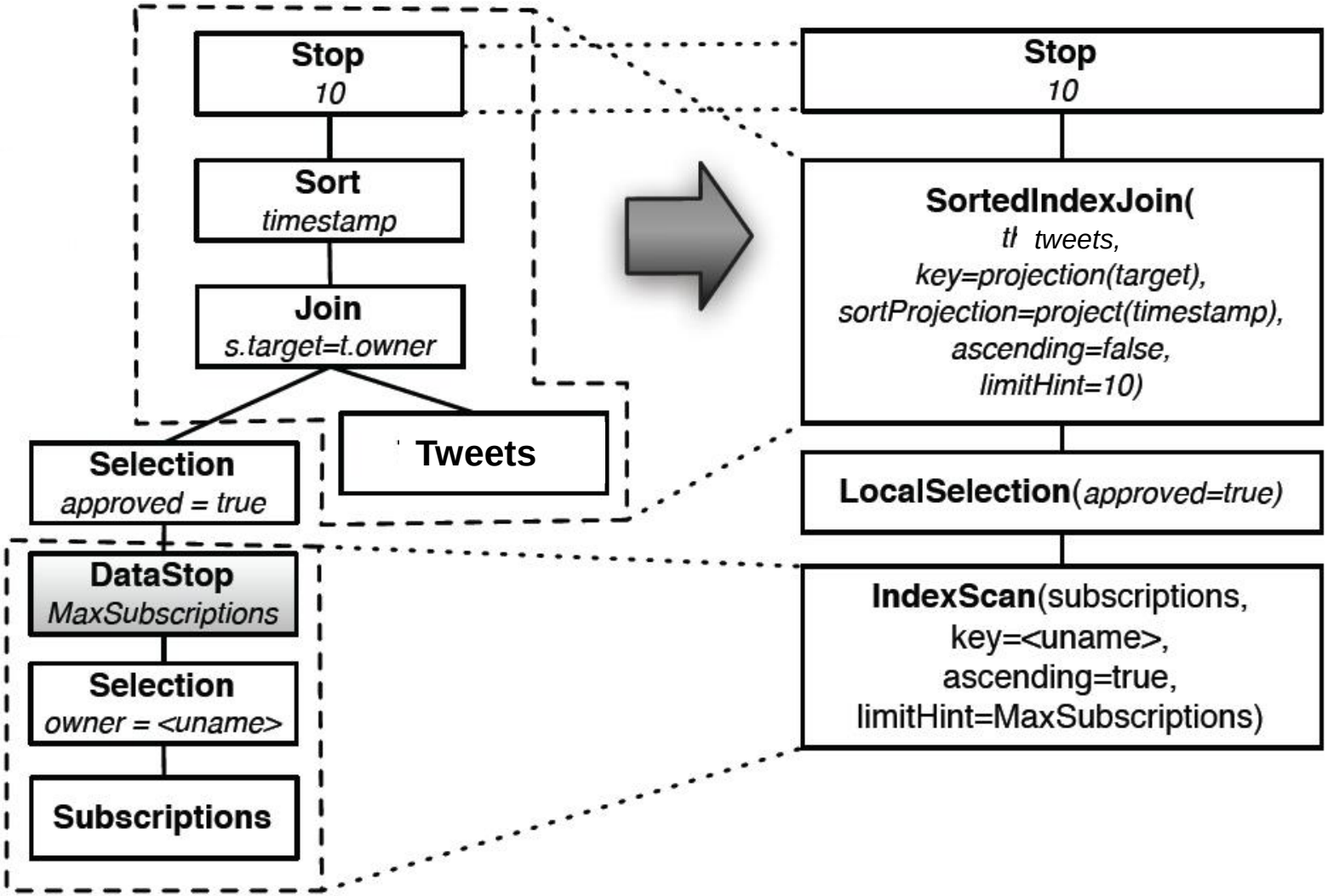
Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Optimierung auf der physischen Ebene (7)

- Verfügbare remote Operatoren (Forts.):
  - **SortedIndexJoin**
    - Bei Sortierung des Inputs nach Join Key lässt sich aus dem limit hint Begrenzung der Anzahl Datenobjekte pro Schlüssel ableiten.
    - Im laufenden Beispiel reichen 10 Tweets pro subscription als Input. Nur leicht realisierbar, wenn Tweets pro subscriber nach tweet.timestamp vorsortiert sind.
    - Vorsortierung durch Indexierung gegeben (s. b. Folie 24 f.).

Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Anfrageübersetzung/-Optimierung (4)



Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

SLO Compliance Vorhersage

- (SLO = ‚Service-Level Objectives‘)
- Beschränkung der Größe der Zwischenergebnisse
  noch keine Garantie dafür,
  dass Aufwand insgesamt wirklich beschränkt ist.
- Wenn anliegende Last sehr groß,
  kann Ausführung von IndexScan beliebig lange dauern.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

SLO Compliance – einzelne Operatoren (1)

- Nur Betrachtung der **remote** Operatoren.

- Hier recht einfaches Vorgehen,
  um das Ziel ‚beschränkter Aufwand' zu erreichen.
- Annahme:
  Ausführungsdauer einzelner Operatoren
  hängt nur ab von Anzahl und Größe der Tupel.
- Anliegende Last fällt (angeblich) nicht ins Gewicht,
  da key-value Stores über gute Lastbalancierung verfügen.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

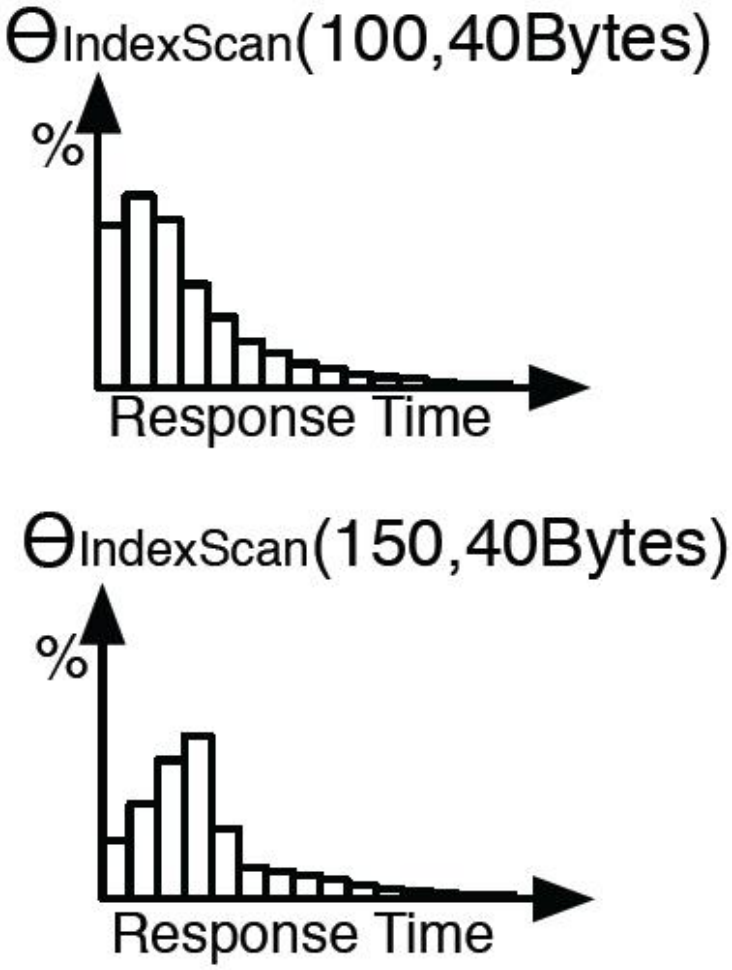
- Umsetzung

- SLO Compliance

Schluss

SLO Compliance – einzelne Operatoren (2)

- Ausführungszeit ist Zufallsvariable, z. B. IndexScan – $\theta(\alpha, \beta)$
  - α – Anzahl erwarteter Tupel (Limit Wert)
  - β – Größe eines Tupels.
- α, β – spannen zweidimensionalen Raum auf.
- Histogramm vereinfachte Darstellung rechts (für einen Punkt in dem Raum).
- Histogramm Lookup für Vorhersage.



Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

- Umsetzung

- SLO Compliance

Schluss

Query Plan insgesamt (1)

- Vereinfachung:
  Modellierung aufeinanderfolgender Operatoren
  als blockierende Operatoren.
  D. h. ein Operator nach dem anderen.
  (Konservativ, hier aber OK.)
- Bzw. ansonsten Unterscheidung
  zwischen seriellen und parallelen Operatoren.

Grundlagen

Dynamo

Begrenzung Erg.größe

-Intro

- Architektur

- Prinzip

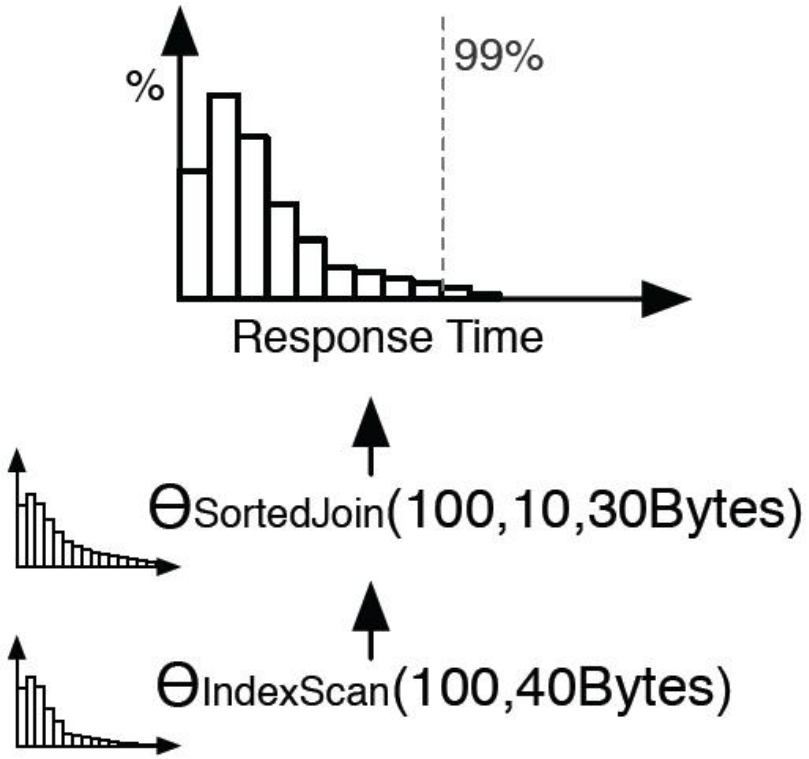
- Umsetzung

- SLO Compliance

Schluss

Query Plan insgesamt (2)

- Abschätzung Ausführungsdauer zweier aufeinanderfolgender physischer Operatoren – Illustration:



Grundlagen
Dynamo
Begrenzung Erg.größe
-Intro
- Architektur
- Prinzip
- Umsetzung
- SLO Compliance
Schluss

Schluss

- Funktioniert gut in Laborexperimenten.
- Ansatz ‚beschränkte Querysprache‘.
- Klassische relationale Datenbank-Technologie nicht in allen Fällen Mittel der Wahl.
- Forschungsthema derzeit, relationale Konzepte ‚für die Cloud‘ verfügbar zu machen, so gut wie möglich.

Grundlagen
Dynamo
Begrenzung Erg.größe
Schluss

Literatur

- Ion Stoica et al. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications.
  SIGCOMM 2001.
- Giuseppe DeCandia et al. Dynamo: Amazon‘s Highly Available Key-Value Store. SOSP 2007.
- Michael Armbrust et al. PIQL: Success-Tolerant Query Processing in the Cloud. PVLDB 2012.

Mögliche Prüfungsfragen – Beispiele

- Was für Möglichkeiten kennen Sie, den Join zu implementieren? Welche Komplexität haben sie?
- Welche Möglichkeiten kennen Sie, den Aufwand, den eine Anfrage verursacht, zu reduzieren/zu begrenzen?