Verteilte Informationssysteme WS 2019/20
Übungsblatt 1

Aaron Winziers - 1176638

20. November 2019

Aufgabe 1

a)

1. Kosten und Skalierbarkeit
2. Replikation zur Verbesserung der Verfügbarkeit
3. Integration verschiedener Softwaremodule
4. Integration von Legacy Systemen
5. Neue Anwendungen
6. Zwänge des Marktes

b) Homogene -

- Verwenden die gleiche Hardware
- Verwenden gleiches DBMS
- Datenstrukturen sind gleich
- Leicht erweiterbar durch weitere Instanzen
- Inflexibel

Heterogene -

- Kann aus verschiedene DBMS bestehen
- Erlaubt das integreieren von verschiedene bereits existierende DBMS
- Erlaubt späteres separieren von integrierte DBMS
- Kann auf verschiedener Hardware laufen

c)

- Keine spezialisierte Server
- Alle Peers speichern Daten und erlauben Zugriff darauf
- Es gibt begrenzte Informationen über das Netz selber
  - Peers kenne nur ihre direkte Nachbarn
  - Es gibt kein globales Wissen
  - keine Zentrale Koordination

d) Die Begriffe erläutern welche Resourcen von verteilten Datenbanksystemen geteilt werden Shared Everything - Mehrere Rechner teilen sich Hauptspeicher und Sekundärspeicher

- Pro - Niedriger Komplexitätsgrad
- Kontra - Skalierbarkeit schwieriger zu realisieren

Shared Disk - Mehrere Rechner - jeder mit eigenem Speicher - teilen sich gemeinsamen Sekundärspeicher

- Pro - Einheitliche Daten
- Kontra - Zugriffe auf die Disk müssen koordiniert werden

Shared Nothing - Weder Hauptspeicher noch Sekundärspeicher werden zwischen den Rechnern geteilt

- Pro - Es gibt kein Single Point of Failure
- Kontra - Daten müssen aktiv geprüft und korrigiert werden um einheitlich zu bleiben

Aufgabe 2

a) $\pi_{name,fachgebiet}(\sigma_{vorname=\text{'Philosoph'}}(assistenten))$

b) $\pi_{vorname,name}(\sigma_{(s.semester\geq 3)\wedge(p.note=1.0)}(studenten \bowtie pruefen))$ (Natürlicher Join, wegen s.matrnr und p.matrnr)

c) Ausgegeben werden die Vornamen und Namen aller Studenten die die Prüfung zur Vorlesung Ëthik"geschrieben haben

d)

```
SELECT professoren.vorname,
       professoren.name,
       vorlesungen.titel
FROM professoren
   JOIN vorlesungen
      ON professoren.persnr = vorlesungen.gelesenvon
   JOIN voraussetzen
      ON vorlesungen.vorlnr = voraussetzen.fortgeschrittennr
WHERE professoren.schwerpunkt = 'Philosophie'
   AND vorlesungen.sws = 4
```

e) $\pi_{titel}(\sigma_{vorlesungen.vorlnr=(\pi_{vorlnr}(vorlesungen)-(\pi_{vorlnr}(hoeren)\cup\pi_{vorlnr}(pruefen)))}(vorlesungen))$

```
SELECT pruefungen.matrnr,
       pruefungen.titel,
       leser.lesername.name,
       pruefungen.pruefername
FROM
   (SELECT persnr AS leserpersnr,
           name AS lesername
    FROM professoren) AS leser
   JOIN leser
      ON
      (SELECT professoren.persnr AS prueferpersnr,
              professoren.name AS pruefername,
              pruefen.matrnr,
              vorlesungen.titel
       FROM professoren
         JOIN pruefen
            ON professoren.persnr = pruefen.persnr
         JOIN vorlesungen
            ON vorlesungen.gelesenvon = professoren.persnr
            ) AS pruefungen
WHERE leserpersnr <> prueferpersnr
```

Aufgabe 3

a) Fragmentierung ist die Aufteilung einer Relation in kleinere Teile. Dies kann entweder Horizontal(Tupel werden aufgeteilt), Vertikal(Attribute werden aufgeteilt), oder Hybrid(eine Kombination aus Vertikal und Horizontal) erfolgen.

b) Allokation ist das Vertielen von Datenfragmenten auf verschiedene Knoten in einem DBS.

c) Wenn die Vollständigkeitsregel verletzt ist, gibt es ein oder mehrere Tupel die in keinem Fragment sind. Damit sind die Daten nicht mehr *Vollständig*

d) Die Disjunktheitsregel ist deswegen wichtig weil mit steigender Anzahl an Tupel die auf mehrere Knoten verteilt sind, steigt auch die Komplexität der Updates auf diese Einträge.

e)

1. Redundanz eliminieren
2. Anzahl der Zugriffe auf einzelne Knoten
3. Menge der Daten pro Knoten
4. Geografische Nähe zum Ort an dem Daten benötigt werden
5. Reduktion der Daten die zurückgegeben werden bei Anfragen