

## Gruppe G Erklärung der CA 3 Lessons Learned

### Datenbank

Für die Umsetzung der CA-Abgabe mussten wir zunächst eine geeignete NoSQL-Datenbank finden, die die benötigten Features bereitstellt. Hierfür haben wir uns drei verschiedene Datenbanken angeschaut. Nach der Recherche fiel unsere Wahl auf die Couchbase NoSQL-Datenbank.

DynamoDB von Amazon:

### Lessons-Learned

Die Datenbank DynamoDB von Amazon bietet zwar die Funktionalität, mehrere Nodes zu verwenden, leider wird diese Funktion nur bei einem kostenpflichtigen Abo bereitgestellt.

Riak:

### Lessons-Learned

- Riak wurde 2009 entwickelt und fühlte sich nach genauerer Recherche nicht mehr nach der State of the Art an. Weshalb wir bedenken hatten diese Datenbank zu verwenden.
- Docker Images recht alt.
- Letztes Update gab es vor 5 Jahren.

### Docker-Compose

Beim Hochfahren des Servers mittels Docker-Compose-File werden den einzelnen Nodes IP-Adressen zugewiesen. Anschließend müssen beim Erstellen des Clusters diese IP-Adressen auf der Startseite des Webinterfaces von Couchbase hinterlegt werden.

Wurde in dem Compose-File die Startreihen der einzelnen Nodes nicht fest vorgegeben, kann es passieren, dass die Nodes beim nächsten Start in einer anderen Reihenfolge hochgefahren werden. Hierdurch können Konflikte mit dem angelegten Cluster auftreten da die IP-Adressen nicht mehr übereinstimmen. → Nodes können nicht mehr angesprochen werden.

### Lessons-learned

- Es muss eine festen Startreihenfolge vergeben werden, damit jeder Node wieder die richtige IP-Adresse zugewiesen bekommt.
- Logging kann den Nutzer in die Irre führen.
- Der Docker Container meldet wenige Sekunden nach dem Start dem Nutzer, dass er bereit ist. Jedoch kann es passieren, dass dieser erst nach ein paar Minuten wirklich erreichbar ist.

### Docker Port-Mapping

### Lessons-learned

- Normalerweise wird nur eine Instanz eines Docker Container pro Rechner verwendet,
- Bei der Verwendung von mehreren Instanzen auf einem Rechner muss das Portmapping von Nutzer vorgenommen werden. **Sehr aufwendig und kompliziert.**

## Vorverarbeitung der Daten

Für die Aufbereitung der Daten, um diese in unser gewählten Datenmodelle zu bringen. Werden die Daten in der Python Datei **preprocessing.py** vorverarbeitet.

```
1 {
2   "user_id": "100005680",
3   "followers_id": [
4     "50681216",
5     "109031507",
6     "38967935",
7     "237752987",
8     "51073543",
9     "59676554"
10  ],
11  "following_id": [
12    "259394077",
13    "50681216",
14    "37527136",
15    "109031507"
16  ]
17 }
```

Zuvor hatten wir versucht die Daten während der Laufzeit für den Upload anzupassen. Für die Überprüfung, ob die Daten bereits vorhanden sind, wurden bei jedem Element die kompletten Daten geladen und jedes Mal geprüft, ob sich das Element bereits also Value abgelegt wurde.

➔ **Verursacht sehr lange Verarbeitungszeiten!**

### Lessions-learned

Daten sollten auf jeden Fall vorher schon in das passende Format gebracht werden, um vielfachen Zugriffe auf die Datenbank zu umgehen.

## Flask - Web Development

### Lessions-learned

- Für das builden des Flask images wird wheel benötigt, es kann passieren, dass das Builden abstürzt.
- Pip install wheel schafft hier abhilfe.

## Couchbase – NoSQL Abfragen

### Lessions-learned

- Die Couchbase Datenbank besitzt eine eigene SQL-Like-Query-Sprache für JSON namens N1QL.
- N1QL sehr ähnlich zu SQL.
- Queries dadurch intuitiv und leichter umstieg von SQL zu NOSQL möglich.

## Tabel-Plus

### Lessions-learned

Während unserer Recherchen für das Aufsetzen des Couchbase Clusters, sind wir auf die Anwendung TablePlus gestoßen. TablePlus ist eine moderne, native App mit einer sauberen Benutzeroberfläche, die es Entwicklern ermöglicht, Datenbanken gleichzeitig sehr schnell und sicher zu verwalten. TablePlus unterstützt die meisten gängigen Datenbanken NO-SQL wie Mongo DB, Cassandra und viele mehr. Leider wird Couchbase nicht unterstützt.

MySQL 8.0.12 : TLS : Myloc : employees : employees

Search for item: "name\$..."

Items Favorites History

Functions

hello

Tables

- 3\_departments
- current\_dept\_emp
- departments
- dept\_emp
- dept\_emp\_latest\_date
- dept\_manager
- employees
- salaries
- titles
- url\_formats

emp_no	birth_date	first_name	last_name	gender	hire_date	email
10037	1963-07-22	Pradeep	Makrucki	M	1990-12-06	NULL
10038	1960-07-20	Huan	Lortz	M	1989-09-20	NULL
10039	1959-10-01	Alejandro	Brender	M	1988-01-19	NULL
10040	1959-09-13	Welyi	Meriste	F	1993-02-14	NULL
10041	1959-08-27	Uri	Lenart	F	1989-11-12	NULL
10042	1956-02-26	Magy	Stamatou	F	1993-03-21	NULL
10043	1960-09-19	Yishay	Tzvieli	M	1990-10-20	NULL
10044	1961-09-21	Mingsen	Casley	F	1994-05-21	NULL
10045	1957-08-14	Moss	Shanhogue	M	1989-09-02	NULL
10046	1960-07-23	Lucien	Rosenbaum	M	1992-06-20	NULL
10047	1952-06-29	Zvonko	Nyanchama	M	1989-03-31	NULL
10048	1963-07-11	Florian	Syrotluk	M	1985-02-24	NULL
10049	1961-04-24	Basil	Tramer	F	1992-05-04	NULL
10050	1958-05-21	Yinghua	Dredge	M	1990-12-26	NULL
10051	1953-07-28	Hidefumi	Caine	M	1992-10-15	NULL
10052	1961-02-26	Heping	Nitsch	M	1988-05-21	NULL
10053	1954-09-13	Sanjiv	Zschoche	F	1986-02-04	NULL
10054	1957-04-04	Mayumi	Schueler	M	1995-03-13	NULL
10055	1956-06-06	Georgy	Dredge	M	1992-04-27	NULL
10056	1961-09-01	Brendon	Bernini	F	1990-02-01	NULL
10057	1954-06-30	Ebbe	Callaway	F	1992-01-16	NULL
10058	1954-10-01	Berhard	McFarlin	M	1987-04-13	NULL
10059	1953-09-19	Alejandro	McAlpine	F	1991-06-26	NULL
10060	1961-10-16	Breannnda	Billingsley	M	1987-11-02	NULL
10061	1962-10-19	Tse	Herber	M	1985-09-17	NULL

Search for field: "name\$..."

emp\_no long  
10014  
birth\_date date  
1956-02-12  
first\_name var\_string  
Berni  
last\_name var\_string  
Genin  
gender ENUM  
M  
hire\_date date  
1987-03-11  
email blob  
NULL

+ Data Structure Row 1 of 300,027 rows selected Columns Filters

Abbildung 1: Ausschnitt von TablePlus