# Kernaufgabe

Festlegen eines Datenbank Modells (SQL/NoSql/Hadoop)

# Vorgehen

Analyse Datenmodell, Einlesen in relevante Produkte, Performanceanalyse

# Einleitung

Beim Meeting mit der Projektbetreuung am 27.1.2017 wurde der Hinweis gegeben, dass Hadoop für dieses Projekt interessant sein könnte und wir uns das näher ansehen sollten. Zudem wurde erwähnt, dass es von Hortonworks eine Sandbox gibt, auf der ein fertig konfiguriertes Hadoop System mit unterschiedlichsten Tools verfügbar ist.

Ein großer Teil dieses Arbeitspaketes bestand darin, sich in Hadoop einzuarbeiten, Tutorials durchzumachen und erste Erfahrungen mit Big Data Systemen zu machen.

Bei den Gesprächen mit der Projektbetreuung hat sich allerdings auch herausgestellt, dass ein Weiterverwenden des bestehenden Datenmodells [Christians ER Modell] wünschenswert ist, da es bereits einiges an Software dafür gibt.

Des Weiteren spricht dafür, dass es eine BAC1 Gruppe gibt, welche sich mit dem Datenimport von frei verfügbaren Smartmeter Datensätzen beschäftigt hat. Der aktuelle Stand dieser Arbeit ist nach Aussage der Gruppe soweit, dass Daten normiert importiert werden können, die Performance allerdings noch nicht optimal ist. Das ist zum aktuellen Zeitpunkt noch nicht verifiziert, allerdings gehen wir davon aus, dass dieses Projekt nach Performanceoptimierung einsatzbereit ist. Da die verwendetet Programmiersprache allerdings C# ist, ist auf jeden Fall eine Portierung notwendig, da wir uns für das Projekt auf Java geeinigt haben.

# Analyse Datenmodell

Bei Analyse des ER Modells stellt sich heraus, dass nur in der Tabelle ‚meter\_data‘ wirklich große Datenmengen vorhanden sind und performancekritische Abfragen ausgeführt werden. Daher konzentriert sich die erste Analyse ausschließlich auf diese Tabelle beziehungsweise eine Teil-Version davon.

# Performanceanalyse

Für erste Tests wurde ein Teil der REDD ‚low\_freq‘ Daten verwendet. Diese wurde in verschiedene Datenbanken importiert und es wurden darauf Abfragen ausgeführt. Ziel dieser Analyse war es ein Gefühl zu bekommen, wie sich die Performance mit Zunahme an Daten verhält und ob eine SQL Datenbank überhaupt in Frage kommen kann.

## Testsystem

### Hadoop Tests

Für die Tests wurde eine virtuelle Maschine mit der ‚Hortonworks Hadoop Sandbox‘ aufgesetzt. Der Maschine wurden alle Cores des Hosts[[1]](#footnote-1) sowie 8GB Arbeitsspeicher zur Verfügung gestellt.

### MySQL Tests

Für die Tests wurde ein MySQL Server auf dem o.g. Testsystem aufgesetzt. Im Gegensatz zu den Hadoop Tests allerdings direkt auf dem Host-Betriebssystem.

## Testdaten

Als Testdaten wurden REDD Daten eines Hauses importiert. Die Abfrage, die darauf abgesetzt wurde, ergibt den Durchschnittsverbrauch pro Tag pro Kanal.

Die Daten liegen in CSV Dateien mit folgendem Format vor.

|  |  |
| --- | --- |
| timestamp | power |
| int | double |

Tabelle 1: Datenformat CSV

Der Timestamp lässt sich mit FROM\_UNIXTIME(timestamp) in ein Datum umwandeln womit gerechnet werden kann.

Folgende Abfrage liefert den Durchschnittsverbrauch gruppiert nach Kanal, Monat und Tag. Die Dauer ist unser Performance Index.



Abbildung : Abfrage Durchschnitt pro Tag

Es wurden Schritt für Schritt Kanäle hinzugefügt um einen wachsenden Datensatz zu simulieren.

## Testergebnisse

### MySQL Datenbank

Zuerst wird die Dauer des Datenimports gemessen.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Import Dauer** | | | | **Vorhanden** | **Hinzugefügt** | **Dauer [s]** | | - | 1.561.660 | 27 | | 1.561.660 | 1.561.660 | 28 | | 3.123.320 | 745.878 | 10 | | 3.869.198 | 745.878 | 11 | | 4.615.076 | 745.878 | 10 | | 5.360.954 | 3.729.390 | 62 | | 9.090.344 |  |  |   Tabelle 2: Dauer des Datenimports  Abbildung 2: Dauer des Datenimports |  |

Hierbei ist zu sehen, dass die Dauer unabhängig von den bereits vorhandenen Datensätzen linear abhängig zu der Menge der Importierten Datensätze ist.

Als nächsten Schritt wird die Dauer des Query aus Abbildung 1 gemessen.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Berechnung des Durchschnitts** | | | **Anzahl Zeilen** | **Dauer** | | 1.561.660 | 6 | | 3.123.320 | 16 | | 3.869.198 | 19 | | 4.615.076 | 23 | | 5.360.954 | 27 | | 9.090.344 | 42 |   Tabelle : Dauer Berechnung des Durchschnittsverbrauchs |

Abbildung 3: Dauer Berechnung des Durchschnittsverbrauchs

Abbildung 3 zeigt, dass die auch die Dauer der Abfrage linear abhängig zur Anzahl der vorhandenen Datensätze ist.

Abschließend wurden Tag und Monat aus dem Timestamp fix in die Tabelle geschrieben und mit einem Index versehen. Dadurch konnte die Berechnungsdauer bei ~ 10 Millionen Datensätzen von 42 Sekunden auf 8 reduziert werden.

### Hadoop Datenbank

Erste Tests haben gezeigt, dass die Abfragen mit HIVE annähernd gleich funktionieren wie mit SQL, die Performance allerdings auf dem Testsystem schlechter ist. Aufgrund der zufriedenstellenden Ergebnisse mit SQL wurde auf ebenso detaillierte Tests verzichtet. Die Performance lässt sich durch Hinzufügen von mehreren Nodes nahezu beliebig steigern.

# Conclusio

Beim Import der Datensätze fällt auf, dass die Anzahl der bereits vorhandenen Daten keine Auswirkung auf die Dauer des Vorgangs hat, was uns entgegenkommt.

Die Dauer einer Berechnung des Durchschnitts nimmt proportional mit der Anzahl der Datensätze zu. Der REDD Datensatz mit niedriger Auflösung beinhaltet ungefähr 52 Millionen Datensätze. Hier würde die Abfrage hochgerechnet 210 Sekunden dauern. Es hat sich aber gezeigt, dass sich durch die geschickte Verwendung von Indizes, die Performance des getesteten MySQL System sehr stark steigern lässt. Aus diesem Grund, und auch um die Kompatibilität zu den bestehenden Applikationen zu wahren, wird für dieses Projekt bis auf weiteres ein Relationales Datenbanksystem verwendet.

Falls sich im weiteren Verlauf des Projekts herausstellen sollte, dass eine SQL Datenbank nicht ausreichend ist, kann eine Hybrid Lösung angestrebt werden. Hier wäre vorstellbar, dass die ganze Meterverwaltung weiterhin in einer SQL Datenbank verbleibt, die Messdaten allerdings in Hadoop abgespeichert werden.

1. Intel Core i5-4690K @ 3.5Ghz [↑](#footnote-ref-1)