WELT Plus Artikel Recommendation Engine

Siehe auch Jira: **DIP-497** - WELT Plus Artikel Recommendation Engine Waiting for feedback

Folgende Idee wie mit möglichst geringen Aufwänden und Kosten eine Lösung erstellt werden kann.

- CDP:
  - stellt wie gehabt eine täglich aktuelle CSV mit den Artikel-IDs, dem „**visitor_cnt**“ und dem „jaccard_idx“ bereit.
  - Diese Datei ist bereits schon optimiert. D.h. sie enthält nur noch **visitor_cnt**Einträge von >100 (? to be defined)
- Q-Team:
  - erstellt ein Lambda welches alle 24 h ausgeführt wird.
  - das Lambda ladet die CSV Datei vom S3 (hier*könnte* es Timeout Probleme geben, wenn die CSV zu groß ist)
  - das Lambda parst die Daten und erstellt für jeden Artikel eine JSON-Datei mit den Top-5 Artikel IDs nach jaccard_idx. Diese JSON wird auf einen S3 erstellt und über Akamai ge-cached.
  - wir erstellen einen AB-Test in Kameleoon der in allen WELTplus Artikeln ausgespielt wird:
    - Wenn ein User z.B. auf diesem Artikel ist: **157519185**, ruft er von dieser Adresse: https://www.welt.de/ab-testing-spectre/ab-tests/articleRecom/*157519185*.json die JSON Datei mit den Top-5 Empfehlungen auf.
    - das gleiche passiert bei diesem WELTplus Artikel: **160721934** - mit der Adresse: https://www.welt.de/ab-testing-spectre/ab-tests/articleRecom/*160721934*.json u.s.w.
    - d.h. für jeden WELTplus Artikel der in der CSV-Liste ist, erstellen wir mit dem Lambda alle 24h eine kleine JSON Datei die für diesen Artikel die Empfehlungen bereit hält.
    - da die Daten unterhalb dieses Verzeichnisses:https://www.welt.de/ab-testing-spectre/ab-tests von Akamai gechached sind, können auch zig-tausend User darauf zugreifen ohne uns zusätzliche Kosten zu verursachen.
    - Wenn ein User auf einem Artikel ist, der nicht Bestandteil der Datenlieferung in der CSV von CDP war, wird der Request auf https://www.welt.de/ab-testing-spectre/ab-tests/articleRecom/*artikelIDdienichtimCSVist*.jsonein 404-Fehler auslösen und wir zeigen für diesen Artikel keine Recommendations an.

Dieser Ansatz ist relativ robust, performant und dennoch „Kostenneutral“.

Die Recommendation wird auf Basis der Tealium Events berechnet. Dazu wird ein Overlap zwischen allen WELT Plus Artikeln per SQL ermittelt. Dieser wird dann wieder mit dem Jaccard-Index normiert.

Die Berechnung wird am Ende des dip-welt-test-emr-imp-01.sh Skriptes gestartet. Hierfür wird AWS Athena verwendet, da die SQL Performance deutlich besser ist als per MapReduce auf dem EMR Cluster.

dip-welt-test-emr-imp-01.sh:

```
## AWS Athena wird aktualisiert und die SQL für die Recommendation ausgeführt
## Das Ergebnis wird aus dem Tmp-Verzeichnis in das Export Verzeichnis kopiert

/home/aws-welt-test/bin/dip-welt-test-msck-repair-table.sh dip_welt_mart_01.
dm1_f_event_events_map_visitorid_30day
/home/aws-welt-test/bin/dip-welt-test-reco.sh
```

dip-welt-test-reco.sh

```
#!/bin/bash

# Athena queries are fundamentally Asynchronous. So we have to :
# 1) Make the query, and tell Athena where in s3 to put the results (tell it the same place as the UI
uses).
# 2) Wait for the query to finish
# 3) Pull down the results and un-wacky-Jsonify them.
```

```
# run the query, use jq to capture the QueryExecutionId, and then capture that into bash variable
queryExecutionId=$(
 aws athena start-query-execution \
 --query-string "WITH tbl_article_plus_events
AS
(
-- Die Kombination Visitor_Id und Article_Id darf nur einmal gezählt werden
SELECT DISTINCT
 visitor_id,
 event_map['udo_page_escenicid'] AS article_id
FROM dip_welt_mart_01.dm1_f_event_events_map_visitorid_30day a
--FROM dip_global_stage.co_tl_visitor_s_event_events_welt a
WHERE event_map['udo_page_escenicid'] IS NOT NULL
AND event_map['udo_page_ispremium'] = 'true'
AND effectiv_date >= '2018-08-17'
--AND visitor_id = '01658f733175001cca8bc5a57df900087008807f00398'
),
tbl_article_plus_data
AS
(
SELECT article_id,
 count(*) AS article_id_cnt
FROM tbl_article_plus_events
group
by article_id
),
tbl_article_plus_overlap
AS
(
SELECT a.article_id AS article_id_1,
 b.article_id AS article_id_2,
 count(*) AS intersection_cnt
FROM tbl_article_plus_events a,
 tbl_article_plus_events b
WHERE a.visitor_id = b.visitor_id
AND a.article_id != b.article_id
GROUP
BY a.article_id, b.article_id
HAVING COUNT(*) > 50
)
SELECT article_id_1,
 article_id_2,
-- article_data_1.article_id_cnt as article_cnt_1,
-- article_data_2.article_id_cnt as article_cnt_2,
 intersection_cnt,
 cast(1.0000 * intersection_cnt /
 (article_data_1.article_id_cnt + article_data_2.article_id_cnt - intersection_cnt) AS DECIMAL(5,4)) AS
jaccard_idx
FROM tbl_article_plus_overlap AS article_overlap,
 tbl_article_plus_data AS article_data_1,
 tbl_article_plus_data AS article_data_2
WHERE article_overlap.article_id_1 = article_data_1.article_id
AND article_overlap.article_id_2 = article_data_2.article_id
--AND cast(1.0000 * intersection_cnt /
-- (article_data_1.article_id_cnt + article_data_2.article_id_cnt - intersection_cnt) AS DECIMAL(5,4)) >
0.005
ORDER
BY article_id_1,
 cast(1.0000 * intersection_cnt /
 (article_data_1.article_id_cnt + article_data_2.article_id_cnt - intersection_cnt) AS DECIMAL(5,4)) DESC;
" \
 --result-configuration "OutputLocation"="s3://dip-welt-test-s3-data-01/athena/reco_welt_plus/" \
 --region eu-central-1 --output json | jq -r ".QueryExecutionId"
)

echo "queryExecutionId was ${queryExecutionId}"


# Wait for the query to finish running.
# This will wait for up to 60 seconds (30 * 2)
for i in $(seq 1 30); do
```

```
queryState=$(
 aws athena get-query-execution --query-execution-id "${queryExecutionId}" --region eu-central-1 --output
json | jq -r ".QueryExecution.Status.State"
 );
 if [[ "${queryState}" == "SUCCEEDED" ]]; then
 break;
 fi;

echo " Awaiting queryExecutionId ${queryExecutionId} - state was ${queryState}"

if [[ "${queryState}" == "FAILED" ]]; then
 # exit with "bad" error code
 exit 1;
 fi;

sleep 5
done


# Get the results.
## aws athena get-query-results \
## --query-execution-id "${queryExecutionId}" \
## --region eu-central-1 > numberOfBooks_wacky.json

aws s3 cp s3://dip-welt-test-s3-data-01/athena/reco_welt_plus/${queryExecutionId}.csv s3://dip-welt-test-
s3-data-01/data/reco_welt_plus/WeltPlusRecoLatest.csv --region eu-central-1

# Todo un-wacky the json with jq or something
# cat numberOfBooks_wacky.json | jq -r ".ResultSet.Rows[] | .Data[0].VarCharValue"
```