**Homework 2**

Group 6: Logan Fish, Lanyu He, Ahmet Burak Kucukserim

Part 1

1. The title of the document is “ Battle of Sudoměř “
2. There are 15 Unique Sentences within the Document

Unique Types Found within the Document.

{

Process\_start, Know, Agree\_or\_refuse\_to\_act, Surrounding, Killing, Catastrophe, Motion, Supply, Process\_end, Arranging, Attack, Preventing\_or\_letting, Escaping, Building, Confronting\_problem, Surrendering, Causation, Conquering.

}

1. There are 18 unique types found within the document.
2. There are 12 Hostile\_encounter types found within the document.
3. Hostile\_encounter event is first mentioned within the sentences marked by sentence id’s

0, 3, and 11.

1. Storing JSON into Oracle database.

**DATABASE STRUCTURE:**

CREATE TABLE DOCUMENT(

DOCID CHAR(32),

DOCTITLE VARCHAR2(50),

PRIMARY KEY (DOCID)

);

CREATE TABLE SENTENCE (

SENTID VARCHAR2(5),

DOCID CHAR(32),

SENTCONTENT VARCHAR2(2000),

PRIMARY KEY (SENTID, DOCID),

FOREIGN KEY (DOCID) REFERENCES DOCUMENT(DOCID)

);

CREATE TABLE TOKEN (

TOKEN VARCHAR2(100),

SENTID VARCHAR2(5),

DOCID CHAR(32),

FOREIGN KEY (SENTID, DOCID) REFERENCES SENTENCE (SENTID, DOCID),

PRIMARY KEY (SENTID, DOCID, TOKEN)

);

CREATE TABLE EVENT\_TYPE (

EVENTTYPEID VARCHAR2(5),

EVENTTYPE VARCHAR2(50),

PRIMARY KEY (EVENTTYPEID)

);

CREATE TABLE EVENT (

EVENTID CHAR(32),

EVENTTYPEID VARCHAR2(5),

PRIMARY KEY (EVENTID),

FOREIGN KEY (EVENTTYPEID) REFERENCES EVENT\_TYPE (EVENTTYPEID)

);

CREATE TABLE MENTION (

MENTIONID CHAR(32),

TRIGGERWORD VARCHAR2(50),

OFFSETS NUMBER,

OFFSETE NUMBER,

EVENTID CHAR (32),

SENTID VARCHAR2(5),

DOCID CHAR(32),

PRIMARY KEY (MENTIONID),

FOREIGN KEY (EVENTID) REFERENCES EVENT (EVENTID),

FOREIGN KEY (SENTID, DOCID) REFERENCES SENTENCE (SENTID, DOCID)

);

**IMPORT DATA FROM valid.jasonl**

1. Use the DDL above create tables
2. Import files in other tables
3. Use **insert into** to insert data from file\_tables to the tables we created. for example:



**Part 2**

**-- Question 1**

The table maven\_content contains the contents for various historical events and news events, contains their title and classifies them by assigning a doc id and a sentence id for each sentence.

The table maven\_events is a list of annotated events, with each item being a dict for an event. event\_type arranges the events by type, and type\_id is the numerical id for event type. event\_id itself is the unique string for each event.

The table maven\_mentions is a list for the event mentions of the events, with each item being a dict. Lastly, The maven\_sentence\_tokens table contains a list of each word and punctuation of all events separated by sentence id and doc id from the maven\_content table.

**-- Question 2**

CREATE OR REPLACE FUNCTION doctitle

(p\_docid IN maven\_content.DOC\_ID%TYPE)

RETURN maven\_content.DOC\_TITLE%TYPE

IS

v\_title maven\_content.DOC\_TITLE%TYPE;

BEGIN

select distinct DOC\_TITLE into v\_title from maven\_content

where DOC\_ID = p\_docid;

return v\_title;

END doctitle;

declare

title varchar2(50);

begin

title := doctitle('f83d3376bd0aba55f1499ce31a506d1f');

DBMS\_OUTPUT.PUT\_LINE(title);

end;

**-- Question 3**

CREATE OR REPLACE FUNCTION sentcont

(p\_sentid IN maven\_content.SENT\_ID%TYPE,

p\_docid IN maven\_content.DOC\_ID%TYPE)

RETURN maven\_content.SENT\_CONTENT%TYPE

IS

v\_cont maven\_content.SENT\_CONTENT%TYPE;

BEGIN

select distinct SENT\_CONTENT into v\_cont from maven\_content

where DOC\_ID = p\_docid and SENT\_ID = p\_sentid;

return v\_cont;

END sentcont;

declare

content varchar2(5000);

begin

content := sentcont(56,'f83d3376bd0aba55f1499ce31a506d1f');

DBMS\_OUTPUT.PUT\_LINE(content);

end;

**-- Question 4**

SELECT DOC\_ID, SENT\_ID, COUNT(MENTION\_ID) FROM MAVEN\_MENTIONS

WHERE EVENT\_ID IS NULL

GROUP BY DOC\_ID, SENT\_ID

ORDER BY COUNT(MENTION\_ID) DESC;

**-- Question 5**

SELECT doctitle(DOC\_ID), sentcont(SENT\_ID, DOC\_ID), COUNT(MENTION\_ID)

FROM MAVEN\_MENTIONS

WHERE EVENT\_ID IS NULL

GROUP BY doctitle(DOC\_ID), sentcont(SENT\_ID, DOC\_ID)

ORDER BY COUNT(MENTION\_ID) DESC;

**-- Question 6**

CREATE OR REPLACE PROCEDURE display\_sentence\_by\_neg\_event\_count (

neg\_event\_count INTEGER

) AS

BEGIN

FOR n IN (

SELECT

doc\_id,

doc\_title,

sent\_id,

sent\_content

FROM

maven\_content

WHERE

sent\_id || doc\_id IN (

SELECT

sent\_id || doc\_id

FROM

maven\_mentions

WHERE

event\_id IS NULL

GROUP BY

sent\_id || doc\_id

HAVING

COUNT(mention\_id) = neg\_event\_count

)

) LOOP

dbms\_output.put\_line(n.doc\_id

|| '; ' || n.doc\_title

|| '; ' || n.sent\_id

|| '; ' || n.sent\_content);

END LOOP;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put(sqlcode);

dbms\_output.put(': ');

dbms\_output.put\_line(substr(

sqlerrm,

1,

100

));

END;

**-- Question 7**

EXEC display\_sentence\_by\_neg\_event\_count(43);

/\*

b6329831617f85440419c44be9ea98c6; Reading and Leeds Festivals; 5; Many of the UK's most successful rock and pop bands have played at the festival, including The Rolling Stones, Fleetwood Mac, The Kinks, Pink Floyd, Deep Purple, The Who, Cream, Black Sabbath, Judas Priest, Genesis, Iron Maiden, The Jam, The Police, Status Quo, The Pogues, Blur, Pulp, Muse, The Cure, Radiohead, The Libertines, Arctic Monkeys, Biffy Clyro and Oasis.

\*/

EXEC display\_sentence\_by\_neg\_event\_count(-996);

/\*

Nothing is displayed

\*/

**-- Question 8**

DECLARE

v\_maxnum NUMBER;

BEGIN

SELECT MAX(CNT)INTO v\_maxnum

FROM (SELECT DOC\_ID, SENT\_ID, COUNT(MENTION\_ID) CNT FROM MAVEN\_MENTIONS

WHERE EVENT\_ID IS NULL

GROUP BY DOC\_ID, SENT\_ID);

display\_sentence\_by\_neg\_event\_count(v\_maxnum);

END;

**-- Question 10**

CREATE OR REPLACE PROCEDURE display\_negative\_triggers (

p\_doc\_id maven\_mentions.doc\_id%TYPE,

p\_sent\_id maven\_mentions.sent\_id%TYPE

) AS

BEGIN

FOR n IN (

SELECT

trigger\_word

FROM

maven\_mentions

WHERE

sent\_id = p\_sent\_id

AND doc\_id = p\_doc\_id

AND event\_id IS NULL

) LOOP

dbms\_output.put\_line(n.trigger\_word);

END LOOP;

END;

execute DISPLAY\_NEGATIVE\_TRIGGERS('4d542bd5c22638a234ecee32b1142309', '4');

**-- Question 11**

CREATE OR REPLACE PROCEDURE display\_non\_negative\_triggers (

p\_doc\_id maven\_mentions.doc\_id%TYPE,

p\_sent\_id maven\_mentions.sent\_id%TYPE

) AS

BEGIN

FOR n IN (

SELECT

trigger\_word

FROM

maven\_mentions

WHERE

sent\_id = p\_sent\_id

AND doc\_id = p\_doc\_id

AND event\_id IS NOT NULL

) LOOP

dbms\_output.put\_line(n.trigger\_word);

END LOOP;

END;

execute DISPLAY\_NON\_NEGATIVE\_TRIGGERS('4d542bd5c22638a234ecee32b1142309', '4');

**-- Question 12**

SELECT distinct trigger\_word

FROM maven\_mentions

WHERE

event\_id IS NOT NULL

AND trigger\_word IN (

SELECT DISTINCT

( trigger\_word )

FROM

maven\_mentions

WHERE

event\_id IS NULL

);

**-- Question 13**

SELECT DISTINCT

( token )

FROM

maven\_sentence\_tokens

WHERE

token NOT IN (

SELECT DISTINCT

( trigger\_word )

FROM

maven\_mentions

WHERE

event\_id IS NULL

);

**-- Question 14**

CREATE OR REPLACE PROCEDURE display\_all\_trigger\_word

(p\_event\_type\_id maven\_events.event\_type\_id%TYPE)

AS

BEGIN

FOR n IN (

SELECT distinct

trigger\_word

FROM

maven\_events inner join maven\_mentions

ON

maven\_events.EVENT\_ID = maven\_mentions.EVENT\_ID

WHERE

event\_type\_id = p\_event\_type\_id

) LOOP

dbms\_output.put\_line(n.trigger\_word);

END LOOP;

END;

execute display\_all\_trigger\_word(167);