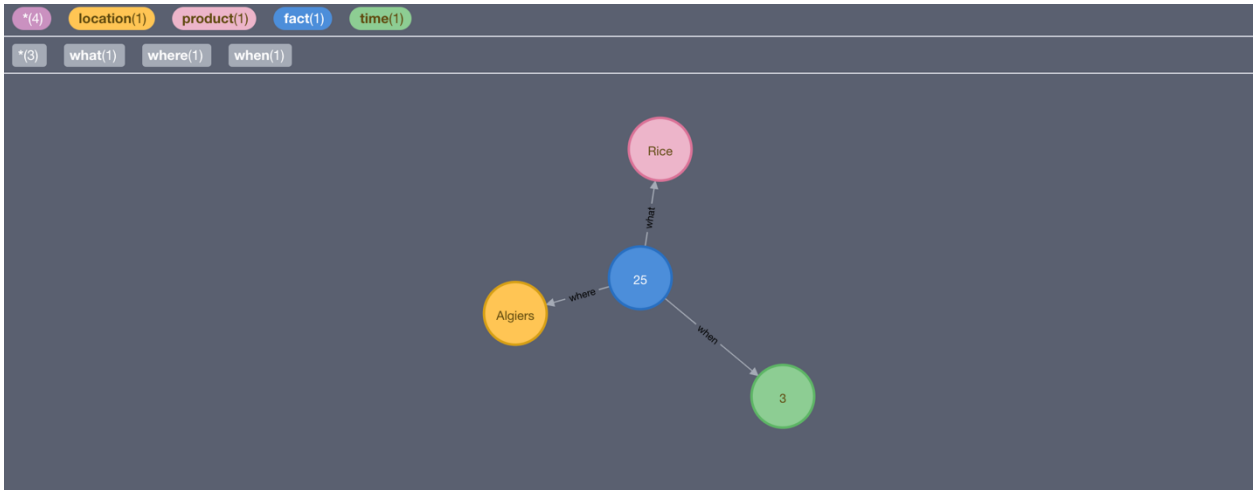
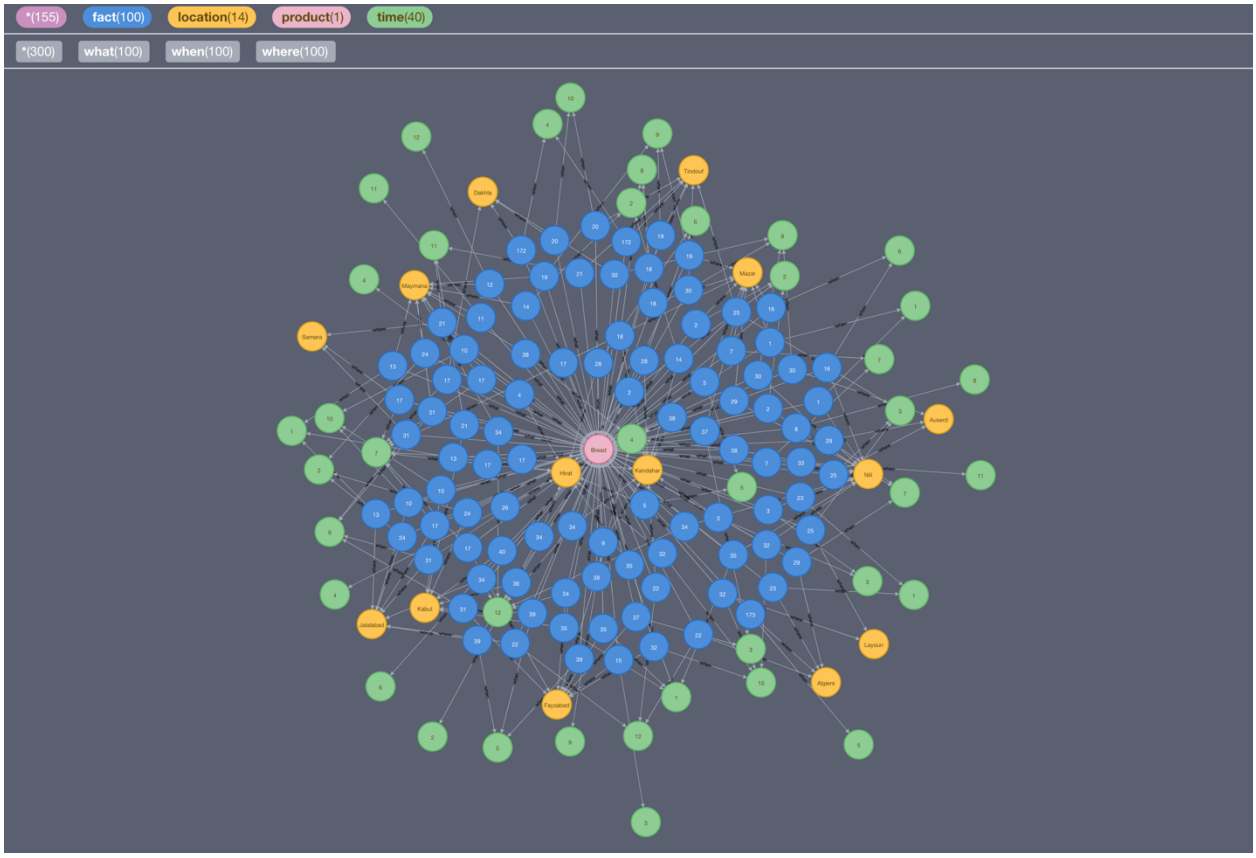
Structure:


*(4)
location(1)
product(1)
fact(1)
time(1)
*(3)
what(1)
where(1)
when(1)
Rice
what
25
where
Algiers
when
3


*(155)
fact(100)
location(14)
product(1)
time(40)
*(300)
what(100)
when(100)
where(100)

Important:

- Please use the neo4j 3.5.14 version database!!!
- change dbms.memory.heap.max_size=(maximum memory space of your system)G, or the memory would leak out!!!

Import data (create nodes):

Important: Put your csv dataset into the correct path before importing datasets!!!

```
LOAD CSV WITH HEADERS  FROM "file:///ADB_Project_fact.csv" AS line
MERGE
(f:fact{fact_id:line.fact_id,location_id:line.location_id,time_id:line.time_id,product_id:line.product_id,price:line.price})
```

If you cannot import the whole fact dataset with one operation, please use the commands below within { }.

```
{

LOAD CSV WITH HEADERS  FROM "file:///fact_1.csv" AS line
MERGE
(f:fact{fact_id:line.fact_id,location_id:line.location_id,time_id:line.time_id,product_id:line.product_id,price:line.price})

LOAD CSV WITH HEADERS  FROM "file:///fact_2.csv" AS line
MERGE
(f:fact{fact_id:line.fact_id,location_id:line.location_id,time_id:line.time_id,product_id:line.product_id,price:line.price})

LOAD CSV WITH HEADERS  FROM "file:///fact_3.csv" AS line
MERGE
(f:fact{fact_id:line.fact_id,location_id:line.location_id,time_id:line.time_id,product_id:line.product_id,price:line.price})

LOAD CSV WITH HEADERS  FROM "file:///fact_4.csv" AS line
MERGE
(f:fact{fact_id:line.fact_id,location_id:line.location_id,time_id:line.time_id,product_id:line.product_id,price:line.price})

LOAD CSV WITH HEADERS  FROM "file:///fact_5.csv" AS line
```

```
MERGE
(f:fact{fact_id:line.fact_id,location_id:line.location_id,time_id:line.time_id,product_id:line.product_id,price:line.price})

}
LOAD CSV WITH HEADERS  FROM "file:///ADB_Project_location.csv" AS line
MERGE (l:location{location_id:line.location_id,country:line.country,
locality:line.locality,market:line.market,market_type:line.market_type,currency:line.currency})

LOAD CSV WITH HEADERS  FROM "file:///ADB_Project_product.csv" AS line
MERGE (p:product{product_id:line.product_id,product_name:line.product_name})

LOAD CSV WITH HEADERS  FROM "file:///ADB_Project_time.csv" AS line
MERGE (t:time{time_id:line.time_id,year:line.year,month:line.month})
```

create relations:

```
match (f:fact),(t:time) where f.time_id=t.time_id create (f)-[rel:when]->(t)

match (f:fact),(l:location) where f.location_id=l.location_id create (f)-[rel:where]->(l)

match (f:fact),(p:product) where f.product_id=p.product_id create (f)-[rel:what]->(p)
```

queries:

Get product_name of specific country:

```
match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel3:where]->(l:location)) where
l.country='Afghanistan' return distinct p.product_name as product_name
```

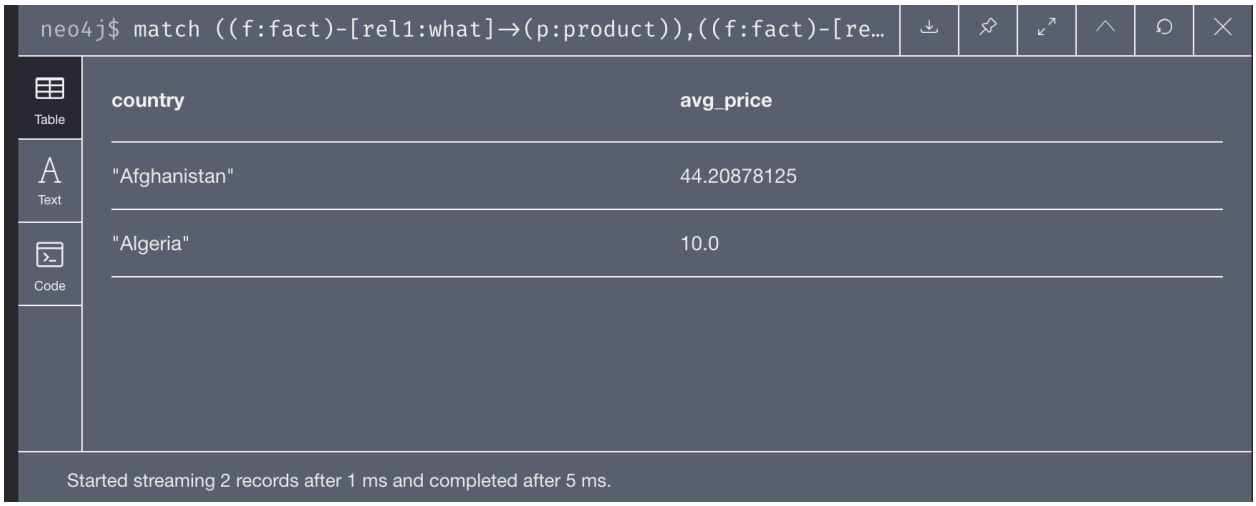
Get market of specific country:

```
match ((f:fact)-[rel3:where]->(l:location)) where l.country='Afghanistan' return distinct l.market
as market
```

Place Aggregations:

1. The average price of commodity “Bread” in each country

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where p.product_name='Bread' return l.country as country,avg(toFloat(f.price)) as avg_price



country	avg_price
"Afghanistan"	44.20878125
"Algeria"	10.0

2. The average price of commodity “Wheat” in each locality of the country “Afghanistan”

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where p.product_name='Wheat' and l.country='Afghanistan' return l.locality as locality,avg(toFloat (f.price)) as avg_price

locality	avg_price
"$Daykundi"	27.432033600000004
"Kabul"	18.50475260115608
"Faryab"	18.061036363636365
"Hirat"	15.192136994219648
"Nangarhar"	16.834626589595384
"Balkh"	15.85119883040937
"Kandahar"	18.76173953488372
"Badakhshan"	18.86864393063584
"Bamyan"	10.539076190476191
"Paktya"	12.362915000000001
"Ghor"	16.460526315789473

3. The highest price of commodity “Rice” in each market type in each country

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where p.product_name='Rice' return l.market_type as market_type,l.country as country,max(toFloat (f.price)) as max_price

neo4j$ match ((f:fact)-[rel1:what]→(p:product)),((f:fact)-[re…

Table | Text | Code

market_type	country	max_price
"Wholesale"	"Burkina Faso"	545.0
"Retail"	"Central African Republic"	833.7137
"Retail"	"Algeria"	120.0

Started streaming 3 records in less than 1 ms and completed after 3 ms.

4. The lowest price of commodity “Bread” in each market in the country “Afghanistan”

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where p.product_name='Bread' and l.country='Afghanistan' return l.market as market,l.country,min(toFloat (f.price)) as min_price

market	l.country	min_price
"Maymana"	"Afghanistan"	50.0
"Nili"	"Afghanistan"	51.7
"Mazar"	"Afghanistan"	45.3375
"Hirat"	"Afghanistan"	29.0
"Kandahar"	"Afghanistan"	31.25
"Jalalabad"	"Afghanistan"	33.3
"Fayzabad"	"Afghanistan"	50.0
"Kabul"	"Afghanistan"	38.475

Time Aggregations:

1. The average price of commodity “Bread” each year in the country “Afghanistan”

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where p.product_name='Bread' and l.country='Afghanistan' return t.year as year,avg(toFloat (f.price)) as avg_price

neo4j$ match ((f:fact)-[rel1:what]→(p:product)),((f:fact)-[re…

Table | Text | Code

year	avg_price
"2015"	44.98606250000001
"2014"	42.84026041666666
"2016"	44.749374999999986
"2017"	44.56916666666667

Started streaming 4 records in less than 1 ms and completed after 3 ms.

2. The highest price of commodity “Rice” each month in the country “Algeria”

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where p.product_name='Rice' and l.country='Algeria' return t.month as month,max(toFloat (f.price)) as max_price

month	max_price
"10"	120.0
"7"	120.0
"3"	100.0
"9"	120.0
"8"	120.0
"12"	100.0
"6"	100.0
"4"	100.0
"11"	90.0
"2"	120.0
"5"	100.0
"1"	90.0

Commodity Aggregations:

1. The lowest price of each commodity in the market “Algiers”

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where l.market='Algiers' return p.product_name as product_name,min(toFloat (f.price)) as min_price

product_name	min_price
"Fish (canned)"	60.0
"Potatoes"	33.0
"Bread"	10.0
"Rice"	80.0
"Tomatoes"	40.0
"Sugar"	75.0
"Carrots"	45.0
"Eggs"	210.0
"Fuel (diesel)"	13.0
"Oil"	112.0
"Tea"	330.0
"Bananas"	170.0
"Cheese (dry)"	200.0
"Pasta"	30.0
"Onions"	30.0
"Lentils"	133.0
"Apples"	140.0
"Meat (chicken)"	250.0
"Fuel (petrol-gasoline)"	21.0
"Beans (white)"	150.0
"Milk"	25.0

2. The commodity (name) which has highest price in the market “Algiers”

match ((f:fact)-[rel1:what]->(p:product)),((f:fact)-[rel2:when]->(t:time)),((f:fact)-[rel3:where]->(l:location)) where l.market='Algiers' return p.product_name as product_name,max(toFloat (f.price)) as max_price order by max_price DESC limit 1

product_name	max_price
"Tea"	750.0

Dump database:
Use your terminal go to the bin folder of neo4j which has the neo4j-admin

```
./neo4j-admin  dump --database=neo4j --to=<path>/neo4j
```

Load database:

```
./neo4j-admin load --from=<path>/neo4j --database=neo4j --force
```

Use the Project graph
Password: project