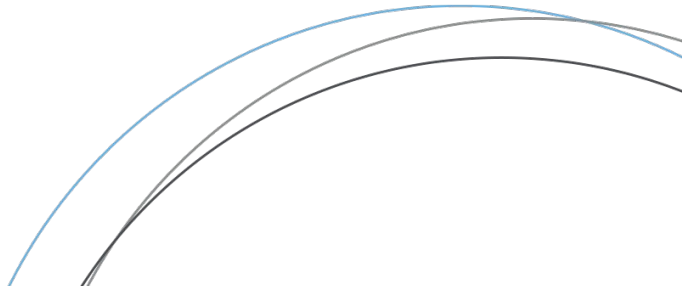
Making a time series database smart. Human and machine communication towards conversational analytics

RosaeNLG Community Meetup

February 2021



Radicalbit

Founded in 2016, **Radicalbit** is an italian deep tech company highly specialized in the development of products and solutions blending **streaming data analysis and Artificial Intelligence**.

Products

RNA (Radicalbit Natural Analytics)
DataOps and MLOps platform able to manage the entire data lifecycle and to integrate Artificial Intelligence algorithms

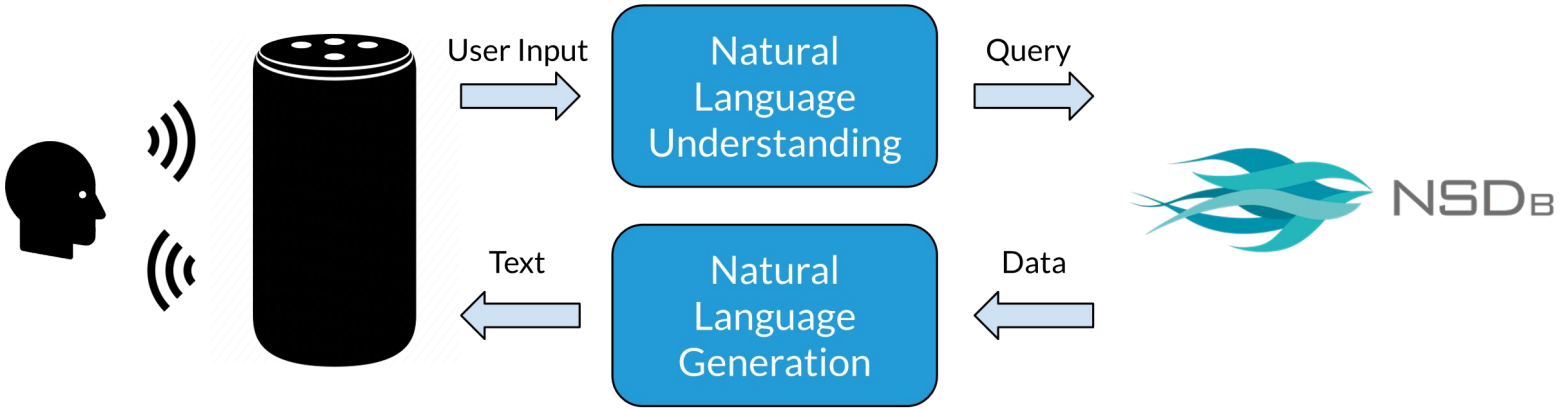
NSDb (Natural Series Database)
Open-source Time Series Database optimized for streaming technologies and real-time data visualization

GoLive - Advanced Stream Commerce
GoLive is an end-to-end platform developed for Retailers capable of combining the innovative power of Live Stream Shopping with Customer Behavioural Analytics

Introduction

The objective is to build an end-to-end system that is capable of:

- understanding human language through Alexa to query a database,
- provide a coherent response in natural language based on the gathered data.



NLU and NLG

"*Find me the number of tweets since yesterday*"

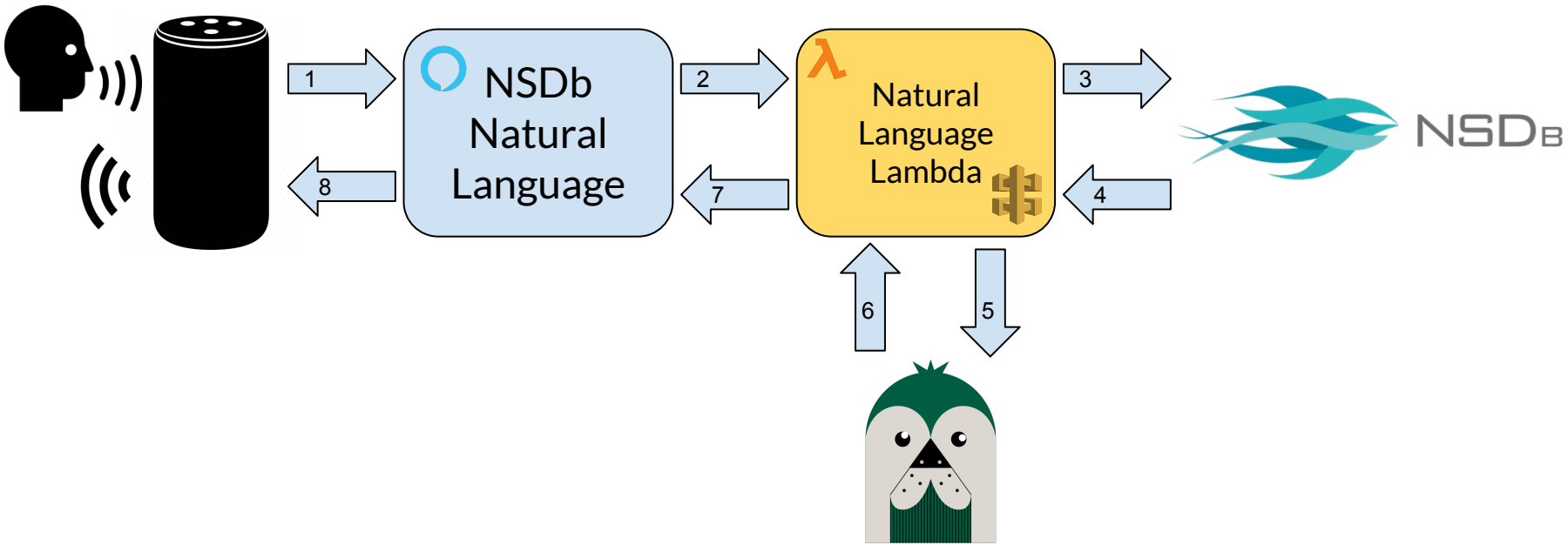
select count(value) from tweets
where timestamp < now – 24h

count(value)

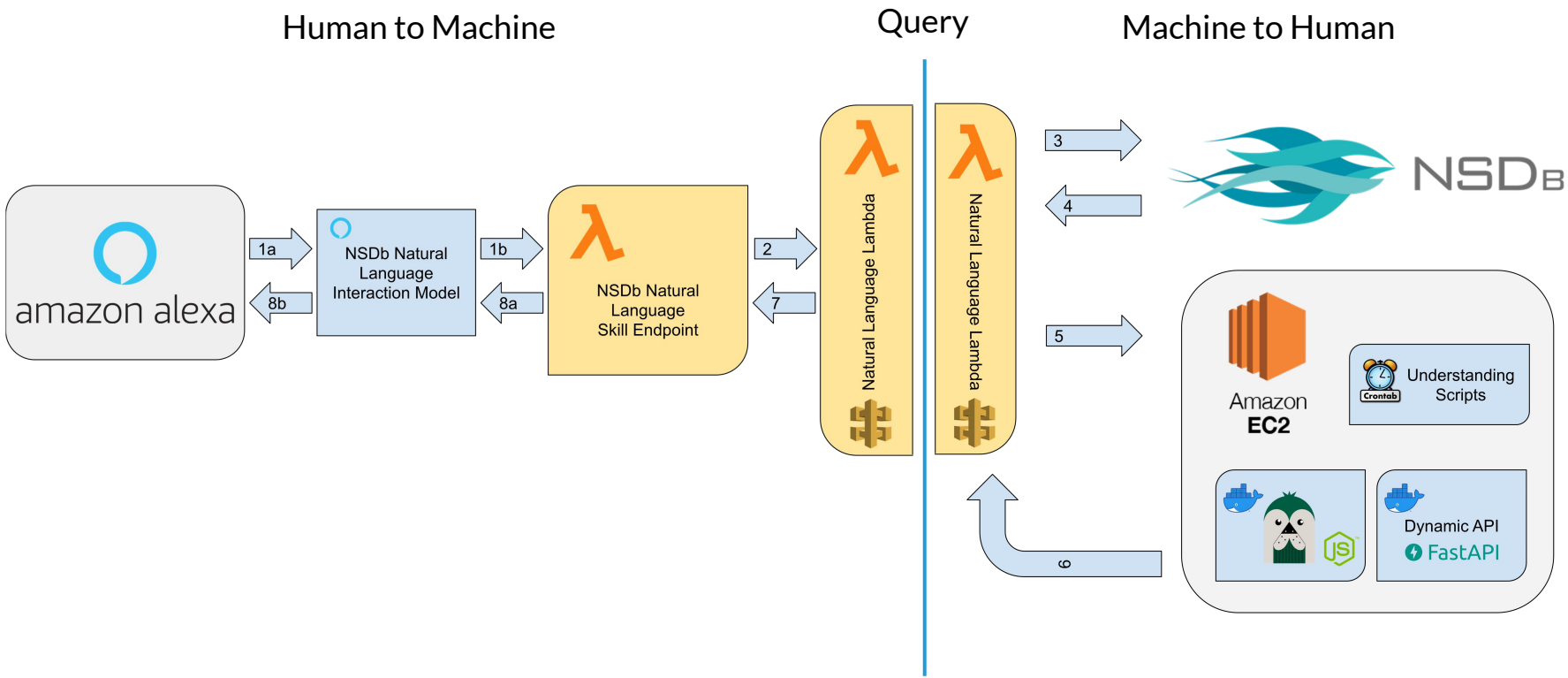
1250

"*The number of tweets is 1250 since yesterday*"

Architecture



Two separated parts



Advantages

The modular architecture potentially allows to:

- Change the NLU (Google Home, Siri, chatbot, …)
- ~~Change the NLG (different tool, Neural Networks, …)~~ 🤭
- Change database (MySQL, PostgreSQL, …)
- Change language (Italian, French, German, ...)

We can also use the two parts separately:

- Standard queries with natural language answers
- Natural language queries with raw data

Why RosaeNLG?

- Open source
- Easy to use
- No beforehand data is needed
- Reliability
- Syntactic variations

RosaeNLG.org

Alternatives?

Neural Networks (especially **RNN** and **Seq2Seq**) are cool for language generation but they are not reliable and require a lot of data that we don't have to train the network.

Templates

Different templates to handle different types of queries:

- *metadata queries*
  - used to know which databases, namespaces, metrics, dimensions we have inside the instance of NSDb we are referring to
- *plain queries*
  - standard one shot SQL queries, they can be plain, temporal or grouped
- *special queries*
  - aggregation of some queries that are used to build a phrase that describe the *global view* of the metric (for example the minimum and maximum value, …)

RADICALBIT

Templates - examples

```
mixin top_bottom_all
  if nsdb.distinct_query.value == 2
    -
      top_percent = Math.floor(nsdb.top_all.value / (nsdb.top_all.value + nsdb.bottom_all.value) * 100)
      bottom_percent = 100 -  top_percent
    synz
      syn
        | the #{dimension} with most #{metric} is #{nsdb.top_all.dimension_value}, with #{nsdb.top_all.value},
        | that is the #{top_percent} percent,
        | while #{nsdb.bottom_all.dimension_value} is at #{bottom_percent} percent, with #{nsdb.bottom_all.value} #{metric}
      syn
        | #{dimension} #{nsdb.top_all.dimension_value} occurred #{nsdb.top_all.value} times,
        | that is the #{top_percent} percent,
        | while #{dimension} #{nsdb.bottom_all.dimension_value} occurred #{bottom_percent} percent, with #{nsdb.bottom_all.value} #{metric},
  else
    synz
      syn
        | the #{dimension} occurred the most is #{nsdb.top_all.dimension_value}, with #{nsdb.top_all.value} #{metric},
        | while the less occurred is #{nsdb.bottom_all.dimension_value}, with #{nsdb.bottom_all.value},
```

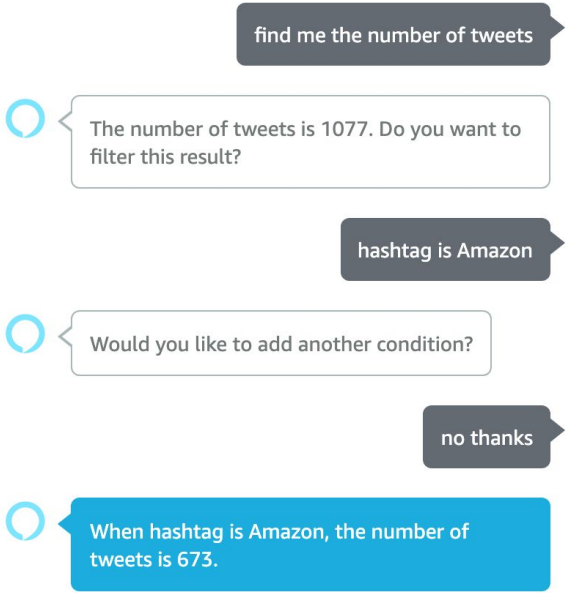
Templates - examples

```
mixin distinct_query
  if nsdb.distinct_query.value > 1
    synz
      syn
        | there are #{nsdb.distinct_query.value} different values of #{dimension},
      syn
        | I've found #{nsdb.distinct_query.value} different values of #{dimension},
  else if nsdb.distinct_query.value == 1
    synz
      syn
        | there is only #{nsdb.distinct_query.value} value for #{dimension},
      syn
        | I've found only #{nsdb.distinct_query.value} value for #{dimension},
  else if nsdb.distinct_query.value == 0
    synz
      syn
        | there aren't values for #{dimension}
```

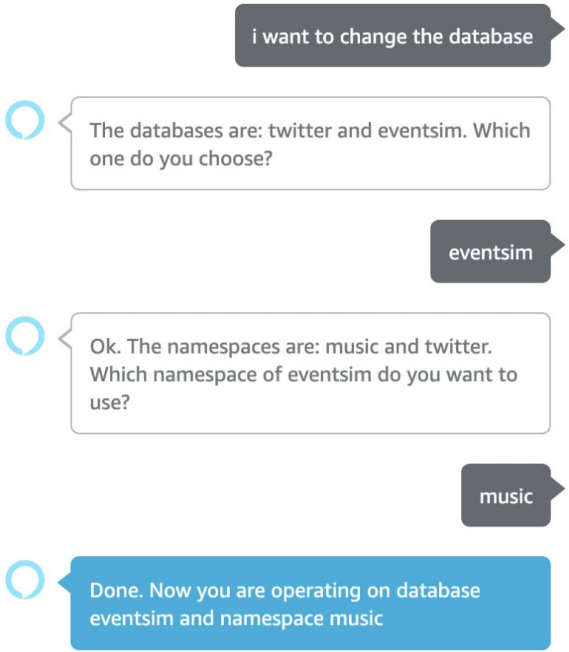
Templates - examples

```
mixin order_by
  if nsdb.parsed.order.direction == 'desc'
    if nsdb.parsed.order.order_by == 'value'
      synz
        syn
          | ranking by #{nsdb.parsed.order.order_by}
        syn
          | from the greatest to the lowest #{nsdb.parsed.order.order_by}
    else
      synz
        syn
          | ordering #{nsdb.parsed.order.order_by} descending
        syn
          | ordering #{nsdb.parsed.order.order_by} descending
        syn
          | in descending order according to #{nsdb.parsed.order.order_by}
        syn
          | ordering the #{nsdb.parsed.order.order_by} in a descending way
        syn
          if nsdb.parsed.order.order_by == 'value'
            | ranking by #{nsdb.parsed.order.order_by}
        syn
          if nsdb.parsed.order.order_by == 'timestamp'
            | from the #[+syn('latest', 'oldest')] to the most recent
```

End-to-end examples

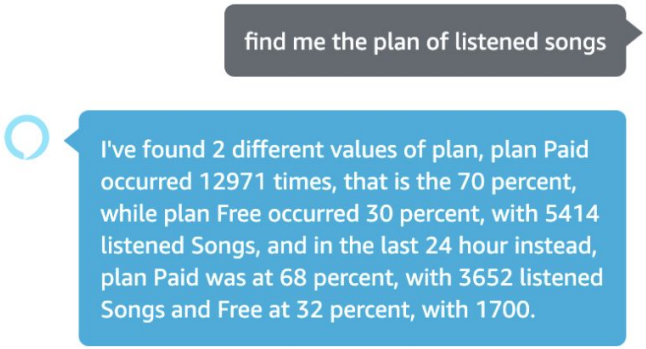


select count(value) from tweets where hashtag = “Amazon”

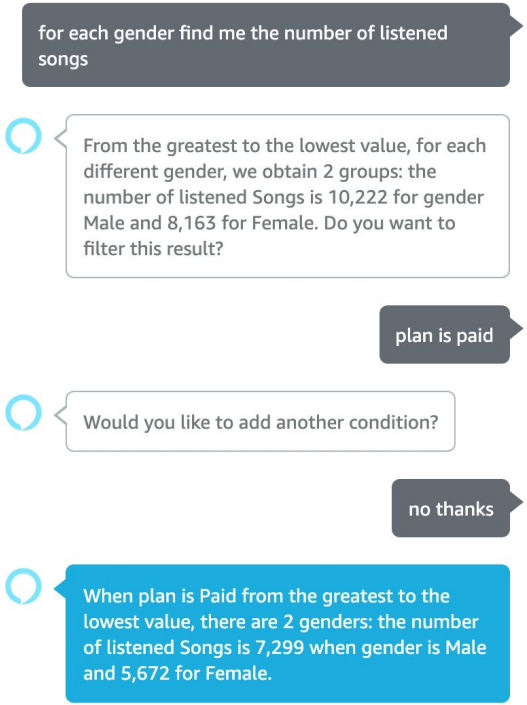


Database selection and namespace description

End-to-end examples



Special query with useful informations



select count(value) from listenedSongs

where plan = “paid” group by gender

RADICALBIT
radicalbit.io

<thanks/>

marco.riva@radicalbit.io