

# Crowdsourced Query Understanding and Optimization



Joachim Hugonot (team leader), François Farquet (speaker), Florian Chlan, Xinyi Guo, Simon Rodriguez, Kristof Szabo, Florian Vessaz, Vincent Zellweger

Supervisor : Immanuel Trummer

# Motivation : Limitations of databases

---

- Join pictures showing the same person.
  - Answer natural language queries.
  - Retrieve data from unstructured data sources (i.e. internet).
-

# Motivation : Limitations of databases

---

- Join pictures showing the same person.
  - Answer natural language queries.
  - Retrieve data from unstructured data sources (i.e. internet).
  - Solution : use **human workers**.
-

# Crowdsourcing

---

- What is crowdsourcing ?
    - Requesters **pay human workers** to do simple tasks.
  - Biggest platform : Amazon Mechanical Turk.
    - Thousands of workers.
    - Typical tasks for 0.01\$ to 0.1\$.
-

# Amazon Mechanical Turk

---

Find website on the internet

**Requester:** Immanuel Trummer

**Reward:** \$0.01 per HIT

**HITs Available:** 1

**Duration:** 60 minutes

**Qualifications Required:** None

What is the most relevant website to find [Presidents of USA] ?

Note that we are interested in : name, political party

http://

Answer a simple Yes/No question

**Requester:** Immanuel Trummer

**Reward:** \$0.01 per HIT

**HITs Available:** 20

**Duration:** 60 minutes

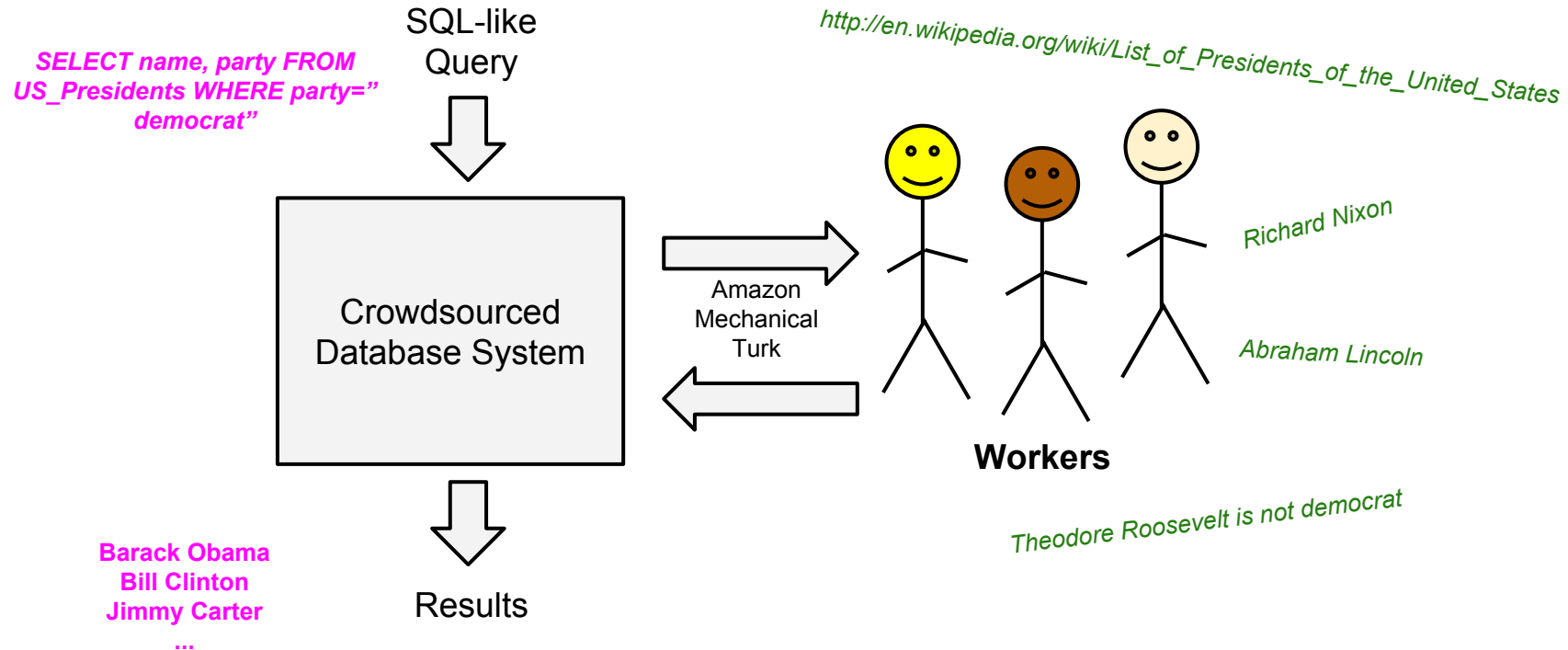
**Qualifications Required:** None

"(Richard Nixon, Republican)": [party is democrat]. Is this true?

☐ yes

☐ no

# Crowdsourced database system



# Example Query

---

(SELECT (*name*, *political party*) FROM [*Presidents of USA*] WHERE [*party is democrat*])

- *attributes*
- *Data source*
- *Predicate*

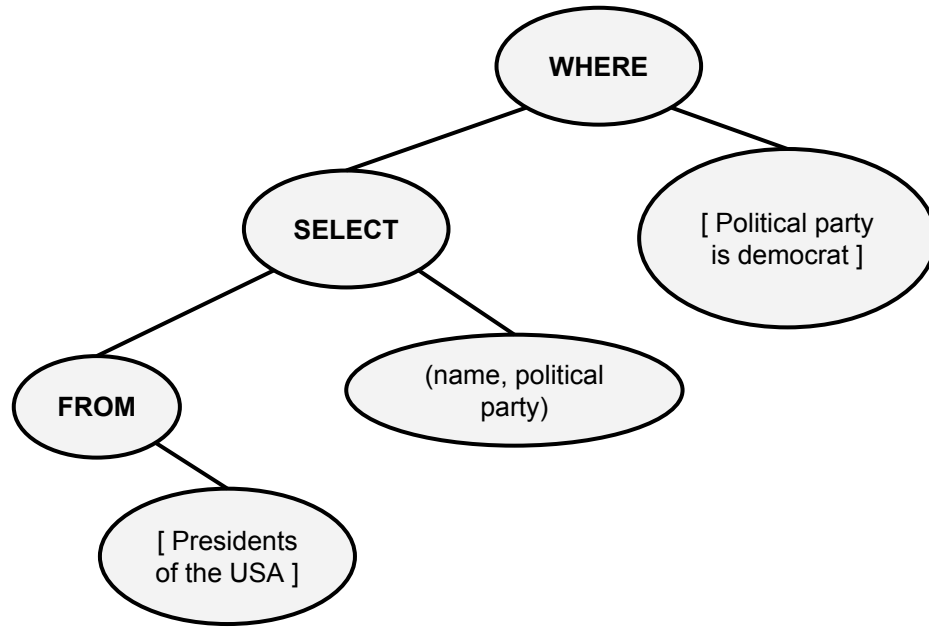
Additionally supported crowd-operators :

- JOIN
  - GROUP BY
  - ORDER BY
-

# Query tree

---

(*SELECT* (*name*, *political party*) *FROM* [*Presidents of USA*] *WHERE* [*party is democrat*])

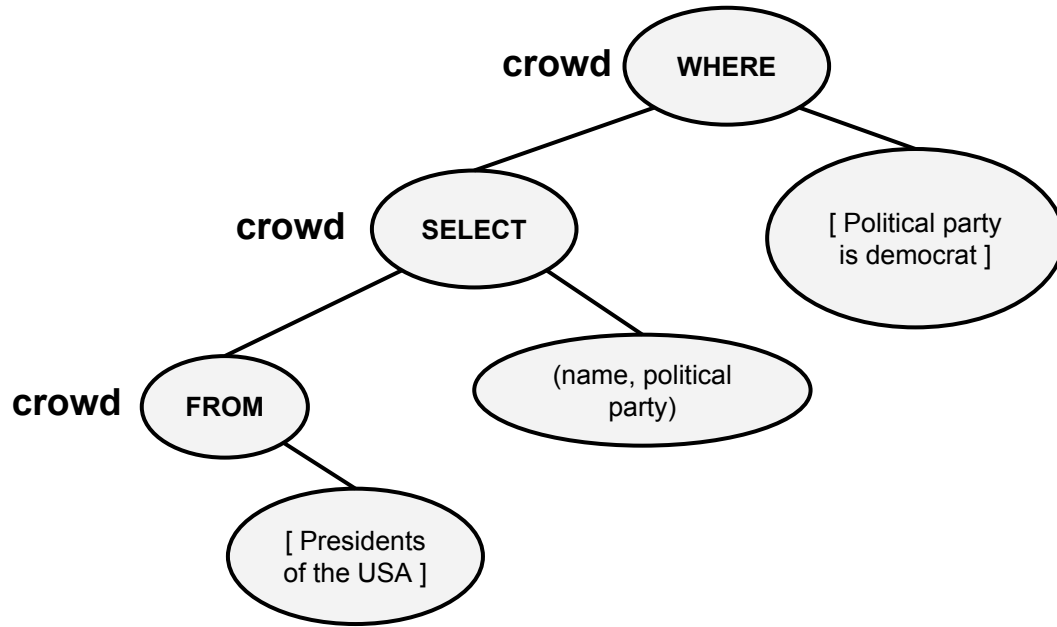




# Query tree

---

(SELECT (*name*, *political party*) FROM [*Presidents of USA*]) WHERE [*party is democrat*]



# Live demo

---

- We build a software (in Java/Scala) that runs on a server.
  - The core parts are :
    - The parser for our SQL-like language
    - API and communication with *Amazon Mechanical Turk* service
    - Query execution strategy, parallelization and pipelining.
    - Operators implementation (select, where, join, group by, order by)
    - Web interface for simple and assisted usage
-

# Conclusions

---

- Fun and trending topic !
  - Endless future work :
    - Query language enhancement
    - different algorithms implementations (JOIN, GROUP BY, ...)
    - results analytics (price, time, accuracy)
    - ...
  - We thank Immanuel Trummer for his suggestions and comments !
-