

# JUSTINE (JUST-INsert Engine): Demonstrating Self-organizing Data Schemas

Benjamin Hättasch, Leon Krüger & Carsten Binnig

## Relational DBs

- ✓ Great tool for data analysis & exploration:  
Efficient querying and easy browsing
- ✗ Manual design: Difficult to come up with a good schema
- ✗ High overhead for schema adaptation

## Schemaless DBs

- ✗ Exploration complex – what is stored and how?  
Requires schema-checking on read
- ✓ Just insert any data  
→ No need for schema design
- ✓ New kinds of data can be stored

## Our Approach

- ✓ Store data in a structured way  
→ Easy usage and exploration
- ✓ Derive schema autonomously from the inserts  
→ No overhead for schema design
- ✓ System automatically adapts to new needs

## Store data in a structured representation – without designing a schema

### Supported Scenarios

#### Complete query, matching schema:

```
INSERT INTO full_bottle_banks (location, timestamp) VALUES (...)
```

Full Bottle Banks	
Location	Report date
Katherine Johnson Sq	2025-02-28 12:34
...	...

#### Synonyms used:

```
INSERT INTO bottle_banks (place, timestamp) VALUES (...)
```

Match between query and schema

#### Table and or column information missing:

```
INSERT INTO full_bottle_bank VALUES (...)  
INSERT (place, timestamp) VALUES (...)  
INSERT VALUES (...)
```

By allowing users to omit table and/or column names, we truly relieve them from the need to think about the schema.

#### Additional columns not yet in schema:

```
INSERT (tour, date, capacity, full) VALUES (...)
```

Derive/choose column names and extend existing table

Garbage Collection Tours			
Tour	Scheduled for	Truck Capacity	Exceeded
5	2025-03-02	8	
...	...	...	

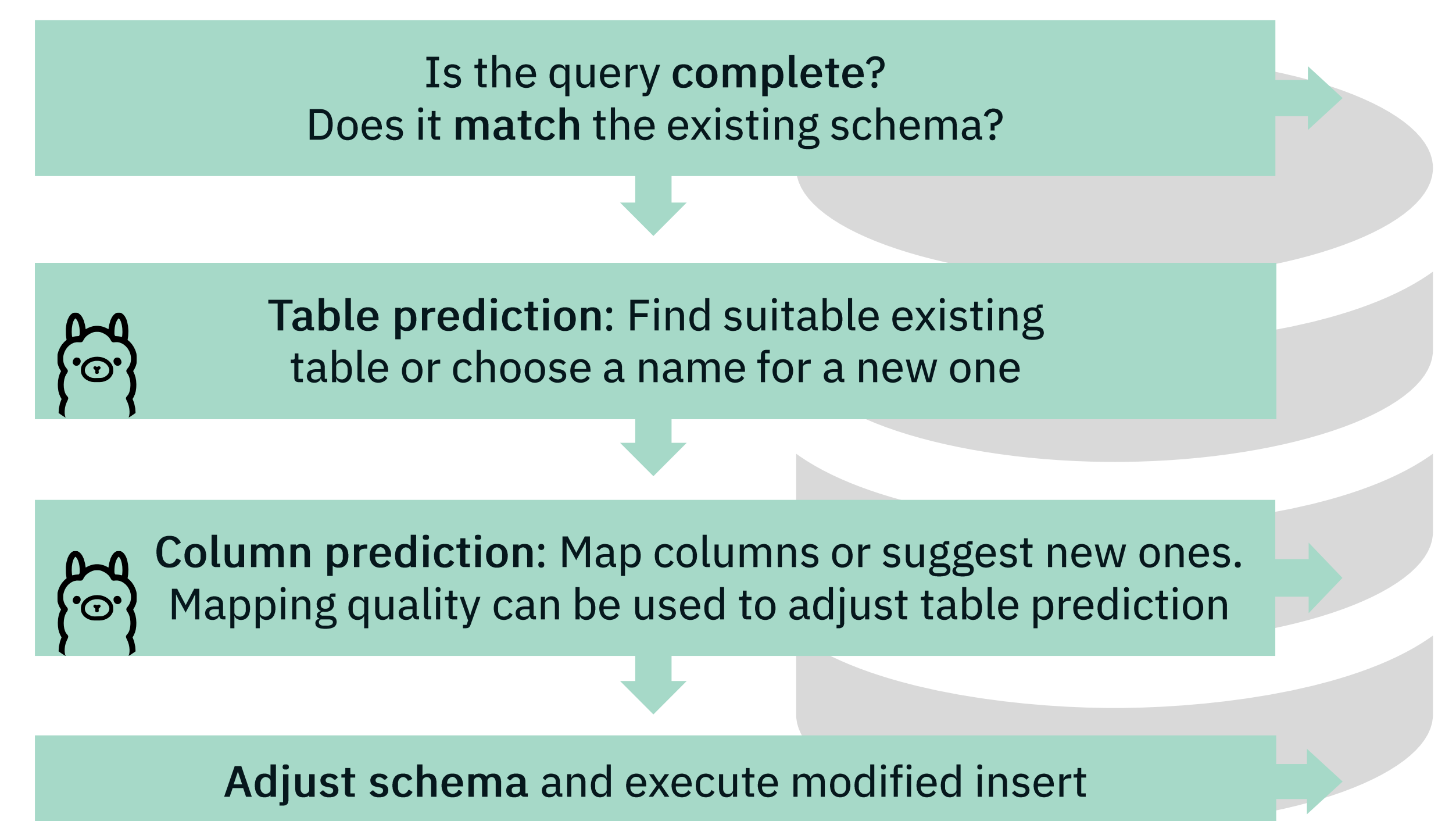
#### Entirely new data:

```
INSERT (street, no, company, timestamp) VALUES (...)
```

Derive design (including names) for a new table and add it to the schema

Scooter Reports			
Street	No	Company	Report date
...	...	...	...

### Insertion Flow of JUSTINE



### Our Models

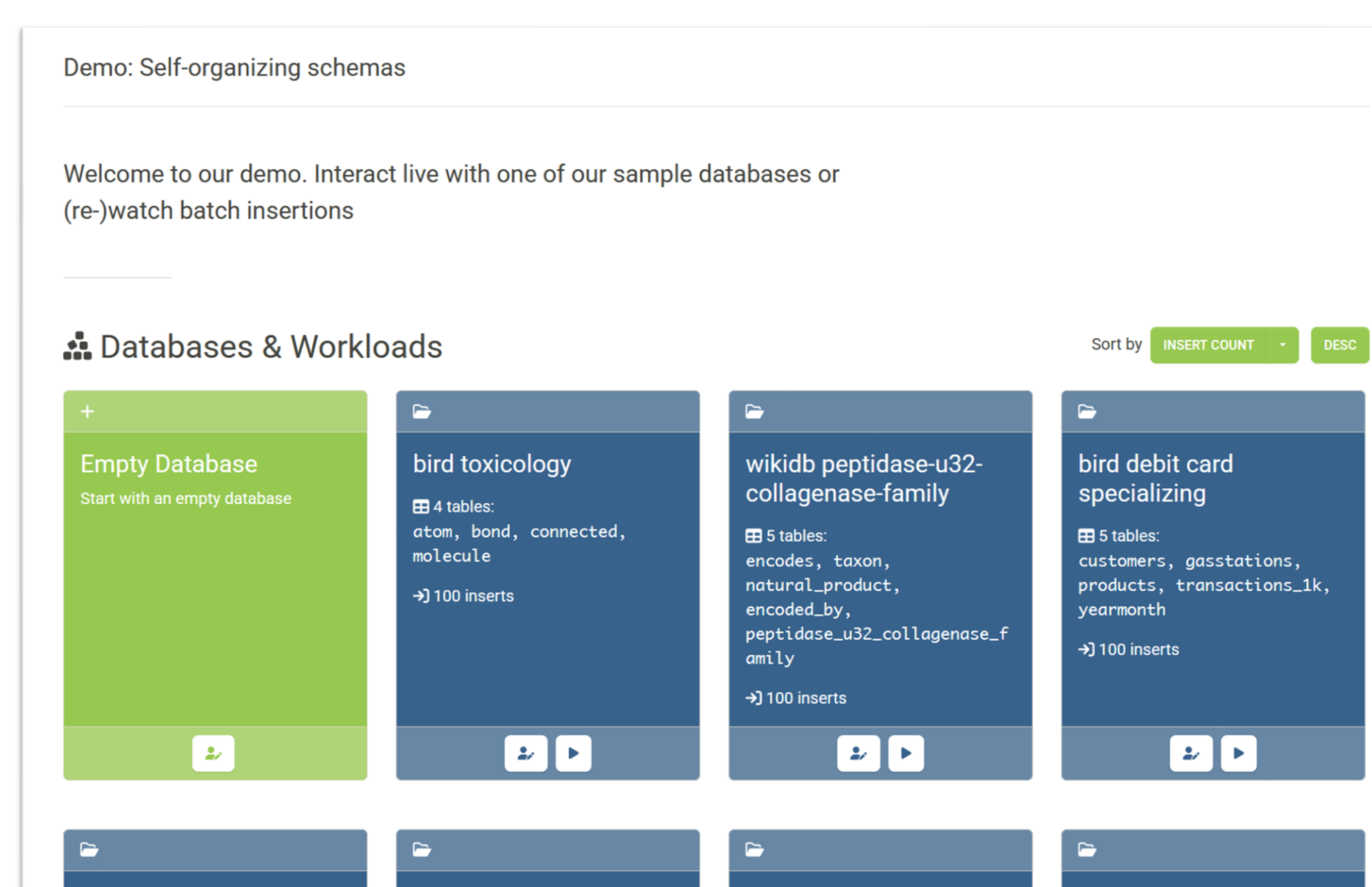
- Two Llama 3 models (8B parameters)
- Finetuned for table and column prediction
- Finetuned on a dataset curated from
  - 52 DBs from BIRD
  - 151 DBs from SPIDER
  - 99 DBs from WikiDBs
- Wide range of naming conventions
- Training cost reduction via QLoRA
- CSV-formatting for table representation in prompts

### Our Demo

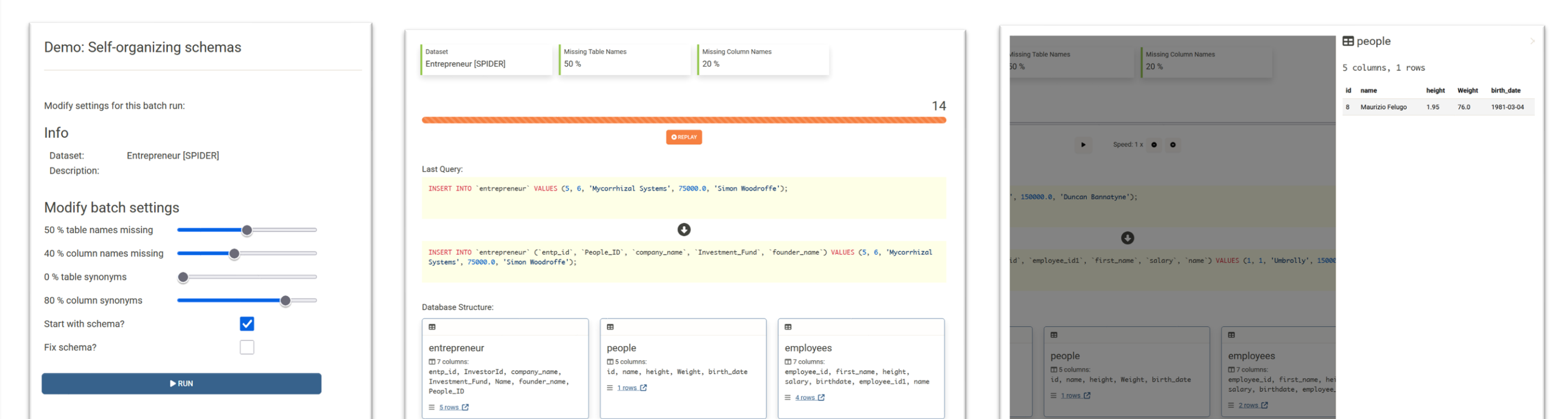
Our demo allows to try out JUSTINE interactively.

Users can issue own insertion queries, either on an initially empty database or for one of several existing DBs.

Or they can choose an existing workload and watch the schema evolve.



Users can modify properties of the workload (e.g., the percentage of inserts w/o table or column names) to steer the difficulty. The system then executes the workload, visualizing the database state after each query. Using the replay feature, users can return to a specific query to inspect the translated query or schema in detail.



This research and development project is partially funded by the German Federal Ministry of Education and Research (BMBF) within the “The Future of Value Creation – Research on Production, Services and Work” program (funding number 02L19C150). It has benefited from early-stage funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC-3057; funding will begin in 2026.

<https://link.tuda.systems/JUSTINE>

Dr. Benjamin Hättasch

FB SAIDE, DFKI Darmstadt & Systems Group, TU Darmstadt

benjamin.haettasch@dfki.de