SingleStore™

SingleStoreDB Python UDFs

Bringing the AI and ML Capability Closer To Data

Prabhat Sachdeva
Software Engineer

Agenda

- SingleStoreDB
- Python UDFs
- Aura Compute
- Vector UDFs
- Table Valued Functions
- AI Functions
- ML Functions
- JSON and more interesting types
- Performance and Scalability

Along with some Demos…

SingleStoreDB

SingleStore is a **real-time**, **distributed** SQL database that combines *transactional* and *analytical* workloads in **one** system to provide fast performance for intelligent applications.

It achieves high speed through in-memory processing, a **distributed** architecture, and universal storage that handles both row-based and column-based data needs.

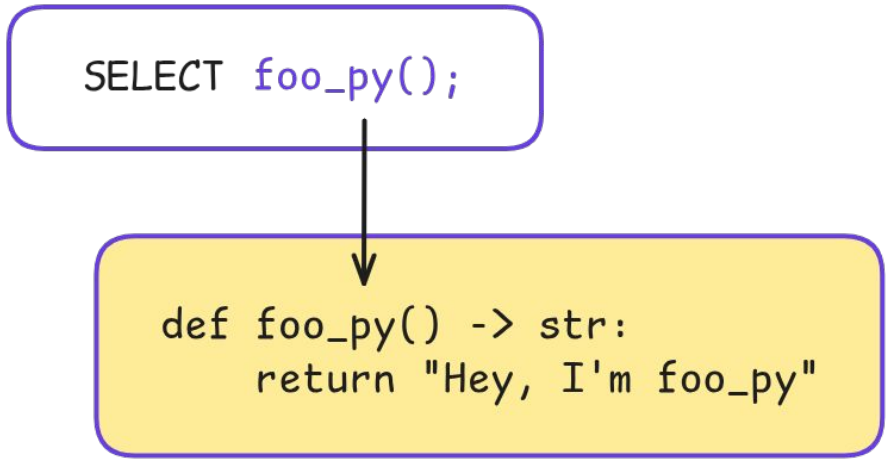
Python UDFs

Call Python functions directly from SQL :)

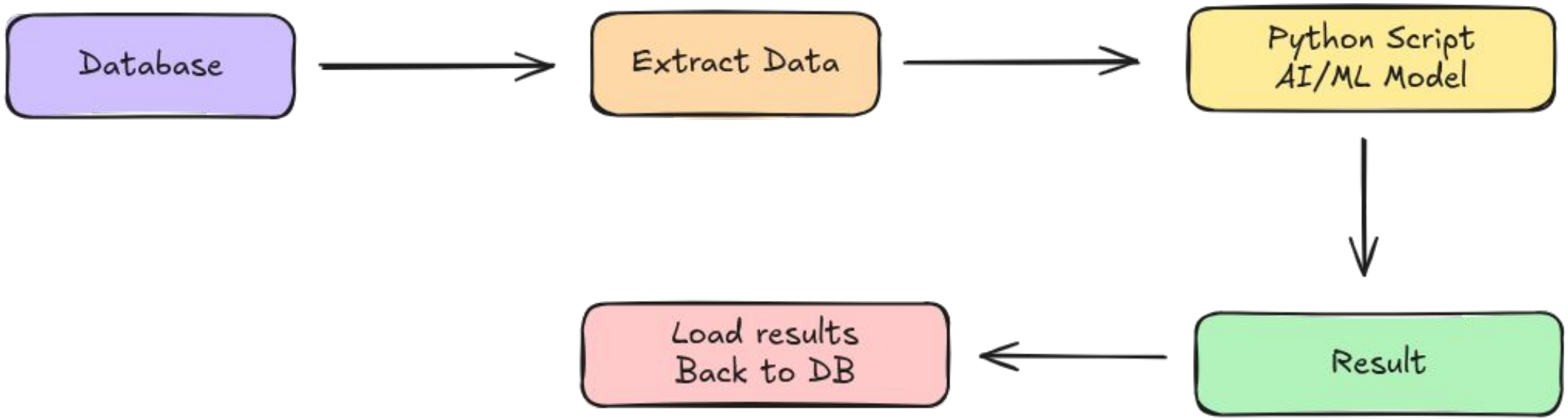
Brings AI/ML Capabilities right into the database.

Custom business logic that's too complex for SQL?
Too easy for Python!

Reduces (almost eliminates) data movement.



Without Python UDFs

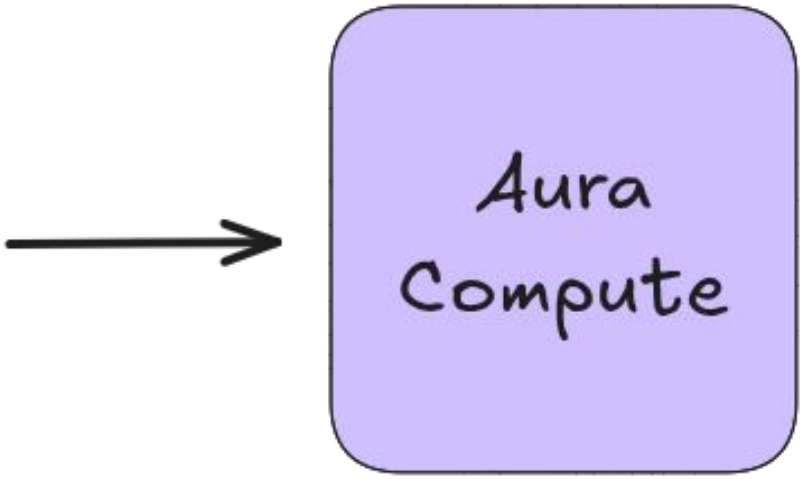


That’s too much data movement!

SingleStore™

With Python UDFs

```
SELECT
    product_id,
    predict_churn(features)
FROM customers;
```



Simple :)

SingleStore™

Make it more interesting and practical?

```
CREATE PIPELINE video_game_wikipedia_pipeline
AS LOAD DATA STAGE 'data.csv'
BATCH_INTERVAL 2500
DISABLE OFFSETS METADATA GC
SKIP DUPLICATE KEY ERRORS
INTO TABLE video_game_data_v2
FIELDS TERMINATED BY ',' ENCLOSED BY '"' ESCAPED BY '\\'
LINES TERMINATED BY '\n' STARTING BY ''
IGNORE 1 LINES
(id, url, @description)
SET
    description = @description,
    description_embedding = cluster.EMBED_TEXT(@description);
```

How to create one?

1. Define the Python UDFs in a Notebook
2. Try it out interactively :)
3. Publish It!
4. Use the UDF from within the SQL.

Demo Time 1 🎀

SingleStore™

Aura Compute

Our serverless platform

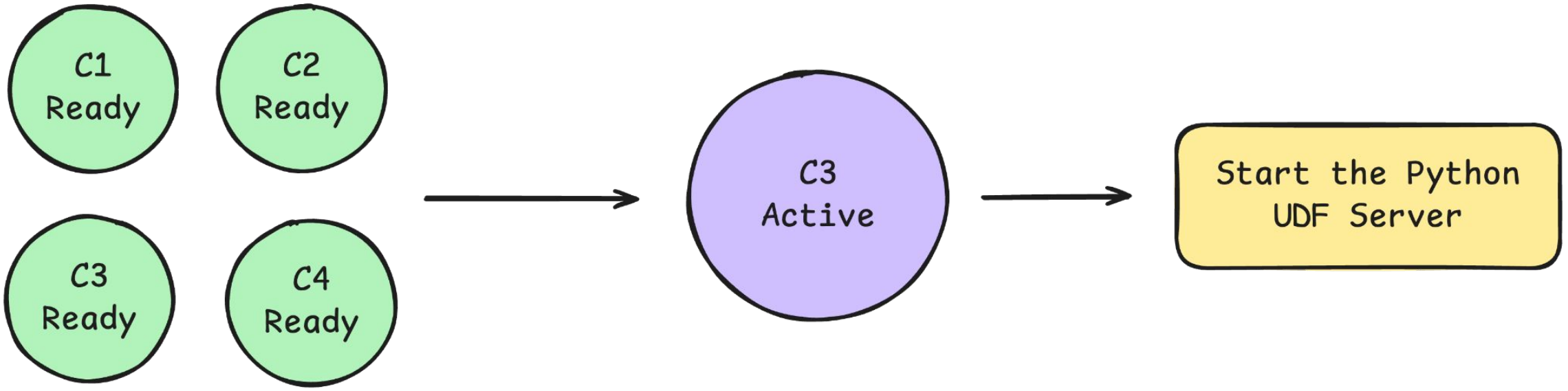
Enables running code securely in a sandboxed environment

Supports scaling!

- Let’s you choose the compute.
  - Even lets you choose a GPU 🔥
- Supports multi-replication.

Powering - Scheduled Jobs, Cloud Functions and more :)

How did that work!?

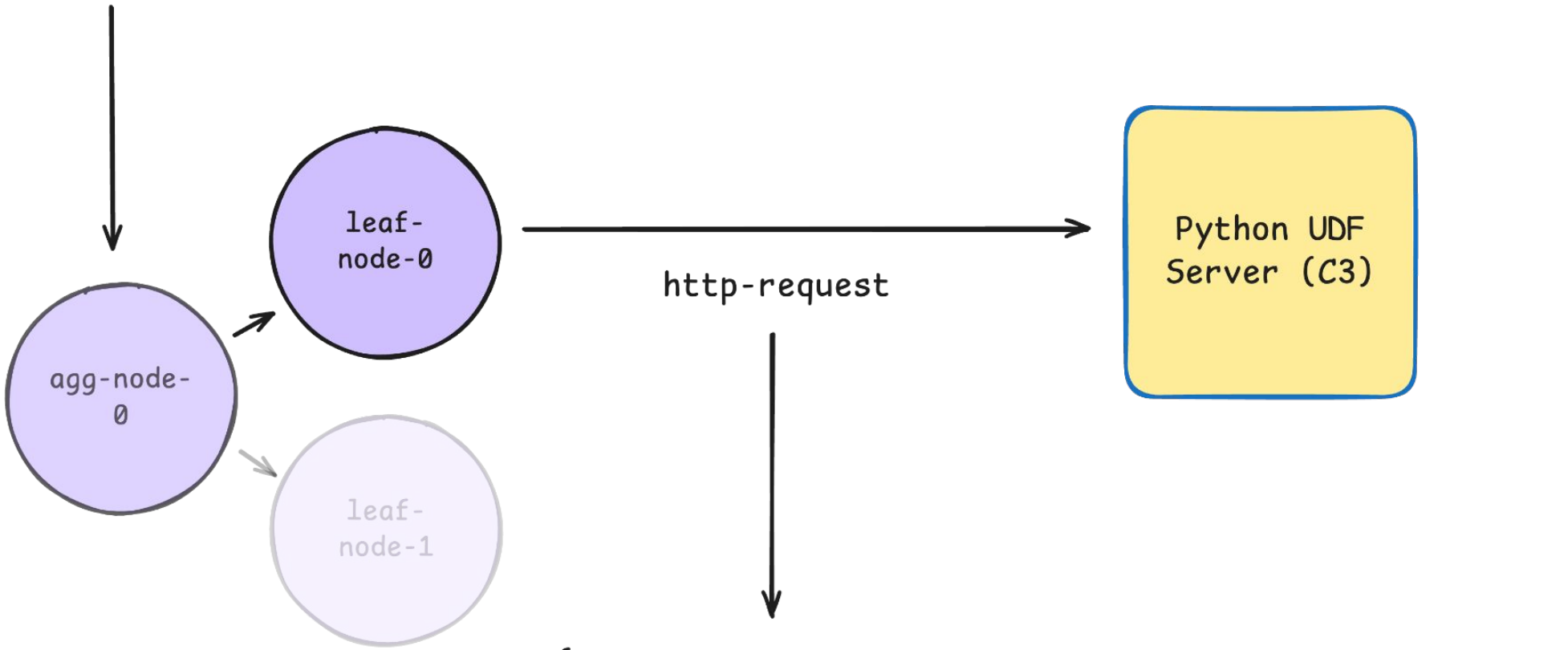


SingleStore™

```
@udf
def x_multiply(a: int, b: int) -> int:
    return a * b
```

```
CREATE EXTERNAL FUNCTION db.x_multiply(a INT, b INT)
RETURNS INT ... 'https://singlestore.com/udf-id'
FORMAT ROWDAT_1
...
```

```
SELECT x_multiply(a, b) from table;
```




```
{
    body: [[rowID, columns], ... by default 511 more]
}
```


SingleStore™

Vector UDFs

```
@udf
async def foo(arr: npt.NDArray[np.float16]) -> npt.NDArray[np.float16]:
    # This let's you process the whole batch in a single function
    # invocation.
    return arr * 2
```

Table Valued Functions (TVFs)

Returns a Table as a value.

```
@udf
def abc() -> Table[List[str, str, str]]:
    return [“a”, “b”, “c”]


SQL> SELECT a, b from abc();
```


SingleStore™

AI Functions

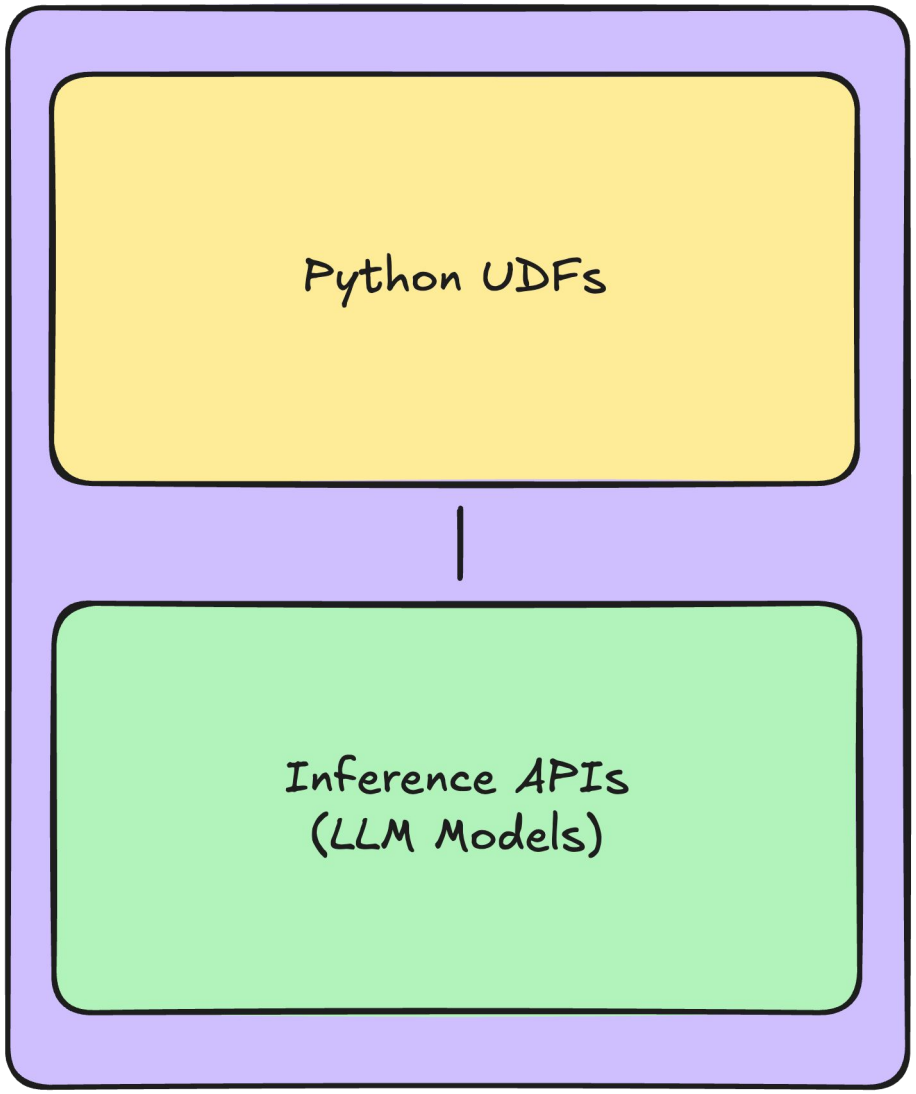
Pseudo builtins, that provide AI capabilities to the engine right out of the box.

EMBED_TEXT

AI_TRANSLATE

AI_SUMMARIZE

and more…



Demo Time 2 

ML Functions

Pseudo builtins, that provide ML capabilities to the engine right out of the box.

ML_CLASSIFY

ML_ANOMALY_DETECT

Demo Time 3 

JSON Type Support For Function Parameters

```
from singlestoredb.functions.typing import JSON

@udf

def transform_json_add_field(json_data: JSON, new_field: str, new_value: str) -> JSON:

    if isinstance(json_data, dict):

        result = json_data.copy()

        result[new_field] = new_value

        return result

    return json_data
```

Performance and Scalability

1. Choose the compute!
2. Multi-replicas :)
    a. Let’s you choose minimum and maximum replicas and supports auto-scaling 🔥
    b. GPU instances for heavy computation!

Thank You