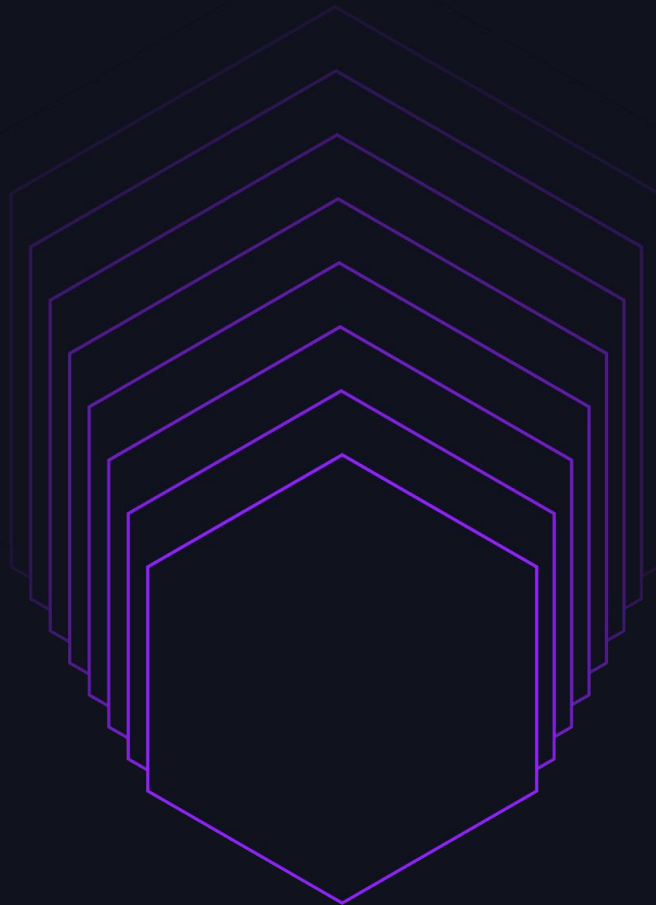


Streaming Cross-sectional Visualization

with

PERSPECTIVE

Tim Bess / Tim Paine



Talk Overview

- Background and Motivation
- Perspective – What it is, and what it does
- Connecting Spark Streaming / Perspective
- Introducing *Prospective*



Background and Motivation



Background – Ecosystem

Web frameworks and data visualization tools abound!



Despite the abundance...

...somethings are still...

clunky

Three Major Pain points

Virtualization

- Many tables require pagination
- Pagination not always convenient or suitable

“Dumb grids”

- Grids are often insufficient on their own
- Sorting / Filtering / Pivoting
- Other chart types

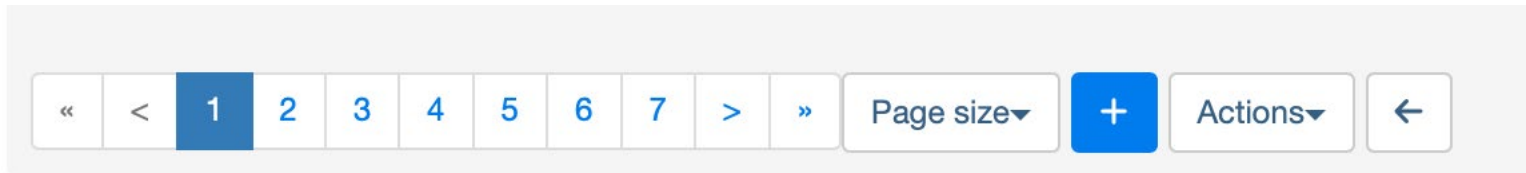
Streaming

- Static data sources / Streaming data sources / both
- Many clients – snapshot + update?

Pain Point 1: Virtualization

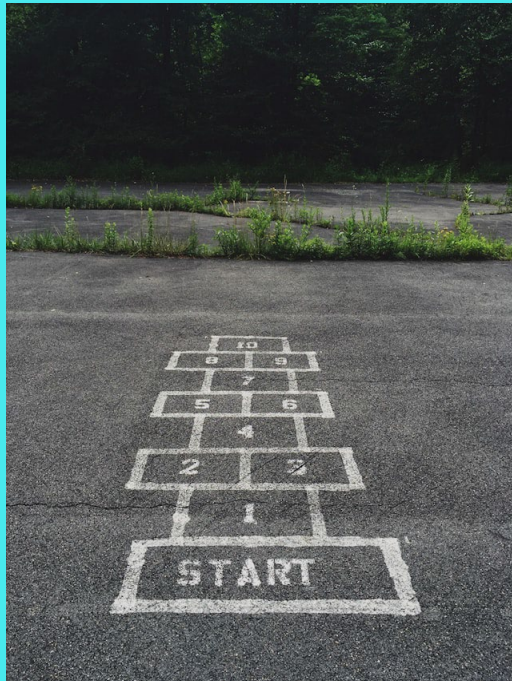


Pain Point 1: Virtualization



- Virtual tables allow for “infinite scroll”, lazy loading
- Absence of Virtualization leads to Pagination
- Pagination is often ill-suited to the task, can be annoying
- Most tools and frameworks **do not** support virtualization!

Pain Point 2: “Dumb Grids”



Pain Point 2: "Dumb Grids"

Grids need to do more

- Grids need to support basic features:

- Sort
- Filter

- But also some advanced features

- Row Pivots
- Column Pivots
- Aggregation Customization
- Computed Columns
- Spark Charts

- This is tricky to get right, both visually and in terms of the user experience

Group	2000			2002		
	sum(Gold)	sum(Silver)	sum(Bronze)	sum(Gold)	sum(Silver)	sum(Bronze)
▼ United St... (8)	7	1	5			
Nata... (2)						
Rya... (1)						
Allis... (1)						
Dara... (1)	2	0	3			
Gary... (1)	2	1	1			
Jenn... (1)	3	0	1			
Sha... (1)						
> Australia (6)	3	5	0			
> Canada (1)						
> Norway (3)				4	0	0

Pain Point 2: "Dumb Grids"

The problem with "dumb grids"

- Grids also should become more than grids
- When you have to configure everything on behalf of your user, it leads to unuseable "vanity" dashboards
- "Visualization Explosion"
- A good data visualization tool should allow for users to explore!



Pain Point 3: Streaming



Pain Point 3: Streaming

Streaming data is everywhere – build with it in mind

- All of the previous problems also apply, and need to work, on streaming data
- How do you manage N users, each with their own pivoted / sorted grid with custom computed columns?
- Difficult to implement strategies –
 - Polling + Pagination?
 - Snapshot + Updates?

Sample FastAPI Stream Server

```
from fastapi import FastAPI, WebSocket, WebSocketDisconnect
from fastapi.responses import HTMLResponse

app = FastAPI()
```

Sample FastAPI Stream Server

```
html = """
<!DOCTYPE html>
<html>
  <head>
    <title>Chat</title>
  </head>
  <body>
    <h1>WebSocket Chat</h1>
    <h2>Your ID: <span id="ws-id"></span></h2>
    <form action="" onsubmit="sendMessage(event)">
      <input type="text" id="messageText" autocomplete="off"/>
      <button>Send</button>
    </form>
    <ul id='messages'>
    </ul>

```

...

Sample FastAPI Stream Server

```
<script>
  var client_id = Date.now()
  document.querySelector("#ws-id").textContent = client_id;
  var ws = new WebSocket(`ws://localhost:8000/ws/${client_id}`);
  ws.onmessage = function(event) {
    var messages = document.getElementById('messages')
    var message = document.createElement('li')
    var content = document.createTextNode(event.data)
    message.appendChild(content)
    messages.appendChild(message)
  };
  function sendMessage(event) {
    var input = document.getElementById("messageText")
    ws.send(input.value)
    input.value = ''
    event.preventDefault()
  }
</script>
</body>
</html>
"""
```


Sample FastAPI Stream Server

```
class ConnectionManager:
    def __init__(self):
        self.active_connections: list[WebSocket] = []

    async def connect(self, websocket: WebSocket):
        await websocket.accept()
        self.active_connections.append(websocket)

    def disconnect(self, websocket: WebSocket):
        self.active_connections.remove(websocket)

    async def send_personal_message(self, message: str, websocket: WebSocket):
        await websocket.send_text(message)

    async def broadcast(self, message: str):
        for connection in self.active_connections:
            await connection.send_text(message)

manager = ConnectionManager()
```

Sample FastAPI Stream Server

```
@app.get("/")
async def get():
    return HTMLResponse(html)
```

```
@app.websocket("/ws/{client_id}")
async def websocket_endpoint(websocket: WebSocket, client_id: int):
    await manager.connect(websocket)
    try:
        while True:
            data = await websocket.receive_text()
            await manager.send_personal_message(f"You wrote: {data}", websocket)
            await manager.broadcast(f"Client #{client_id} says: {data}")
    except WebSocketDisconnect:
        manager.disconnect(websocket)
        await manager.broadcast(f"Client #{client_id} left the chat")
```

That's a lot!
And it doesn't do most of what we need...

PERSPECTIVE

What it is

What it does

PERSPECTIVE

Interactive analytics
and data visualization
component, which is
especially well-suited
for large and/or
streaming datasets



[finos/perspective](#)

Perspective

Key Facts and Features

Implementation

- C++ Engine
- Symmetric compilation on frontend (WebAssembly) and backend (Python)
- Client-only and Client/Server Architectures
- Rust/TS/JS Based UI Plugin System

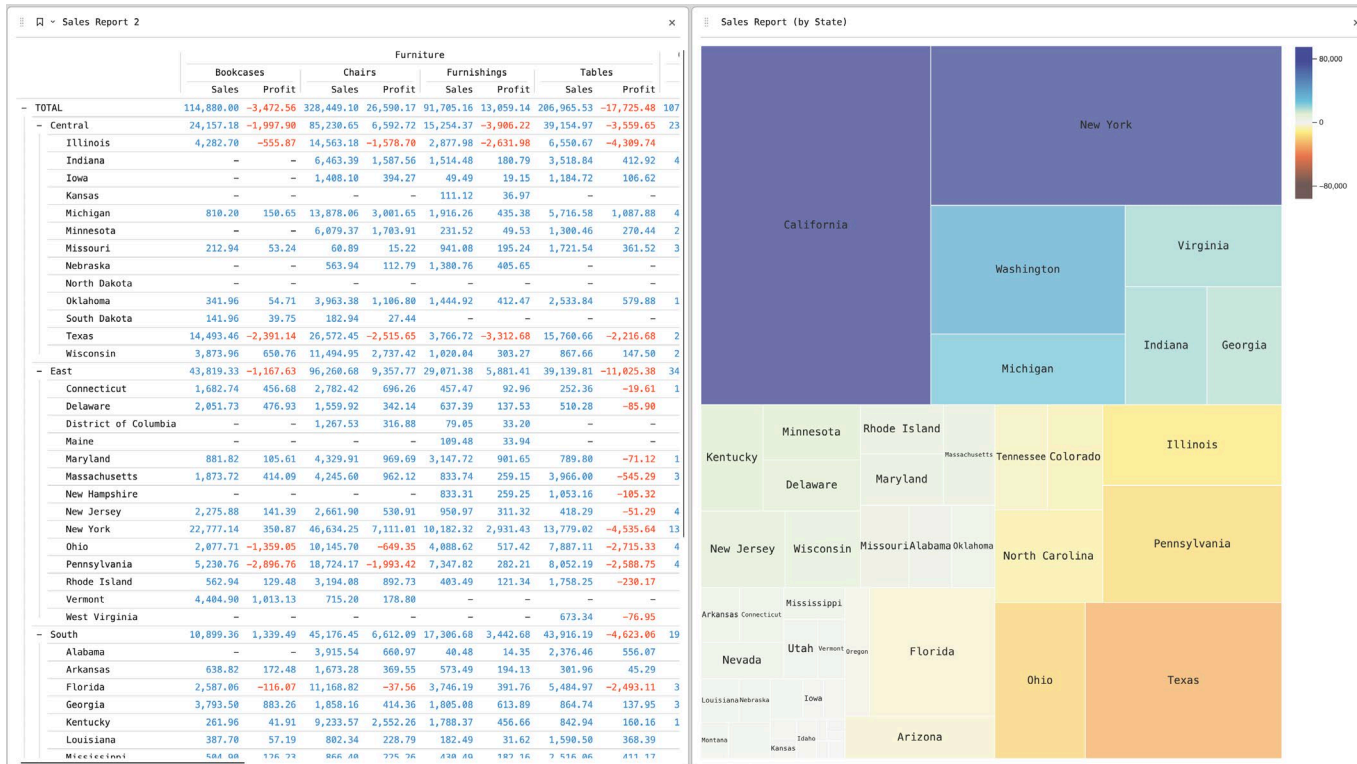
Frontend

- Web Components for ease-of-use
- JSON-based configuration, data and configuration are separate
- UI Plugins – Grid, Scatter, Bars, Lines, Maps, and more

Features

- Fast virtualization via Apache Arrow diffs
- Client/Server messaging protocol allows for alternative backends
- Exprtk-based expression engine for computed columns

Perspective



Let's run through the main features with an example dataset

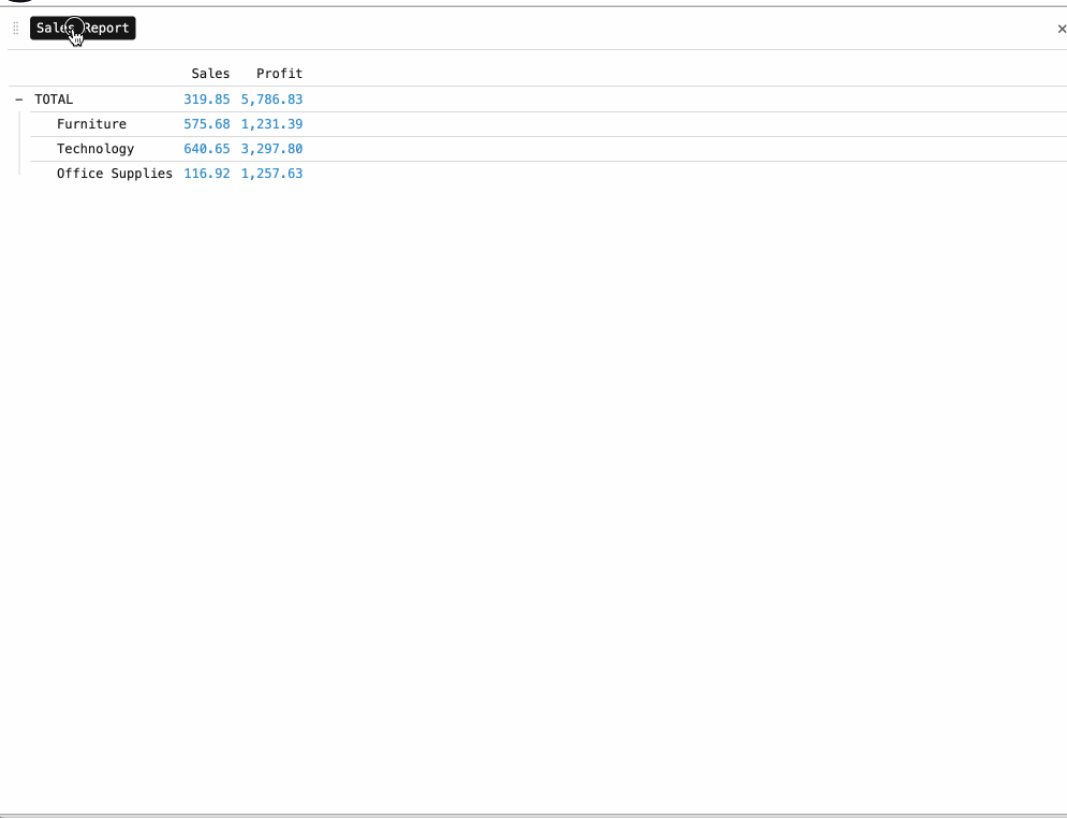
Sorting/Filtering/Pivoting

☰ Sales Report x

Order Date	Product Name	Category	State	Sales	Profit
11/7/16	Bush Somerset Collection Bookcase	Furniture	Kentucky	261.96	41.91
11/7/16	Hon Deluxe Fabric Upholstered Stacking Chairs, Rounded Back	Furniture	Kentucky	731.94	219.58
6/11/16	Self-Adhesive Address Labels for Typewriters by Universal	Office Supplies	California	14.62	6.87
10/10/15	Bretford CR4500 Series Slim Rectangular Table	Furniture	Florida	957.58	-383.03
10/10/15	Eldon Fold 'N Roll Cart System	Office Supplies	Florida	22.37	2.52
6/8/14	Eldon Expressions Wood and Plastic Desk Accessories, Cherry Wood	Furniture	California	48.86	14.17
6/8/14	Newell 322	Office Supplies	California	7.28	1.97
6/8/14	Mitel 5320 IP Phone VoIP phone	Technology	California	907.15	90.72
6/8/14	DXL Angle-View Binders with Locking Rings by Samsill	Office Supplies	California	18.50	5.78
6/8/14	Belkin F5C206VTEL 6 Outlet Surge	Office Supplies	California	114.90	34.47
6/8/14	Chromcraft Rectangular Conference Tables	Furniture	California	1,706.18	85.31
6/8/14	Konftel 250 Conference phone - Charcoal black	Technology	California	911.42	68.36
4/14/17	Xerox 1967	Office Supplies	North Carolina	15.55	5.44
12/4/16	Fellowes PB200 Plastic Comb Binding Machine	Office Supplies	Washington	407.98	132.59
11/21/15	Holmes Replacement Filter for HEPA Air Cleaner, Very Large Room, HEPA Filter	Office Supplies	Texas	68.81	-123.86
11/21/15	Storex DuraTech Recycled Plastic Frosted Binders	Office Supplies	Texas	2.54	-3.82
11/10/14	Stur-D-Stor Shelving, Vertical 5-Shelf: 72"H x 36"W x 18 1/2"D	Office Supplies	Wisconsin	665.88	13.32
5/12/14	Fellowes Super Stor/Drawer	Office Supplies	Utah	55.50	9.99
8/26/14	Newell 341	Office Supplies	California	8.56	2.48
8/26/14	Cisco SPA 501G IP Phone	Technology	California	213.48	16.01
8/26/14	Wilson Jones Hanging View Binder, White, 1"	Office Supplies	California	22.72	7.38
12/8/16	Newell 318	Office Supplies	Nebraska	19.46	5.06
12/8/16	Acco Six-Outlet Power Strip, 4' Cord Length	Office Supplies	Nebraska	60.34	15.69
7/15/17	Global Deluxe Stacking Chair, Gray	Furniture	Pennsylvania	71.37	-1.02
9/24/15	Bretford CR4500 Series Slim Rectangular Table	Furniture	Utah	1,044.63	240.26
1/15/16	Wilson Jones Active Use Binders	Office Supplies	California	11.65	4.22
1/15/16	Imation 8GB Mini TravelDrive USB 2.0 Flash Drive	Technology	California	90.57	11.77



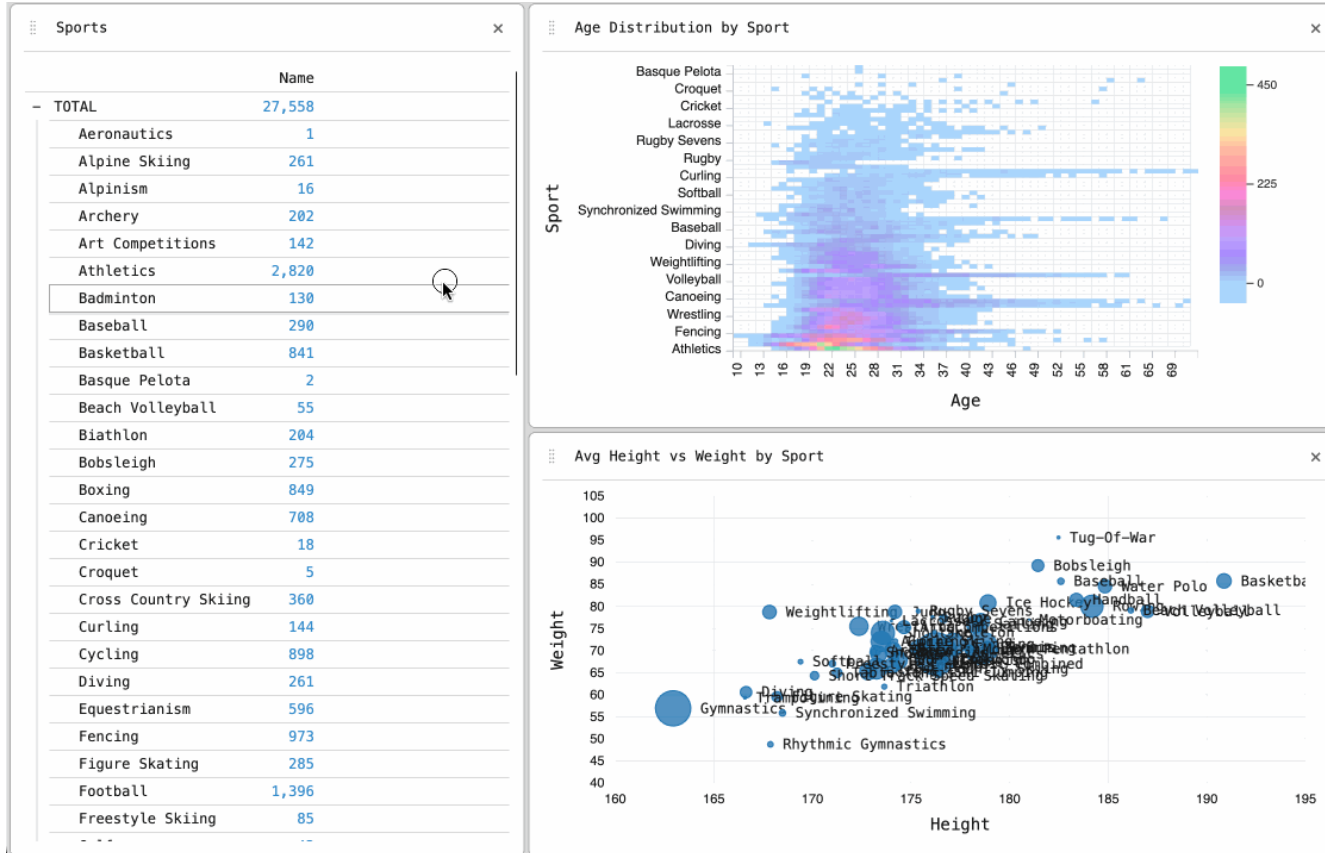
UI Plugins



The screenshot shows a UI plugin window titled "Sales Report" with a close button (x) in the top right corner. The window contains a table with the following data:

	Sales	Profit
- TOTAL	319.85	5,786.83
Furniture	575.68	1,231.39
Technology	640.65	3,297.80
Office Supplies	116.92	1,257.63

Fancy Stuff: Spark Bars / Crossfilter



Now let's bring it all together with a
full demo

Spark Streaming +

PERSPECTIVE

Spark Streaming + Perspective

An end-to-end example

- Let's build a simple but high-performance application
- Dummy dataset – Machines, Utilization, and Jobs
- Stream and aggregate data with Spark Streaming
- Feed Data into Perspective / FastAPI Server

Demo time...

Demo Code:



Introducing...

* PROSPECTIVE

* PROSPECTIVE

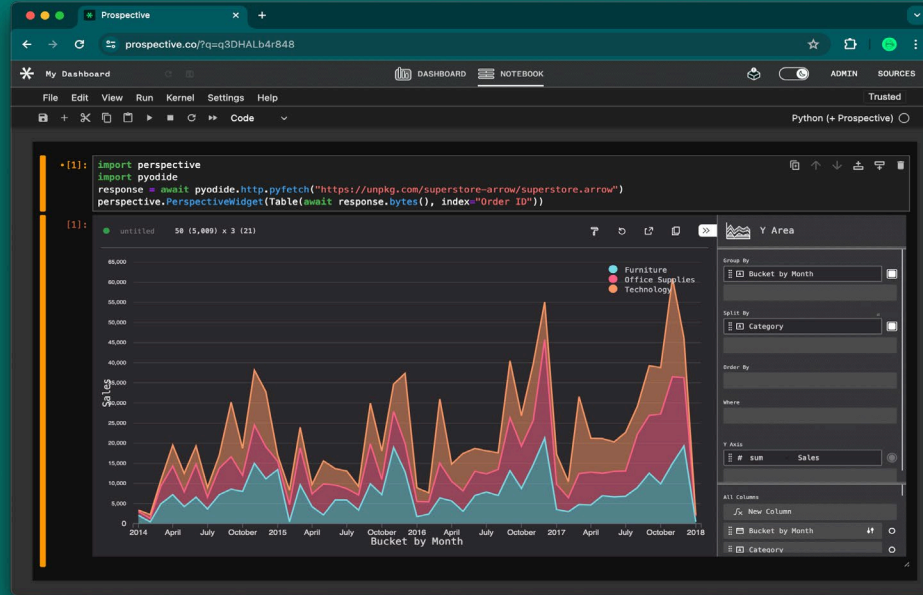
Data Adaptors for real-time, relational and archival data stores, e.g. Kafka, Postgres, S3, HTTP, Parquet, Excel and more.

Securely connect outside of the browser sandbox, without leaking access credentials to the cloud, via trustless proxy.

Persistent dashboards are shareable, forkable, downloadable, embeddable, portable.

Extend Perspective itself with Python in a browser-local, Pyodide-based Jupyter environment

NO CLOUD REQUIRED



* PROSPECTIVE

Data Adaptors for real-time, relational and archival data stores, e.g. Kafka, Postgres, S3, HTTP, Parquet, Excel and more.

Securely connect outside of the browser sandbox, without leaking access credentials to the cloud, via trustless proxy.

Persistent dashboards are shareable, forkable, downloadable, embeddable, portable.

Extend Perspective itself with Python in a browser-local, Pyodide-based Jupyter environment

NO CLOUD REQUIRED



DATA+AI SUMMIT

Q/A

Streaming Cross-sectional Visualization

with

PERSPECTIVE

Tim Bess / Tim Paine

