Peak ETL

Open-source Technology Success

Jason Holden

CEO, PRESENTOR

JASON@PEAKETL.COM | (918) 409-CODE (2633) | (719) 310-5240

About
Me

The Story (so far):

I started working with computers at the age of 13. Growing up in a small town in eastern Oklahoma, I acquired a reputation as the "computer kid." Shortly after graduating high school, I got my first software development job working with Visual FoxPro. Since then, I've used numerous other languages and platforms to develop countless database applications and services. My newest endeavor is Peak ETL. At Peak ETL, we're creating a subscription-based platform to aggregate and analyze public oil and gas data and deliver it to industry professionals for use in their operations.

Jason Holden
CEO & Founder

Open-source Development

Why Open Source?

- **Cost:** Products like MySQL and PostgreSQL reduce licensing costs compared to commercial solutions like Microsoft SQL Server and Oracle.
- **Flexibility:** Provides unparalleled flexibility, allowing developers to modify source code, integrate diverse technologies, and deploy applications across various platforms without vendor lock-in
- **Community:** Active and talented communities contribute to rapid development, troubleshooting, and feature enhancements
- **Transparency:** Community members can review and audit the source for security vulnerabilities
- **Everywhere:** Open source is everywhere, from tech giants (including Microsoft) to little startups (like Peak ETL)

Cost

Reduce licensing costs compared to commercial solutions.

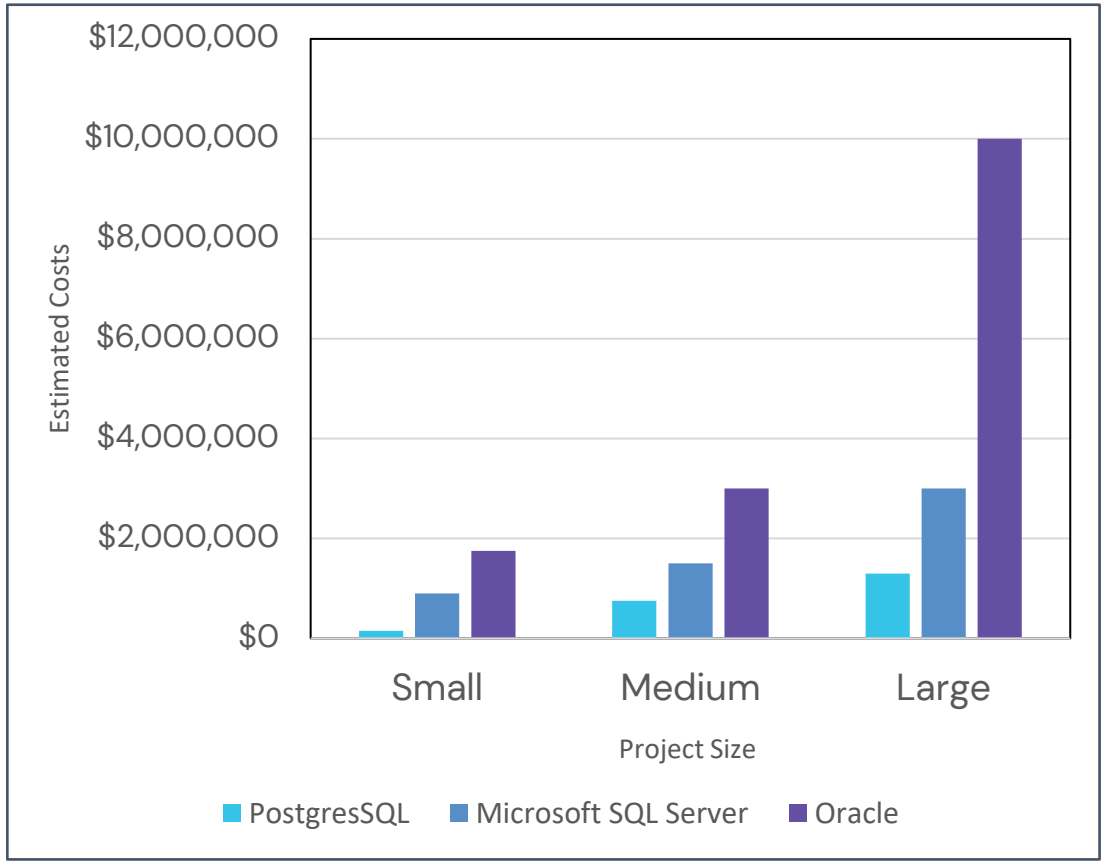
Advancement

Open-source projects advance faster.

Customizable

Modify for specific business needs.

Open Source vs. Commercial Comparison



Sources: servermania.com and redresscompliance.com

Small: < 100 Users / 1 – 2 TB | **Medium:** 100 – 1000 Users / 5 – 10 TB | **Large:** 1000+ Users / 10+ TB

Pros

- Lower licensing cost (usually no licensing costs)
- Highly customizable
- Community-driven for fast-paced advancement
- Open access to source code for auditing
- Easily scalable and modular components

Cons

- Higher cost for talent driven by a smaller pool
- Lack of formal support channels
- Multiple vendors can add complexity and challenge integration efforts

Our Stack

The Best of All Worlds

- **Code:** C#, Java, Python, and JavaScript
- **Database:** Postgres, Microsoft SQL, and MongoDB
- **Source control:** Github.com
- **OS:** Linux and Windows

Demo
WSL

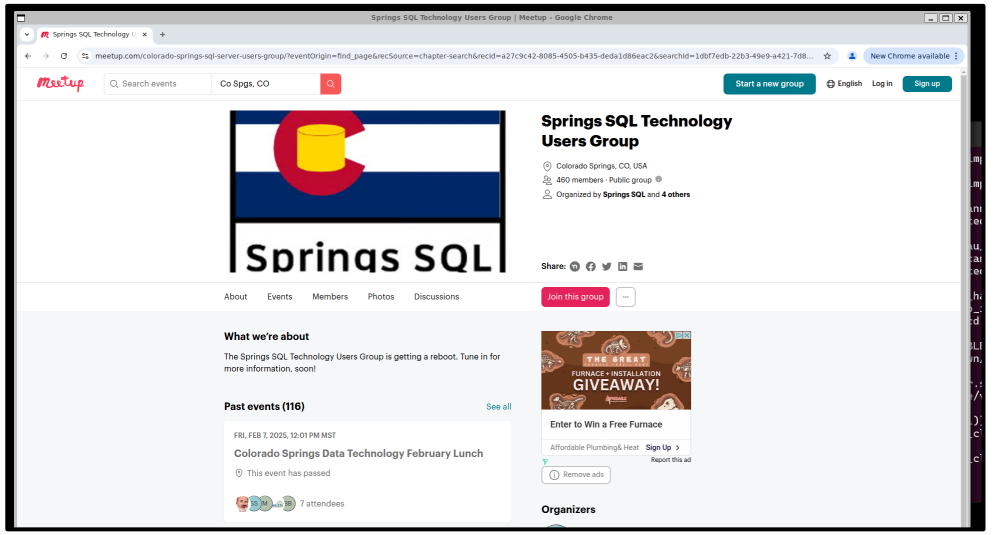
Windows Subsystem for Linux

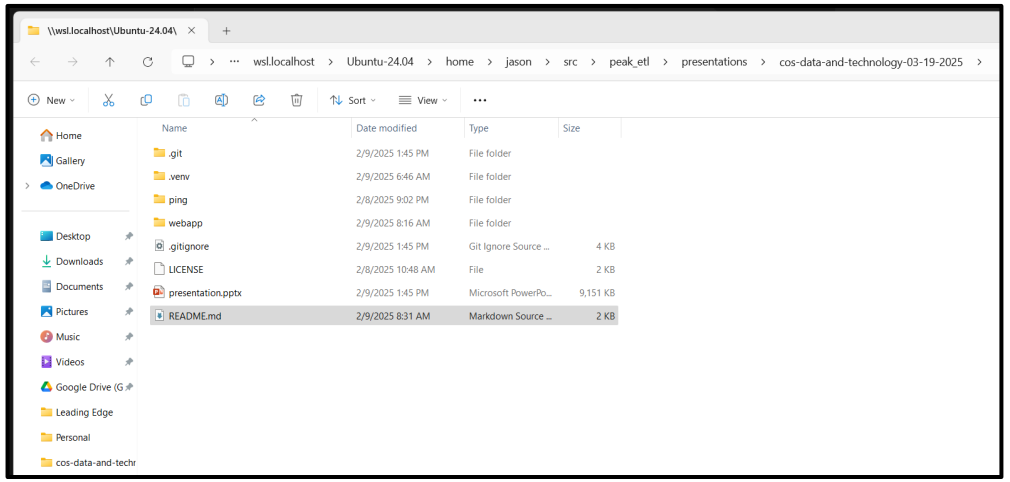
- **Integration:** Seamless integration between Windows and Linux
- **Fast:** Fast startup with low resource usage
- **GPUs**: Use the latest AI toolsets directly from Windows with full hardware acceleration (WSL 2)
- **Support**: Supports lots of popular Linux distributions
- **Multi-Instance**: Install more than one distribution
- **Linux Apps:** Run Linux GUI applications through Windows

NOT FOR PRODUCTION USE

System Requirements

- Recommend using WSL 2 for a genuine Linux kernel and GPU support
- Windows 10 (build 1909) or Windows 11
- **DO NOT USE** the Windows App Store





Demo
PING

PING a remote server

- **Code:** `ping.sh` to execute PING and log to a file
- **Container:** `Dockerfile` maps /app folder on the host for easy access to log files

```
$ ping.sh
#!/bin/sh

# Start the Python script, use full paths
ping "google.com" -i 5 >> /app/logs/ping.log 2>&1
```

ping.log

ping > app > logs > ping.log

```
1  PING google.com (142.250.191.142): 56 data bytes
2  64 bytes from 142.250.191.142: seq=0 ttl=63 time=35.637 ms
3  64 bytes from 142.250.191.142: seq=1 ttl=63 time=38.328 ms
4  64 bytes from 142.250.191.142: seq=2 ttl=63 time=38.118 ms
5  64 bytes from 142.250.191.142: seq=3 ttl=63 time=40.066 ms
6  64 bytes from 142.250.191.142: seq=4 ttl=63 time=37.255 ms
7  64 bytes from 142.250.191.142: seq=5 ttl=63 time=36.508 ms
8  64 bytes from 142.250.191.142: seq=6 ttl=63 time=40.346 ms
9  64 bytes from 142.250.191.142: seq=7 ttl=63 time=38.428 ms
10 64 bytes from 142.250.191.142: seq=8 ttl=63 time=37.815 ms
11
```

Demo
Web Application

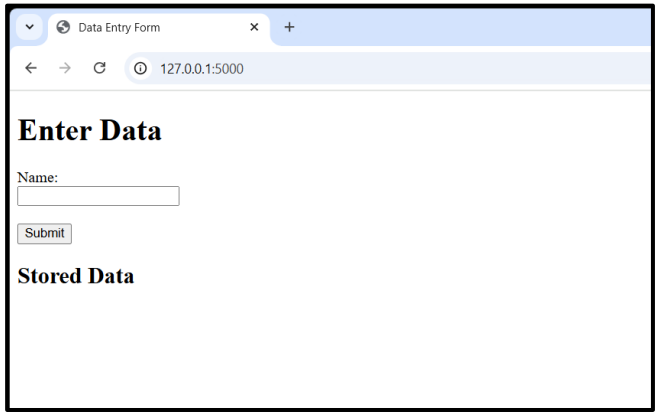
Flask Web Application

- **Code:** `app.py` Flask application with single data entry form
- **Containers:** `App_Dockerfile` (Python) and `DB_Dockerfile` (Postgres)

app.py

webapp > app.py > ...

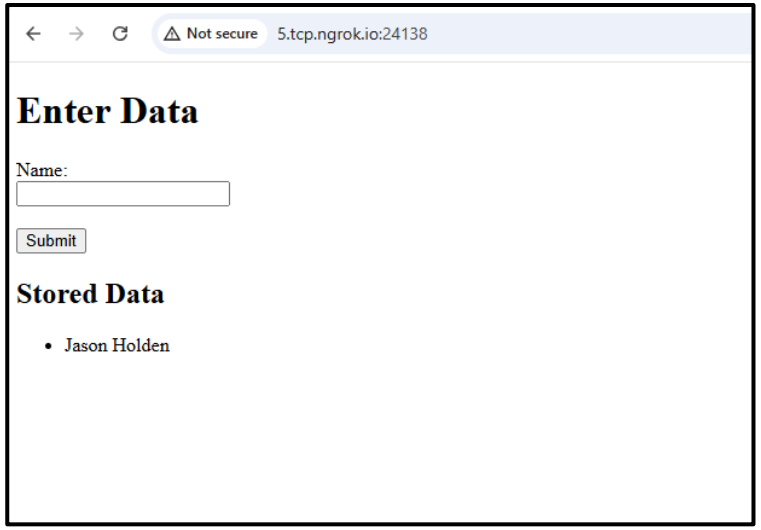
```python
import os
from flask import Flask, request, jsonify, render_template_string
import psycopg

app = Flask(__name__)

# Database connection parameters
# This is a simple example, in production you should use environment variables
db_params = {
    "dbname": "postgres",
    "user": "cos_data_tech",
    "password": "cos_data_tech",
    "host": os.getenv("POSTGRES_SERVER", '127.0.0.1'),
    "port": 5432
}

# HTML Template for the form
form_template = """
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
```



Demo
Publishing

Publishing Containers

- **Build.sh**
  - Build App and DB docker images into portable `tar` files
- **Publish.sh**
  - Call `./Build.sh`
  - Use `scp` to copy the necessary files to the remote server
  - Use `ssh` to automate Docker on remote server and publish images



Demo
GPU

Publishing Containers

- **app.py**
  - PyTorch demo showing tensor benchmarks
- **app.sh**
  - Runs `app.py`
  - Logs results `/app/logs/gpu.log`
- **loop.sh**
  - Executes `./app.sh` in a loop
- **docker-compose.yml**
  - Special directives to enable GPU

```
CUDA is available.
GPU Name: NVIDIA RTX 4000 Ada Generation Laptop GPU
Total GPU Memory: 12281 MB
Initial Memory Allocated: 0 MB

Running on device: cuda
Starting benchmark with 10 outer iterations, 30 inner loops each...

[1/10] Time: 0.81s | GPU Mem: 422 MB
[2/10] Time: 0.71s | GPU Mem: 422 MB
[3/10] Time: 0.73s | GPU Mem: 422 MB
[4/10] Time: 0.75s | GPU Mem: 422 MB
[5/10] Time: 0.74s | GPU Mem: 422 MB
[6/10] Time: 0.73s | GPU Mem: 422 MB
[7/10] Time: 0.71s | GPU Mem: 422 MB
[8/10] Time: 0.70s | GPU Mem: 422 MB
[9/10] Time: 0.71s | GPU Mem: 422 MB
[10/10] Time: 0.70s | GPU Mem: 422 MB

Finished 10 iterations.
Total time: 7.30s | Avg per iteration: 0.73s
```

Notes

- Requires WSL2
- GPU driver needs to be greater than or equal to the version installed in the Docker image

Key
Take Aways

Open Source is the Future

- **Development:**
  - Open source can jump-start your development
  - Allows developers to define their infrastructure, significantly speeding up development
- **Not mutually exclusive:**
  - Run Linux application on Windows
  - Run Windows applications, including SQL Server, on Linux
- **Future proof:**
  - Leading AI toolsets are built on open-source technology
  - Microsoft uses Linux extensively in their Azure cloud platform

Contact Us

GitHub

https://github.com/peaketl/open-source-success-presentation

CONTACT

(719) 310-5240
(918) 409-CODE (2633)

www.peaketl.com
jason@peaketl.com