Chester "Chet" Weger

chester.weger@gmail.com || (909)706-2026 || chet-weger.herokuapp.com

Mindset: Develop the cheapest adequate solution. Understand best practices. Use DataFrames as composable SQL! Setup turnkey CI/CD.

Summary

Software/data engineer with broad experience, passionate about delivering ***simple***, ***maintainable***, and ***correct*** Data-Pipelines, and Reporting and Insights products. Strong understanding and experience applying *relational data modeling* and *data warehousing* concepts such as *normalization*, *star schema*, and *constraints*. Experience developing *performance critical data pipelines* processing billions of transaction logs per hour, and performance tuning *Druid*. Quickly research, discover, and evangelize idioms and best practices for newly adopted frameworks and tools. Ability to provide and receive constructive criticism and admit if the current implementation of a broader strategy isn't working! Involved in a series of successful migrations. Migrated a series of legacy Hadoop jobs to Spark. Helped rewrite a reporting application backed by MySQL to use Druid. Migrated a suite of data pipelines scheduled from Azkaban with limited CI/CD to a new platform based on a stack including Databricks & *Spark*, *Airflow*, and Gitlab *CI/CD*.

Career Highlights

- Databricks training – completed 3-day onsite training course and discussed best practices and approaches to solving business needs.
- CI/CD wins – helped evangelized a shift from a "Monorepo" deployment philosophy using the *Pants* tool to a *variant* of the *12-factor App* using Gitlab with mainstream, off-the-shelf build and deploy tools, and application-specific credentials encrypted using sops and added directly to Git.
- Functional Programming in Scala, – completed five part Coursera specialization, improving my understanding of functional programming and Spark.
- Druid Performance Tuning – worked with Imply.io to implement optimizations for Druid querying and indexing. Benchmarked potential performance optimizations to evaluate cost effectiveness. Achieved over 40% reduction in average report query time.
- Won Internal Hackathon "Automation Prize" – used Sankey chart to visualize the flow of money among marketplace entities.
- MongoDB Fiasco – patched an urgent fire preventing output from Hadoop job from successfully being inserted into MongoDB caused by documents exceeding the maximum BSON document size. Used GZIP to compress bid-distribution objects to *~10%* of the original size.

Experience

Senior Data Engineer – *Walt Disney Studio Technologies* via *Insight Global* – Burbank, CA Sept 2018 – Present

- Supported and addressed feature requests for legacy jobs scheduled using Azkaban with ETL logic in Spark1.6 or Pandas.
- When I joined, Studio Technologies had not yet finalized a full contract with Databricks and was in the process of evaluating a POC Databricks environment. I acted as a guinea pig on the POC Databricks and helped raise important questions around issues such as library management, access control, deploy methods and CI/CD, and best approaches to migrate existing jobs to Databricks. The Databricks contract was finalized and became a part of the new platform along with *Apache Airflow* for job scheduling and *Gitlab* for CI/CD.
  - Used the Python SDK to copy Google Sheets to S3, rather than a legacy solution that was difficult to integrated with Databricks.
  - Applied scrutiny to a proposed "orchestrator" framework which mixed Apache Airflow and Kafka and raised questions such as what business needs the support for streaming would solve.
  - Proved the usefulness of Databricks notebooks as a first-class deployment artifact and a building block on the new platform.
  - Reviewed design documents for the new platform. Suggested ideas, some of which were later used or addressed, such as enabling object versioning on S3 buckets storing raw data, enabling library isolation for PySpark notebooks.
- Migrated and developed new features for existing jobs that used sources including Movio, Adobe Analytics, Innovid, and other data providers. Performed rigorous comparisons during the migration process to avoid bugs. Added common functionality to a new helper library *etl_lib.*
- Developed new data pipelines for marketing data and wrote ingests pulling from google Display Video 360 and Facebook Insights API.
- Worked with product to apply data transformation logic and data cleaning. Implemented simple and creative ways to get around external API limitations while satisfying business requirements such as frequently updated metadata.

Software Development Engineer – *The Rubicon Project* – Playa Vista, CA July 2014 – Sept 2018

- Wrote new Spark job for reporting on Rubicon's "Prebid" traffic. Used Amazon EMR for rapid deployment and scaling.
- Designed & Developed Spark job for "First-Party" dataset persisted in Druid to sunset an older reporting product and add new functionality.
- Migrated several legacy Hadoop jobs to Spark using Spark SQL and Dataset API, increasing performance and maintainability.
- Contributed to data-pipeline, API, and application layer for next-gen performance reporting product based on Druid capable of ingesting over a billion aggregated rows per month and allowing an arbitrary selection of over 40 dimensions and over 30 base and derived metrics.
  - Designed new tables, built REST API, and developed application for managing *saved*, *standard*, and *scheduled* reports.
  - Developed new dimension columns such as *Site Groups* that represent many-to-many relationships to deprecate legacy reporting.
  - Worked with *Imply.io* to improve Druid performance. Worked with DevOps engineers to upgrade Druid to the next version.
- Contributed to *Blocked Advertiser Revenue* and *Optimized Price Floors* which produce recommendations based on historical data.
- Developed ETL to ingest financial data into MySQL from a new buy-side product. Worked with QA to ensure data accuracy.
- Performed maintenance and performance optimization on a legacy OLAP reporting product based on *MySQL*.
- Helped build new API and application for managing "seats" which represent the buyer/customer on Rubicon's exchange.

Student – *HMC* Clinic Program with *The Rubicon Project – Respectful Cross Device Attribution* *Sept 2013 – May 2014*

- Worked with three other students, an academic advisor, and a liaison from *The Rubicon Project* to gain industry experience and explore novel approaches to tackling a big-data problem.
- Project's goal is to be able to match devices – to be able to say these two devices might belong to the same person.
- Played a key role in creating the design for the project, pushed for similarity hashing as a technique to group device profiles.
- Researched, designed, and implemented similarity measures that operate on pairs of device profiles.

Software Engineering Intern – Pagewoo/Nearwoo (now defunct) June 2013 – Aug 2013

- Developed a module to automatically generate information about a user's business. Performed integration with the yelp business search API as well as building a scraping component to find, among other things, relevant pictures that would be useful in banner ad creation.
- Performed other miscellaneous scraping and web development tasks.

Technical Assistant – *Pomona College*; CS052: Foundations of Computer Science Aug 2012 – Dec 2012

- Responsibilities included grading homework using an automated script and holding office hours to answer questions on homework problems.

Skills

- Design, Architecture, and Fundamentals
  - Ability to *identify* & communicate *urgent* or *important* information clearly & precisely; and give and receive *constructive criticism*.
  - Strong knowledge of *data-structures*, *algorithms*, and CS fundamentals. Ability to understand and apply new concepts.
  - Pragmatic ability to balance the benefits of *functional programming*, without harming code readability and comprehension.
  - Experience applying Object-Oriented concepts such as *Dependency Inversion* to write *testable* and *maintainable* code.
  - Ability to write readable and idiomatic code in a variety of languages including *Python*, *Scala*, *JavaScript*, *Java*, *Bash*, and *SQL*.
- Data Engineering
  - Experience applying Data Engineering best practices such as making tasks idempotent, and separating ingestion from processing.
  - Experience using *Apache Airflow* to schedule jobs and effortlessly perform common tasks such as running backfills.
  - Extensive experience using *Spark DataFrames API* including understanding of idioms and best practices. Ability to troubleshoot Spark Jobs and fix performance issues; knowledge of *Spark's* architecture and internals such as the *Hive Metastore*.
- DevOps & CI/CD
  - Experience using Gitlab and Jenkins to automate testing and deployment. Used build tools including *setuptools*, *sbt*, and *maven*.
  - Familiar using a local Kubernetes environment with *minikube*, *kustomize*, and *skaffold*. Proficient *Dockerizing* python apps.
- Web Development
  - Experience using *Django*, *Flask*, and other frameworks to develop *RESTful* HTTP services following best practices.
  - Experience using *ReactJS*, and components such as *datatables*, *daterangepicker*, and *highcharts* to develop SPAs.
- Experience using *Druid* for OLAP reporting on extremely high-volume transaction logs.
- Relational Databases – *MySQL* & *PostgreSQL*
  - Experience with *data-modeling* for OLAP & OLTP apps. Experience applying *data warehouse* design and concepts.
  - Experience troubleshooting *performance* issues with complex application-generated *OLAP* queries.

Education

Bachelor of Arts, *Computer Science*, *Minor in Mathematics*, *2014*, **Pomona College**, Claremont, California

Personal Projects

- Interactive Tic-Tac-Toe: can you beat the AI?
- Self Learning Meta Tic-Tac-Toe: watch the AI improve its state-evaluation constants using temporal difference learning.
- Interactive Meta Tic-Tac-Toe: test your abilities against the AI. Adjust its search depth parameter to change its skill level. Adjust its evaluation constants to change its behavior.
- Using the Copy Model for Edge Attachment to Evaluate the Stability of Graph Clustering: reproducing and extending results from a paper in graph/social-network analysis.
- E-Flashcards: create, organize, and review digital flashcards.

Lessons and Philosophy

- Gain inspiration from the philosophy behind great software projects. For example – *Unix philosophy*, *Zen of Python.*
- Whenever possible, reduce amount of state the program must manage. *"It is better to have 100 functions operate on one data structure than 10 functions on 10 data structures"* – Alan Perlis.
- Make architectural design choices extremely carefully and use the right tool for the job! The code can change, but the data structure & architecture is much harder to change!
- YAGNI – don't add abstractions, features, more datastores etc., unless they are truly needed. Focus on delivering solutions to business needs rather than solving interesting problems. Champion simplicity and understand the overall goal of minimizing costs!

Key Abilities, Responsibilities, and Values

Domain	Ability/Value
Design	**Simplicity** – Value the most direct and straightforward approach. Minimize dependencies. Choose the right *abstraction* after understanding the problem domain. Understand the overall goal of reducing development, maintenance, and operational costs. **Innovation** – Quickly implement a *proof of concept* or *prototype* to evaluate the feasibility of an idea. Value and respond to feedback. **Rigor** – Make the numbers match. Use the appropriate lens to reduce risk and achieve *correctness* in areas such as *concurrency* and *scalability*. Scrutinize design choices by asking ***why***, and be wary of the potentially high unforeseen costs of adding new technology.
Develop	**Continuous Learning** – Quickly learn new code bases, frameworks, and libraries. Value the instincts of insiders and domain experts. **Function Programming** – Use a *pragmatic* functional programming style to reduce costs. Combine *functional programming* with *DataFrame* libraries such as *Spark* or *Pandas* to apply SQL concepts and idioms while building composable, testable data pipelines. **Open Source** – Achieve simplicity, minimize development and maintenance costs, and avoid reinventing the wheel by finding, using, and contributing to open source tools, platforms, and libraries.
Maintain	**Manage Tech Debt** – Ship rapidly in order to solve urgent business needs. Follow the boy-scout rule when possible, and keep the style of new code consistent with that of existing code. Apply refactoring strategically to "payoff" the tech debt with the highest cost. **Firefighter** – Identify root causes and rapidly implement solutions to critical *production fires* and bugs. **Anticipate** how changes in one component might cause breaks in connected or downstream components. Setup monitoring to identify issues proactively. **CI/CD** – Dramatically reduce maintenance costs by setting up simple and consistent turnkey or at least reproducible and well-defined CI/CD. Value frequent deploys with small changes to reduce downtime and the number of bugs exposed to the end user.
Teamwork	**Documentation** – Rigorously document high-level *design*, *algorithms*, *data-flow*, and *usage*. Follow the Light in the Darkness! **Peer-Review** – Apply internally. *Assume good faith*. Communicate with candor, discretion, & sincerity. Speed up the Feedback Cycles! **Truth** – *Focus* on what's really important. Avoid *bias*. Qualify one's *level of certainty*. *Differentiate* among factual statements, personal preference, and conjecture. Do the research to be *really certain* when answering important questions. Have Faith in the Team!