

Tips and Tricks for Data Engineering in the #GOshop

Miriah Peterson

co weave

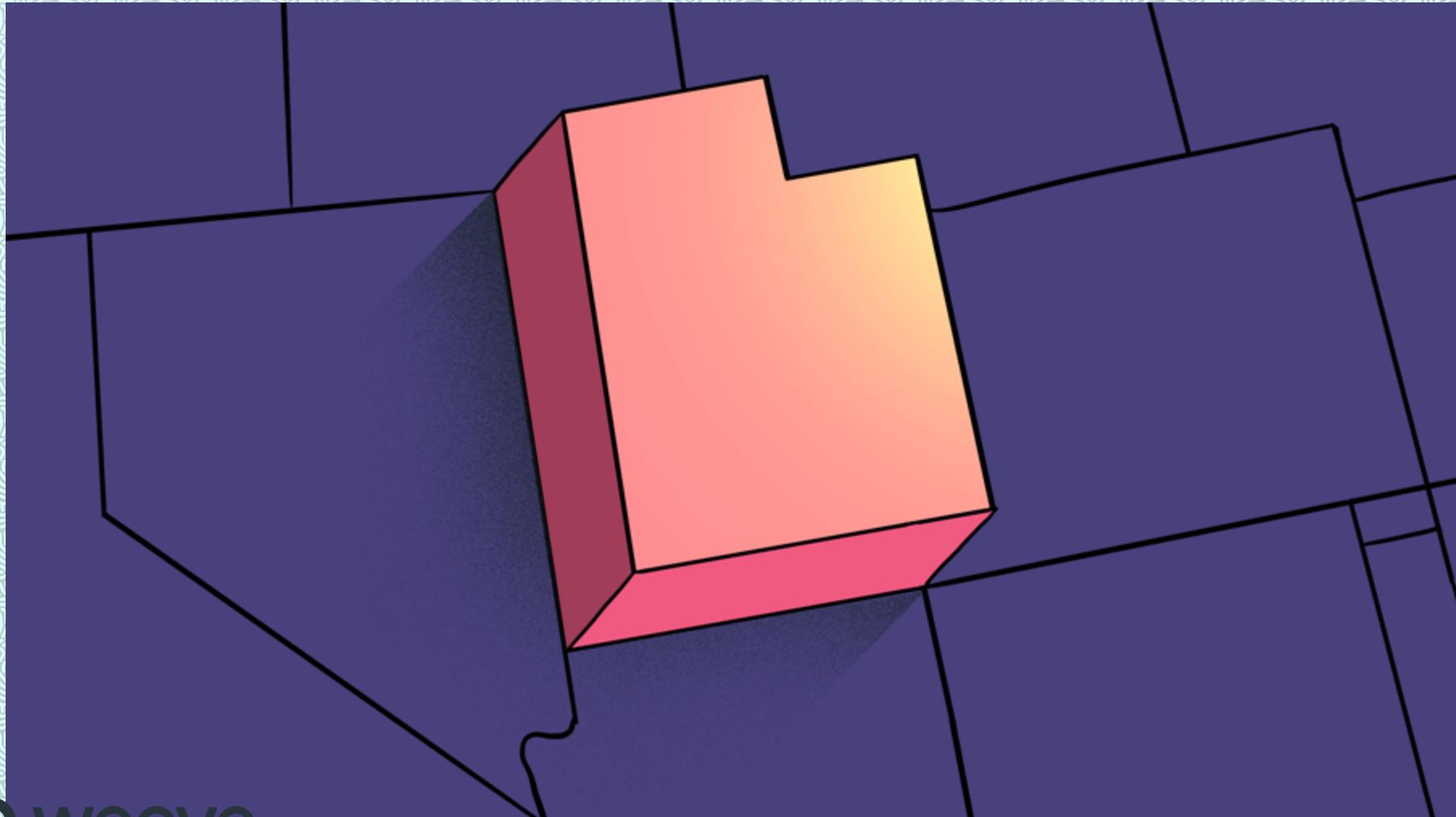


Bio

- Data Engineer at Weave in Lehi, Utah
- Board Member with Forge Foundation
- Proud Dog Mom
- [Twitter @captainnobody1](https://twitter.com/captainnobody1)
- [GitHub soypete](https://github.com/soypete)
- [LinkedIn](https://www.linkedin.com/in/soypete)



Intro



co weave



Intro



- <https://www.crunchbase.com/organization/weave>

CO weave



Intro

Tips and Tricks I have learned doing Data Engineering in Go:

- Platform
- Extract Data
- Load Data
- Transform Data



Platform

Platform: Infrastructure for storing, transferring, analyzing, and transforming data.



CO weave



Platform

If possible don't build your own home grown solution



Platform

Do not over engineer it

- Anticipate constraints and security
- Anticipate compliance
- Budget time to iterate on your design

Platform

Plan for Monolith or Microservice upfront. *No Hybrids*

- You don't want to maintain/create unnecessary services
- Use Caching Layers sparingly
- Consider Number of transfers that the data makes
- Don't cheat yourself with Go routines and channels



Platform

Use a messaging/streaming services to manage data flow

- APIs slow things down
- Allows for multiple worker consumption
- Manage batch processing jobs
- Services like Kafka allow for replaying jobs



Platform

Use a schema repository

- Standardize data types across services (microservice)
- Leverage Generation for all the repos
- Protects againsts stupidity
- (protobuf + go = ❤)



Extraction

Pick the Data type at Extraction and stick to it

- Avoid non-typed data
- Optimize type for storage
- Leverage Scraping as Needed

Extraction

Create Struct types for complex data structures

- Pointers make things lighter
- Memory doesn't matter until it does
- Leverage interfaces for complete picture

Extraction

Fail Fast isn't always your friend

- Always pass errors up to stop workers/routines/sub-processes
- Leverage back-off and retry
- Track failed data

Load

Find a good sql library

- [SQLX](#)
- [SQLC](#)
- Logic in SQL can help protect data



Load

Find the datastore that fits your needs

- Do you need a database, data lake, data warehouse or something else?
- Do you need more than one?



Load

Manage Your Migrations

- Letting migrations get out of hand is gonna kill your data store

Load

Monitor your data change. It makes debugging easier.

- Distributed data systems increase complications

Transform

Do not infuse business data model into your storage

- Optimize for Storage
- Pick a type and stick to it
- Make data discoverable

Transform

Don't be afraid of generating code

```
type example{{ $Type }} struct {
{{- range .TypeDef.TypeMembers}}
    {{ .Name }} {{ .DataType }} {{ if (ne .JSONName "")}}`json:"{{ .JSONName }}"`{{ end }}
{{- end }}
}
func (ex *example{{ $Type }}) Convert(ctx context.Context) (record.Record, error) {
    var dst = record.{{ $Type }}{
{{- range .Conversions }}
        {{ .ProtoMember }} : {{ if (eq .ConversionType "null.UUID")}}{{ ToPrivate .Cast }}{{ else }}ex.{{ .Cast }}{{ end }},
{{- end}}
    }
    if ex.ID == "" {
        return nil, werror.New("id must be supplied").SetCode(werror.CodeInvalidArgument)
    }
    return &dst, nil
}
```



Transform

DON'T Add Feature Logic Here!



CO weave



Transform

Live Data insights



co weave



In Summary

Tips and Tricks Go:

- Build a Smart Data Platform
- Know what Data you Extract
- Load the right Data to the right place
- Transform Data carefully



Conclusions

- There are lots of pitfalls when engineering for data
- If you have to do it yourself do not!
- I do recommend doing using all the tools out there, but Go is the best!



Contact

- [Twitter @captainnobody1](#)
- [GitHub soypete](#)
- [LinkedIn](#)

REFERENCE:

- <https://news.crunchbase.com/news/utahs-tech-scene-its-rising-profile-and-the-third-generation-of-startups/>
- <https://blog.gopheracademy.com/advent-2018/apache-beam/>
- <https://www.pachyderm.com/>

