# Big data & OCaml @ Ahrefs

# Hello!

**I am Javier Chávarri**

Software Engineer at Ahrefs.

You can find me at @javierwchavarri

# Ahrefs

WHAT WE DO

SaaS for SEO agencies

Transforming big data into relevant indicators

5 main tools to visualize and work with that data

More recently, also a search engine: yep.com

WHO WE ARE

Founded in 2010

HQ in Singapore

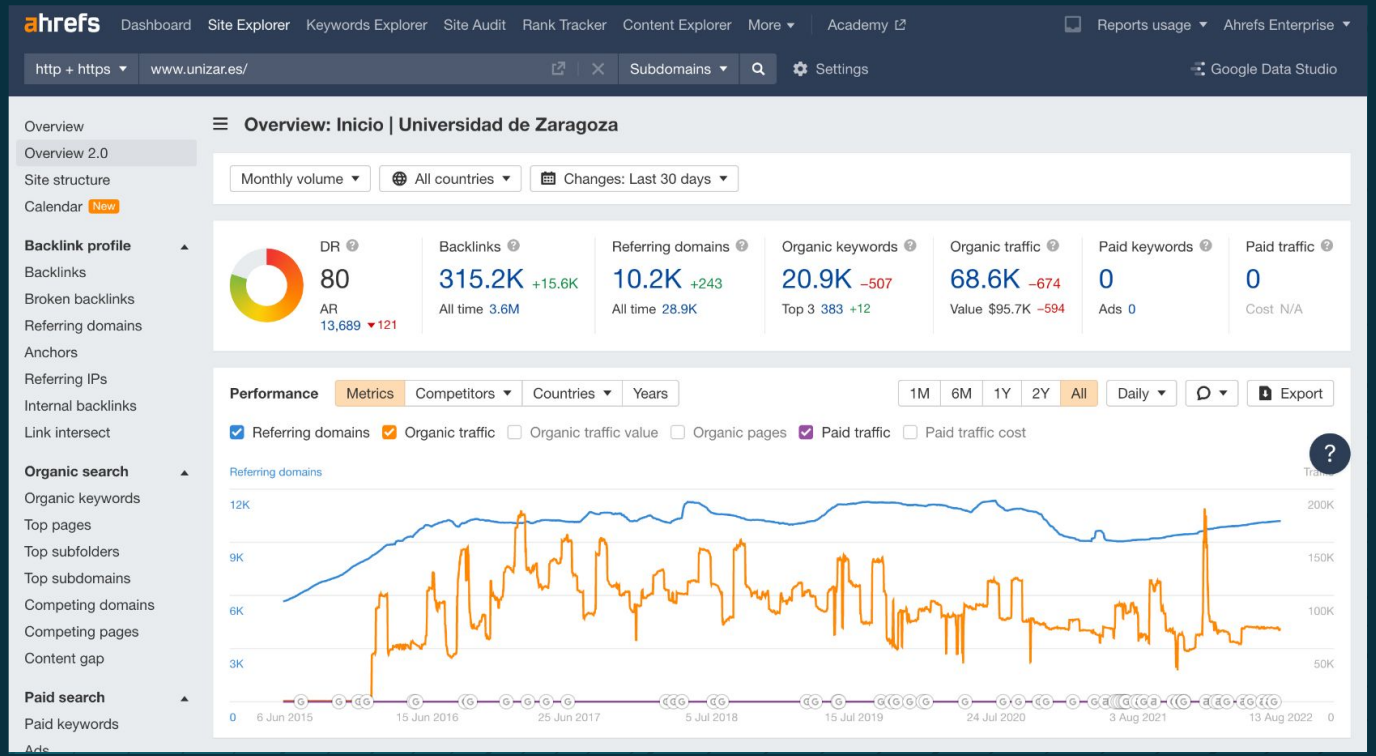
Almost 100 people

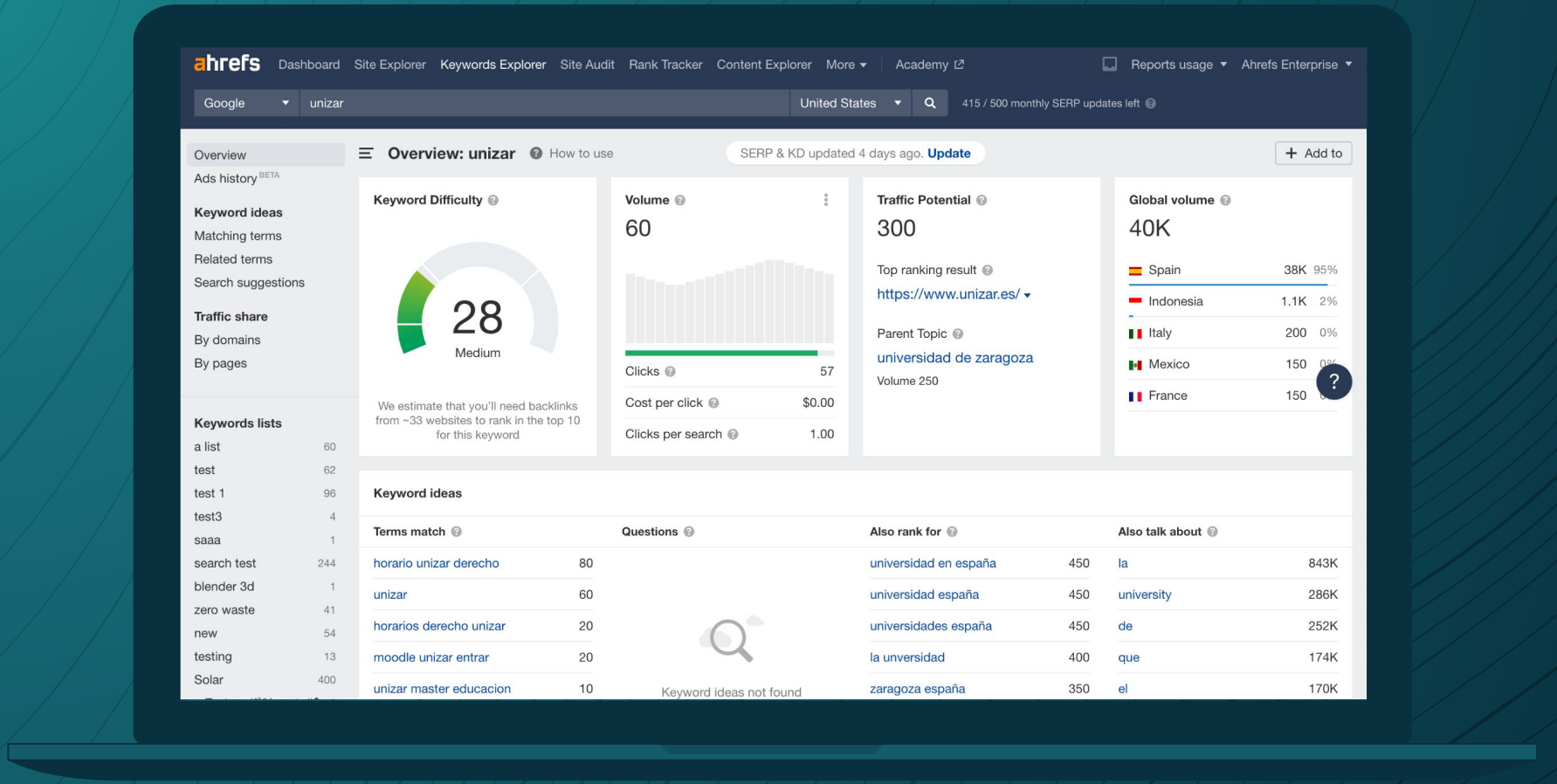
Two thirds work remotely

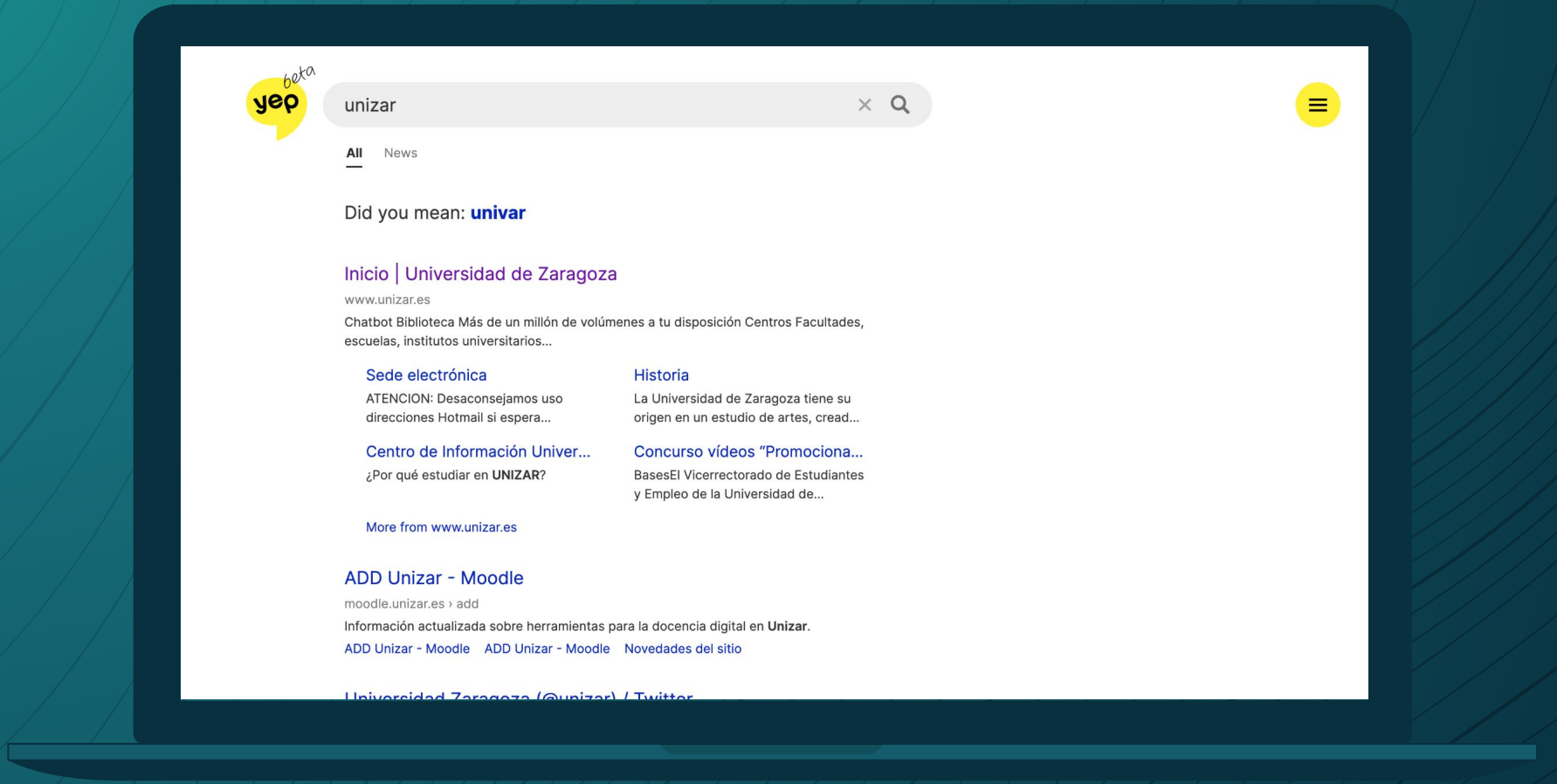
14 OCaml developers, 18 ReasonML* developers

Some numbers:

- 3000 servers, 344K CPU cores, 2.5PB RAM, 51PB HDD, 202PB SSD
- Every 24 hours we update metrics for 420 million pages
- Backlinks index contains data from 218 million domains
- More at https://ahrefs.com/big-data


ahrefs
Dashboard
Site Explorer
Keywords Explorer
Site Audit
Rank Tracker
Content Explorer
More
Academy
Reports usage
Ahrefs Enterprise
http + https
www.unizar.es/
Subdomains
Settings
Google Data Studio
Overview
Overview 2.0
Site structure
Calendar
New
Backlink profile
Backlinks
Broken backlinks
Referring domains
Anchors
Referring IPs
Internal backlinks
Link intersect
Organic search
Organic keywords
Top pages
Top subfolders
Top subdomains
Competing domains
Competing pages
Content gap
Paid search
Paid keywords
Ads
Overview: Inicio | Universidad de Zaragoza
Monthly volume
All countries
Changes: Last 30 days
DR
80
AR
13,689 ▼121
Backlinks
315.2K +15.6K
All time 3.6M
Referring domains
10.2K +243
All time 28.9K
Organic keywords
20.9K −507
Top 3 383 +12
Organic traffic
68.6K −674
Value $95.7K −594
Paid keywords
0
Ads 0
Paid traffic
0
Cost N/A
Performance
Metrics
Competitors
Countries
Years
1M
6M
1Y
2Y
All
Daily
Export
Referring domains
Organic traffic
Organic traffic value
Organic pages
Paid traffic
Paid traffic cost
Referring domains
12K
9K
6K
3K
0
200K
150K
100K
50K
6 Jun 2015
15 Jun 2016
25 Jun 2017
5 Jul 2018
15 Jul 2019
24 Jul 2020
3 Aug 2021
13 Aug 2022


ahrefs
Dashboard
Site Explorer
Keywords Explorer
Site Audit
Rank Tracker
Content Explorer
More
Academy
Reports usage
Ahrefs Enterprise
Google
unizar
United States
415 / 500 monthly SERP updates left
Overview
Ads history BETA
Keyword ideas
Matching terms
Related terms
Search suggestions
Traffic share
By domains
By pages
Keywords lists
a list 60
test 62
test 1 96
test3 4
saaa 1
search test 244
blender 3d 1
zero waste 41
new 54
testing 13
Solar 400
Overview: unizar
How to use
SERP & KD updated 4 days ago. Update
Add to
Keyword Difficulty
28
Medium
We estimate that you'll need backlinks from ~33 websites to rank in the top 10 for this keyword
Volume
60
Clicks 57
Cost per click $0.00
Clicks per search 1.00
Traffic Potential
300
Top ranking result
https://www.unizar.es/
Parent Topic
universidad de zaragoza
Volume 250
Global volume
40K
Spain 38K 95%
Indonesia 1.1K 2%
Italy 200 0%
Mexico 150
France 150
Keyword ideas
Terms match
horario unizar derecho 80
unizar 60
horarios derecho unizar 20
moodle unizar entrar 20
unizar master educacion 10
Questions
Keyword ideas not found
Also rank for
universidad en españa 450
universidad españa 450
universidades españa 450
la unversidad 400
zaragoza españa 350
Also talk about
la 843K
university 286K
de 252K
que 174K
el 170K


yep beta
unizar
All
News
Did you mean: univar
Inicio | Universidad de Zaragoza
www.unizar.es
Chatbot Biblioteca Más de un millón de volúmenes a tu disposición Centros Facultades, escuelas, institutos universitarios...
Sede electrónica
ATENCION: Desaconsejamos uso direcciones Hotmail si espera...
Historia
La Universidad de Zaragoza tiene su origen en un estudio de artes, cread...
Centro de Información Univer...
¿Por qué estudiar en UNIZAR?
Concurso vídeos "Promociona...
BasesEl Vicerrectorado de Estudiantes y Empleo de la Universidad de...
More from www.unizar.es
ADD Unizar - Moodle
moodle.unizar.es › add
Información actualizada sobre herramientas para la docencia digital en Unizar.
ADD Unizar - Moodle
ADD Unizar - Moodle
Novedades del sitio

# Eagle-eye view

# Challenges

- Crawl pages fast
- Funnel data from 3000 servers to 1 browser
- Increasing number of ways to read data: ahrefs.com, public API, search engine, 3rd party integrations
- Remote team, working across multiple time zones

# ONE LANGUAGE

We mostly use OCaml
everywhere we can

# One language

- Shared tooling* and code
- Same semantics across domains
- Reduced friction, easier for developers to explore other parts of the stack

# Why OCaml

- Good ratio dev time/features
- Expressive language
- Great performance
- Type inference => Easy maintenance
- Available in many environments, including the browser
- Stability

# Eagle-eye view

# Data backend

The data is produced by “offline” tools / background jobs

- Crawling the web
- Refreshing keywords
- Crawl clients’ websites
- …

Tools are independent from each other

# Data backend

Many different kind of databases (SQL, Elasticsearch, ClickHouse, Redis, custom dbs, ...)

We need a way to describe all the data we store (similar to a database schema)

- Mappings: generated data types
- Typed DSLs

# Mappings (esgg, github.com/ahrefs/esgg)

**Source file (api_ahrefs_rank.all.json)**

```
{
 "size": 1,
 "query": {
   "match_all": {}
 },
 "_source": [
   "ahrefs_rank",
   "url"
 ]
}
```

**Generated file (api_ahrefs_rank_all_j.ml)**

```
[@@@ocaml.warning "-3-33"]
[@@@alert "-codegen_only"]
open Es_api_ahrefs_rank_all_t
open Aa_esMapping
(* Generated by esgg input_j from
./es/api/api_ahrefs_rank.all.json based on
./mapping/api_ahrefs_rank.json *)
let type_ = `Search
let make ~index:(__esgg_index,__esgg_kind) () =
 `POST,
 [__esgg_index;"_search"],
 [],
 Some
("{\"aggregations\":{},\"query\":{\"match_all\":{}},\"
size\":1,\"_source\":[\"ahrefs_rank\",\"url\"]}"
 )
```

# Website backend

- Http server
- Data transformation
- Data types generated from mappings
- Static definitions of all queries to databases (well typed)
- Typed payloads (request body + response value)
- Typed DSLs

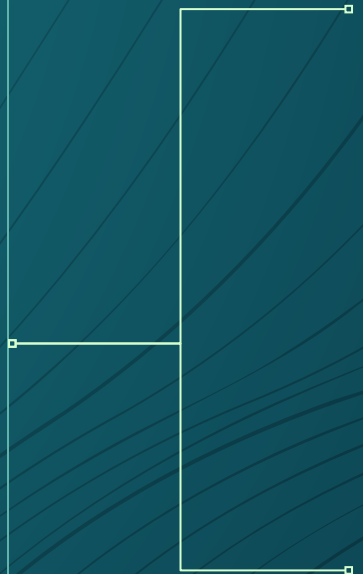
# Frontend

- Multiple ReasonReact applications
- Calls to http endpoints
- Data types generated from mappings and website backend routes
- Sharing code with website backend

# Interfaces (ATD, atd.readthedocs.io)

**Source file (interface.atd)**

```
type keywordsListId <ocaml
from="KeywordsExplorer"> = abstract
type keywords <ocaml from="KeywordsExplorer"> =
abstract

type input = {
 listId <ocaml name="list_id">: keywordsListId;
 keywords: keywords;
}

type keywordsAmount = int

type output = keywordsAmount
```



OCaml native

ReasonML (compiled to JavaScript)

# Now vs before

Before: OCaml - PHP - JavaScript

- All communication layers are written by hand and do validation on their own
- Some code is written in multiple languages (validation, encoding & decoding…)

# Now vs before

Now: OCaml - OCaml - OCaml

- Communication layers are generated
- Code is shared between all layers
- Cross-team shared knowledge compounds over time

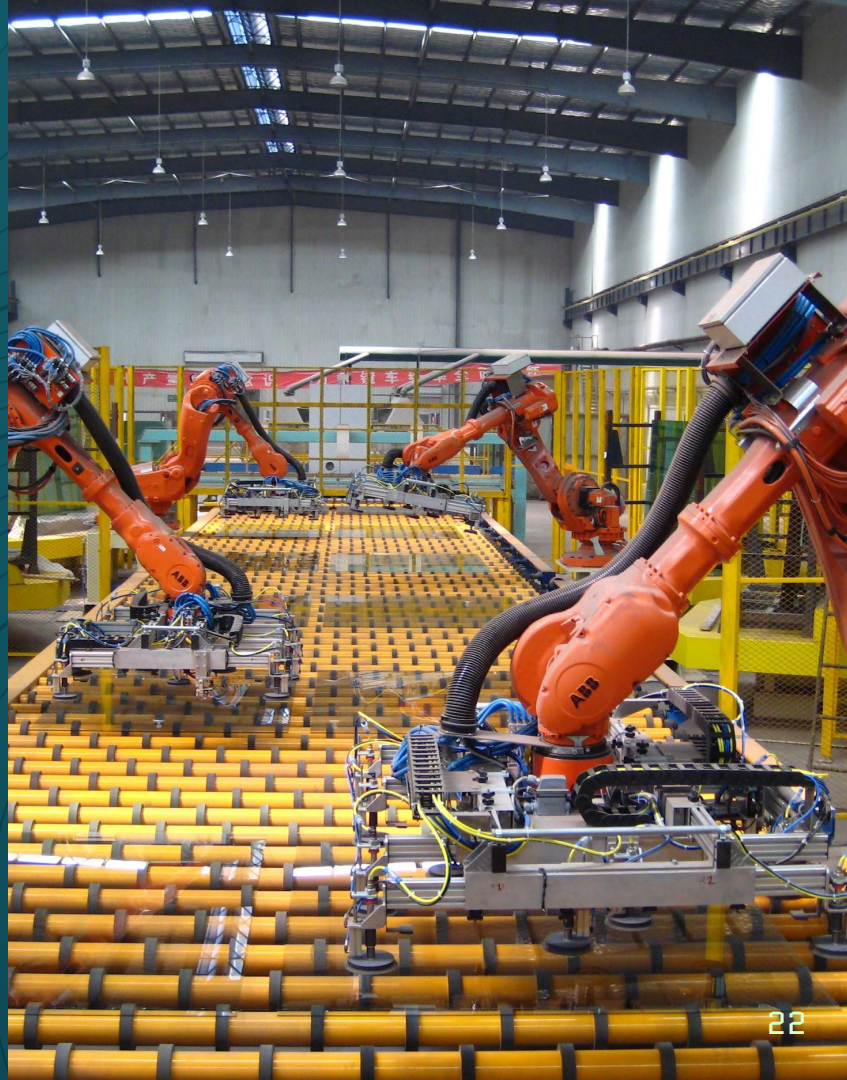
# One language technical benefits

Shared types from data generation in data backend to the browser

- No bad transformation
- Semantic is preserved
- Less mental overhead
- Changes are automatically propagated and verified by the compiler
- Can generate a lot of code and helpers easily

# Code generation

- Save time
- Avoid bugs
- Make coding more enjoyable



# ATD (atd.readthedocs.io)

**Source file (interface.atd)**

```
type version = [
  | K1
]

type input = {
  v: version;
  n: string;
  u: string;
}

type output = string
```

**Generated file (interface_j.ml)**

```
let write_output = Atdgen_codec_runtime.Encode.string
let read_output = Atdgen_codec_runtime.Decode.string
let write_input =
  Atdgen_codec_runtime.Encode.make (fun (t : input) ->
    Atdgen_codec_runtime.Encode.obj
      [
        Atdgen_codec_runtime.Encode.field write_version
~name:"v" t.v;
        Atdgen_codec_runtime.Encode.field
Atdgen_codec_runtime.Encode.string ~name:"n" t.n;
        Atdgen_codec_runtime.Encode.field
Atdgen_codec_runtime.Encode.string ~name:"u" t.u;
      ])
let read_input =
  Atdgen_codec_runtime.Decode.make (fun json : input ->
    {
      v = Atdgen_codec_runtime.Decode.decode (read_version |>
Atdgen_codec_runtime.Decode.field "v") json;
      n =
        Atdgen_codec_runtime.Decode.decode
          (Atdgen_codec_runtime.Decode.string |>
Atdgen_codec_runtime.Decode.field "n")
          json;
      u =
        Atdgen_codec_runtime.Decode.decode
          (Atdgen_codec_runtime.Decode.string |>
Atdgen_codec_runtime.Decode.field "u")
          json;
    })
```

# sqlgg [ygrek.org/p/sqlgg]

**Source file (gen_permissions.sql)**

```
-- @list
SELECT `permission_id` FROM
`roles_permissions` WHERE `id` = @role_id;
```

**Usage**

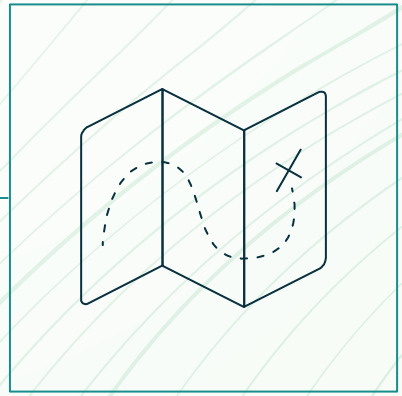
```
module Sql = Sql_permissions.Make(Conn)

let list dbd role_id =
 Sql.list dbd ~role_id (fun ~permission_id ->
permission_of_enum (Int64.to_int permission_id))
```

**Generated file (sql_permissions.ml)**

```
(* DO NOT EDIT MANUALLY *)
(*  *)
(* generated by sqlgg *)

module Make (T : Sqlgg_traits.M_io) = struct

 let list db ~role_id callback =
   let invoke_callback stmt =
     callback
       ~permission_id:(T.get_column_Int stmt 0)
   in
   let set_params stmt =
     let p = T.start_params stmt (1) in
     T.set_param_Int p role_id;
     T.finish_params p
   in
   T.select db ("SELECT `permission_id` FROM
`roles_permissions` WHERE `id` = ?") set_params
invoke_callback

end (* module Make *)
```

We are hiring! :)

ahrefs.com/jobs



# Thanks!

**Any questions?**

You can find me at @javierwchavarri &
javier.chavarri@ahrefs.com