

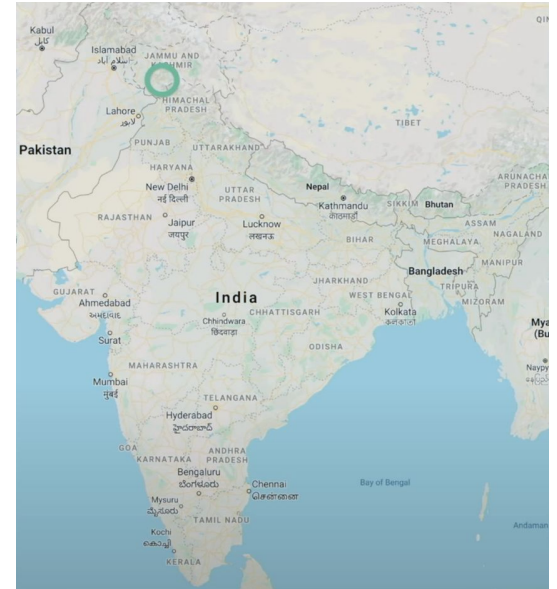
# PLV8-ify

TypeScript to PostgreSQL Functions

# Divyendu Singh



- Aurumfit.com (Engineering Lead)
- Source: <https://github.com/divyenduz/plv8ify>
- Previously, @prisma, @yara
- Twitter: @divyenduz
- Email: [mail@divyendusingh.com](mailto:mail@divyendusingh.com)
- Plays Football, loves tinkering/side projects, food
- Open to a bit of freelancing!





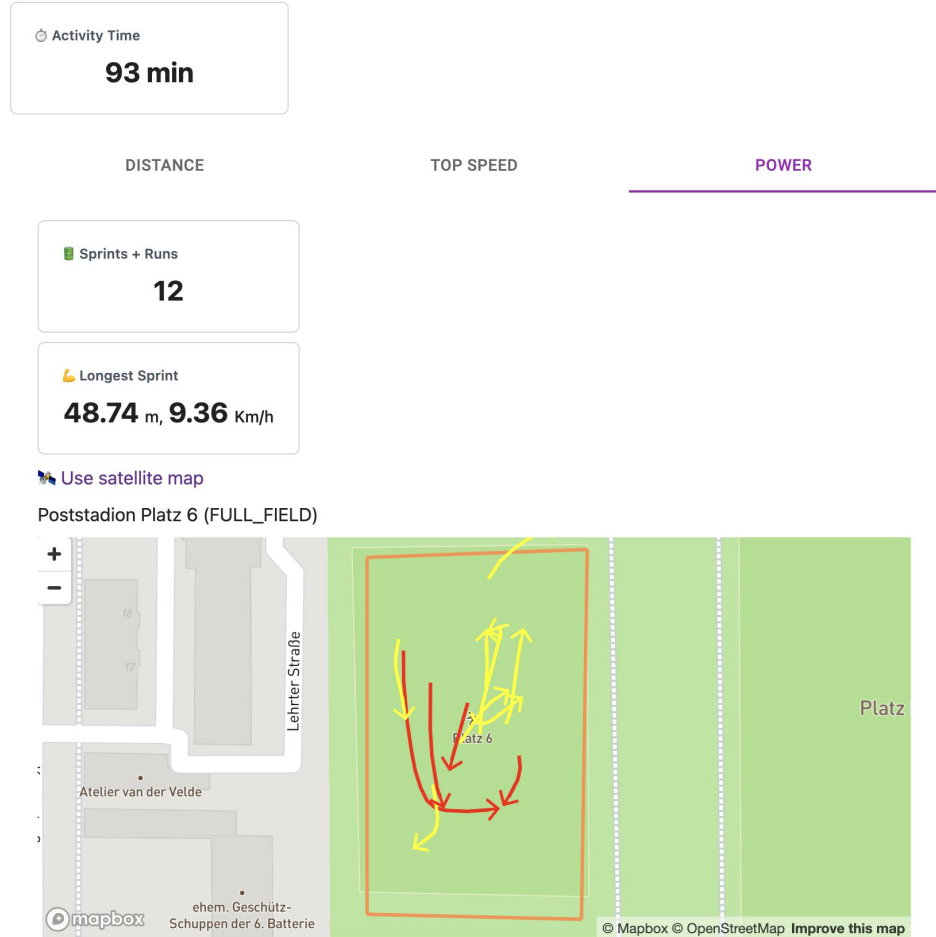
## Isokinetic Biofeedback - technology for optimal resistance

Use software to digitize weights for the safest and most effective workout ever.

[Learn more >](#)

# Why?

- Well 😊, side project in side project
- TrackFootball
- Algorithm in TypeScript
  - Why move to DB?
    - Easier to iterate/tweak
    - Better developer experience (more on that later)
- Powered by PLV8 💪🙏
  - Esbuild + ts-morph



# PLV8ify

- CLI that bundles TypeScript function to PostgreSQL (+PLV8) function

```
export function hello() {  
  return 'hello'  
}
```



```
1 select plv8ify_hello()  
  
line 1, column 23, location 22  
  
plv8ify_hello  
hello
```

```
plv8ify > examples > hello > plv8ify-dist > plv8ify_hello.plv8.sql  
1 DROP FUNCTION IF EXISTS plv8ify_hello();  
2 CREATE OR REPLACE FUNCTION plv8ify_hello() RETURNS text AS $plv8ify$  
3 var plv8ify = () => {  
4   var __defProp = Object.defineProperty;  
5   var __getOwnPropDesc = Object.getOwnPropertyDescriptor;  
6   var __getOwnPropNames = Object.getOwnPropertyNames;  
7   var __hasOwnProp = Object.prototype.hasOwnProperty;  
8   var __markAsModule = (target) => __defProp(target, "__esModule", { value: true });  
9   var __export = (target, all) => {  
10    for (var name in all)  
11      __defProp(target, name, { get: all[name], enumerable: true });  
12  };  
13  var __reExport = (target, module, copyDefault, desc) => {  
14    if (module && typeof module === "object" || typeof module === "function") {  
15      for (let key of __getOwnPropNames(module))  
16        if (!__hasOwnProp.call(target, key) && (copyDefault || key !== "default"))  
17          __defProp(target, key, { get: () => module[key], enumerable: !(desc = __getOwnPropDesc(module, key)) || desc.enumerable });  
18    }  
19    return target;  
20  };  
21  var __toCommonJS = /* @PURE */ ((cache) => {  
22    return (module, temp) => {  
23      return cache && cache.get(module) || (temp = __reExport(__markAsModule({}), module, 1), cache && cache.set(module, temp), temp);  
24    };  
25  })(typeof WeakMap !== "undefined" ? /* @PURE */ new WeakMap() : 0);  
26  
27 // examples/hello/input.ts  
28 var input_exports = {};  
29 __export(input_exports, {  
30   hello: () => hello  
31 });  
32 function hello() {  
33   return "hello";  
34 }  
35 return __toCommonJS(input_exports);  
36})();  
37  
38 return plv8ify.hello()  
39  
40 $plv8ify$ LANGUAGE plv8 IMMUTABLE STRICT;
```

# React also works (Kinda)

plv8ify > examples > react-js > TS input.tsx > ...

You, 8 months ago | 1 author (You)

```
1 import React from 'react'
2 import ReactDOMServer from 'react-dom/server'
3
4 function Header({ children }) {
5   return <h1>{children}</h1>
6 }
7
```

0 references

```
8 export function component(text: string) {
9   return ReactDOMServer.renderToStaticMarkup(
10     <div>
11       <Header>Hello</Header> {text}
12     </div>
13   )
14 }
15
```



```
plv8ify > examples > react-js > plv8ify-dist > plv8ify_component.plv8.sql
3 var plv8ify = (() => {
4975 var input_exports = {};
4976   __export(input_exports, {
4977     component: () => component
4978   });
4979   var import_react = __toESM(require_react());
4980   var import_server = __toESM(require_server_browser());
4981   function Header({ children }) {
4982     return /* @PURE */ import_react.default.createElement("h1", null, children);
4983   }
4984   function component(text) {
4985     return import_server.default.renderToStaticMarkup(/* @PURE */ import_react.default.createElement("div", null, /* @PURE */ import_react.default.createElement(Header, null, "Hello
4986   )
4987   return __toCommonJS(input_exports);
4988   })();
4989   /*
4990   object-assign
4991   (c) Sindre Sorhus
4992   @license MIT
4993   */
4994   /* @license React v17.0.2
4995   * react-dom-server.browser.development.js
4996   *
4997   * Copyright (c) Facebook, Inc. and its affiliates.
4998   *
4999   * This source code is licensed under the MIT license found in the
5000   * LICENSE file in the root directory of this source tree.
5001   */
5002   /* @license React v17.0.2
5003   * react.development.js
5004   *
5005   * Copyright (c) Facebook, Inc. and its affiliates.
5006   *
5007   * This source code is licensed under the MIT license found in the
5008   * LICENSE file in the root directory of this source tree.
5009   */
5010   return plv8ify.component(text)
5011 }
5012
5013 plv8ify LANGUAGE plv8 IMMUTABLE STRICT;
```

SQL Query

```
1 select plv8ify_component(['world'])
```

⚙ line 1, column 34, location 33

plv8ify\_component

```
["<div><h1>Hello</h1> world</div>"]
```

# Controlling the generated code

```
const { point: turfPoint } = require('@turf/helpers')

export function point(lat: number, long: number) {
  const pt = turfPoint([lat, long])
  return pt
}

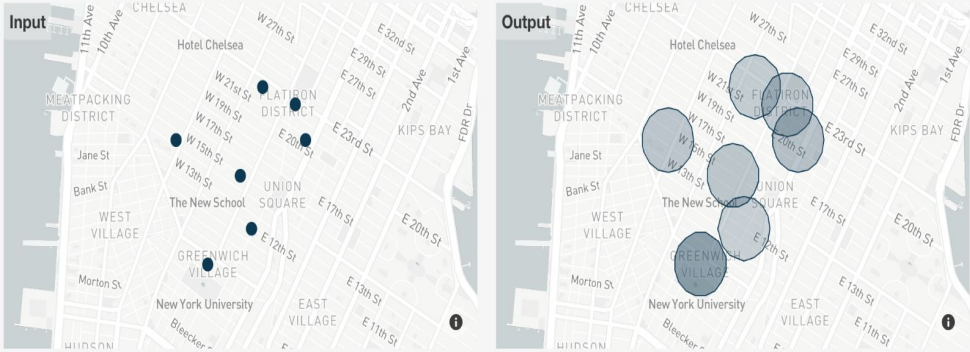
//@plv8ify-volatility-STABLE
export function stablePoint(lat: number, long: number) {
  const pt = turfPoint([lat, long])
  return pt
}

export function stablePointAsString(lat: number, long: number) {
  const pt = JSON.stringify(turfPoint([lat, long]))
  return pt
}
```

# Turf.js (+ the test cases/portability use case)

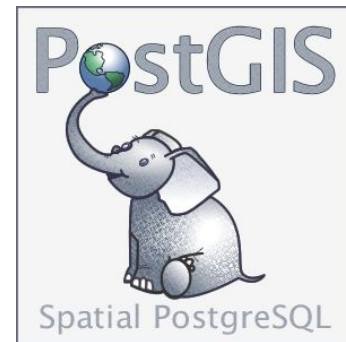
Welcome to Turf.js

Choose an operation:



Advanced geospatial analysis for browsers and Node.js

Simple	Modular	Fast
Modular, simple-to-understand JavaScript functions that speak GeoJSON	Turf is a collection of small modules, you only need to take what you want to use	Takes advantage of the newest algorithms and doesn't require you to send data to a server





# Turf.js (+ the test cases for “field detection”)



Divyendu Singh  
@divyenduz

...

</StartWeekendHacking>

Improved automatic field detection in TrackFootball.

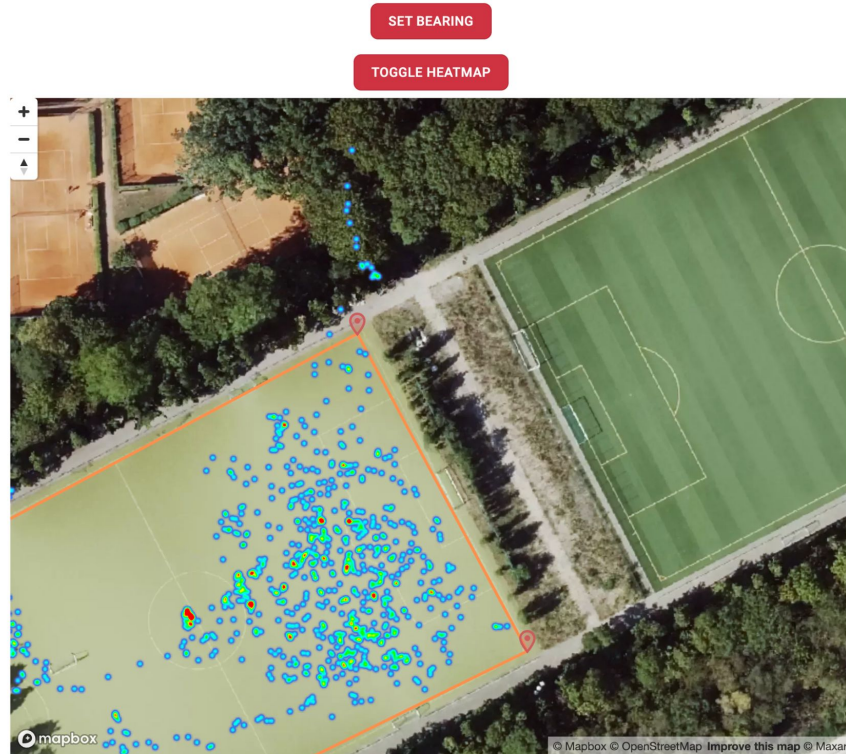
Problem: unlike "organized" Football, unorganized football can happen on half pitches, so for each field we have

TOP\_HALF  
BOTTOM\_HALF  
FULL\_FIELD



8:57 PM · May 6, 2023 · 700 Views

# Turf.js (+ the test cases for “field detection”)



# Turf.js (+ the test cases for “sprint detection”)

From <https://trackfootball.app/blog/engineering/physical-world-test-cases>

```
Test Suites: 5 passed, 5 total
Tests:      6 passed, 6 total
Snapshots:  1 passed, 1 total
Time:       2.027 s
```

Results of the tests run from jest test runner.

This gives me the confidence that I didn't significantly break the algorithm when I tweak it a bit. Of course, this is linked to my running profile, and it might break for other athletes.

Here are these activities in TrackFootball

- Sprint → <https://trackfootball.app/activity/60>
- Run → <https://trackfootball.app/activity/75>
- Jog → <https://trackfootball.app/activity/74>

# Turf.js (+ the test cases for “match detection”)

```
export class Overlap {
  private data1: Post
  private data2: Post
  constructor(data1: Post, data2: Post) {
    this.data1 = data1
    this.data2 = data2
  }

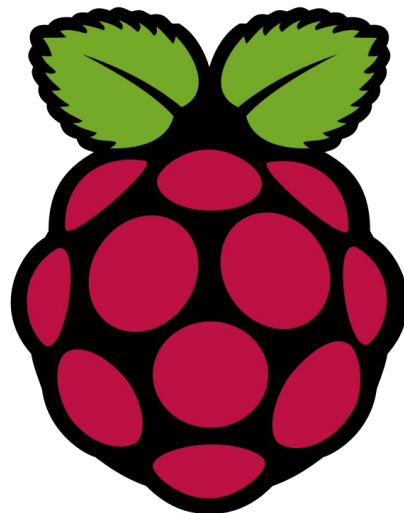
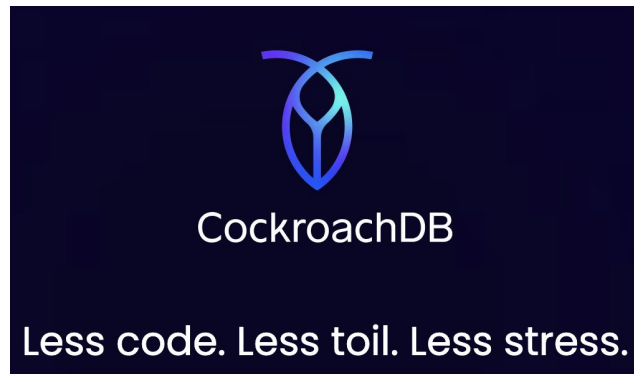
  isSameMatch() {
    return this.hasSpatialOverlap() && this.hasTemporalOverlap()
  }

  hasSpatialOverlap() {
    const envelope1 = envelope(
      this.data1.geojson as unknown as FeatureCollection<LineString>
    )
    const envelope2 = envelope(
      this.data2.geojson as unknown as FeatureCollection<LineString>
    )
    const intersection = intersect(envelope1, envelope2)
    return Boolean(intersection)
  }

  hasTemporalOverlap() {
```

# Turf.js (+ portability)

\*PLV8 is required though!



# Package Manager for PostgreSQL

<https://database.dev/> by supabase

# The Database Package Manager

For PostgreSQL trusted language extensions (TLEs)

Getting started

Read the blog post [↗](#)

# pg\_tle



**Divyendu Singh**

@divyenduz



This was exactly my motivation to write PLV8ify  
([github.com/divyenduz/plv8...](https://github.com/divyenduz/plv8...))

I was using PostGIS a lot but not all cloud providers supported it, so I ended up porting [@turfjs](#) via PLV8ify and used that instead.



**Sam Willis** @samwillis · Apr 14

With the just announced @superbase Trusted Language Extensions and dbdev package manager, it should be possible to build a similar Yjs extension with plv8. And it would be easily installable on any cloud Postgres with the pg\_tle extension. Going to have to try this!

[Show this thread](#)

11:46 AM · Apr 20, 2023 · **190** Views

# Real World Usage - AWS Request Signing

## Share your use cases #5



Open

divyenduz opened this issue on Mar 10 · 3 comments



divyenduz commented on Mar 10

Owner



How are you using/experimenting with `plv8ify` ?

Here is mine: <https://trackfootball.app/blog/engineering/typescript-in-postgres-with-plv8>



divyenduz added the `question` label on Mar 10



divyenduz pinned this issue on Mar 10



easel commented on Mar 16



I'm using `plv8ify` to implement AWS request signing as a Postgres function. This allows applications built using no-code API tools such as `postgrest`, `postgraphile`, and `hasura` to easily generate upload/download urls for assets on S3.



1



1



1



# Real World Usage - Math.js



ENHANCED BY Google



HOME

DOWNLOAD

GET STARTED

DOCS

EXAMPLES

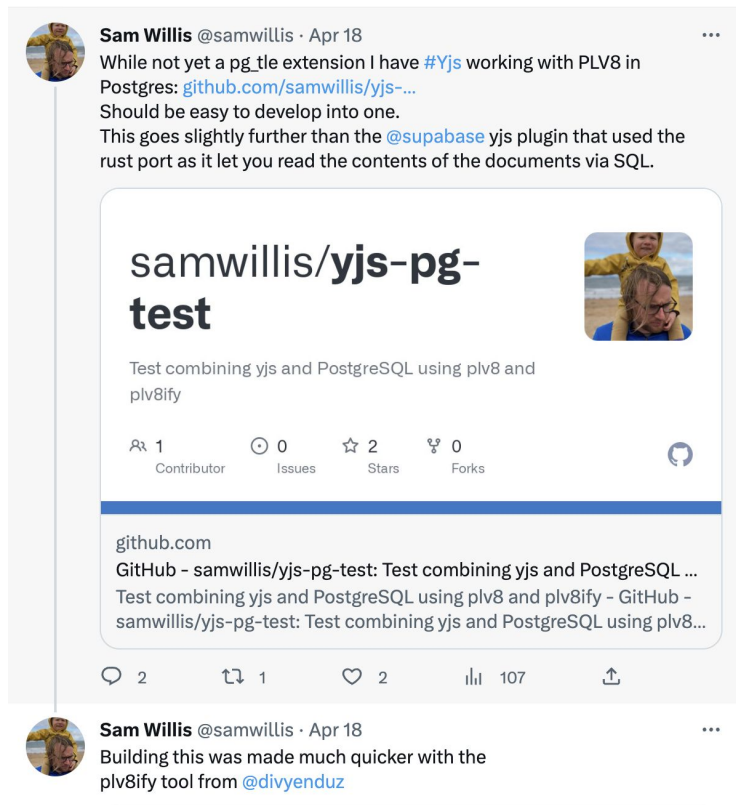
## An extensive math library for JavaScript and Node.js

Math.js is an extensive math library for JavaScript and Node.js. It features a flexible expression parser with support for symbolic computation, comes with a large set of built-in functions and constants, and offers an integrated solution to work with different data types like numbers, big numbers, complex numbers, fractions, units, and matrices. Powerful and easy to use.

### Features

- Supports numbers, big numbers, complex numbers, fractions, units, strings, arrays, and matrices.
- Is compatible with JavaScript's built-in Math library.
- Contains a flexible expression parser.
- Does symbolic computation.
- Comes with a large set of built-in functions and constants.
- Can be used as a command line application as well.
- Runs on any JavaScript engine.
- Is easily extensible.
- Open source.

# Real World Usage - Y.js (CRDT w/ JS)



**Sam Willis** @samwillis · Apr 18

While not yet a pg\_tle extension I have #Yjs working with PLV8 in Postgres: [github.com/samwillis/yjs-...](https://github.com/samwillis/yjs-pg-test)  
Should be easy to develop into one.  
This goes slightly further than the @supabase yjs plugin that used the rust port as it let you read the contents of the documents via SQL.

## samwillis/yjs-pg-test

Test combining yjs and PostgreSQL using plv8 and plv8ify

Contributor 1   Issues 0   Stars 2   Forks 0

github.com

GitHub - samwillis/yjs-pg-test: Test combining yjs and PostgreSQL ...  
Test combining yjs and PostgreSQL using plv8 and plv8ify - GitHub - samwillis/yjs-pg-test: Test combining yjs and PostgreSQL using plv8...

2   1   2   107

**Sam Willis** @samwillis · Apr 18

Building this was made much quicker with the plv8ify tool from @divyenduz

# Real World Usage - CloudWatch Log Processing

---

```
/**
 * This function removes varying parts of a message to make it hashable.
 * Varying parts can be user data like email, business data like location
 * or system data like request ID
 */
export function getFingerprintableMessage(message: string) {
  let msg = message;
  if (!msg.startsWith("ERROR")) {
    msg = message.substring(62);
  }
  msg = msg.replace(new RegExp(`([-\\+\\.a-zA-Z0-9\\d]+@[a-zA-Z\\d\\.]+)`), "");
  msg = msg.replace(new RegExp("(customer-app:)([a-z0-9-]+)", "$1");
  msg = msg.replace(new RegExp("(coaching-app:)([a-z0-9-]+)", "$1");
  msg = msg.replace(
    new RegExp("(Session Type :)([\\s\\w\\d\\-:~+()]+)(at)",
      "$1 <session-type-removed> $3"
    );
  msg = msg.replace(
    new RegExp("(at)([\\s\\w\\d\\-()ÄäÖöÜüß]+)(on (Web|Mobile))",
      "$1 <location-removed> $3"
    );
  msg = msg.replace(
    new RegExp("(Time-slot)([\\s\\w\\d-:~+()ÄäÖöÜüß]+)(as slot)",
      "$1 <time-removed> $3"
    );
  return msg;
}
```

# North Star

- TypeScript development workflows
  - Autocompletion, type-safety, test-cases etc.
- CLI to deploy to the database, list functions etc. (better API)
- pg\_tle extensions (WIP)
- <https://github.com/wasmerio/wasmer-postgres>

# Revenge of the SQL - Advanced SQL Training

```
1 WITH SpeedDistanceTimeForRegexp AS (  
2 WITH SpeedDistanceTime AS (  
3   WITH GeoJson AS (  
4     WITH _GeoJson AS (  
5       SELECT  
6         jsonb_array_elements("Post","geoJson" -> 'features' -> 0 -> 'geometry' -> 'coordinates') as "coordinate",  
7         jsonb_array_elements("Post","geoJson" -> 'features' -> 0 -> 'properties' -> 'coordTimes')::text::timestamp as "coordTime"  
8       FROM  
9         "Post"  
10      WHERE  
11        "id" = 61  
12    )  
13    SELECT *, lag("coordinate") OVER () as "lastCoordinate", lag("coordTime") OVER () as "lastCoordTime" FROM _GeoJson  
14  )  
15  SELECT  
16  *, |  
17  plv8ify_distance("coordinate", "lastCoordinate") * 1000 as "distance",  
18  EXTRACT(EPOCH FROM "coordTime" - "lastCoordTime")::integer AS "time",  
19  ((plv8ify_distance("coordinate", "lastCoordinate") * 1000) / (EXTRACT(EPOCH FROM "coordTime" - "lastCoordTime")::integer)) as "speed"  
20  FROM GeoJson  
21  WHERE  
22    "lastCoordinate" IS NOT NULL  
23 )  
24 SELECT  
25   "speed",  
26   "time",  
27   sum("time") over (ROWS BETWEEN 10 PRECEDING AND CURRENT ROW) as "elapsedTimeOfTheSegment",  
28   -- array_agg("speed") OVER(ROWS BETWEEN 10 PRECEDING AND CURRENT ROW) as "rawSpeedsInSegment",  
29   string_agg(CASE WHEN "speed" > 2.78 AND "speed" <= 12.50 THEN 's' ELSE 'n' END, '') OVER(ROWS BETWEEN 10 PRECEDING AND CURRENT ROW) as "speedsOverThreshold"  
30  FROM SpeedDistanceTime  
31 )  
32 SELECT * from SpeedDistanceTimeForRegexp  
33 WHERE "speedsOverThreshold" ~ '.*s{7}.*$' AND "elapsedTimeOfTheSegment" >= 10;
```

line 16, column 6, location 531

No limit

speed	time	elapsedTimeOfTheSegment	speedsOverThreshold
3.74987741987445	2	30	ssnnsssssss
3.50520960657979	3	31	snnssssssss
1.53629299199146	3	31	nnsssssssn
0.923467735215683	2	30	ns:ssssssn
3.98802012464983	8	36	sssssssnns
4.42965852217533	2	33	ssssssnns
3.92221108817110	2	30	nnsssssss

Data Message Chart 266 ms

39 rows

# Thank You / Q&A

- Aurumfit.com (Engineering Lead)
- Source: <https://github.com/divyenduz/plv8ify>
- **trackfootball.app**
- Previously, @prisma, @yara
- Twitter: @divyenduz
- Email: [mail@divyendusingh.com](mailto:mail@divyendusingh.com)
- Plays Football, loves tinkering/side projects, food
- Open to a bit of freelancing!

