

Prueba Técnica - Ingeniero de datos

SQL, Python/Pandas y Django

Objetivo de la prueba

El objetivo de de la prueba es ver cómo te desenvuelves en un ejercicio práctico que abarque la extracción de datos, la generación de unas simples web y API de consulta y por último la generación de un entregable en Python:

1. Poblar una base de datos y realizar unas consultas
2. Desarrollo con Django
 - a. Web
 - b. API
3. Generación de un informe en Python

Nos gustaría recibir la solución a la prueba más o menos dentro de unos 7 días desde la recepción de este mensaje en la dirección procesoseleccion@████████.org, pero si tienes algún inconveniente somos flexibles, avísanos mandándonos un correo a esa misma dirección y busquemos una alternativa.

Aunque idealmente busquemos a alguien que domine las 3 fases, si en alguna de ellas no te desenvuelves tanto, por favor, ámate a intentar sacar lo que puedas durante estos días. Valoraremos muy positivamente lo que seas capaz de aprender sobre la marcha y el esfuerzo por completar la tarea.

Si te atascas con algo, tienes dudas o necesitas consejo no dudes en escribirnos a procesoseleccion@████████.org. Te responderemos muy rápido.

Como referencia te listamos las cosas que nos interesa valorar de tu prueba:

- Claridad, organización y eficiencia del código.
- Modularidad del código: estrategias implementadas que lo hagan fácilmente exportable a otros proyectos y fácilmente adaptable a nuevos requisitos o infraestructuras.
- Selección de estrategias, funciones y librerías adecuadas para resolver cada fase del ejercicio.

- Proactividad a la hora de implementar funcionalidades ideadas por ti mismo, con su justificación y poniendo de relieve las ventajas.
- Calidad técnica de la web resultante: aplicación de estándares de buenas prácticas, diseño adaptativo...
- Apariencia final de la web resultante: diseño elegante y atractivo, usabilidad, profesionalidad...

No es obligatorio completar la prueba al 100%, llega hasta donde puedas o hasta donde consideres apropiado para mostrar tus habilidades. ¡Muchas gracias y suerte!

Descripción de la prueba

La ciudad de Nueva York deja como Open Data la información de su servicio de taxis, vamos a emplear esos datos a lo largo de toda la prueba práctica, los puedes descargar desde aquí:

<https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page>

Los ficheros que vamos a usar son los del Yellow Taxi Trip Records correspondientes a enero, febrero y marzo de 2020.

En este PDF

https://www1.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf puedes consultar la definición de cada uno de los campos del fichero.

Esta prueba está diseñada para evaluar habilidades muy parecidas a las que luego se usarán en el puesto de trabajo, pero en ningún caso el resultado de la esta prueba va a formar parte de la base de código de Frogtek, tampoco el set de datos empleado en la prueba guarda relación con la tipología de datos que utiliza Frogtek.

La prueba está estructurada en tres partes:

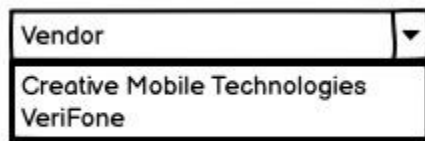
- Responder una serie de preguntas usando SQL. Te pediremos que nos compartas la sentencias utilizadas y el resultado final. Tienes libertad para usar la base de datos de tu preferencia con la única restricción de ser una base de datos que podamos instantar gratuitamente en un equipo con Linux o macOS
- Elaborar un panel web (en Django) para gestionar los servicios que no tienen VendorID
 - Aunque no es obligatorio preferiríamos que usases una de las últimas versiones de Django y python3
 - Para esta parte el entregable sería el código con el proyecto Django con todo lo necesario para que podamos desplegarlo en nuestros equipos. Valoraremos la inclusión de una mínima documentación para hacer el despliegue
- Escribir un script python que genere un informe basado en un determinado layout
 - Para esta parte te pedimos que uses python3 y una versión de Pandas posterior a la 1
 - Para esta parte el entregable sería el código que resuelva el problema planteado y el fichero excel resultante

Parte 1. SQL

1. ¿Puedes describirnos el procedimiento que has seguido para cargar los tres ficheros? ¿Puedes compartirnos la definición de la tabla que estás usando para contestar las preguntas planteadas?
2. Una vez cargados los tres ficheros en tu base de datos, ¿cuántos registros tiene la tabla? Compártenos la SQL que has usado.
3. Para cada uno de los tres meses con datos ¿podrías decirnos cuál es el trayecto más corto, más largo y el trayecto medio?, en distancia y en tiempo. Compártenos la sentencia SQL que has usado.
4. Para cada uno de los tres meses con datos ¿podrías decirnos cuánta es la variación porcentual en el número de servicios registrados con respecto al mes anterior? Por ejemplo, en esta pregunta nos gustaría obtener un resultado así. Compártenos la sentencia SQL que has usado.

mes	servicios	variacion_mes_anterior
2020-01	1000	NULL
2020-02	2000	100%

- Al entrar en la web se llenará el datagrid con todos los registros que tengan nulo el campo `vendor_id`
- El datagrid mostrará los campos `vendor`, `tpep_pickup_datetime`, `trip_distance` y `payment_type`
- No es necesario usar el ORM de Django, si lo prefieres puedes usar directamente una consulta SQL contra la base de datos
- El único campo editable del datagrid deberá ser `Vendor`. En cada campo `vendor` habrá un listbox como el que se muestra en el ejemplo para que el usuario seleccione uno de los dos posibles valores.



A screenshot of a web form element. It consists of a rectangular box with a thin border. Inside the box, the word 'Vendor' is written in a standard font. To the right of 'Vendor' is a small downward-pointing triangle, indicating a dropdown menu. Below 'Vendor', the text 'Creative Mobile Technologies' and 'VeriFone' are listed, separated by a horizontal line, representing the available options for selection.

- Cuando se pulse en el botón aplicar cambios se actualizarán los registros de la base de datos con los nuevos valores del campo vendor, debería ser posible cambiar varios registros del datagrid y aplicarlos todos simultáneamente al pulsar el botón “aplicar cambios”.
- El datagrid se paginará en bloques de 100 registros.

Parte 2.2. API Endpoint para consultar los servicios

Requisitos:

- Escribir un endpoint `longest_trips` que reciba como parámetros de entrada el `vendor_id` (numérico) y un `limit` (numérico)
- Devolverá un JSON con los viajes del vendor seleccionado ordenados por duración en millas descendente hasta completar el número de servicios que le hayamos pasado en el `limit`

Ejemplo del JSON devuelto, si le hubiéramos preguntado por el vendor 1 y un limit de 2:

```
[{
  "VendorID": 1,
  "tpep_pickup_datetime": "2020-01-01 12:00:01",
  "tpep_dropoff_datetime": "2020-01-01 12:30:00",
  "Trip_distance": 40
},
{
  "VendorID": 1,
  "tpep_pickup_datetime": "2020-02-01 12:00:01",
  "tpep_dropoff_datetime": "2020-02-01 12:30:00",
  "Trip_distance": 35
}]
```

[illegible]