Prueba Técnica Data Engineer Manager

El rol de Data Engineer es crear y administrar la Arquitectura para disponibilizar procesos de Data. Esta prueba técnica evalúa tu capacidad de aprender nuevos conceptos y tecnologías, tu capacidad de leer documentación y aplicarla para lograr resultados.

Descripción: Después del análisis, el equipo de Data te envía una matriz de relaciones entre personas, un 1 cuando tienen relación y un 0 cuando no tienen ningún tipo de relación. (Archivo de Excel adjunto a este email).

		1	2	3	4	5	6	7	8	9	10	11
		A	B	C	D	E	F	G	H	I	J	K
1	A	0	0	0	0	0	1	0	0	0	1	0
2	B	0	0	0	0	0	0	0	0	0	0	0
3	C	0	0	0	1	1	1	1	1	1	1	1
4	D	0	0	1	0	1	1	1	1	1	1	1
5	E	0	0	1	1	0	1	1	1	1	1	1
6	F	0	0	1	1	1	0	1	1	1	1	1
7	G	0	0	1	1	1	1	0	1	1	1	1
8	H	0	0	1	1	1	1	1	0	1	1	1
9	I	0	0	1	1	1	1	1	1	0	1	1
10	J	0	0	1	1	1	1	1	1	1	0	1
11	K	0	0	1	1	1	1	1	1	1	1	0
12	L	0	0	1	1	1	1	1	1	1	1	1
13	M	0	0	0	0	0	0	0	0	0	0	0
14	N	0	1	1	1	1	1	1	1	1	1	1
15	Ñ	0	0	0	0	0	0	0	0	0	1	0
16	O	0	0	0	0	0	0	0	0	0	0	0
17	P	0	0	0	0	0	0	0	0	0	0	0
18	Q											

El archivo tiene información hasta la row 17, así que faltan relaciones. El equipo de Data nos enviará este archivo actualizado diariamente, y te solicitan montar un pipeline de datos usando Dagster (Orquestador de Pipelines de Datos).

Reto: Como Data Engineer debes tomar estos datos y generar un Pipeline que cargue esta info en una nueva tabla de mysql (genera una Base de Datos Mysql local) y que este proceso se ejecute cada 24h actualizando la tabla de Mysql utilizando Dagster.

Resultado esperado:

Código en Python con la configuración del Pipeline de datos, junto con los querys utilizados para generar la DB y tablas. En este código esperamos se evidencien los pasos donde se lee el excel, organizan los datos y carga a mysql en una nueva tabla junto con su respectiva automatización empleando Dagster.

Documentación Recomendada:

- Adjunto a este email encontrarás el paso a paso detallando de como configurar Dagster en local y crear tu propio ETL.
- Documentación Dagster job: https://docs.dagster.io/tutorial/intro-tutorial/single-op-job

Código de Ejemplo de Automatización:

```python
from dagster import schedule
from datetime import datetime, time, date

@schedule(
        cron_schedule= "*/20 * * * *",
        pipeline_name= "load_bigquery_table_pipeline",
        execution_timezone="America/Bogota"
)
```

Modalidad de Calificación:

Tomaremos el código y lo ejecutaremos en un nuevo entorno, ejecutaremos los querys para generar la DB y el pipeline, esperamos que el pipeline se ejecute cada 24 h y actualice la tabla en Mysql, cargaremos diferentes versiones del archivo de excel y esperamos que la tabla de la base de datos se actualice con los cambios.

Punto Opcional: Puede complementar su pipeline de datos utilizando otras tecnologías y librerías. Adjuntando la documentación correspondiente explicando el proceso.

Ejercicio 2 API

Implementarás un pequeño proyecto basado en uno de nuestros problemas actuales. Creará un mecanismo para ingerir datos de un servidor a una base de datos de postgres y creará una API REST para obtener los registros de la base de datos.

El ejercicio consiste en diseñar la base de datos basada en los archivos adjuntos, ingerir los archivos adjuntos en la base de datos de postgres y construir una API REST para obtener los registros de la base de datos. Se adjuntan plantillas de docker-compose para comenzar de inmediato con postgres y una instancia de centos donde puede cargar los archivos. **Puede elegir el lenguaje de programación para desarrollar el código.**

La funcionalidad central para este desafío es:

1. Diseñar el E-R a partir de la base de datos y crear la estructura en base a los archivos adjuntos.
2. Ingiera los datos del servidor centos a la base de datos postgres.
3. Al menos la solicitud de "lectura" debe ser compatible con la API
4. El servidor donde se va a implementar la API debe tener acceso solo a la base de datos de postgres. Y el servidor centos también debe tener acceso solo a la base de datos de postgres.

Algunas funcionalidades opcionales que puede implementar:
1. Valide que el estado de la columna tenga una longitud de 2 y solo contenga letras. 2. Su código también podría ejecutarse en Docker.
3. Admite más solicitudes que solo 'leer', como 'crear o 'actualizar'.
4. Prueba unitaria y cobertura de tu código
5. Implementar CI/CD
6. Implementa un mecanismo para crear o actualizar el esquema

Algunas de las cosas que revisaremos de tu desafío:

- Software de trabajo. Primero revisaremos si su aplicación está funcionando como se especifica en este desafío.
- Código de limpieza. Esperamos que escriba un código limpio que sea comprensible y fácil de mantener.
- Bajo mantenimiento. No creemos en soluciones que impliquen grandes esfuerzos de mantenimiento, por lo que si hay un cambio como agregar nuevos campos, nuevas tablas, los cambios deben ser económicos.

Recursos

docker-compose.yml

```
# Use postgres/example user/password credentials
version: '3.1'
services: db:
image: postgres
restart: always
environment:
POSTGRES_PASSWORD: example
ports:
- 5432:5432
networks:
- postgres
centos:
image: centos
networks:
- postgres
networks:
postgres:
driver: bridge
```

Notas

Tenga en cuenta que solo revisaremos proyectos que estén en Github o GitLab.

EJERCICIO 3 DISEÑO DE SISTEMAS

Una empresa hotelera posee actualmente un sitio web para gestionar reservas, incluyendo fotos, ubicación y detalles de los hoteles. Como también tratan con clientes internacionales, una experiencia de usuario rápida y segura es un punto a considerar. Ellos están planeando migrar su sitio web tradicional a una infraestructura basada en la nube y tu labor es generar la propuesta de una arquitectura para este proceso.

Estos son los requerimientos que debes considerar.

El uso de los servicios es el siguiente:
- Los servicios de pago permiten a los usuarios realizar pagos en línea.
- Los servicios de visualización hacen que los datos estén disponibles para el huésped.
- Los servicios para huéspedes mantienen los favoritos del huésped, los últimos registros revisados, etc.
- Los servicios de búsqueda permiten al usuario buscar en diferentes propiedades.
- El diseño debe incluir todos los servicios y recursos necesarios para lograr la funcionalidad según los requisitos anteriores.

Tenga en cuenta los siguientes puntos al diseñar el sistema:
- No es necesario incluir el código de los servicios. Asume que ya está disponible. Asigne los servicios de acuerdo con el caso de uso y el diseño básico dado.
- El framework del front-end y el lenguaje del back-end no son de interés. Sólo hay que diseñar una arquitectura que muestre la conexión con los servicios mencionados, así como el repositorio de datos al que el sistema debe acceder.
- Muestra la manera de comunicar los servicios y los componentes de la nube que alojarán cada uno de los elementos que consideres.

Entregables

- Diagrama que muestre los componentes, la infraestructura necesaria para el despliegue y la forma de comunicarse.

Comentarios

Sabemos que 3 dias no es mucho tiempo, pero esperamos que haga su mejor esfuerzo y recuerde que usted es el dueño de lo que entrega y puede decidir qué quiere y qué no quiere entregar dadas las limitaciones de tiempo.

Asegúrese de que revisaremos cuidadosamente su desafío y nos pondremos en contacto con usted lo antes posible. ¡Te agradecemos de antemano tu interes y esperamos que te diviertas haciendo este desafío!

Si tienes alguna duda sobre el reto, puedes enviarnos un e-mai a julilopez@theapalacecompany

Mucho exito ¡!