Cuestionario sobre Arquitecturas de Almacenamiento

Elige la opción que consideres correcta, justificando tu respuesta y el porqué de lo inadecuado de las otras opciones.

Cuestión 1: Optimización de consultas analíticas

Una empresa necesita realizar consultas complejas sobre billones de registros de llamadas telefónicas, pero en cada consulta solo se accede a 3 o 4 atributos específicos (como la duración y el destino) de los cientos disponibles. ¿Qué formato de archivo y organización sería la más eficiente?

- **A)** Formato **Avro**, debido a su orientación a filas que permite leer el registro completo rápidamente.

Su orientación a filas ralentizaría demasiado las consultas al procesar todas las columnas de esa fila.

- **B)** Formato **CSV**, por ser un estándar de texto que permite la interoperabilidad entre cualquier sistema.

CSV no es optimo para consultas de tal calibre.

- **C)** Formato **Parquet o ORC**, debido a su orientación a columnas y capacidad de comprimir bloques de datos.

Gracias a su orientación en columnas, Parquet o ORC son perfecto para esta tarea.

Cuestión 2: Aplicación del Teorema CAP en sistemas distribuidos

Un sistema de monitorización de flotas globales requiere que el servicio esté siempre disponible para recibir datos de los sensores, incluso si hay fallos en los enlaces de red entre continentes. ¿Qué combinación del teorema CAP es la más adecuada para este caso?

- **A) CA (Consistencia y Disponibilidad)**, asegurando que todos los nodos vean lo mismo al mismo tiempo.

Es indispensable que se muestre la misma información a todos los usuarios y que esté siempre disponible.

- **B) CP (Consistencia y Tolerancia a Particiones)**, sacrificando la disponibilidad del sistema para evitar datos erróneos.

La tolerancia a particiones no es prioritaria en este caso y menos ante la indispensable disponibilidad.

- **C) AP (Disponibilidad y Tolerancia a Particiones)**, priorizando que el sistema siga dando servicio, aunque la consistencia sea eventual.

La tolerancia a particiones no es prioritaria en este caso y menos ante la indispensable consistencia.

Cuestión 3: Gestión de metadatos en HDFS

En un clúster de **Apache Hadoop HDFS**, ¿cuál es la función crítica del **NameNode** primario respecto a los datos almacenados?

- **A)** Almacenar físicamente los bloques de datos y replicarlos en otros servidores del clúster.

NameNode no se ocupa de esto.

- **B)** Gestionar exclusivamente el espacio de nombres (*namespace*) y la ubicación de los bloques sin que los datos reales pasen por él.

Esta es la función principal de NameNode.

- **C)** Ejecutar la lógica de negocio de las aplicaciones directamente sobre los discos locales.

No es la función crítica de NameNode.

Cuestión 4: Evolución hacia el Data Lakehouse

¿Qué ventaja principal aporta la implementación de capas como **Delta Lake o Apache Iceberg** sobre un almacenamiento de objetos (como Amazon S3)?

- **A)** Convertir el almacenamiento de objetos en un sistema jerárquico de carpetas tradicional.

Una jerarquía tradicional de carpetas no es la ventaja principal.

- **B)** Reducir el coste de almacenamiento eliminando la necesidad de metadatos.

Delta Lake y Apache Iceberg funcionan por medio de metadatos.

- **C)** Proporcionar características **ACID**, acceso SQL y control de transacciones sobre datos no estructurados.

Esta es la opción que retrata la ventaja principal.

Cuestión 5: Selección de base de datos para relaciones complejas

Si una organización necesita detectar tramas de fraude analizando cómo se conectan conductores, médicos y abogados mediante múltiples vínculos directos e indirectos, ¿qué tipo de gestor NoSQL es el más indicado?

- **A) Clave-Valor**, por su extrema rapidez en búsquedas simples.

No es necesaria la velocidad en este caso.

- **B) Grafos**, ya que representa entidades como nodos y relaciones como arcos.

La complejidad de las consultas hace de esta la mejor opción.

- **C) Documentos**, para anidar toda la información en estructuras JSON complejas.

Esta opción no es eficiente para realizar la tarea.

Cuestión 6: Identificación de los estados del dato

Un banco decide mover todos los registros de transacciones de hace más de cinco años, que legalmente debe conservar pero que raramente consulta, a un sistema de cintas de respaldo. Según la fuente, ¿en qué estado se encuentran estos datos?

- **A) Datos en tránsito (*data in motion*)**, ya que se están moviendo hacia el respaldo.

Damos por sentado que esta migración ya se ha realizado.

- **B) Datos en reposo (*data at rest*)**, ya que se encuentran fuera del acceso habitual y son inmutables.

Esta opción describe el caso en cuestión.

- **C) Datos en uso (*data in use*)**, porque siguen siendo consultables bajo solicitud.

Son consultables, pero no con la flexibilidad de data in use.

Cuestión 7: Limitaciones del modelo de escritura en HDFS

Un equipo de desarrollo intenta implementar una aplicación que requiere actualizar constantemente registros específicos (modificar una línea en medio del archivo) dentro de un fichero de 5 TB almacenado en **HDFS**. ¿Es esta una arquitectura adecuada?

- **A) Sí**, HDFS permite el acceso aleatorio y la edición de cualquier bloque del archivo de forma eficiente.

No, no lo permite.

- **B) No**, HDFS utiliza un modelo **WORM** (*Write Once Read Many*), donde los archivos no pueden ser actualizados una vez creados, soportando solo el anexo al final.

Esta es correcta. El modelo WORM complica esta acción.

- **C) Sí**, siempre que el **NameNode** coordine la reescritura de los metadatos del bloque afectado.

El modelo WORM.

Cuestión 8: Virtualización mediante Federación en RDBMS

Una analista de datos necesita cruzar una tabla de clientes en una base de datos Oracle con un archivo de logs en formato **JSON** que reside en **MongoDB** y un archivo histórico en **Parquet** en **Amazon S3**. ¿Cuál es la solución más eficiente según las capacidades modernas de los RDBMS?

- **A)** Utilizar la **Federación**, que permite acceder a fuentes heterogéneas con una única sentencia SQL como si los datos estuvieran juntos.

Esta es la opción más eficiente.

- **B)** Mover físicamente todos los datos a un único clúster de **Hadoop** para procesarlos.

Es una opción viable pero no preferible.

- **C)** Convertir todos los datos a formato **CSV** e importarlos manualmente a una tabla relacional.

Es una opción viable pero no preferible.

Cuestión 9: Estructura del Almacenamiento de Objetos

¿Cuál es la diferencia fundamental en la organización de los datos entre un sistema de archivos tradicional (como HDFS o NFS) y un **Almacenamiento de Objetos**?

- **A)** El almacenamiento de objetos utiliza una **estructura jerárquica** compleja de directorios y subdirectorios.

Esto describe el sistema de archivos tradicional.

- **B)** El almacenamiento de objetos gestiona los datos de forma **plana**, donde cada unidad es un objeto autocontenido con un identificador único y metadatos enriquecidos.

Así es como gestiona HDFS o NFS el almacenamiento.

- **C)** El almacenamiento de objetos divide los archivos en bloques físicos que el sistema operativo debe ensamblar manualmente.

No. No describe el almacenamiento de objetos.

Cuestión 10: Elección de base de datos NoSQL para sesiones

Una plataforma de comercio electrónico necesita almacenar los "carritos de la compra" y las sesiones de usuario. La prioridad es que la recuperación sea extremadamente ágil utilizando únicamente el ID de la sesión. ¿Qué categoría de NoSQL es la más recomendada?

- **A) Series temporales**, para registrar cada clic del usuario cronológicamente.

No aporta a la velocidad de las consultas.

- **B) Clave-Valor**, por su simplicidad y rapidez en búsquedas simples recuperando todo el valor.

Así se debería de gestionar la información para acometer la tarea planteada en el enunciado del ejercicio concreto.

- **C) Documentos**, para poder realizar consultas complejas sobre los productos dentro del carrito.

Es la peor opción, la más lenta.

Cuestión 11: Consistencia en el *Data Lakehouse*

Una organización utiliza un almacén de objetos en la nube para su data lake, pero experimenta problemas de inconsistencia cuando varios procesos intentan modificar los mismos datos simultáneamente. ¿Cuál es la solución tecnológica recomendada en las fuentes para resolver esto sin abandonar el almacenamiento de objetos?

- **A)** Migrar todos los datos a un sistema de archivos distribuido como HDFS para forzar el modelo WORM.

Podría ser una opción, pero no es la más oportuna.

- **B)** Implementar una capa de gestión de metadatos como **Delta Lake, Apache Iceberg o Apache Hudi**.

Es la implementación correcta para realizar esta tarea.

- **C)** Aumentar el número de réplicas del objeto en diferentes regiones geográficas.

Esta opción ni siquiera resuelve el problema.

Cuestión 12: Disponibilidad y "Rack Awareness" en HDFS

En la configuración por defecto de **HDFS**, cuando un archivo tiene un factor de replicación de 3, ¿cómo distribuye el **NameNode** los bloques para balancear la disponibilidad y el rendimiento?

- **A)** Los tres bloques se colocan en nodos aleatorios de tres bastidores (*racks*) diferentes.

No es la realidad.

- **B)** Las tres copias se guardan en el mismo nodo para minimizar la transferencia de datos por red.

Podría ser la opción correcta, pero no.

- **C)** Dos copias se crean en el mismo bastidor (en nodos distintos) y la tercera en un bastidor diferente.

Así es, esta es la opción correcta.

Cuestión 13: Escalado de sistemas en Big Data

Un administrador de base de datos nota que el servidor actual ha llegado al límite de su capacidad de CPU y RAM. Según las fuentes, ¿cuál es el enfoque de crecimiento preferible para un entorno de Big Data?

- **A) Escalado Vertical**, añadiendo más recursos (CPU, RAM, disco) al servidor existente.

Es una concepción errónea.

- **B) Escalado Horizontal**, añadiendo más servidores (nodos) al clúster y fragmentando los datos (*sharding*).

La cuestión correcta.

- **C)** Reemplazar el hardware por sistemas de almacenamiento de bloques (SAN) de alto rendimiento.

No se ha concretado.

Cuestión 14: Flexibilidad de esquema en NoSQL

Una aplicación de comercio electrónico necesita añadir nuevos atributos a sus productos (como "color", "talla" o "voltaje") de forma dinámica y diferente para cada artículo. ¿Por qué una base de datos de **Documentos** es más apta que una **Relacional**?

- **A)** Porque el modelo relacional es rígido y requiere que todas las filas tengan las mismas columnas, mientras que NoSQL permite libertad de esquema (*schemaless*).

Se realiza la respuesta de la cuestión.

- **B)** Porque las bases relacionales no permiten almacenar datos en formato JSON.

No ha sido concretado correctamente.

- **C)** Porque las bases de documentos eliminan la necesidad de realizar copias de seguridad.

Lo realizan, pero no con tal función relacional.

Cuestión 15: Diferencia entre Tiempo Real y Series Temporales

¿Cuál es la diferencia principal en el enfoque de uso entre una base de datos de **Tiempo Real** y una de **Series Temporales** según su taxonomía NoSQL?

- **A)** No hay diferencia; son dos nombres para la misma tecnología.

Sí, la hay.

- **B)** Las de tiempo real solo almacenan datos de menos de una hora, mientras que las de series temporales son para datos de años.

Es lo contrario.

- **C)** Las de tiempo real se enfocan en una alta velocidad de procesamiento y sistemas de alerta, mientras que las de series temporales priorizan el análisis retrospectivo y el pronóstico.

Es la diferencia principal entre ambos sistemas.