TIPO TEST FBD

Al aplicar DROP TABLE sobre una tabla T:

Se borrarán las tuplas de la tabla, aunque se mantenga la tabla en nuestro esquema.

Es posible que se produzca un error, aunque la tabla T exista en nuestro esquema.

Ninguna de las otras opciones es cierta.

Considere la tabla PIEZA (codpie, nompie, color), cuya clave primaria es codpie; considere la consulta: encontrar los colores que cumplen que el número de piezas de la tabla que tienen ese color es exactamente 2:

La consulta se puede resolver tanto con SQL como con Álgebra Relacional.

La consulta no se puede resolver con SQL.

La consulta se puede resolver con SQL, pero no se puede resolver con Álgebra Relacional.

Considere una tabla T(A,B). En relación con los operadores del Algebra Relacional:

La consulta $\pi_A(T) \cup \pi_B(T)$ equivale a $\pi_{A,B}(T)$

La consulta $\sigma_{A=a_1}(T) \cap \sigma_{B=b_1}(T)$ se puede plantear con una sola selección sobre la tabla T.

Ninguna de las otras opciones es cierta.

Cuando operamos con dos tablas que están conectadas por una clave externa formada por dos atributos:

Es posible que su producto cartesiano devuelva menos tuplas que su reunión.

Es imposible que su producto cartesiano devuelva más tuplas que su reunión.

Ninguna de las otras es cierta.

Considere la siguiente consulta en SQL: SELECT codpro FROM ventas.

El resultado de dicha consulta, coincidirá, en cualquier caso, con la proyyecion por el campo codpro de la tabla ventas.

El resultado de dicha consulta coincidirá,en cualquier caso, con el de la siguiente consulta:
SELECT codpro FROM ventas WHERE cantidad >= 1000

UNION

SELECT codpro FROM ventas WHERE cantidad < 1000;

Ninguna de las otras opciones es cierta.

Considere dos tablas T1 y T2 con esquemas equivalentes. Considere la siguiente expresión en Álgebra Relacional: (T1 – T2)UT2. El resultado de aplicar dicha operación sobre dos instancias t1 y t2 de las tablas:

Producirá siempre como resultado una instancia t contenida en o igual a t2.

Producira siempre como resultado la instancia t1.

Ninguna de las otras opciones es cierta.

En relación con la capacidad de consulta:

Todas las consultas que se pueden resolver con Algebra Relacional, se pueden resolver con SQL.

Todas las consultas que se pueden resolver con SQL, se pueden resolver con Álgebra Relacional.

No sabe o no contesta.

Considere dos tablas T1 y T2 tales que el esquema de T2 está contenido en el de T1. Considere la siguiente expresión en Álgebra Relacional: (T1/T2) x T2. El resultado de aplicar dicha operación sobre dos instancias t1 y t2 de las tablas:

Producirá como resultado una instancia t contenida en o igual a t1.

Producirá siempre como resultado la instancia t1.

Ninguna de las otras opciones es cierta.

Cuando operamos con dos tablas que están conectadas por una clave externa:

Su producto cartesiano siempre devuelve la misma cantidad de tuplas que su reunión.

Su producto cartesiano puede devolver más tuplas que la reunión, aunque no siempre.

Su producto cartesiano puede devolver menos tuplas que su reunión, aunque no siempre.

En relación con los operadores fundamentales y no fundamentales del Algebra Relacional:

Solo una parte de los operadores no fundamentales pueden reproducirse utilizando operadores fundamentales.

Todos los operadores no fundamentales pueden reproducirse utilizando operadores fundamentales.

Todos los operadores fundamentales pueden reproducirse utilizando operadores no fundamentales.

En general en Oracle si añadimos un índice a una tabla:

Habrá que reescribir todas las sentencias de consulta que hayamos planteado previamente sobre dicha tabla, puesto que pueden dar errores sintácticos.

Habrá que reescribir solo las sentencias de consulta en las que aparezca la tabla más de una vez, puesto que darán errores sintácticos.

Ninguna de las otras opciones es cierta

Piensa en el esquema de suministros que hemos utilizado en las prácticas con Oracle. En relación con el comando DESCRIBE:

Se trata de un comando fundamental, puesto que la información que proporciona sobre dicas tablas es imposible de obtener de otra manera.

La información que proporciona el comando DESCRIBE sobre dichas tablas esta almacenada en el catalogo de la base de datos.

Ninguna de las otras opciones es cierta.

Un índice primario:

Nunca puede ser denso

Nunca puede ser no denso

Ninguna de las otras es cierta.

En general para acceder a los datos:

Ni índices ni acceso directo son mejores en términos absolutos frente al otro.

El acceso directo es siempre la mejor alternativa si está disponible en el SGBD.

Los índices son siempre la mejor alternativa si está disponible en el SGBD.

En un índice de mapa de bits:

El numero de entradas es el doble del número de valores que tiene la clave por la que se quiere indexar el fichero.

El numero de entradas coincide con el numero de registros que hay en el fichero que se quiere indexar.

El numero de entradas coincide con el numero de valores que tiene la clave por la que se quiere indexar el fichero.

En la aproximación del método de acceso a la base de datos vista en clase:

El gestor de archivos y el gestor de disco son dos elementos del S.O que permiten la transformación entre páginas almacenadas y sectores de disco.

El gestor de archivos y el gestor de disco son dos formas de decir lo mismo

Ninguna de las otras es cierta.

Cuando se utilizan técnicas de Hashing básico:

Deja de ser necesario usar estrategias para solventar colisiones, puesto que estas no pueden presentarse.

En general, no ayuda a conocer zonas del dominio del campo clave donde pueden presentarse más valores.

Es posible que se produzcan huecos.

En relación con los índices:

No se pueden utilizar si el fichero de datos no está ordenador por la clave del índice.

Cuantos más campos añadamos a la clave de búsqueda, más rápido será el proceso de búsqueda en el índice por uno de esos campos.

Ninguna de las otras es cierta

En general, cuanto mayor es el número de colisiones que produce una función hash

Las búsquedas tenderán a ser más lentas

Menos huecos se producirán

Ninguna de las otras es cierta

Un objetivo primordial en relación con el método de acceso es:

Evitar la aparición de valores nulos

Ocultar al usuario el verdadero valor de la clave física

Ninguna de las otras es cierta

Considere las tablas organizadas por índice (IOT):

Son la forma de implementar los arboles B+ en bases de datos

La clave de búsqueda del índice no tiene nada que ver con la clave física

Ninguna de las otras es cierta

En el nivel interno

El almacenamiento persistente de los datos se hace con dispositivos de memoria de las primeras posiciones de la jerarquía de memoria para que las operaciones sean más rápidas.

El almacenamiento persistente de los datos se hace con dispositivos de memoria de las primeras posiciones de la jerarquía de memoria para que sea más barata la implantación del sistema.

Ninguna de las otras es cierta

Considere que se está usando Hashing dinámico. En un momento dado, al insertar un nuevo registro en un cubo con profundidad local igual a la profundidad global:

En ningún caso habrá que desdoblar la tabla índice.

Es posible que no haya que desdoblar la tabla índice

Necesariamente habrá que desdoblar la tabla índice.

Considere un fichero secuencial indexado:

Cuanto mas grandes sean los registros del fichero índice, más se ayudará a acelerar el proceso de búsqueda.

Cuando más pequeños sean los registros del fichero índice, más se ayudará a acelerar el proceso de búsqueda.

Ninguna de las otras es cierta

El Hashing dinámico:

Va asignando más espacio en disco a zonas del dominio de la clave donde se van presentando más valores en la instancia de la base de datos.

Asigna mas espacio en disco a aquellas zonas del dominio de la clave que teóricamente van a presentar más valores en la instancia de la base de datos.

No utiliza ninguna estructura auxiliar, aparte del propio fichero que almacena los registros.

En un árbol B+:

Cuanto menos es M, mayor tiende a ser el número de niveles.

Cuanto mayor es M, mayor tiende a ser el número de niveles

Ninguna de las otras es cierta.

La BD en el nivel interno se puede representar de distintas formas, pero:

Nunca deben ubicarse juntos registros de distinto tipo para facilitar operaciones.

Cuando se ponen juntos los registros del mismo tipo, se optimizan operaciones como las de reunión de tablas.

Ninguna de las otras es cierta.

Cuando se utiliza Hashing dinamico:

No se puede utilizar una función de direccionamiento.

Es imposible que se produzcan colisiones.

Ninguna de las otras es cierta.

En relación con el método de acceso a la BD, las páginas o bloques de la BD deben tener un tamaño múltiplo de las páginas o bloques del sistema operativo

Para garantizar que la memoria y el disco duro sean compatibles.

Para aprovechar bien cada operación de E/S en disco.

Para que sea fácil hacer cuentas a la hora de organizar los datos.

Cuando se organiza el acceso a los datos de un fichero mediante el uso de índices:

En general, lo mejor es usar tantos índices como configuraciones de consulta pueden plantearse.

Si el espacio de disco no es un problema, lo mejor es usar tantos índices como configuraciones de consulta puedan plantearse.

Ninguna de las otras es cierta.

Al respecto de los índices jerárquicos:

El numero de niveles depende, entre otras cosas, del número de registros del fichero de datos.

Los arboles B+ no son un ejemplo de índice jerárquico.

Los arboles B no son un ejemplo de índice jerárquico.

En general, cuando se utiliza un índice denso:

Al realizar una operación de borrado de un registro en el fichero nunca hay que actualizar el índice.

Al realizar una operación de inserción de un nuevo registro en el fichero nunca hay que actualizar el índice.

Ninguna de las otras es cierta.

En la indexación con arboles B:

El orden de un árbol influye directamente en el número de niveles.

El orden de un árbol se determina por el tamaño de página/bloque que se asigna a los nodos.

Se puede montar un árbol B sobre cualquier campo.

Las otras respuestas indicadas son ciertas.

La técnica de acceso directo:

No utiliza área de desbordamiento.

Garantiza siempre que encuentro una fila con una sola lectura de bloque.

No permite realizar la lectura secuencial de datos en un rango.

No necesita una estimación previa del número de registros.

El objetivo principal de los mecanismo de indexación y métodos de acceso es:

Localizar datos requeridos con el número mínimo de operaciones de lectura en disco.

Garantizar que no se duplique la clave primaria

Acceder a los datos de una tabla de forma ordenada.

El clúster:

Perjudica la lectura individual de las tablas que contiene

Acelera las consultas que involucran la reunion natural de las tablas que contiene.

Es una estructura inter-archivo.

Todo lo anterior es cierto.

En un índice denso:

El índice ocupa lo mismo que la tabla indexada.

El numero de registros del índice es menor que el numero de registros de la tabla indexada.

El numero de registros del índice es igual al numero de registros de la tabla indexada.

Ninguna de las otras respuestas propuestas es cierta.

En el Hashing extendido:

Lo mejor es que la pseudollave se ajuste al tamaño del índice que se guarda en memoria.

Lo mejor es que la pseudollave tenga muchos dígitos.

Hay que reservar de antemano un número fijo de bloques.

No hay que tener una estimación del número de registros a almacenar.

Las paginas que componen un archivo almacenado:

Contienen siempre registros de una misma tabla.

Pueden ser de distinto tamaño dependiendo del tamaño de los registros que se almacenen en ellas.

Tienen que estar consecutivas en disco.

Todo lo anterior es falso.

Sean F y D las tablas procedentes de una entidad fuerte y una débil, respectivamente. Las filas de D se recuperan con las de F, y rara vez, por separado. La mejor opción sería:

Indexar D por el atributo que tiene en común con F.

Crear una vista en la que aparezcan los datos de ambas tablas en el formato adecuado.

Almacenarlas conjuntamente en un clúster.

Poner en D como clave externa el atributo que tiene en común con F.

Indica la afirmación verdadera:

Se pueden montar tantos índices densos como se necesiten.

En ficheros no ordenados físicamente se pueden montar índices no densos

El índice no denso es el único mecanismo de indexación posible cuando los datos están ordenados físicamente.

Un árbol B solo se puede montar sobre la clave física de un archivo.

La sentencia CREATE TABLE provoca:

La creación de u n nuevo archivo almacenado

La creación de un nuevo fichero en disco.

Un índice no denso:

Permite realizar preguntas de tipo existencial sin acceder al fichero de datos.

Exige que los registros esten ordenados físicamente

El rendimiento desciende considerablemente cuando se realizan muchas inserciones o borrados en la tabla.

Es adecuado para consultas por rangos de valores del campo clave.

El record identifier (RID):

Es un campo de un índice denso

Es un campo de un indice no denso

Puede servir para identificar varios registros.

Se calculan mediante un algoritmo de direccionamiento.

El índice por clave invertida:

Se puede montar sobre cualquier campo de la tabla.

Ayuda a distribuir mejor los datos en el espacio de almacenamiento.

Es útil para recorrer una tabla por el campo clave en orden inverso al establecido.

Todo lo anterior es cierto.

En Hashing dinámico si el numero de registros por bloque es 4 y tengo alrededor de 1000 registros, el numero de bits necesario para la tabla hash es:

8 4 16 5

Cuando se necesita acceder a la tabla alumnos por rangos de notas el mejor mecanismo es:

Un índice no denso.

Un índice denso

Un índice de mapa de bits

Hashing básico.

Cuando la cardinalidad del campo por el que se indexa una tabla es muy baja, el mejor mecanismo de indexación es:

Un árbol B donde el conjunto secuencia sea denso.

Un árbol B donde el conjunto secuencia sea no denso.

Algún mecanismo de acceso directo.

Un índice de mapa de bits.

Indica cual de estas afirmaciones es verdadera:

El tamaño de los bloques físicos y de las páginas deben ser independientes.

Cada archivo almacenado del nivel interno debe estar en un fichero físico separado.

El nivel interno de una base de datos está enteramente gestionado por el SO del ordenador.

Las paginas que componen un archivo almacenado no tienen porque estar consecutivas en disco.

Con la consulta SELECT codpro, sum(cantidad) FROM ventas GROUP BY codpro:

Se puede crear una vista, pero no será actualizado.

No se puede crear una vista por estar agrupada.

Se puede crear una vista y será actualizable porque solo usa una tabla.

Ninguna de las anteriores es cierta.

Las tablas organizadas por índice (IOT):

No admiten ningún tipo de índice

No tienen llave primaria

No pueden recuperarse de forma ordenada.

Se organizan como un árbol B cuyas hojas contienen las tuplas.

Indica cual de estas afirmaciones es FALSA:

En el Hashing extendido una mala elección en el tamaño de las paginas puede obligar a reorganizar completamente la estructura.

El Hashing extendido es muy eficaz porque la tabla hash va en memoria principal

El índice no denso es el único mecanismo de indexación posible cuando los datos están ordenados físicamente

La actualización de los archivos puede no influir en la actualización de los índices no densos.